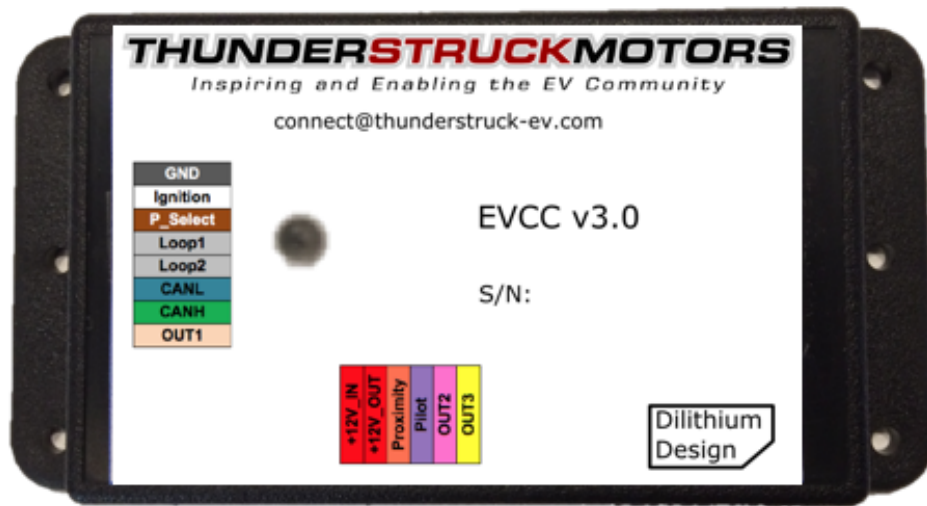


# ThunderStruck Motors EV Charge Controller v3.0

EVCC v3.0



Document Revision 3.1.3

© 2018, 2025 Dilithium Design

## Contents

OVERVIEW.....	4
INSTALLATION and THEORY OF OPERATION .....	6
Mechanical.....	6
Powering .....	7
CAN Bus (CANH, CANL) .....	7
Profile Selection (P_Select) .....	9
Cell Loop (Loop1, Loop2).....	9
Mappable Outputs (OUT1, OUT2, OUT3).....	10
LED Operation.....	11
Serial Port.....	11
Firmware Upgrade .....	12
CHARGER SUPPORT .....	12
TSM2500.....	13
ELCON and TC.....	13
STEALTH.....	13
LEAR .....	14
NLG5 / BRUSA .....	14
VOLT .....	14
BASIC (non-CAN charger).....	14
CHARGING .....	14
J1772 Type 1 .....	15
J1772 Type 2.....	16
Three-phase Power .....	18
Charge Parameters .....	19
Charge Profiles.....	20
Time of Use Charging.....	20
Driveaway Protection.....	21
DC/DC CONVERTER SUPPORT .....	21
START-UP CHECKLIST .....	21
SERIAL INTERFACE .....	22
Startup Banner.....	22
help.....	22
show .....	23
show version .....	24
show config.....	24
show history.....	26
set .....	27

set <> .....	27
set bms .....	29
set plug.....	29
set canbr.....	29
set out1 (set out2, set out3).....	29
set charger<n> .....	29
set evsewait.....	30
set profile <n>.....	30
set map.....	32
set linev, set linec.....	33
set maxv, set maxc.....	33
set maxbc .....	33
set termc.....	33
set termt .....	33
set fin_maxv, set fin_maxc, set fin_termt.....	33
set flt_maxv, set flt_maxc, set flt_termt .....	33
set ddtype, set ddvoltage.....	33
reset .....	34
reset history.....	34
reset profile <n>.....	34
reset config.....	34
enable   disable .....	34
enable   disable canterm.....	34
enable   disable topbalance .....	34
enable   disable loopground .....	35
enable   disable slowstart .....	35
enable   disable ignorethcensus.....	35
trace .....	35
trace charger.....	35
trace can.....	36
trace state .....	36
trace verbose .....	37
trace off.....	37
measure .....	37
measure loop.....	38
measure proximity .....	38
measure pselect.....	38
upgrade.....	39

APPLICATION NOTES .....	40
Configuring the EVCC with Multiple Chargers .....	40
Setting the CAN address of a TSM2500 Charger .....	42
Integration with a CAN Enabled BMS .....	43
Integration with Motor Controllers .....	44
Integration with DCDC Converters .....	44
Charging Lead Acid Batteries .....	44
Bulk Charge .....	45
Finishing Charge .....	45
Float Charge .....	46
Limitations .....	46
Charging Lithium-Ion Batteries .....	47
WARRANTY and SUPPORT .....	49
Document History .....	49

## Figures

Figure 1 – Standard EVCC System Diagram .....	5
Figure 2 – Standard EVCC Connections .....	5
Figure 3 – EVCC Enclosure .....	6
Figure 4 – Push-In Connector .....	7
Figure 5 – CAN Network Diagram .....	8
Figure 6 – External Face of J1772 Type 1 Inlet Port .....	15
Figure 7 – EVCC Output Conditions .....	10
Figure 8 – EVCC LED Blink Patterns .....	11
Figure 9 – Multiple Chargers - System Diagram .....	40
Figure 10 – Flooded Lead Acid Charging Profile (Trojan) .....	45
Figure 11 – Face of J1772 Type 2 Female Cord Connector .....	16
Figure 12 - Wiring J1772 Type 2 with One Phase Power .....	18
Figure 13 - Wiring J1772 Type 2 with Three Phase Power .....	18

## OVERVIEW

The Electric Vehicle Charge Controller (EVCC) controls the operation of a CAN enabled charger. Parameters including maximum voltage, maximum current, and total charge time are configured, saved in nonvolatile memory, and used during charging.

This document covers the EVCC 3.0 which has been optimized for use in Electric Vehicles using the J1772 Level 2 charge protocol (either J1772 Type 1 or Type 2). Note that the EVCC basic model, which did not include J1772 charging capabilities, has been discontinued.

The EVCC supports several CAN enabled chargers, including the ThunderStruck TSM2500 charger. Charge voltage, charge current, and overall charge time are controlled by the EVCC. A Constant Current to Constant Voltage (CCCV) charge curve is supported for Lithium Batteries, and a three-phase charge cycle is supported for Lead Acid Batteries.

The EVCC can control up to four parallel chargers for faster charging. When multiple chargers are configured, each charger is individually CAN addressed. Work is divided evenly between the chargers and statistics are gathered and recorded on each charger separately. Chargers may be configured for J1772 Type 2 multiple phase charging.

Determining cell overvoltage errors and cell undervoltage errors within the pack are functions of a Battery Management System (BMS). The EVCC can be configured to interface with a BMS either by a cell loop or by CAN messages. When a CAN BMS is used, the EVCC can also be configured to lower the charging current when a cell exceeds a configurable threshold.

Charging normally completes at the end of a charge cycle, which, for Lithium batteries, occurs when the pack voltage approaches the target maximum voltage and the charging current drops below a minimum configured charge current. The EVCC will also stop charging if the J1772 plug becomes unplugged, the BMS reports an error, there is loss of communication between the EVCC and Charger or BMS, or the maximum configured charge time is reached.

Charging history is recorded for each charger. History includes the reason that charging stopped, total charge time, maximum voltage, maximum current, final current, and watt hours delivered.

The EVCC enclosure contains a serial port jack, an LED, and two system connectors.

The figures below show the system diagram and connections for the EVCC:

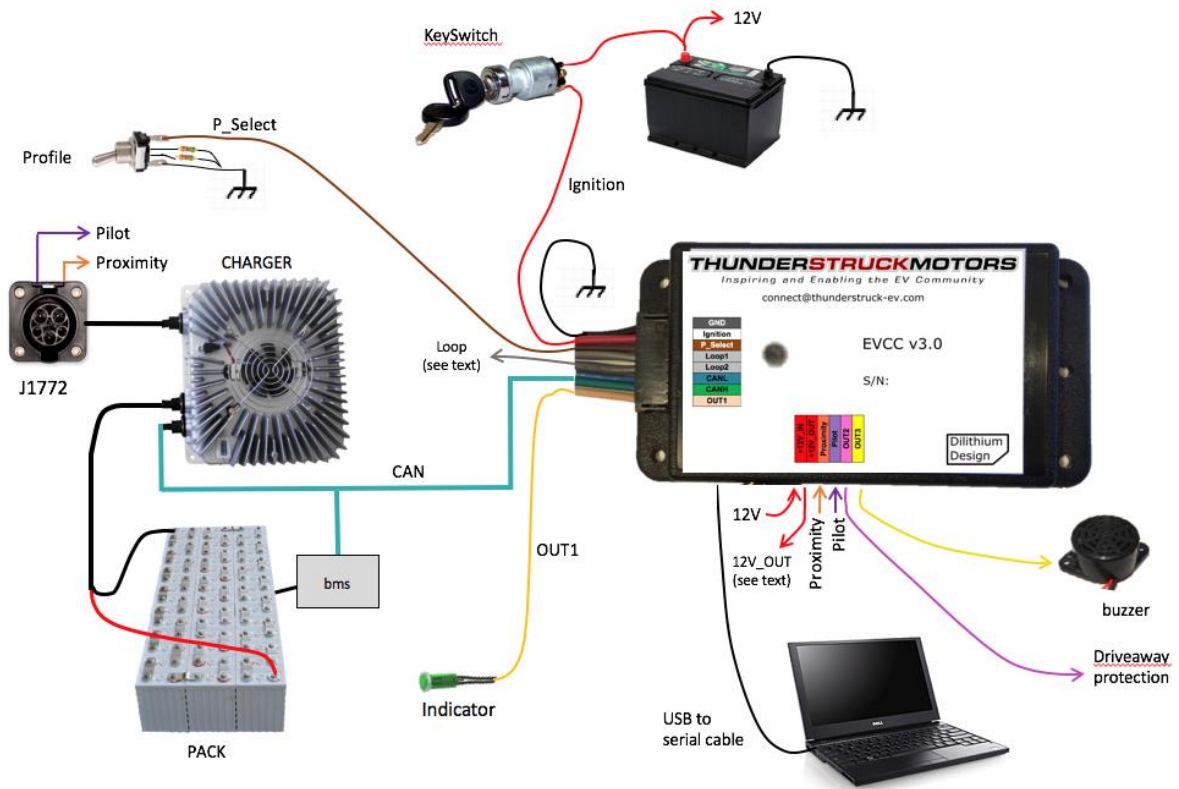


Figure 1 – Standard EVCC System Diagram

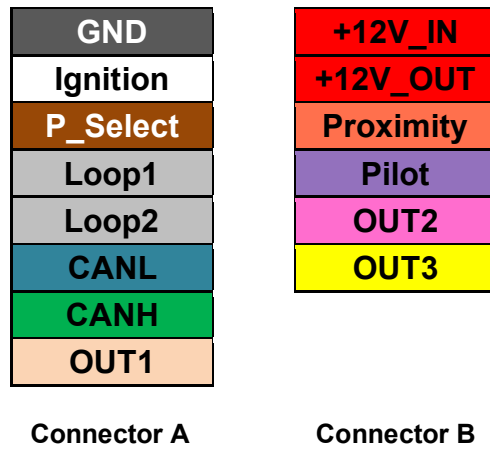


Figure 2 – EVCC Connections

Connector A has the following connections:

- **GND** connects to Ground
- **Ignition** connects to switched 12V power<sup>1</sup>
- **P\_Select** is a “charge profile” selection input
- **Loop1** and **Loop2** are used for the cell loop supervision circuit
- **CANH, CANL** connects to the CAN network
- **OUT1** is a mappable output

Connector B has the following connections:

- **+12V\_IN** connects to “always on” 12V power
- **+12V\_OUT** provides switched 12V power
- **Proximity** connects to the J1772 Proximity signal
- **Pilot** connects to the J1772 Pilot signal
- **OUT2** and **OUT3** are mappable outputs

## INSTALLATION and THEORY OF OPERATION

### Mechanical

The EVCC enclosure is a Serpac WM010I, a 4.61 x 2.32 x 0.6 plastic enclosure with mounting flanges.

