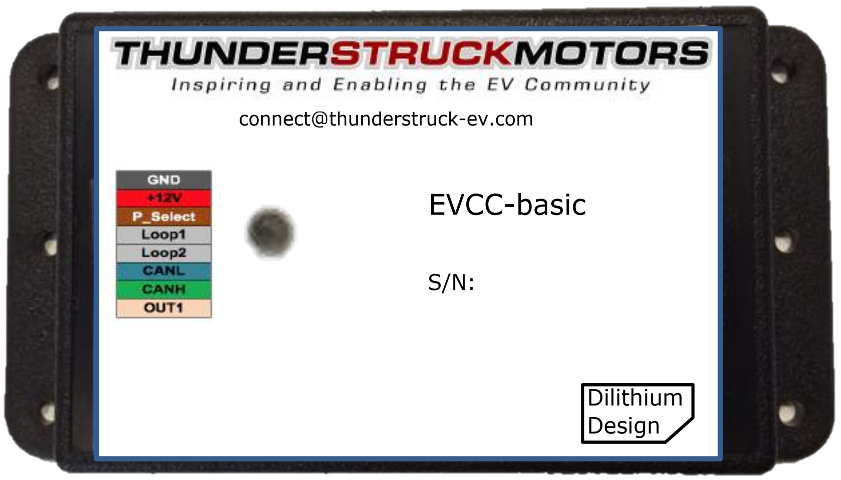
Thunderstruck Motors
EVCC-basic



Contents

Figures

Overview

The EVCC-basic is an electric vehicle charger controller (EVCC) which controls charging of a CAN enabled charger. The EVCC-basic has been optimized for applications that do not require an SAE-J1772 charge interface. See below for a system diagram.