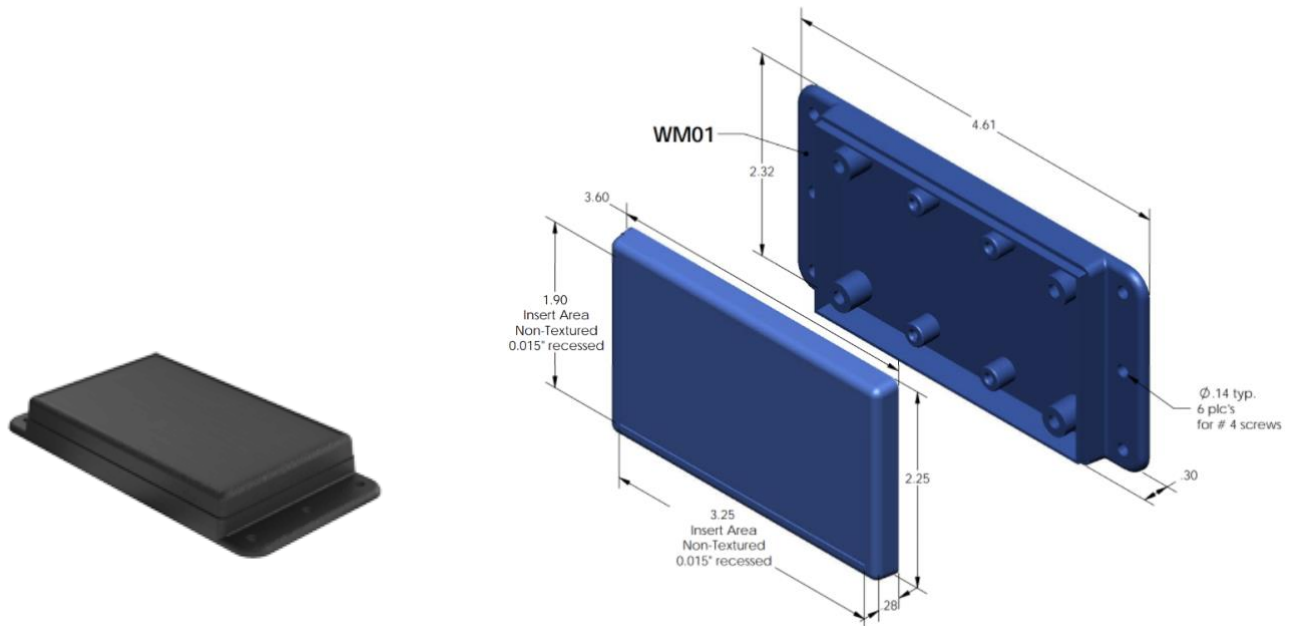
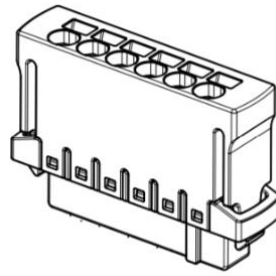


Figure 3 – EVCC Enclosure

Connector A and Connector B are “push-in” connectors. Stranded 20-gauge TXL wire is recommended for the best fit and is included with the EVCC kit. However, the connectors will also accept 20-24 gauge stranded or solid wire. To make a connection, strip the wire back 1/4". Twist the wire end and insert. Be sure

<sup>1</sup> **Ignition** input is connected to switched 12V power, such as a 12V EV keyswitch. This is NOT connected to keyswitch systems using traction pack voltage, which would damage the EVCC.

that all strands of wire get correctly inserted to prevent shorting between adjacent wires. When properly inserted, the wire insulation is slightly recessed in the connector body.



**Figure 4 – Push-In Connector**

Removing the wire from the connector requires a removal tool, supplied. To remove a wire, insert the tool into the associated slot above the wire and wiggle it in. This will collapse the spring holding the wire and the wire can be removed.

The connector part numbers are

- 8p Connector A                      Harting 14310813101000
- 6p Connector B                      Harting 14310613101000

### Powering

The **GND** input is connected to chassis ground or 12v power negative.

The **Ignition** input is normally connected to 12V keyswitch. This will power the EVCC whenever the EV keyswitch is “on”.

The **+12V\_IN** connection is connected to a source of “always-on” 12V power. The EVCC has an “autostart” feature which powers up the EVCC when the J1772 charge plug is inserted.

When the EVCC powers up, it will attempt a new charge cycle. If a charge completes, either normally or due to an error, it will remain powered up (if the 12V\_IN connection is powered) or will power itself off (if not). To attempt a new charge cycle, the J1772 charge cable must be removed and then re-inserted.

The EVCC provides a **+12V\_OUT** output which is enabled when the EVCC is powered up. This output may be used to power downstream devices such as a BMS, instrumentation, or a relay. This output is rated at 500ma.

The EVCC draws approximately 3ma when powered off, and approximately 35ma when powered on.

### CAN Bus (CANH, CANL)

CAN is a robust communications protocol designed for automotive applications. A CAN Bus uses a two-wire interface; the signals are designated CANH (“CAN high”) and CANL (“CAN low”). Not shown, but necessary, is that each node on the CAN network must share a common ground (e.g., chassis ground). A

CAN network is a daisy-chain, multi-station network that must be terminated on both ends of the string by 120ohm termination resistors.

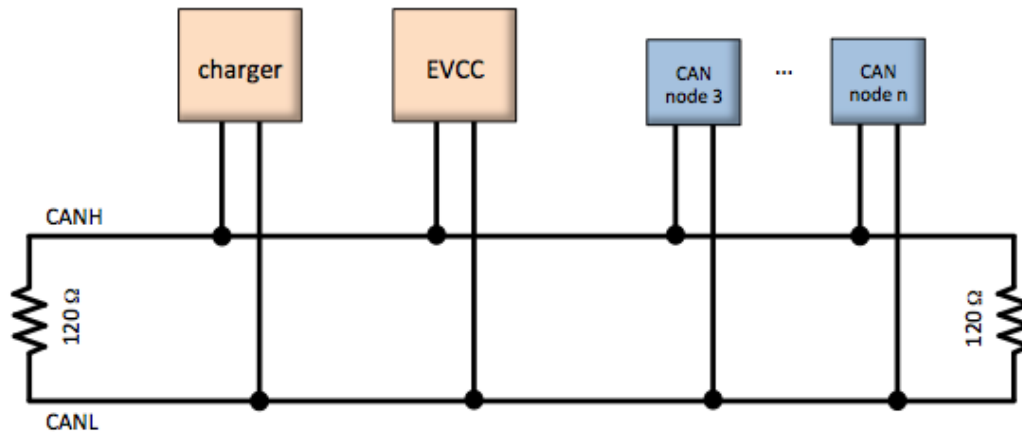


Figure 5 – CAN Network Diagram

CAN wiring should be kept short and the conductor pair should be twisted together along its entire length. Wiring stubs between the CAN network and a node should be kept as short as possible, ideally less than a few inches. Network wiring should be placed away from EMI (Electro-Magnetic Interference) such as the motor and controller, and parallel runs next to EV 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, a short run of twisted wire normally works fine. For longer runs, more nodes, or cases where EMI may be an issue, shielded cable may be used. In that case, the shield should be connected to chassis ground at one end only.

To simplify CAN network wiring, the EVCC contains an internal, configurable, CAN termination resistor. By default, this termination is enabled. (When the EVCC is used as an intermediate node, the termination resistor may be disabled by using the CLI command “**disable canterm**”).

Some CAN devices contain an internal termination resistor and must be installed at the end of the CAN string. In order to verify the proper CAN terminations, make all connections and measure the resistance between CANH and CANL. The resistance between CANH and CANL should be 60 ohms, indicating the presence of two 120-ohm resistors in parallel.

Note that the internal EVCC termination resistor is not bridged onto the CAN network unless the EVCC is powered up. This design choice covers the vast majority of installation scenarios<sup>2</sup>.

The EVCC CAN interface supports 125kbps, 250kbps and 500kbp data rates. The TSM2500 charger has a default datarate of 250kbps, and ELCON chargers may use either 250kbps or 500kbps, depending on their factory settings. Support is also available for the Lear, Volt and Stealth chargers which, require a CAN data rate of 500kbps. The EVCC CAN data rate can be set to match the charger using the command “**set canbr.**”

<sup>2</sup> One exception is if the EVCC is a terminal node and the network needs to be active when the EVCC is powered down. In this case, the EVCC CAN termination resistor would not be enabled and network quality may suffer. In this case, the recommendation is to disable the termination resistor in the EVCC and connect a physical resistor instead.

**NOTE:** All nodes on a CAN network must be configured with the same CAN data rate. Otherwise message corruption will occur, and the CAN network will perform poorly, if at all. If the CAN network is not performing correctly, it is recommended that the user first check for proper network termination. Next, remove all but two nodes from the network to verify correct operation and add nodes back one by one.

Two types of CAN messages are used to control a CAN enabled charger. The first, sent to the charger, provides the charger with the desired values of charge voltage and charge current, and the second message, sent from the charger, reports actual charging voltage, current, watts, status and temperature.

In most cases, EVCC/Charger CAN messages are sent approximately twice a second, both from the EVCC to the charger and from the charger to the EVCC. If the EVCC stops receiving periodic messages from the charger, charging will terminate with a termination reason of “Charger Rx Timeout.”

The EVCC can also interpret BMS status messages over CAN. BMS status can be used as an alternate to (or in addition with) the cell loop to indicate pack fault conditions. See *Integration with a CAN enabled BMS*, below, for more details.

CAN messages may be lost or corrupted as the result of EMI, stubs that are too long, or improperly terminated cables. In order to verify correct operation, there is both low level tracing (“**trace can**”) as well as high level tracing (“**trace charger**”) to show CAN message traffic. Enter “**tr off**” to stop the trace.

### Profile Selection (P\_Select)

Up to four charging profiles may be defined with the EVCC, numbered from 1 to 4. Each profile selects a distinct copy of charging parameters. By default, Profile 1 is created and is used by default. A new profile may be defined using the “**set profile**” command and then editing the charge parameters for that profile.

When charging, the **P\_Select** input is used to select the profile. The EVCC measures the resistance to GND at this input and determines four possible selections: “inf,” “20K,” “5K,” and “0”. If the **P\_Select** input is left unconnected, it will read “open” (or “infinite” resistance), and maps to “inf.” If the **P\_Select** input is shorted to ground it will measure “0”. If a resistor is connected between the **P\_Select** input and ground, the remaining two choices “20K” and “5K” can be selected. The EV designer may decide to leave this feature unused, connect the input to a switch to GND to enable two profiles, or connect it to a multi-position switch and a resistor network and enable up to four profiles.

Note that the four input values represent a “switch setting” and not a “profile number.” The mapping from input value to profile number is done using the command “**set map**.”

### Cell Loop (Loop1, Loop2)

The EVCC is intended to be used with a Battery Management System that monitors per-cell over voltage conditions when charging and per-cell undervoltage when driving. A simple BMS interface may be provided using the Cell Loop “go/no-go” contact closure. The EVCC can monitor pack health using the CAN interface, instead of, or in addition to, using the Cell Loop. See the command “**set bms**.”

**WARNING:** It is strongly recommended that per-cell monitoring be performed on the pack by a Battery Management System (BMS) for any lithium-ion pack, so that charging and discharging can

be stopped if any cell exceeds a high voltage or low voltage cutoff. Lithium batteries can be damaged and dangerous if overcharged or undercharged.

The Cell Loop may be configured in one of two ways.

By default, the Cell Loop circuit measures the resistance between **Loop1** and **Loop2**, by applying +5v to Cell Loop1, and checking for continuity. If there is continuity, then the loop is “good.” If the loop is open, it is “bad.” It is expected that the Cell Loop be implemented using solid state relays or optoisolators. Connecting the cell loop to the contacts of a mechanical relay is not recommended, as the cell loop current is limited to about 2ma. Running low current through the contacts of a mechanical relay may result in erratic operation.

Alternately, the Cell Loop may be configured to use an open collector input, when the **loopground** option is enabled. With this option, Loop1 is left unconnected. The loop is “good” if Loop2 is grounded, and “bad” if Loop2 is disconnected or “high impedance.” Note that the Orion BMS supports this type of output.

A cell loop error will result in the **BUZZER** condition, which may be directed to any of the mappable outputs.

### Mappable Outputs (OUT1, OUT2, OUT3)

The EVCC supports three mappable outputs (OUT1, OUT2 and OUT3).

When OFF, OUT1, OUT2, and OUT3 are all disconnected and high impedance. When ON,

- OUT1 is switched to 12V. This output is rated at 200ma.
- OUT2 is switched to ground. This output is rated to 200ma.
- OUT3 is switched to ground. This output is rated to 500ma.

The EVCC firmware there are several conditions that can be flexibly mapped to these outputs. These are:

Condition	Default	Function
LED	OUT1	tracks the state of the EVCC LED
CHARGE		ON when charging
EVSEDISC	OUT2	ON when the J1772 cable is disconnected
BUZZER	OUT3	ON when the Pack is in HVC or LVC or the cell loop is not good
HVC		ON when the Pack is in HVC (only available when using a CAN BMS)
LVC		ON when the Pack is in LVC (only available when using a CAN BMS)

**Figure 6 – EVCC Output Conditions**

See “**set out<n>**” in *Command Line Interface* below for examples of how to change output mappings.

### LED Operation

The EVCC LED has the following blink patterns:

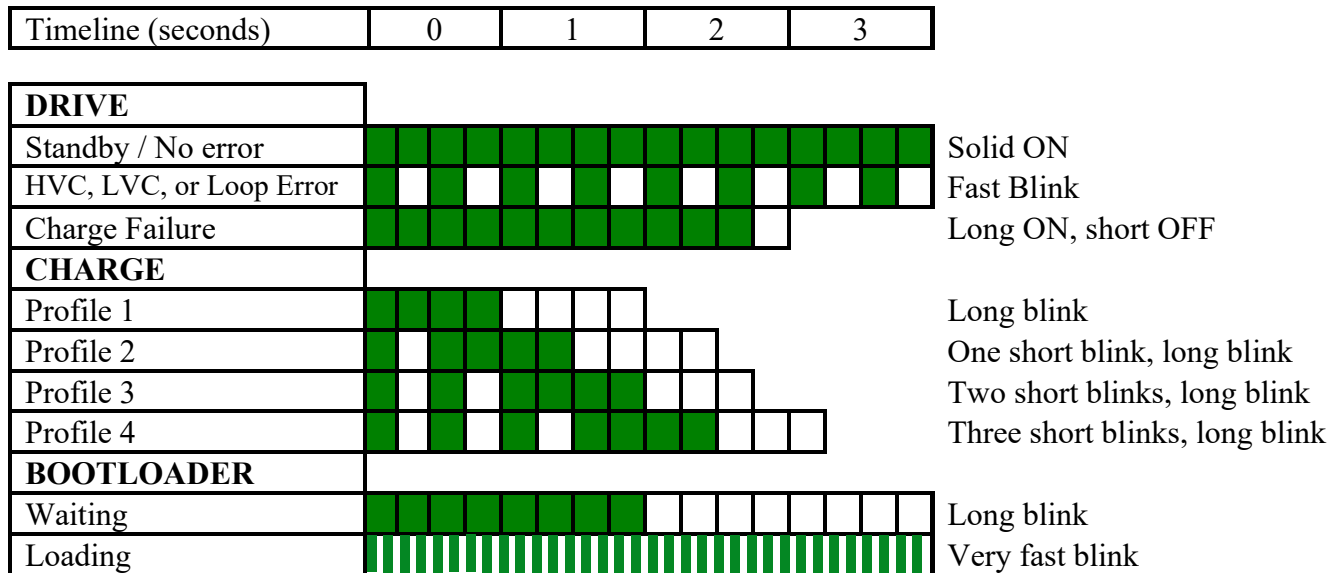


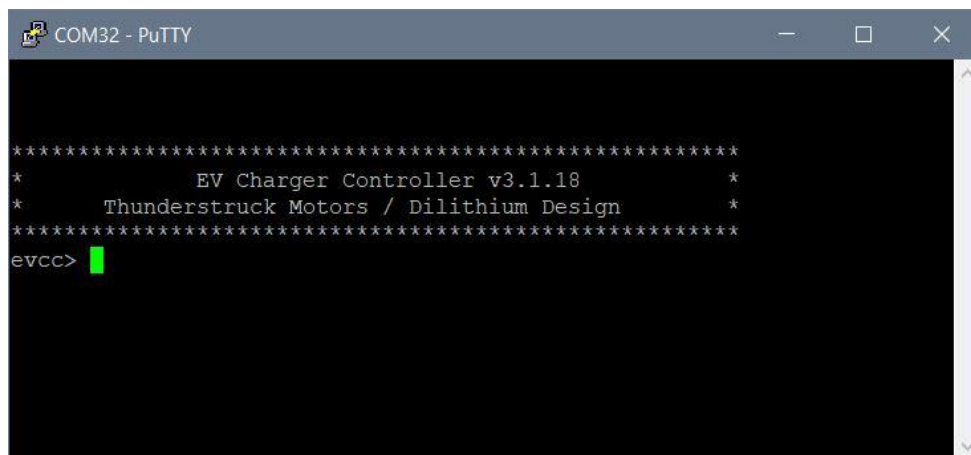
Figure 7 – EVCC LED Blink Patterns

### Serial Port

The EVCC uses a serial interface for configuration, firmware upgrade, and debugging. The serial interface is not required for normal operation.

Before using the serial port, a terminal application must be installed on a host PC such as PuTTY (for Windows) or Coolterm (for Mac). See the document *Serial Port Utilities*, and the Putty Install video for information about installing this application. These documents are located at the EVCC page tab *Manuals & Downloads* on the ThunderStruck website.

Connect the serial cable to the EVCC. Apply power to the EVCC by providing a 12V supply to **12V\_IN** and **GND**. Connect +12V to **Ignition**. The EVCC LED should start blinking (assuming the cell loop has not been hooked up yet). If Putty is the terminal program used on the host computer, the following banner should be displayed:



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 the commands and their syntax.

The serial interface supports several diagnostic commands: to verify proper wiring (“**measure**”), to trace CAN messages (“**trace can**”), to trace EVCC internal state changes (“**trace state**”) to trace charger operation (“**trace charger**”), and to retrieve charging history (“**show history**”).

### Firmware Upgrade

Firmware upgrade is supported which allows for bug fixes and feature addition. The customer should contact Thunderstruck Motors to receive the appropriate firmware release binary file.

A firmware upgrade is initiated at the EVCC by the “**upgrade**” command. Once this command is entered, a bootloader utility is run on a host PC which downloads the new binary file. See the document *Serial Port Utilities* on the ThunderStruck website for detailed instructions.

## CHARGER SUPPORT

This section describes the chargers supported by the EVCC.

When wiring the charger, required connections are the AC line input, the DC output to the pack and the CAN network connection. Note that a charger typically does not have an integrated CAN termination resistor; see *CAN Bus*, above, for design guidelines.

To charge, Charge Request messages are periodically sent from the EVCC to the charger which indicates the desired charge voltage and current and Charge Status messages are sent from the charger indicating pack voltage and pack current. Depending on charger type, there may be additional status or control fields in these messages. For example, the EVCC can control the LED blinking on some versions of the TSM2500 chargers.

A charger is configured in the EVCC using the command “**set charger<n> <chargertype>**”. The charger type determines the CAN data rate, message set and the CAN IDs that are used. See below for the Charger Types supported by the EVCC and CAN ID usage:

Charger Type	EVCC→Charger		Charger→EVCC	
	CAN Id	Rate	CAN Id	Timeout
<b>tsm2500</b>	0x18e54024	500ms	0x18eb2440	5 sec
<b>tsm2500_41</b>	0x18e54124	500ms	0x18eb2441	5 sec
<b>tsm2500_42</b>	0x18e54224	500ms	0x18eb2442	5 sec
<b>tsm2500_43</b>	0x18e54324	500ms	0x18eb2443	5 sec
<b>elcon</b>	0x1806e5f4	1 sec	0x18ff50e5	5 sec
<b>elcon_e7</b>	0x1806e7f4	1 sec	0x18ff50e7	5 sec
<b>elcon_e8</b>	0x1806e8f4	1 sec	0x18ff50e8	5 sec
<b>elcon_e9</b>	0x1806e9f4	1 sec	0x18ff50e9	5 sec
<b>stealth</b>	0x1806e0f4	1 sec	0x18ff50e0	5 sec
<b>lear</b>	0x0050	500ms	0x0617	5 sec
<b>nlg5</b>	0x0618	100ms	0x05f1	5 sec
<b>volt</b>	0x0304	200ms	0x0212	5 sec
	0x030e	200ms		

**Figure 8 - Charger CAN Id Usage**

Up to four chargers may be configured in the EVCC to support parallel charging.

### TSM2500

See the *Manuals & Downloads* tab on the TSM2500 Charger page at the ThunderStruck website for the manual and connector information. The TSM2500 CAN connector pinouts vary between manufacture dates. The CANL connection is commonly wired with a blue/white wire, and CANH is wired with a green/white wire. Although the TSM2500 charger does not have an integrated termination resistor, a 120-ohm termination resistor is provided by ThunderStruck Motors in the equipment box. The default CAN rate for TSM2500 chargers is 250kbps.

TSM2500 chargers are manufactured with default CAN IDs. The chargers support CAN Address reprogramming so the EVCC can communicate with parallel chargers. See below, *Setting the CAN address of a TSM2500 Charger*. However, note that CAN address programming for multiple chargers may have been done at ThunderStruck as part of an order.

### ELCON and TC

For more information on Elcon chargers, see: <https://elconchargers.com/>. Elcon chargers may be ordered with alternate CAN Ids, which allows parallel chargers to be controlled by the EVCC. They can also be ordered with CAN rates set to either 250kbps or 500kbps.

Modern versions of Elcon chargers are CAN-enabled, but older models like Elcon PFC chargers used an add-on module to enable CAN communication.

Elcon and TC Chargers are made by the same manufacturer and use the charger type “**elcon.**” For the purposes of the EVCC, these chargers are functionally identical.

### STEALTH

The Stealth charger uses different CAN Ids but the same CAN message format as the Elcon charger. The Stealth charger runs at a CAN rate of 500kbps.

The Stealth charger has less accurate voltage control and special accommodations are made in the code to work acceptably with this charger.

## LEAR

The EVCC supports Lear chargers repurposed from Coda EVs. See the “Lear Hi-V Charger” in the *Chargers & Accessories* section of the ThunderStruck website for more information about these chargers.

## NLG5 / BRUSA

The EVCC supports some NLG5 chargers.

## VOLT

The EVCC supports some Chevy Volt chargers. Note that the Volt charger expects two separate CAN messages, one which enables the charger and one that provides the charger parameters.

## BASIC (non-CAN charger)

The EVCC supports a “basic” non-CAN controlled charger. This type of charger is configured by entering “**set charger basic**” in the EVCC.

When the J1772 plug is inserted, the EVCC will operate the J1772 pilot diode to enable the EV Supply Equipment (EVSE). The EVCC will stop charging if a BMS error occurs (such as cell HVC), the charge plug is disconnected, the charger reaches an internal termination voltage, or the `termt` limit is exceeded.

When the EVCC is configured with both “**set charger basic**” and “**set plug direct**” settings, a configurable EVCC output can control a charger power relay in the event of a fault posted by the BMS. For example:

```
evcc> set out1 hvc
```

When performing a “basic” charge, the configurable charge parameters `maxv`, `maxc`, and `termt` have no effect. A charge history will be kept however, statistics such as charge voltage, charge current, and watt-hours will all be reported as zero.

## CHARGING

The EVCC supports AC charging using SAE J1772 Type 1, SAE J1772 Type 2 and “Direct Plug” charging.

SAE J1772 Type 1 and SAE J1772 Type 2 charging are functionally similar. The charge station / EVSE provides high voltage AC power to a charger onboard the EV. The charge connector provides AC power as well as “proximity” and “pilot” signals which indicate EVSE connectivity and charge current capability and allow the EV to enable or disable charging.

The “Direct Plug” option allows for a direct connection from external power directly to an onboard charger. The J1772 proximity and J1772 pilot signals are not used. Plug connectivity is signaled by the presence (or lack) of CAN messages from the charger. There is no negotiation of charge station capability, and all charge parameters must be configured in the EVCC beforehand.

The command “**set plug**” configures the charge plug type; this command supports three options **J1772**, **J1772T2**, and **direct**.

Configuring the charger type determines the CAN messages that the EVCC will use to control the charging process. Up to four chargers may be defined (**charger**, **charger2**, **charger3** and **charger4**). However, each

defined charger must be a unique type, with a unique CAN Id (see Figure 8). J1772 Type 2 charging supports three phase charging. In this case, each charger may be assigned to a specific phase (L1, L2 or L3).

If multiple chargers are defined, the EVCC will divide requested charge current evenly between chargers.

When charging, the EVCC sends periodic CAN messages to the chargers that contains Target Charge Voltage and Target Charge Current. It is the responsibility of the charger to limit charge voltage and charge current to these values and periodically report the pack voltage and charge current. While charging, pack voltage will rise and as the measured pack voltage approaches the Target Charge Voltage, the charger will lower the charging current. Normally, charging stops when reported charge current drops below a configured termination current.

### J1772 Type 1

SAE J1772 Type 1 charging is commonly used in North America and supports single phase 120V charging up to 16A and single phase 240V charging up to 80A. The Electric Vehicle Supply Equipment (EVSE) denotes the charging station. For J1772 Type 1 EVSEs, the EVSE has an integrated charging cable and plug.

The J1772 Type 1 charge plug is a latching plug. When the charge cable is fully inserted into the EV Inlet and latched, it is “locked.” When the charger release button is pressed (by thumb on the charge plug), the charge plug becomes “unlocked,” or simply “connected.” This is implemented by an internal resistor network in the charge plug. The J1772 Proximity to J1772 Ground resistance is 480R if the cable is connected, 150R if it is locked, and “infinite” if the cable is not connected.

The following picture depicts the EV Inlet Connector and its connections.

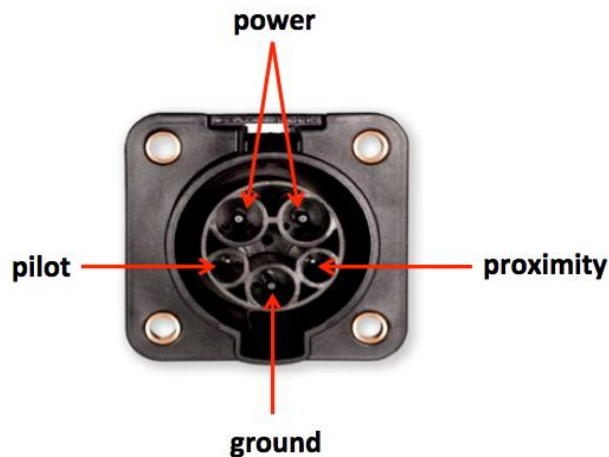


Figure 9 – J1772 EV Connector / Type 1 Inlet

When wiring the EVCC, the **J1772 Proximity** and **J1772 Pilot** signals from the J1772 Inlet are connected directly to corresponding signals at the EVCC. The **J1772 Ground** signal must be connected to EV Chassis Ground as well as EVCC GND.

**IMPORTANT: A poor ground connection between the J1772 Inlet and the EV Chassis or the EVCC GND connection may result in erratic operation.**

The EVCC has a supervisory circuit which monitors for a change in the **J1772 Proximity** resistor value and the EVCC can automatically power itself up when the charge cable is inserted.

The **J1772 Proximity** signal allows the EVCC to determine whether the J1772 charge plug is disconnected, connected or locked. Should the plug become “unlocked” or “disconnected” while charging, charging will stop.