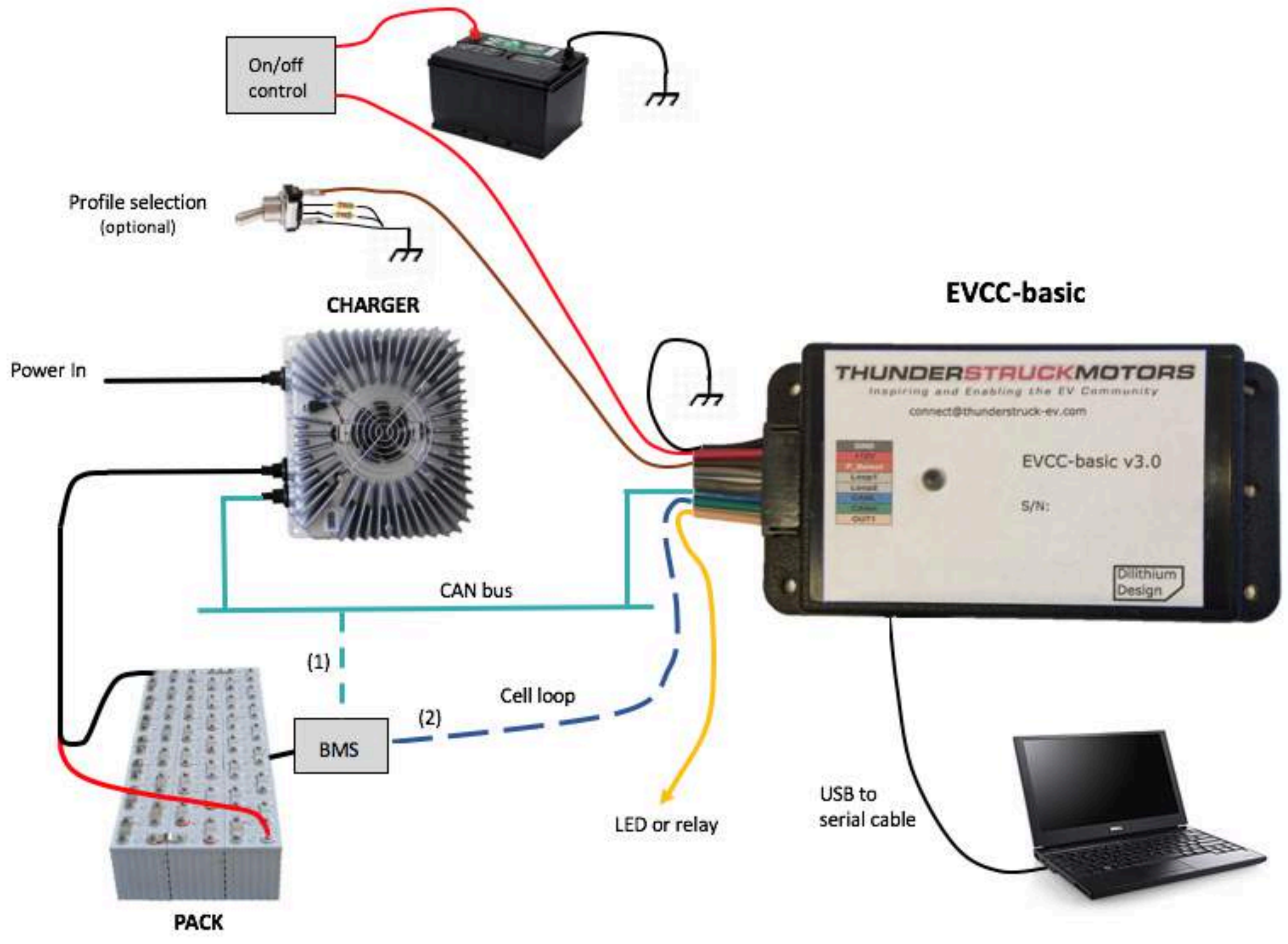


Figure 1 – EVCC-basic System Diagram

The EVCC-basic shares the hardware design and firmware of the Thunderstruck EVCC and many features of the standard EVCC are available unchanged. In this document, the terms EVCC and EVCC-basic are both used: the term "EVCC" is used for features in common, and the term "EVCC-basic" is used to call out features or operation specific to the EVCC-basic.

The EVCC supports several CAN enabled chargers, including the Thunderstruck TSM2500 charger. Charge voltage, charge current and maximum charge time are controlled by the EVCC. A constant current/constant voltage charge curve is supported for Lithium Batteries; and a three-phase charge cycle is supported for Lead Acid Batteries.

Charge history is provided which records the reason that charging stopped, total charge time, maximum voltage, maximum current, final current, and watt hours delivered.

The EVCC supports up to four parallel chargers for faster charging. When multiple chargers are configured, they are individually CAN addressed. Work is divided evenly between the chargers and statistics are gathered and recorded for each charger individually.

Determining cell overvoltage and undervoltage errors are functions of an EV Battery Management System such as the Dilithium Design BMS. The EVCC can be configured to use a BMS that supports a cell loop or CAN messaging to report pack status. If a CAN BMS is used, the EVCC can be configured to lower the charging current when a cell exceeds a "balancing threshold".

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

Mechanical

The EVCC-basic is enclosed in a Serpac WM010I enclosure, measuring 4.61 x 2.32 x 0.6. The datasheet for the enclosure is at http://www.serpac.com/userprints/wm010i_up_reva.pdf.

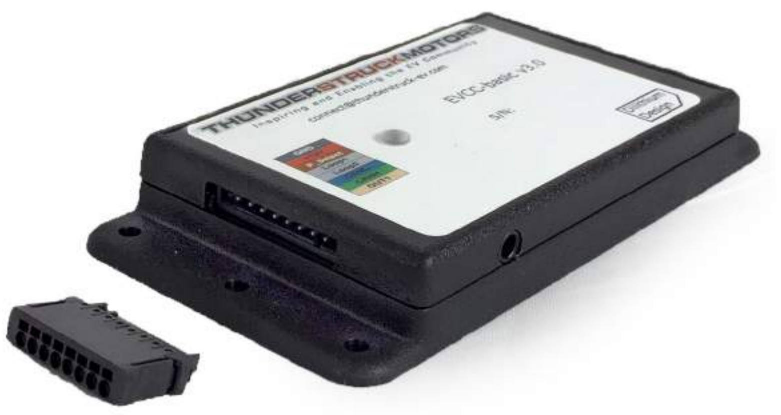


Figure 2 – EVCC-basic Housing and Connector

The EVCC-basic has a serial port, an LED, and one 8-pin connector.

Connections

The EVCC-basic uses an 8 pin push-in connector (Harting 14310813101000) that will accept from 20-24 gauge solid or stranded wire; 20ga stranded wire is recommended. To make a connection, strip the wire back approximately 1/2" and insert into the associated hole in the connector housing. Make sure that all conductors are correctly inserted so that adjacent wires do not inadvertently short together.

Removing the wire from the connector requires a removal tool. The recommended removal tool is Harting 14990000001. Alternately, the Molex KK terminal removal tool, part number W-HT-1884, is widely available and has been found to work well.

To remove a wire, insert the tool into the associated slot above the wire and wiggle it in. This will compress the spring holding the wire and the wire can be removed.

The figure below shows the pinout of the EVCC Connector

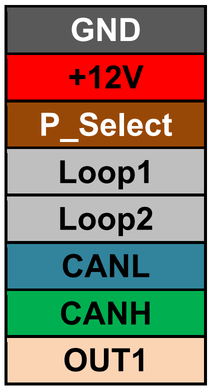


Figure 3 – Connector Pinout

- **GND** connects to the EV Chassis Ground.
- **+12V** connects to switched 12V power.
- **P_Select** is the profile selection input.
- **Loop1** and **Loop2** are used for the cell loop supervision circuit
- **CANH, CANL** connects to the CAN network
- **OUT1** is an output, which can be connected to an indicator light or a relay.

Power (GND, +12V)

GND and **+12V** are Power Inputs. Applying +12V will power the EVCC-basic on. The EVCC-basic draws approximately 25ma when on.

Profile Selection (P_Select)

The **P_Select** input is used to select an optional charge profile. This input may be left unconnected, connected to a switch to GND to enable two profiles, or connected to a multi-position switch and a resistor network and enable up to four profiles.

Electrically, **P_Select** input measures the resistance to chassis GND and deduces four possible selections: “inf”, 20K, 5K, and “0”. If **P_Select** is left unconnected, it will read “open” (or “infinite” resistance), and maps to “inf”. If **P_Select** is shorted to ground it will measure “0”. And, if a resistor is connected between **P_Select** and ground, the remaining two choices “20K” and “5K” are selected).

Each charge profile contains a complete copy of all charging parameters. Profile 1 is always created, used by default, and cannot be deleted. Additional profiles may be defined using the “**set profile**” command. The mapping from P_Select input value to profile number is accomplished with the command “**set map**”.

Note: the **measure pselect** command can be used to verify that this input has been connected correctly.

Cell Loop (Loop1, Loop2)

The EVCC has been designed for use with a Battery Management System, and the Cell Loop allows the BMS to enable or disable charging, as necessary.

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

By default, the EVCC applies +5v to **Loop1**, and verifies continuity between **Loop1** and **Loop2** by measuring the voltage at **Loop2**. The EVCC will allow charging only if the circuit is "closed", i.e., with an electrical connection between Loop1 and Loop2.

> It is expected that the user cell loop circuit 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 this may not provide enough "wetting current" for the relay contacts.

Alternately, the EVCC may be configured to detect a closure to ground (see the command "**enable loopground**"). Charging is allowed if the **Loop2** has a connection to chassis ground. In practice, **Loop2** could be connected to an open collector output controlled by a BMS and switched to ground.

> Note: the **measure loop** command can be used to read the value of this input.

CAN (CANH, CANL)

The EVCC uses CAN to communicate with the charger, and optionally, the BMS. CAN is a robust communications protocol originally designed for automotive applications and has error detection and recovery mechanisms.

CAN uses a two-wire interface: the signals are designated CANH ("CAN high") and CANL ("CAN low"). A CAN network is a daisy-chain, multi-station network that should be terminated on both ends of the string by 120ohm termination resistors. See below for a network diagram.

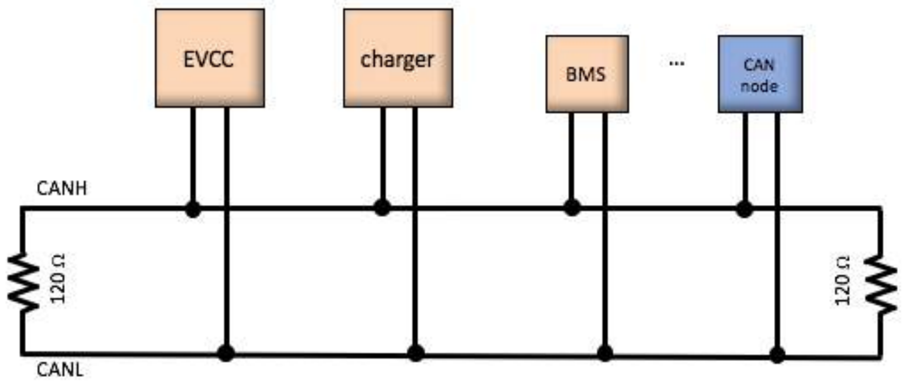


Figure 4 – CAN Network Diagram

CAN wiring should be kept short and the conductors should be twisted. Wiring stubs between the CAN network and the node should be kept as short as possible, ideally less than a few inches. CAN messages may be lost or corrupted as the result of EMI (Electro-Magnetic Interference), stubs that are too long, or improperly terminated cables. Network wiring should be placed away from EMI sources 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 simple case, a short run of hand-twisted wire generally works fine.