When the charge plug is inserted, the EVSE will generate a square wave on the **J1772 Pilot** signal. The duty cycle of this square wave indicates the available charging current from the EVSE. The EVCC can enable and disable charging by switching an appropriate diode and resistor between the **J1772 Pilot** signal to GND which the EVSE monitors.

For more information on J1772 see [http://en.wikipedia.org/wiki/SAE\\_J1772](http://en.wikipedia.org/wiki/SAE_J1772).

## J1772 Type 2

The International Electrotechnical Commission (IEC) has adopted and extended the SAE J1772 standard for use internationally. In its IEC 62196 standard, several charging plug types are defined. IEC 62196 Type 2 uses the same protocols as SAE J1772 but uses a different plug that allows for either one or three phase charging. This plug is commonly known as the “Mennekes” plug, as it was first developed by Mennekes Elektrotechnik GmbH, of Germany.

SAE J1772 Type 2 charging is commonly used outside of North America and supports one or three-phase 240V charging up to 70A.

The face of the female Mennekes cable plug is shown below (vehicle receptacle is the reverse).



**Figure 10 – Face of J1772 Type 2 Female Mennekes Plug**

The Mennekes connector allows the EVSE to provide either single-phase or three-phase power. If only one-phase power is needed, an adapter cable can be used that mechanically converts from one connector to the other. The following table lists the J1772 Type 2 signals and shows how they correspond to the J1772 signals.

<b>J1772 T2 Abbreviation</b>	<b>J1772 T2 Signal</b>	<b>J1772 T1 Signal</b>
PP	Plug Present	Proximity
CP	Control Pilot	Pilot
PE	Protective Earth	Ground
N	Neutral	Power
L1	Phase 1	Power
L2	Phase 2	n/a
L3	Phase 3	n/a

In J1772 Type 2, the charge cable does not support a pushbutton connector lock, so the cable is either “disconnected” or “connected.”

For J1772 Type 2 EVSEs, the EV owner can provide the charge cable. Different charge cables are rated at different line currents; the current rating of the charge cable is indicated to the EV by means of the cable Proximity Pilot resistor.

The standard J1772 Type 2 Proximity resistor values are:

<b>Cable Rating</b>	<b>Proximity Resistor</b>
13A	1.5 K
20A	680 $\Omega$
32A	220 $\Omega$
63A	100 $\Omega$

The EVCC has a supervisory circuit which monitors for a change in the J1772 Proximity resistor value and can automatically power itself up when the charge cable is inserted. When using J1772 Type 2, the autostart circuit will work correctly for proximity resistor values of 680 $\Omega$  or less.

**NOTE: Autostart will not work for a J1772 Type 2 13A cable. To charging with a 13A cable the user must install the cable and then apply power to the KeySwitch EVCC input. This will power up the EVCC and charging will start.**

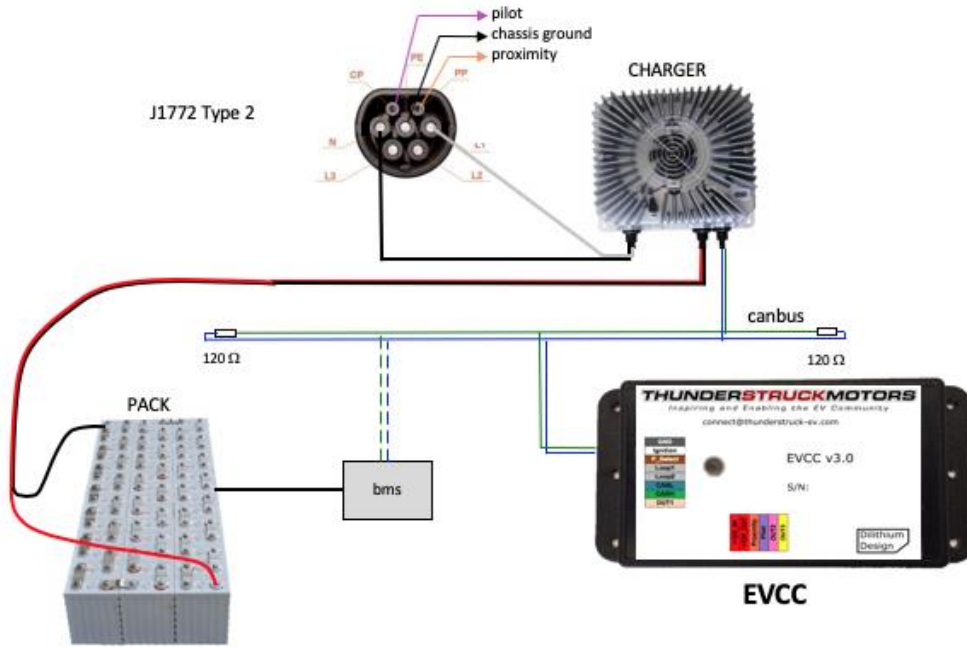


Figure 11 - Wiring J1772 Type 2 with One Phase

**Three-phase Power**

J1772 Type 2 supports three-phase power. When configuring multiple chargers, each charger requires different CAN addresses for control and each charger requires a different type. For example:

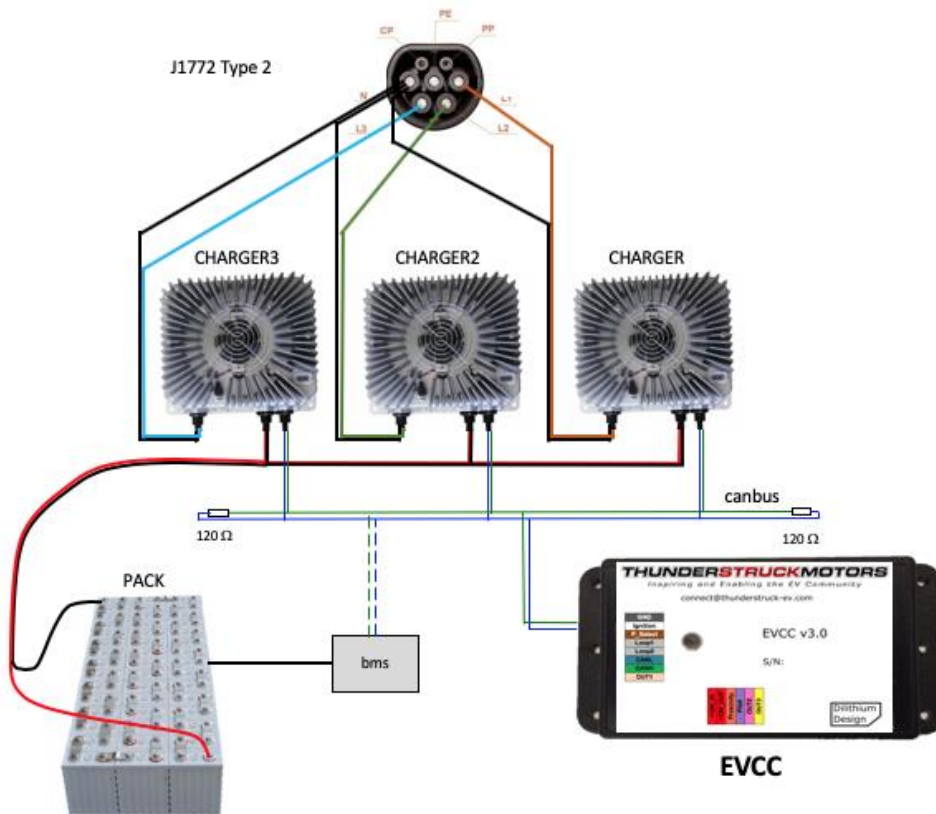


Figure 12 - Wiring J1772 Type 2 with Three Phase Power

## Charge Parameters

In order to configure the EVCC, the user must specify the plug type, the types of charger(s) being used (and if applicable, what phase they are connected to), the charging Voltage and Current, whether a “Finishing” or “Float” charge is configured, and the “charge termination” conditions.

See below, [Charging Lithium-Ion Batteries](#) and [Charging Lead-Acid Batteries](#) for appropriate values of Charging Voltage and Charging Current parameters as well as a discussion of Finishing and Float charge.

**Example 1:** The following example covers a simple case: J1772 Type 1 charging using a single TSM2500 charger, using only the Bulk Charge phase. In this case, the Charge Voltage and Charge Current is explicitly configured:

```
evcc> set plug j1772
evcc> set charger tsm2500
evcc> set maxv 140
evcc> set maxc 20
evcc> set termc 2
```

The `set plug` command chooses J1772 Type1 and the `set charger` command indicates that a TSM2500 charger will be used. Both Target Charge Voltage and Target Charge Current are configured (`maxv` and `maxc`).

In this example, the Charge Current is explicitly configured in the EVCC (`maxc`). The EVCC will measure the J1772 Duty Cycle calculation and *may* reduce charging current if the resulting current exceeds the capability of the EVSE. The J1772 duty cycle limit is determined by the following steps:

1. The EVCC measures the pilot duty cycle and determines maximum Line Current available from the EVSE.
2. Line Voltage and Line Current are used to determine how much power the EVSE can provide. Line Voltage may be explicitly configured using the parameter `linev`, however if not, a heuristic is used<sup>3</sup>. Line Watts is then calculated.
3. Line Watts is then derated by 10% to account for charger efficiency, resulting in Available Charging Watts.
4. Available Charging Watts is then used with Pack Voltage to determine the Available Charge Current<sup>4</sup>.

If Available Charge Current is less than the `maxc` parameter, then the requested Charge Current will be reduced accordingly.

**Example 2:** This example is similar to the previous; the parameter `maxc` is not configured, however `linec` is set to `j1772`:

```
evcc> set plug j1772
evcc> set charger tsm2500
```

---

<sup>3</sup> If the J1772 duty cycle calculation results in a Line Current of less than 18.5A, then EVCC assumes the Line Voltage is 110V, otherwise the EVCC assumes a Line Voltage of 220. This is driven by commonly available EVSE equipment used in the North American market.

<sup>4</sup> Because Pack Voltage is used and not Target Charge Voltage to determine Charge Current, the result is that the charge current will be higher if the pack is discharged.

```

evcc> set maxv 140
evcc> set linec j1772
evcc> set termc 2

```

Using this configuration, **linec** is explicitly configured and so Charging Current is determined from the J1772 Duty Cycle using the power calculation above.

**Example 3:** This example will illustrate three phase charging. In this case **linev** is configured to 220V, the pilot duty cycle indicates 50A of available current and the proximity indicates a 63A cable. Pack **maxv** is configured to 310V.

Three chargers are defined, one for each phase. The calculation is similar to the one above. Available line watts is 11000 (220V \* 50A), derated by 10% leaves 9900W. With a charge voltage of 310V, the maximum charging current is around 32A. This current is then divided among the three chargers which results in about 10.6A each.

```

evcc> set plug j1772t2
evcc> set charger tsm2500 L1
evcc> set charger tsm2500_41 L2
evcc> set charger tsm2500_42 L3
evcc> set maxv 140
evcc> set linec j1772
evcc> set termc 2

```

## Charge Profiles

The EVCC supports up to four charge profiles; each profile contains a copy of EVCC charge parameters that can be used in different charging scenarios. In practice a profile is chosen using the profile select input, and is implemented as an external two to four-way switch. See the section above *Profile Selection*.

As an example, suppose there are two different charging scenarios : a primary EVSE at home capable of 220V at 30A, and secondary EVSE capable of 110V at 15A (a granny charger).

This could be configured in the EVSE with two charge profiles:

- Profile 1: Primary EVSE. Define maxv, linev=220, linec=30
- Profile 2: Secondary EVSE. Define maxv, linev=110, linec=15

## Time of Use Charging

The EVCC supports “Time of Use” charging which allows the user to charge the EV during off-peak hours (where, presumably, electricity costs are lower). The EVCC supports this operation with the configuration parameter **evsewait**. When time of use charging is used, the EVSE is on a timer which only connects it to mains power during off-peak times.

Upon charge plug insertion, the EVCC will autostart and perform a J1772 startup handshake with the EVSE. If the chargers do not power up (e.g., because the EVSE is not switched to mains power), then the EVSE then decides if it should delay charging. If time of use charging requires a delay, the EVSE aborts the charge and then waits for the low-rate period to begin. When the time comes, the EVSE will restart the startup sequence.

If this parameter is set, the EVCC will autostart, will “forgive” a short, aborted, charge such as described above, and then will wait for the charge to start later. The `evsewait` parameter specifies how many hours the EVCC should wait for the EVSE pilot signal before starting a charge.

```
evcc> set evsewait 4.5
```

## Driveaway Protection

The J1772 Proximity signal can be used to support “driveaway protection,” which can prevent the EV from being driven if the charge cable remains connected. The **EVSEDISC** condition is set when the charge plug is unplugged and can be mapped to a mappable output using the command “`set out<n>`”. The output can be used along with low current relays to enable or disable driving the EV. Specific solutions for driveaway protection are application specific.

Example usage:

```
evcc> set out1 evsedisc
```

## DC/DC CONVERTER SUPPORT

The EVCC supports Volt and Delphi DC/DC converters. A DC/DC converter may be enabled by the command: `set ddtype`, and the DC/DC target voltage can be set by the command `set ddvoltage`.

Example usage:

```
evcc> set ddtype volt
evcc> set ddvoltage 13.2
```

## START-UP CHECKLIST

### EV Installation

- 1) Install serial port drivers and terminal program on a host computer. Connect the USB to serial cable, and open the user interface (Putty or Coolterm).
- 2) Connect +12v\_IN to the +12v bus.
- 3) Connect GND. Connect Ignition to the +12v keyswitch circuit (key ON, or connect direct to +12v for tests)
- 4) This should power up the EVCC and the LED will start to blink. Verify that the serial port window opens up and that a startup banner is printed.

### Verify Loop (if using the cell loop inputs)

- 1) With the cell loop circuit open, type “`measure loop`” and verify that the reading is < 0.2V
- 2) Close the cell loop circuit, type “`measure loop`” to verify that the reading is >2.5V

### Verify P\_Select (if using the profile select feature)

- 1) Type “`measure pselect`” with different switch settings to verify that the hardware is correctly reading the inputs.

### Verify J1772 (if using a J1772 EVSE)

- 1) Remove +12V from Ignition to power down the EVCC (leave +12V\_IN connected)
- 2) Connect the EVCC Proximity and Pilot wires to the J1772 inlet port (see Figure 8).
- 3) Connect the J1772 inlet port ground to the 12v negative bus (chassis ground for negative ground systems).
- 4) Plug in J1772 Plug, verify that the EVCC autostarts. The LED should start blinking and the EVCC banner should print to the user intrerface.
- 5) Assuming the CAN bus is not connected to the charger, the charge cycle should stop after 10-15 seconds and the EVCC will power itself down.

### Verify Charger and CAN

- 1) Connect CAN between Charger and EVCC. If no BMS CAN termination is configured, verify proper installation of the charger CAN termination resistor.
- 2) Power up the EVCC and configure initial charging parameters, **maxv** and **maxc**. See **set** instructions below for all configuration options.
- 3) Connect line power to the charger. If J1772 is used, this will come from the J1772 plug.
- 4) Power up the EVCC either by autostarting the EVCC (plug in the J1772 plug) or by applying +12V to Ignition.
- 5) Now verify that a charge cycle is started, with messages exchanged between EVCC and Charger. (Use “**trace charger**” or “**trace can**” to view the messages).
- 6) If the pack is not yet connected to the Charger, the charge cycle will stop after a minute.
- 7) If the charger fan starts and then stops, a Red/Grn/Yel/Yel/Yel light pattern may show on the charger. In this case, check the CAN wires for security and polarity. Check power input and output connections and voltage.

### BMS Integration

- 1) Connect +12V\_OUT to power the BMS.
- 2) Follow the BMS testing and configuration procedures in the BMS manual, including CAN termination.
- 3) Enable the BMS in the EVCC by entering “**set bms bmsc**.”
- 4) Verify proper BMS / EVCC CAN communication by entering “**show**” in the EVCC – note the BMS status.

### Systems Test

- 1) Configure charge parameters, profiles (if used), and verify proper operation.
- 2) If J1772 is being used, verify that pressing the plug release latch stops charging (without unplugging)
- 3) If J1772 is not being used, then configure the EVCC for “plug direct.”
- 4) Run a charge cycle. Check results using “**trace charger**” while charging and “**show history**” when complete.
- 5) Review the charge cycle termination reason in the “**show history**” report.
- 6) Enter “**tr off**” to stop the tracing output.

## SERIAL INTERFACE

### Startup Banner

When the EVCC is powered up after connection to a computer interface, it will print a message similar to the following:

```
*****
*           EV Charger Controller v3.1.19           *
*   Thunderstruck Motors / Dilithium Design         *
*****
```

### help

The **help** command prints out system help information.

```
evcc> help
  SHow [<>|Version|Config|History]
    <>      - status
    version - firmware version
    config  - configuration
    history - charge history

  SSet [
    CANBR|BMS|PLUG|OUT1|OUT2|OUT3
    |CHARGER|CHARGER2|CHARGER3|CHARGER4
    |Profile|MAP
    |EVSEWAIT
    |LINEV|LINEC
    |MAXV|MAXC|MAXBC|TERMC|TERMT
    |FIN_MAXV|FIN_MAXC|FIN_TERMT
```

```

    |FLT_MAXV|FLT_MAXC|FLT_TERMT
    |DDTYPE|DDVOLTAGE
]
REset [History|PROFILE|CONFIG]
    history - reset charge history
    profile<n>- deletes a charge profile
    config - set configuration to defaults
ENable [CANTERM|TOPBALANCE|LOOPGROUND|SLOWSTART|IGNORETHCENSUS]
    canterm - enable CAN termination resistor
    topbalance- cell HVC stops current but not charging
    loopground- loop OK if loop2 grounded
    slowstart - wait up to 40 secs for the charger to start
    ignorethcensus - ignore thermistor census error when charging
DISable [CANTERM|TOPBALANCE|LOOPGROUND|SLOWSTART|IGNORETHCENSUS]
TRace [Charger|CANbus|State|OFF]
    charger - trace charger messages
    canbus - trace canbus messages
    state - trace EVCC state changes
    off - disable all tracing
MEasure [LOOP|PROXimity|PSElect]
    loop - measure Cell Loop A/D
    proximity - measure J1772 Proximity A/D
    pselect - measure Profile Selection A/D
UPGRADE - performs a firmware upgrade
evcc>

```

In many 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, the present status will be displayed.

The following is an example **show** output, when the EVCC is in the DRIVE state:

```

evcc> show
state      : DRIVE
cell loop: OK
proximity: EVSE not connected
profile   : 1
OUT1      : OFF - LED
OUT2      : ON - EVSEDISC
OUT3      : OFF - BUZZER
uptime    : 0 hour(s), 0 minute(s), 33 second(s)

```

If, instead, the BMS configuration is set to “**bmsc**” instead of “**loop**,” the output would be the following:

```

evcc> set bms bmsc
evcc> show
state      : DRIVE
bmsc      : OK
proximity: EVSE not connected
profile   : 1
OUT1      : OFF - LED
OUT2      : ON - EVSEDISC

```

```
OUT3      : OFF - BUZZER
uptime    : 0 hour(s), 0 minute(s), 33 second(s)
```

In the CHARGE state, the pack is actively charging.

```
evcc> show
state      : CHARGE
cell loop: OK
proximity: EVSE Connected and locked
profile    : 1
OUT1       : OFF - LED
OUT2       : ON  - EVSEDISC
OUT3       : OFF - BUZZER
J1772      : duty cycle= 50%, line current available= 30.0A
charger    : tsm2500
  status   : 14 msgs sent; 11 msgs received
  voltage  : 53.4V
  current  : 1.9A
  charge   : 0.12Wh
uptime     : 0 hour(s), 2 minute(s), 0 second(s)
```

### show version

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

```
evcc> show version
version    : v3.1.19; Jul 24 2025 08:13:25
evcc>
```

### show config

The **show config** command displays configuration parameters. The default configuration output of **show config** is the following:

```
evcc> sh config
bms        : loop
canbr      : 250kbps
plug       : j1772
OUT1       : LED
OUT2       : EVSEDISC
OUT3       : BUZZER
charger    : tsm2500
maxv       : 20.0V
maxc       : 2.0A
termc     : 2.0A
termt      : 72.0hr
options    : canterm (CAN termination resistor enabled)
```

The output of **show config** becomes progressively more complex as more features are enabled. If only one charge profile is defined, the full set of configured parameters is given below:

- bms            - the bms type(s) (none or one or more of loop, bmsc, bmsc2, bmsc3, bmsc4)
- canbr         - configured can baud rate
- OUT1          - mappings for OUT1 (none or LED, HVC, LVC, CHARGE, BUZZER, EVSEDISC)
- OUT2          - mapping for OUT2
- OUT3          - mapping for OUT3

- charger - the configured charger type
- charger2 - (if configured) type of charger2
- charger3 - (if configured) type of charger3
- charger4 - (if configured) type of charger4
- plug - AC power connection – direct (wall plug), J1772, or J1772T2
- ddtype - (if configured) dc/dc type
- ddvoltage - (if configured) dc/dc voltage setting
- evsewait - (if configured) time to wait for pilot signal
- linev - (if configured) line voltage of service connection
- linec - (if configured) line current of service connection
- maxv - maximum charging voltage (in Volts)
- maxc - maximum charging current (in Amps)
- maxbc - (if configured) maximum balance current
- termc - terminating charging current (in Amps)
- termt - maximum charging time (in hours)
- fin\_maxv - (if configured) finishing charge voltage (for SLA charging)  
Note: must be at least 1 volt higher than maxv)
- fin\_maxc - (if configured) finishing charge time (in hours, for SLA charging)
- fin\_termt - (if configured) finishing charge current (for SLA charging)
- flt\_maxv - (if configured) float charge voltage (for SLA charging)
- flt\_maxc - (if configured) float charge current (for SLA charging)
- flt\_termt - (if configured) float charge time (in hours, for SLA charging)
- profile map- (if configured) profile setting to profile
- options - canterm (if enabled) can termination resistor enabled  
- topbalance (if enabled) enable 'top balance' feature  
- loopground (if enabled) enables open collector cell loop surveillance  
- slowstart (if enabled) waits up to 40secs for charger to start  
- ignorethcensus (if enabled) thermistor census alerts do not stop charging

An example of a full output with all options is shown below:

```

evcc> show c
bms      : loop
canbr    : 250kbps
plug     : J1772t2
ddtype   : volt
ddvoltage: 13.6
OUT1     : LED
OUT2     : EVSEDISC
OUT3     : BUZZER
charger  : tsm2500
charger2 : tsm2500_42
charger3 : tsm2500_43
charger4 : elcon
evsewait: 4.0hr
linev    : 220.0V
linec    : 30.0A
maxv     : 155.0V
maxc     : 15.0A
maxbc    : 2.0A
termc    : 1.5A

```

```

termt      : 6.0hr
fin_maxv   : 160.0V
fin_maxc   : 2.0A
fin_termt  : 2.0hr
flt_maxv   : 152.0V
flt_maxc   : 1.5A
flt_termt  : inf
options    : canterm (CAN termination resistor enabled)
           : topbalance (cell HVC stops current but not charging)
           : loopground (loop OK if loop2 grounded)
           : slowstart (wait up to 40 secs for charger to start)
           : ignorethcensus (ignore thermistor census errors when charging)

```

If more than one charge profile is defined, **show config** will display the charge profiles in “tabular form.” The charge profile selected for editing is indicated with a “\*.” Also, the profile map is shown.

Example output with multiple charge profiles is shown below:

```

evcc> show c
bms       : loop
canbr     : 250kbps
plug      : J1772
OUT1      : LED
OUT2      : EVSEDISC
OUT3      : BUZZER
charger   : tsm2500
profiles  : 1          2*          3          4
  linev   : 220.0V
  linec   : 30.0A      J1772
  maxv    : 155.0V     152.0V
  maxc    : 15.0A      10.0A
  maxbc   : 2.0A
  termc   : 1.5A      1.5A
  termt   : 6.0hr     8.0hr
  fin_maxv : 160.0V
  fin_maxc : 2.0A
  fin_termt : 2.0hr
  flt_maxv : 152.0V
  flt_maxc : 1.5A
  flt_termt : inf
profile map:
  inf     : x
  20K    : x
  5K     : x
  0      : x
Options  : canterm (CAN termination resistor enabled)

```

### 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.

```

evcc> show history
no charge history

```

The next example shows charge history, with different “termination reasons.” The termination reason contains the reason that the charge cycle stopped. In this example, 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.

```
evcc> show history
```

num	term reason	charge time	charger	watt hours	maximum voltage	maximum current	ending current
last	EVSE disc	1 mins	tsm2500	7Wh	148.9V	7.9A	7.9A
- 1	normal	214 mins	tsm2500	3249Wh	152.9V	7.9A	0.5A
- 2	EVSE disc	1 mins	tsm2500	0Wh	144.8V	0.0A	0.0A
- 3	comm err	0 mins	tsm2500	0Wh	0.0V	0.0A	0.0A

```
evcc>
```

The full set of “term reason” codes is:

- EVSE disc - J1772 charge plug became unlocked while charging
- no pilot - control pilot timeout
- cell loop - a cell loop error was detected
- pack disc - no pack was detected
- not locked - BMS cell configuration not locked
- HVC - BMS reported a cell HVC error
- cellcensus - BMS reported fault: cell census (not all cells present)
- th census - BMS reported fault: thermistor census (not all thermistors present)
- overtemp - BMS reported fault: thermistor over temperature
- bmstimeout - BMS communication error
- comm err - communications error with the charger
- timeout - the maximum charge time was reached
- chargestop - charger message timeout (possibly due to loss of line power)
- normal - normal completion (charge current is less than terminating charging current)
- fin normal - normal termination of finishing charge

When multiple chargers are configured, the format of the charge history is modified to show the contribution of each charger.

```
evcc> show history
```

num	term reason	charge time	charger	watt hours	maximum voltage	maximum current	ending current
last	EVSE disc	2 mins	tsm2500	6Wh	127.8V	2.2A	0.0A
			tsm2500_42	6Wh	127.5V	2.0A	0.0A
			TOTAL	12Wh	127.8V	4.2A	0.0A