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

The CAN transceiver in the EVCC uses chassis ground as its ground reference. This is the convention of many chargers (including the TSM-2500 charger) and the Dilithium Design BMS.

Some CAN devices contain an internal termination resistor and should therefore be installed at the end of the CAN string.

> In order to verify the proper installation of the CAN terminations, one can make all connections and measure the resistance between CANH and CANL. The resistance between CANH and CANL should be 60 ohms, which indicates the presence of two 120 ohm resistors in parallel.

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**").

The internal EVCC termination resistor is not bridged onto the CAN network unless the EVCC is powered on. There might be a case where some CAN nodes are powered, the EVCC is a terminal node, and the EVCC is not powered up. In that (expectedly rare) case, a physical resistor should be used and the EVCC internal termination resistor should be disabled.

The EVCC CAN interface normally runs at 250Kbs. The baud rate is not software configurable, however, both the TSM2500 and ELCON chargers require this rate. Support is also available for the Lear charger which, if enabled, automatically reconfigures the CAN data rate to 500kbps. The EVCC supports both Extended Identifiers and Standard Identifier.

In order to verify correct CAN operation there is a can message tracing capability ("**trace can**") that can show CAN message traffic.

OUT1

The EVCC-basic supports a single output, OUT1. This output follows the operation of the EVCC-basic LED, and can be used to operate an indicator light or a relay. Electrically, this output is 12V output protected by a 200ma resettable fuse.

Note that this output has been renamed from the signal **Ext_Ind**, used in the existing v2.4 EVCC.

Operation

Bringup Checklist

Serial Interface

1) Install driver and terminal application on a PC.
2) Plug in the USB to serial port in the PC and EVCC.
3) Connect +12V, GND to the EVCC.

Verify Analog Inputs

1) If using the cell loop, connect to the cell EV cell loop circuit (or for testing, simply connect the two wires together). Using "**measure loop**", verify readings with cell loop open and closed.
2) If using charge profiles, connect P_Select to a selector switch (and resistor array, if more than two profiles are used). Verify the P_Select input, using the "**measure pselect**" command.

Verify CAN Network

1) Follow CAN network guidelines to connect the EVCC and charger to the EV CAN bus.
2) Verify proper installation of the CAN termination resistors. If the EVCC is not a terminal node, then verify that the **canterm** option is not enabled.
3) Verify CAN operation between EVCC and Charger. Power up the EVCC and charger, type "**trace can"**, to verify can message traffic.
4) If using a CAN BMS, connect and power the BMS. Verify CAN traffic from the BMS.

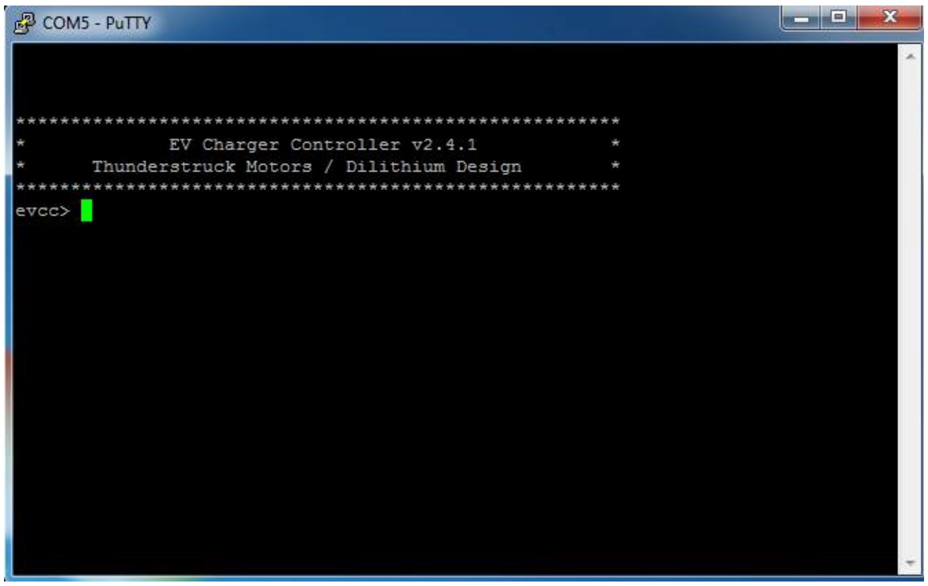
Configure the EVCC; Perform Systems Test

1) Set the bms to **loop** or **bmsc** using "**set bms"**.
2) Configure, at least, **maxv**, **maxc**, **termc** and **termt**.
3) Power the charger, EVCC, and BMS and verify proper charge operation using "**trace charger**".