## set

This command sets the configurable parameters.

```
set <>
```

Using the **set** command with no parameters will print help information for the “set” command. The following example shows this output.

```

evcc> set
  SET [
    BMS|CANBR|PLUG|OUT1|OUT2|OUT3
    |CHARGER|CHARGER2|CHARGER3|CHARGER4
    |PROFILE|MAP
    |EVSEWAIT
    |LINEV|LINEC
    |MAXV|MAXC|MAXBC|TERMC|TERMT
    |FIN_MAXV|FIN_MAXC|FIN_TERMT
    |FLT_MAXV|FLT_MAXC|FLT_TERMT
    |DDTYPE|DDVOLTAGE
  ]
bms configuration
  set bms [NONE|<bmsn>[,<bmsn>[,<bmsn>[,<bmsn>]]]]
    <bmsn> - [LOOP|BMSC|BMSC2|BMSC3|BMSC4]
charge plug
  set plug [J1772|J1772T2|direct]
    J1772 - J1772 charge plug
    J1772T2 - J1772 type 2 charge plug
    direct - charger is directly AC powered
CAN baud rate
  set canbr [125|250|500]
OUT<n> mapping
  set out<n> [NONE|LED|HVC|LVC|CHARGE|BUZZER|EVSEDISC]
evse wait time
  set evsewait <h>
charger configuration
  set <chargern> <type> [<phase>]- defines <chargern>
    <chargern> - [CHARGER|CHARGER2|CHARGER3|CHARGER4]
    <type> - [TSM2500|TSM2500_41|TSM2500_42|TSM2500_43
      |ELCON|ELCON_E7|ELCON_E8|ELCON_E9
      |STEALTH|LEAR|VOLT
      |BASIC|NONE]
    <phase> - [L1|L2|L3|NONE]
  set <chargern> <type> PROGRAM - programs tsm2500 CAN IDs
profile editing
  set profile [1|2|3|4] - set profile for editing
  set map [inf|20K|5K|0|all] <n>
    - set profile mapping
Service parameters
  set linev <v> - available line voltage
  set linec [<a>|J1772] - available line current
BULK charge parameters
  set maxv <v> - maximum charge voltage
  set maxc <a> - maximum charge current
  set maxbc <a> - maximum balancing current
  set termc <a> - charge termination current
  set termt <h> - charge termination timeout
FINISH charge parameters
  set fin_maxv <v> - finishing charge voltage
  set fin_maxc <a> - finishing charge current
  set fin_termt <h> - finishing charge termination timeout
FLOAT charge parameters
  set flt_maxv <v> - float charge voltage
  set flt_maxc <a> - float charge current
  set flt_termt <h> - float charge termination timeout (0=no timeout)
dc/dc converter

```

```

    set ddtype [NONE|VOLT|DELPHI] - set dc/dc converter type
    set ddvoltage <v> - dc/dc output voltage
evcc>

```

### set bms

This command sets the BMS type. The EVCC can use a cell loop and/or up to four CAN BMSs.

The following examples show some different options for setting the BMS type for loop, can messaging, multiple BMS units, or to disable BMS messaging.

```

evcc> set bms loop
evcc> set bms bmsc
evcc> set bms bmsc, bmsc2
evcc> set bms none

```

### set plug

This configures the AC charge plug type. There are three options: **J1772**, **J1772T2**, and **DIRECT**. The **J1772** option is used for J1772 Type 1, and **J1772T2** is used for J1772 Type 2.

The **DIRECT** option is used when the charger is connected directly to an AC power cord. In this case, autostart and duty cycle information are not available. An external mechanism is necessary to power up the EVCC and charger, and the charge will start when the EVCC starts receiving CAN messages from the Charger.

In the EVCC, the plug type only determines the plug behavior. In order to use J1772 duty cycle (or J1772T2 duty cycle and charge cable ampacity) to determine or limit charge current, it is necessary to also configure the **linec** parameter to **j1772**.

```

evcc> set plug direct
evcc> set linec j1772

```

### set canbr

This command sets the can baud rate. The EVCC supports 125kbps, 250kbps, and 500kbps. Each supported charger type has factory settings determining this value. For example, when setting the can baud rate to 500 kbps, the following command is used:

```

evcc> set canbr 500

```

### set out1 (set out2, set out3)

This command sets the output mapping. Note that OUT1, OUT2, and OUT3 have different hardware capabilities and must be wired appropriately.

OUT1, OUT1 and OUT3 are mappable to the functions: LED, HVC, LVC, CHARGE, BUZZER, and EVSEDISC. Each output can map to any single function.

```

evcc> set out1 evsedisc

```

### set charger<n>

This sets the charger type. The first charger is named “charger.” Chargers 2 through 4 are named “charger2,” “charger3,” “charger4.”

The following charger types are supported:

- `none`
- `tsm2500`, `tsm2500_41`, `tsm2500_42`, `tsm2500_43`
- `elcon`, `elcon_e7`, `elcon_e8`, `elcon_e9`
- `stealth`
- `lear`
- `nlg5`
- `volt`
- `basic`

The following command sets a single charger

```
evcc> set charger tsm2500
```

The following command sets a second charger

```
evcc> set charger2 tsm2500_42
```

A charger may be deleted using the command

```
evcc> set charger2 none
```

This command is also used to associate a phase with a charger, which is used with J1772 Type 2 charging. The charger type. The phases are named: `L1`, `L2`, `L3`, `none`. The following example defines three chargers and associates each of them with a different phase.

```
evcc> set charger tsm2500 L1
evcc> set charger2 tsm2500_41 L2
evcc> set charger3 tsm2500_42 L3
```

The phase association may be reset to “none” using the command:

```
evcc> set charger tsm2500 none
```

#### **set evsewait**

This sets the time in hours, to wait for the pilot signal for Time of Use charging. To disable this feature, set the `evsewait` parameter to zero.

```
evcc> set evsewait 4.5
```

#### **set profile <n>**

This command selects a profile for editing. There are four possible profiles: 1-4. Profile 1 is automatically created. It is the default profile and cannot be deleted. If the user types “**set profile <n>**,” then this will both select a profile for editing and create the profile if it does not already exist. Once a profile is selected, then subsequent editing commands (e.g., `set maxv`, etc.) apply to the parameters associated with the profile. Profiles 2-4 may be deleted using the command “**reset profile <n>**.”

Examples of creating and editing profiles:

Start with the default configuration:

```

evcc> show config
bms      : loop
charger  : tsm2500
plug     : j1772
maxv     : 20.0V
maxc     : 2.0A
termc    : 0.2A
termt    : 72.0hr
evcc>

```

Create Profile 2 with default configuration:

```

evcc> set profile 2
evcc> show c
bms      : loop
charger  : tsm2500
plug     : j1772
profiles : 1      2*      3      4
maxv     : 20.0V  20.0V
maxc     : 2.0A   2.0A
termc    : 0.2A   0.2A
termt    : 72.0hr 72.0hr
profile map:
inf      : x
20K      : x
5K       : x
0        : x

```

Change some parameters in Profile 2:

```

evcc> set maxv 150
evcc> set maxc 12
evcc> show config
bms      : loop
charger  : tsm2500
plug     : j1772
profiles : 1      2*      3      4
maxv     : 20.0V  150.0V
maxc     : 2.0A   12.0A
termc    : 0.2A   0.2A
termt    : 72.0hr 72.0hr
profile map:
inf      : x
20K      : x
5K       : x
0        : x

```

Return to Profile 1 and set some parameters in Profile 1:

```

evcc> set profile 1
evcc> set maxv 160
evcc> set maxc 15
evcc> set linec j1772
evcc> show config
bms      : loop
charger  : tsm2500

```

```

plug      : j1772
profiles  :   1*       2       3       4
  linec   :   J1772
  maxv    : 160.0V   150.0V
  maxc    :  15.0A   12.0A
  termc   :   0.2A   0.2A
  termt   :  72.0hr  72.0hr
profile map:
  inf     :   x
  20K    :   x
  5K     :   x
  0      :   x

```

Finally, delete Profile 2:

```

evcc> reset profile 2
evcc> show config
  bms      : loop
  charger  : tsm2500
  plug     : j1772
  linec    : J1772
  maxv     : 160.0V
  maxc     :  15.0A
  termc    :   0.2A
  termt    :  72.0hr

```

#### set map

This command sets the profile map. The EVCC measures resistance to ground at the P\_Select input and from the measurement determines four possible choices, as follows:

$R \geq 30K$	the result is “inf”
$30K > R \geq 10K$	the result is “20K”
$10K > R \geq 2K$	the result is “5K”
$2K > R$	the result is “0”

Each P\_Select range is mapped to a profile using the profile map. Initially everything is mapped to the default profile, Profile 1. If the user wants to use two profiles, a switch to ground may be connected at the P\_Select input. If the switch is open then Profile 1 is used and if the switch is closed, then Profile 2 is used. Once Profile 2 is defined, this may be configured as follows:

```

evcc> set map 0 2
evcc> show config
  bms      : loop
  charger  : tsm2500
  profiles :   1       2*       3       4
  linec    :   J1772
  maxv     : 160.0V   150.0V
  maxc     :  15.0A   12.0A
  termc    :   0.2A   0.2A
  termt    :  72.0hr  72.0hr
profile map:
  inf     :   x
  20K    :   x
  5K     :   x
  0      :           x

```

If a third profile were to be defined, a switchable resistor would be required at the P\_Select input. The following command will enable the mapping from a 5K resistor to Profile 3:

```
evcc> set map 5K 3
```

#### set linev, set linec

This sets the maximum line voltage and line current available. If the J1772 (or J1772T2) current limitations should be applied then **linec** is set to **J1772**.

```
evcc> set linev 110
evcc> set linec 12.5
```

#### set maxv, set maxc

The command **set maxv** sets the maximum charging voltage, in Volts. The command **set maxc** sets the maximum charging current, in Amps.

For voltage and current, whole numbers (145) or decimal numbers (145.2) can be entered. The EVCC supports one decimal digit of precision, for example:

```
evcc> set maxv 155.0
evcc> set maxc 8.5
```

#### set maxbc

This sets the maximum balancing charging current, in Amps. This option is only possible if a CAN BMS is used and it sends a “BVC threshold exceeded” condition to the EVCC. For the ThunderStruck BMS, entering “set bvc 3.5” would tell the EVCC to reduce charge current when the highest cell voltage reached 3.5 volts.

```
evcc> set maxbc 2.0
```

#### 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 hours.

```
evcc> set termt 6.5
```

#### set fin\_maxv, set fin\_maxc, set fin\_termt

These commands are used to define the “finishing charge” phase voltage, current, and charge time for Sealed Lead Acid battery charging. See below, *Charging Lead Acid Batteries*, for examples of use. Note: the finishing phase will be skipped unless **fin\_maxv** is set at least 1 volt above **maxv**.

#### set flt\_maxv, set flt\_maxc, set flt\_termt

These commands are used to define the “float charge” phase voltage, current, and charge time for Sealed Lead Acid battery charging. See below, *Charging Lead Acid Batteries*, for examples.

#### set ddtype, set ddvoltage

These commands are used to enable CAN messages to control a DC/DC converter. The Volt and Delphi converters are supported.

Example commands for configuring a Volt DCDC:

```
evcc> set ddtype volt
evcc> set ddvoltage 13.6
```

A DC/DC converter can be deleted using the command

```
evcc> set ddtype none
```

## reset

### reset history

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

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

### reset profile <n>

The **reset profile** command can be used to delete Profiles 2-4. It is not possible to delete Profile 1.

```
evcc> reset profile 3
```

### reset config

The **reset config** command can be used to return all settings to factory defaults

```
evcc> reset config
done!
evcc>
```

## enable | disable

The EVCC supports several options that may be “enabled” and “disabled.” Options that are “enabled” are shown in the **show config** output.

### enable | disable canterm

The EVCC contains an integrated, and programmable CAN termination resistor. By default, this termination resistor is enabled. In order to enable this termination resistor, use the command:

```
evcc> enable canterm
```

In order to disable this option, use the **disable canterm** command.

### enable | disable topbalance

Normally, the charging cycle terminates when the cell loop opens or when the BMS indicates a pack HVC condition. Some customers may not want to completely stop charging at this stage: instead they may want to suspend charging, allow the pack to “rest” and for the HVC condition to clear, and to resume charging. The theory is that lower charged cells may be topped up with this process. In this case, the cell loop / HVC condition does not stop charging.

```
evcc> enable topbalance
```

In order to disable this option, and return the EVCC to default behavior, use the **disable topbalance** command.

#### enable | disable loopground

Setting this option will change the method of loop supervision. With the “loopground” option the Cell Loop is considered good if there is a ground on Loop2. (High Impedance means that the cell loop is not good). With this option, Loop1 must be left unconnected.

```
evcc> enable loopground
```

In order to disable this option, and return the EVCC to default behavior, use the **disable loopground** command.

#### enable | disable slowstart

Setting this option will enable the “slowstart” option. If this option is enabled, the EVCC will wait up to 40 seconds for the charger to become active. This feature does not function when the charger type is set to “basic.”

```
evcc> enable slowstart
```

In order to disable this option, and return the EVCC to default behavior, use the **disable slowstart** command.

#### enable | disable ignorethcensus

Setting this option will enable the “ignorethcensus” option. If this option is enabled, the EVCC will ignore thermistor census errors from the BMS.

```
evcc> enable ignorethcensus
```

In order to disable this option, and return the EVCC to default behavior, use the **disable ignorethcensus** command.

#### 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 CAN tracing may be independently enabled.

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

#### trace charger

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

```
evcc> trace charger
charger tracing is now ON
evcc> 00:10:28.9 tsm2500 : V=126.3, A= 5.9, W=745, Wh= 0.09, TMP = 26C
00:10:29.3 tsm2500 : V=126.3, A= 5.9, W=745, Wh= 0.11, TMP = 26C
00:10:29.8 tsm2500 : V=126.3, A= 5.9, W=745, Wh= 0.13, TMP = 26C
```

The TSM2500 can report the following errors reported in the “trace charger” output:

- **rxerr**
- **hwfail**
- **overtemp**
- **not charging**
- **input voltage err**
- **pack voltage err**

The Elcon chargers will report the following errors in the “trace charger” output:

- **rxerr**
- **hwfail**
- **overtemp**
- **input voltage err**
- **pack voltage err**

### trace can

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> 00:02:20.9      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:21.4      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:21.9      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:22.4      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:22.5 tsm2500   : 18eb2440 42 f7 41 fd 00 fe 12 dd
00:02:22.9      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:22.9 tsm2500   : 18eb2440 04 fd 13 02 80 0c 3f ff
00:02:23.4      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:23.5 tsm2500   : 18eb2440 00 fc 13 02 80 0c 3f ff
00:02:23.9      evcc: 18e54024 fc c8 00 6c 0c 01 ff ff
00:02:23.9 tsm2500   : 18eb2440 00 fc 13 02 80 0c 3f ff
```

### trace state

The **trace state** command displays internal EVCC and J1772 state transitions.

Here is an example of state trace output that shows the charger plug being plugged in and unplugged, also showing the attempt to charge.

```
evcc> trace state
state tracing is now ON
evcc>
evcc> 00:00:31.7 j1772=LOCKED
00:00:31.7 old state=DRIVE, new state=WARMUP, term rsn=0
00:00:31.8 old state=WARMUP, new state=CHARGE, term rsn=0
00:00:52.2 j1772=CONNECTED
00:00:52.3 j1772=WAITING FOR DISC
00:00:52.4 old state=CHARGE, new state=STANDBY, term rsn=EVSE UNLOCKED
00:00:56.1 j1772=DISCONNECTED
00:00:57.4 old state=STANDBY, new state=DRIVE, term rsn=0
```

The list of EVCC state values is:

- WARMUP state just after power up
- CHARGE charging – bulk charge phase
- FINISH CHARGE charging – finish charge phase
- FLOAT CHARGE charging – float charge phase
- CHARGE/TOPBALANCE topbalance – waiting for cells to drop below HVC
- STANDBY standby state
- EVSE\_WAIT waiting for control Pilot
- DRIVE drive state

The list of J1772 state values is:

- DISCONNECTED charge plug is not plugged in
- CONNECTED charge plug is plugged in but not “locked”
- LOCKED charge plug is plugged in and “locked”
- WAITING FOR DISC charge plug was LOCKED, is now CONNECTED  
now waiting for it to be completely removed

### trace verbose

The **trace verbose** command turns on additional debug trace information.

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

### 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
           V >= 2.5V - OK
proximity - J1772 Proximity A/D
           V >= 4.0V - disconnected
           4.0 > V >= 1.2V - connected
           else          - locked
pselect   - Profile Selection A/D
           R >= 30K      - inf
           30K > R >= 10K - 20K
           10K > R >= 2K  - 5K
           2K > R        - 0
```

```
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 proximity**

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

In the example given below, both the **measure proximity** and **trace state** commands are enabled. Initially the J1772 charge plug is disconnected (measurement is 4.67V), then it becomes locked (measurement is 0.86V), then connected and unlocked (1.97V), and then finally, disconnected.

```
evcc> trace state
state tracing is now ON
evcc> me prox
evcc> Proximity A/D= 4.67V
Proximity A/D= 4.67V
Proximity A/D= 4.67V
Proximity A/D= 4.67V
Proximity A/D= 0.86V
00:05:02.4 j1772=LOCKED
00:05:02.4 old state=DRIVE, new state=WARMUP, term rsn=0
00:05:02.5 old state=WARMUP, new state=CHARGE, term rsn=0
Proximity A/D= 0.86V
Proximity A/D= 0.86V
Proximity A/D= 0.85V
Proximity A/D= 0.86V
Proximity A/D= 1.97V
00:05:05.6 j1772=CONNECTED
00:05:05.7 j1772=WAITING FOR DISC
00:05:05.8 old state=CHARGE, new state=STANDBY, term rsn=EVSE UNLOCKED
Proximity A/D= 1.96V
Proximity A/D= 1.96V
Proximity A/D= 1.97V
Proximity A/D= 4.67V
00:05:07.9 j1772=DISCONNECTED
Proximity A/D= 4.68V
Proximity A/D= 4.67V
Proximity A/D= 4.67V
00:05:10.8 old state=STANDBY, new state=DRIVE, term rsn=0
```

**measure pselect**

The **measure pselect** command gives a real time measurement of the **P\_Select** input. Note that this output is reported in resistance.

```
evcc> me pselect
evcc> Pselect A/D= inf
Pselect A/D= inf
Pselect A/D= inf
Pselect A/D= inf
```

```
Pselect A/D= inf
```

## upgrade

The **upgrade** command is used to perform a firmware upgrade. This command will place the EVCC into the serial bootloader state, waiting for the load to begin. The EVCC must be power cycled in order to leave this state. See *Firmware Upgrade*, for a description of how to perform a firmware upgrade.

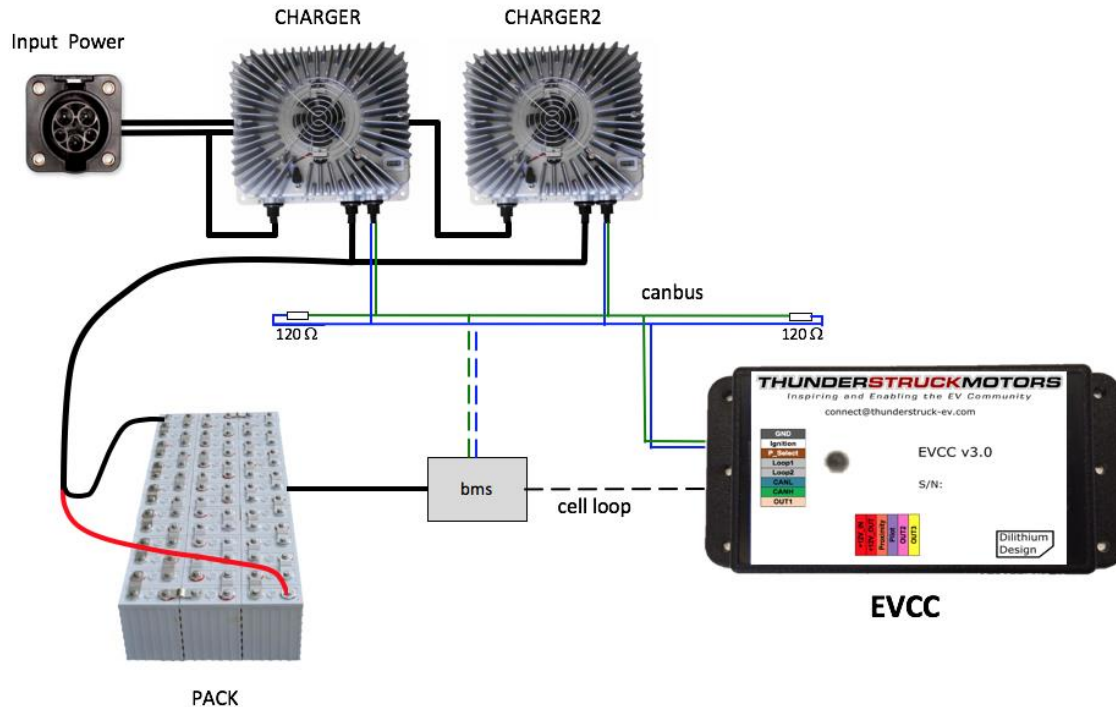
```
evcc> upgrade
```

```
***
***                               Starting EVCC Upgrade                               ***
*** 1) Exit from the terminal application                                         ***
*** 2) Start the bootloader and download a new .hex file                         ***
*** 3) Restart the EVCC                                                         ***
```

## APPLICATION NOTES

### Configuring the EVCC with Multiple Chargers

Up to four chargers can be used in parallel for faster charging. A logical picture is shown in the diagram below.



**Figure 13 – Multiple Chargers - System Diagram**

If J1772 is used, there is a single J1772 interface for line power which feeds all chargers. The chargers are in parallel and they charge a single pack. All chargers are placed on the CAN bus. There is a single EVCC and it communicates with the chargers independently. (Also shown on the CAN bus is a CAN enabled BMS, which is required for any lithium ion pack).

There are several design considerations when installing multiple chargers. Two chargers require more power than a single charger. One must verify that adequate line power is available. With more CAN nodes, the CAN wiring is no longer simply point to point and installation must be done with care. Each charger requires a unique CAN ID. Each charger must be explicitly configured.

When charging, power calculations are needed to make sure that there is sufficient power available for all chargers. As an example, suppose the EVSE provides 220V at 30A. A 220V / 30A circuit has 6600 Watts available. Two 3.0 Kw chargers running at full power can be placed on the line, but three chargers cannot.

When installing multiple chargers, care must be taken that CAN termination resistors are properly placed. Keep in mind that some chargers have a termination resistor installed in the charger, and so that charger must be at the end of the CAN string.

Each charger must have a unique CAN address. See the section *Charger Support* for information on how to determine the charger CAN address and change it if necessary.

The EVCC supports up to four chargers (named: `charger`, `charger2`, `charger3`, and `charger4`). Chargers are defined in the EVCC using the **set charger** command. When a charger is configured, it is set to a “charger type,” which indicates both the manufacturer and its CAN address. It is possible to have chargers from multiple manufacturers (e.g., one ELCON and one TSM2500) operating on the same CAN network, provided all chargers share the same CAN rate.

The following example defines a single charger and sets its type to `tsm2500`:

```
evcc> set charger tsm2500
evcc> show config
  bms      : loop
  canbr    : 250kbps
  plug     : j1772
  charger  : tsm2500
  linec    : j1772
  maxv     : 158.0V
  maxc     : 12.0A
  termc    : 0.5A
  termt    : 72.0hr
evcc>
```

This example defines a second charger, sets its type to `tsm2500_42` and sets the maximum current to twice that of a single charger.

```
evcc> set charger2 tsm2500_42
evcc> set maxc 24
evcc> show config
  bms      : loop
  canbr    : 250kbps
  plug     : j1772
  charger  : tsm2500
  charger2 : tsm2500_42
  linec    : j1772
  maxv     : 158.0V
  maxc     : 24.0A
  termc    : 0.5A
  termt    : 72.0hr
evcc>
```

A charger can be deleted by setting the type to “none,” however the first charger (“charger”) cannot be deleted.

```
evcc> set charger2 none
```

When charging with multiple chargers, `maxc` is divided by the number of chargers and given to each charger. So here is an example of charger tracing when `maxc` is set to 12A. Note that 6A goes to both TSM2500 and TSM2500\_42. Note that “trace charger” reports the status of the charger ... and that voltage, current, watts, and watt hours may be slightly different between chargers.

```
evcc> trace charger
charger tracing is now ON
00:10:28.8 tsm2500_42: V=126.0, A= 5.8, W=730, Wh= 0.10, TMP = 26C
00:10:28.9 tsm2500   : V=126.3, A= 5.9, W=745, Wh= 0.09, TMP = 26C
00:10:29.3 tsm2500_42: V=126.6, A= 5.7, W=721, Wh= 0.19, TMP = 26C
00:10:29.3 tsm2500   : V=126.6, A= 5.8, W=734, Wh= 0.19, TMP = 26C
```

```
00:10:29.8 tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.30, TMP = 26C
```

### Setting the CAN address of a TSM2500 Charger

This section describes how to set the CAN addresses of a tsm2500 charger.

For this procedure, the charger can either be directly connected to power (85-265 volts AC), or indirectly via a J1772 EVSE. Only one charger is connected to the CAN bus during this procedure.

In this example, the second charger will be defined as type tsm2500\_42. The **program** keyword is used to change the canbus address.

To do this, power up the EVCC by keyswitch. Then type the following command:

```
evcc> set charger2 tsm2500_42 program
```

The EVCC will then print

```
***
***                               tsm2500 PROGRAMMING                               ***
*** WARNING: This command changes the CAN IDs of a tsm2500 charger ***
*** ONLY ONE tsm2500 charger should be powered up at this time ***
***
```

```
Proceed [Y/N] ?
```

To continue, press the "y" key, and the EVCC will print:

```
Programming the charger ...
```

Then, 5-10 seconds later it will finish printing the completion message:

```
Programming the charger ... done.
The charger must now be power cycled.
evcc>
```

The charger has now been programmed to tsm2500\_42, and will be listed in the **show config** report as "charger2." To finalize, connect all chargers to the CAN network and charge the pack. If successful, a "show" report will list both chargers showing the expected voltage and current information.

## Integration with a CAN Enabled BMS

The EVCC can be used with a CAN-enabled Battery Management System which shares CAN commands with the EVCC, such as the Dilithium BMSC and Orion BMS2. The following functions are supported:

- **High Voltage Cutoff (HVC) Detection.** In this case, the BMS detects that at least one cell has exceeded its programmed High Voltage Cutoff limit. If this occurs, the BMS sends a message to the EVCC which causes the EVCC to stop charging.
- **Balance Voltage Cutoff (BVC) Detection.** In this case, the BMS detects that at least one cell has exceeded its programmed Balance Voltage Cutoff limit. If this occurs, the BMS sends a message to the EVCC that it should reduce its charging current to the maximum balancing current (**maxbc**). Lowering the charging current allows cell balancing circuits to minimize the net current delivered to the highest cells in the pack.
- **Low Voltage Cutoff (LVC) Detection.** In this case, the BMS detects that at least one cell voltage is less than its programmed Low Voltage Cutoff limit. If this occurs, the BMS sends a message to the EVCC which causes the EVCC to operate the buzzer.
- **Thermistor Overtemp Detection.** In this case, the BMS detects that at least one thermistor is overtemperature. The EVCC will stop charging.

Programming the actual HVC, BVC, and LVC are done in the BMS. The BMS must determine if any cell in the pack meets these conditions and if so, it sets a bit associated with each of these conditions. This information is sent in a message from the BMS to the EVCC; the message must be periodically sent at least once a second.

```

/*
 * The EVCC supports 29 bit identifiers
 */

#define uint8      unsigned char
/*
 * BMS->EVCC message Identifier
 */
#define BMS_EVCC_STATUS_IND      0x01dd0001

/* bBMSStatusFlag bits */
#define BMS_STATUS_CELL_HVC_FLAG      0x01 /* set if a cell is > HVC */
#define BMS_STATUS_CELL_LVC_FLAG      0x02 /* set if a cell is < LVC */
#define BMS_STATUS_CELL_BVC_FLAG      0x04 /* set if a cell is > BVC */

/* bBMSFault bits */
#define BMS_FAULT_OVERTEMP_FLAG      0x04 /* set for thermistor overtemp */

/*
 * BMS->EVCC message body
 */
typedef struct tBMS_EVCC_StatusInd
{
    uint8 bBMSStatusFlags; /* see bit definitions above */
    uint8 bBmsId; /* reserved, set to 0 */
    uint8 bBMSFault;
    uint8 bReserved2; /* reserved, set to 0 */
    uint8 bReserved3; /* reserved, set to 0 */
} tBMS_EVCC_StatusInd;

```

Note that the CAN message only has a 5 byte message body, up to 8 bytes may be sent to the EVCC. These bytes should be set to 0 if so. The EVCC will ignore additional message bytes.

In order to use the CAN interface with the BMS, it must be configured in the EVCC, using the **set bms bmsc** command. It is possible to configure the EVCC to only use **loop**, only use **bmsc**, or use both **loop** and **bmsc**.

If the EVCC is configured to only use **loop**, then if the loop circuit is closed then the pack is error free; if the loop circuit is open, HVC is assumed if in CHARGE state and LVC is assumed if in DRIVE state.

If the EVCC is configured to use only **bmsc**, then pack status is taken from the BMS\_EVCC\_STATUS\_IND message. Note that the message also supports the BVC condition (which the loop does not). If that is reported, then the EVCC will drop back into balance cutback. If there is a message timeout and BMS\_EVCC\_STATUS\_IND does not arrive, then this is treated as a pack error (e.g., HVC and LVC are assumed).

If both **loop** and **bmsc** are configured then an error results if either input reports an error. So, in this case, charging will stop if the loop opens, the CAN message indicates HVC, or there is a CAN message timeout.

### Integration with Motor Controllers

The EVCC ignores most CAN Bus traffic including EV motor controllers. It is possible, however, to use the EVCC outputs to indirectly control enable/disable inputs to a motor controller if available. If a controller has a “limp mode” input then the EVCC could indirectly trigger a low power state if the BMS sends a LVC flag to the EVCC during operation.

For example, the following command would cause Out1 to output 12v if the BMS reported an LVC condition. This could trigger a low current relay to open or close an applicable controller input.

```
evcc> set out1 lvc
```

### Integration with DCDC Converters

The EVCC can control the Chevy Volt and Delphi high voltage DCDC converters using CAN. These are factory configured for a CAN baud rate of 500 kbps.

See the **set ddtype** entry in the *Command Line Interface* section for a configuration example.

If the DCDC converter can be enabled by an analog voltage input, then it is possible to enable the DCDC by using the +12V\_OUT output to activate a control relay for the DCDC enable circuit. This method will enable the DCDC whenever the EVCC is powered: during operation or charging.

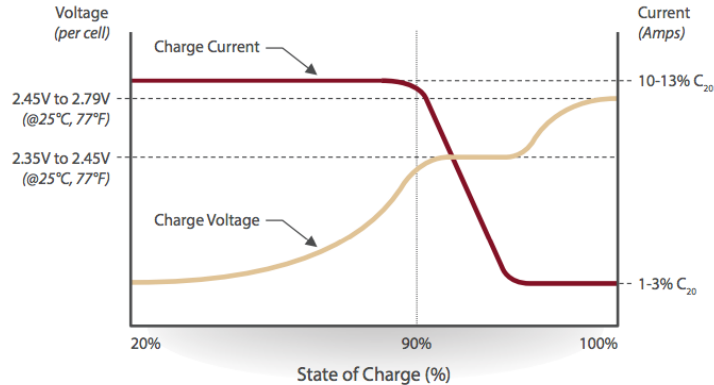
### Charging Lead Acid Batteries

Lead Acid Batteries require a multi-stage charging algorithm. The terminology to describe the algorithms varies in the industry and between manufacturers. Here we follow the documentation and requirements from Trojan. See [www.Trojanbattery.com](http://www.Trojanbattery.com) for information about charging lead acid batteries.

As an example consider a EV pack that consists of 12 Trojan 30XHS deep cycle flooded batteries, charging at 25°C (77°F), with a C<sub>20</sub> rating of 130 Amp hours (AH). When looking up specs for a given battery, the C<sub>20</sub> rating may be used for the battery amp hour (AH) capacity.

Diagram 4

### Recommended Deep-Cycle Flooded/Wet Charging Profile



Note: Charging time will vary depending on battery size, charger output, and depth of discharge.

Figure 14 – Flooded Lead Acid Charging Profile (Trojan)

#### Bulk Charge

The first phase of charging is the Bulk Charge phase. (Note that the Bulk Charge phase is sometimes thought of as two phases: a constant current phase and a constant voltage phase). The EVCC supports this phase by the parameters **maxv** and **maxc**. This phase is used by battery types including flooded, AGM, and Gel.

See Figure 14, above. For flooded cells, the Bulk Charge phase brings the cells to over 90% state of charge. For its cells, Trojan recommends a maximum voltage of 2.35 to 2.45v per cell, and a current of 10-13% C<sub>20</sub>. The bulk charge phase completes when the charging current drops to 1-3% of C<sub>20</sub>.

In the example of twelve 30XHS cells, here are suggested EVCC settings:

- **maxv=172.8**: The charging voltage would be 2.4v \* 6 cells \* 12 batteries = 172.8v.
- **maxc=13**: Since 30XHS cells have a C<sub>20</sub> rating of 130AH, the charging current would be 13A.
- **termc=2.6**: The guidelines are 1-3% of C<sub>20</sub>. 2.6A is 2% of the C<sub>20</sub> rating of 30XHS.
- **termt=480**: (10 hours). This parameter is a failsafe; the actual time of charge will depend on depth of discharge. In 10 hours, this would allow 13A\*10H =130 AH to be delivered to the batteries.

#### Finishing Charge

For Lead Acid batteries, the second phase of charging is the “finishing charge” or “absorption charge” phase. The EVCC will only enter the finishing charge phase if the bulk charging phase completes successfully, if **termc** is reached. If the bulk charge phase terminates because of a charging timeout (**termt**), then this is considered an unsuccessful termination. The finishing phase will be skipped unless **fin\_maxv** is configured to be at least one volt greater than **maxv**.

For its cells, Trojan recommends a maximum voltage of 2.45 to 2.79v per cell, and a current limit of 1-3% of C<sub>20</sub>. This phase completes when the charging voltage rises to the target finishing voltage.

In the example of twelve 30XHS cells, here are suggested EVCC settings:

- **fin\_maxv=187.2**: The finishing voltage would be 2.6v \* 6 cells \* 12 batteries = 187.2v.
- **fin\_maxc=2.6**: Note that this is the same as the termc setting above.
- **fin\_termt=480**: (2 hours). **fin\_termt** may be set to “0” (or “forever”) but if it is set to a finite time, is a failsafe on this charging phase.

### Float Charge

Once Lead Acid batteries are charged, they may be kept on a “float charge” or “trickle charge.” Lead Acid batteries have a relatively high self-discharge rate and this phase keeps them topped up if the EV sits for an extended period of nonuse.

For its cells, Trojan recommends a float voltage of 2.2v per cell. A current limit is not explicitly specified.

In the example of twelve 30XHS cells, here are suggested EVCC settings:

- **flt\_maxv=158.4**: The float voltage would be  $2.2\text{v} * 6 \text{ cells} * 12 \text{ batteries} = 158.4\text{v}$ .
- **flt\_maxc=2.6**: Note that this is the same as the termc setting, above.
- **flt\_termt=0**: No timeout

### Limitations

The EVCC does not support “equalization charge.” This type of charging purposely overcharges the batteries in order to balance the cells. Higher charge cells bubble off excess charge as hydrogen gas, and lower charged cells “catch up.”

Temperature sensors are not supported in the EVCC, so the EVCC does not perform temperature compensated charging. The examples assume charging at a constant 25°C in a well-ventilated area.

**DISCLAIMER: This is an example only. These instructions do not cover all details or variations in the equipment and do not claim to provide for every possible contingency met in connection with installation, operation, or maintenance. It is strongly recommended that the user check with their battery supplier to determine appropriate charging parameters.**

## Charging Lithium-Ion Batteries

Lithium-ion battery packs are typically charged using a Constant Current to Constant Voltage (CCCV) profile. In this profile, a constant current is applied as the voltage rises until a maximum voltage is reached for the pack, and then the charge controller holds the charger output constant while it reduces current to maintain that voltage. The charge is terminated when the charge current has reduced to the termination current setting in the charge controller.

Typically, a lithium-ion pack is made up of individual cells placed in a series string to produce the desired pack voltage. The amp hour rating of a single series string of cells is the same as each individual cell, so a string of 100 amp hour cells will have the same amp hour rating no matter how many cells are in series.

Lithium-ion cells are very efficient energy storage devices when compared to lead acid batteries. However, they are much more sensitive to charge voltage, and can be damaged or catch fire if not maintained properly. A Battery Management System (BMS) is a safety system attached to a lithium ion battery pack to ensure that cell voltages remain within the manufacturer's specifications.

Smaller cells can be combined in parallel (connecting like poles) until the desired amp hour rating is achieved, and then identical groups are added in series to reach the desired pack voltage. In this case, each parallel group can be considered a single cell for BMS connections.

A typical Lithium Iron Phosphate (LiFePO<sub>4</sub>) cell may have a maximum charge voltage of 3.6 volts. A 48v nominal pack is constructed by adding 16 of this type of cell in series. The maximum pack voltage in this case will be: 3.6 volts x 16 cells = 57.6 volts.

Because lithium cells tend to change voltage very quickly at the end of a charge cycle, the charge controller (EVCC) is typically set to a target voltage slightly below the manufacturer's recommended maximum. For instance, if the EVCC is set to a maximum voltage (**maxv**) of 3.45 volts per cell, then the constant voltage phase of the charge cycle may be completed without exceeding the maximum cell voltage of any cell. In this case, the maximum charge voltage is calculated as follows: 3.45 volts x 16 cells = 55.2 volts.

For this example, the following command would be entered into the EVCC command line interface:

```
set maxv 55.2
```

The value used for **maxv** is determined by the cell chemistry and the number of parallel cells in the pack. Typically, the **maxv** setting should be slightly less than the published High Voltage Cutoff (HVC) rating for a cell, multiplied by the number of cells in series.

Example:

Lithium Iron Phosphate (LFP) cells with a 3.6v HVC rating might have a charge voltage of 3.45v. For a well-balanced pack with 32 cells in series using this value, the **maxv** setting would be determined as follows:

$$\mathbf{maxv} = (32 * 3.45\text{v}) = 110.4 \text{ volts}$$

For this example, the following would be entered into the EVCC interface:

```
evcc> set maxv 110.4
```

**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 BMS is required to safely monitor every cell while charging or discharging the battery pack.

## WARRANTY 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](mailto:connect@thunderstruck-ev.com)

For errors in this document, or comments about the product, contact  
[djmdilithium@gmail.com](mailto:djmdilithium@gmail.com)

## Document History

Rev 3.0	Aug 2018	- Initial Version
Rev 3.0.1	Sept 2018	- Minor updates
Rev 3.0.2	Jan 2019	- clarify operation of set <outn>
Rev 3.0.3	Nov 2019	- added J1772 Type 2
Rev 3.1.0	Oct 2020	- J1772, additional charge completion codes
Rev 3.1.1	Sept 2021	- consistency edits, feature updates
Rev 3.1.2	Mar 2022	- editorial clarifications
Rev 3.1.3	Aug 2025	- remove Basic model, add feature and content updates