Serial Interface

Before using the serial port, host computer drivers and a terminal application must be installed. See the document *Driver Installation* for details.

Connect the serial cable to the EVCC and to the host computer. Start the terminal application on the host computer. Apply power to the EVCC by providing a 12V supply to **+12V** and **GND**. The EVCC LED should turn on (see below for the blink patterns), and a banner should be displayed:



```
*********************************************************
*              EV Charger Controller v2.4.1             *
*       Thunderstruck Motors / Dilithium Design         *
*********************************************************
evcc>
```

At this point, the EVCC may be configured. Configuration is stored in non-volatile memory and retained across a power cycle.

Configuration Guidelines

The EVCC supports a three-phase charging algorithm. The first phase is the "bulk" phase. This phase is primarily controlled by the parameters: **maxv**, **maxc**, **termc**, and **termt**. The **maxv** and **maxc** parameters specify the requested target voltage and target charging current from the charger. In a normal charge cycle, the charger will begin charging at **maxc**, the target charging current. As the pack voltage begins to rise and approach **maxv**, the target voltage, the charger will reduce the charging current. The parameters **termc** and **termt** are used to determine the end of the bulk charging phase. The charging will stop when the charge current drops below **termc**, the termination current threshold. The charging will also stop if the charge time exceeds **termt,** the maximum charging time.

The second, optional phase, is the "finishing" phase, which is used to charge Sealed Lead-Acid batteries. It also has charging and termination limits: **fin_maxv**, **fin_maxc**, **fin_termt**. The finishing phase completes when the charging voltage rises to the target finishing voltage **fin_maxv** or exceeds **fin_termt**, the maximum finishing charge time.

The third, optional, phase is the "float" phase. This phase is used in applications such as solar power or long term EV storage, where it is necessary to keep the battery "topped up" and the charger remains running after the pack is full. This phase is controlled by the parameters **flt_maxv**, **flt_maxc**, **flt_termt**. If **flt_termt** is set to 0, then there is no timeout and the charger will remain on as long as the EVCC is powered.

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

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

Two additional parameters, **linev** and **linec** can be configured. These parameters characterize how much voltage and current is available from the service connection (e.g., at the input to the charger). The charge current requested from the charger is determined by several methods, based on what information that the user provides.

Using maxc only: The user can specify the maximum charge current explicitly and set no other parameters. In this case Charge Current Requested is simply **maxc**.

Set linev and linec only: In this case, the EVCC will perform a power calculation to determine a computed **maxc** value. The EVCC computes the power available from the service, derates it by a nominal 90% charger efficiency, and then computes an appropriate value for ChargeCurrent.

Charge Current Requested = (**linev** * **linec** * .9)/ **maxv**

Set linev, linec, and maxc: In this EVCC performs a power calculation and uses the minimum of power available and maxc.

Charge Current Requested = min (**maxc**, (**linev** * **linec** * .9)/ **maxv)**

The EVCC does not have the information on how much power a given charger can actually provide and it is possible to request a charger current that is higher than the charger can actually deliver. In that case, charger firmware will reduce the delivered current in order to stay within its power limits.

Charger Support

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

EVCC/Charger CAN messages are usually sent approximately twice a second, both from EVCC to Charger and from Charger to EVCC. If either the EVCC stops receiving these periodic messages, charging will terminate with a termination reason of "Charger Rx Timeout".

TSM2500

See *TSM2500 Series High Efficiency Intelligent Charger, ThunderStruck User Manual Ver 1.0.5*. http://www.thunderstruck-ev.com/images/ThunderStruck%20TSM2500%20ManualV1.05.pdf.

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

The TSM2500 charger does not have an integrated termination resistor, and is configured with a default CAN address. However, a termination resistor may be installed in the connector and the CAN address may have been reprogrammed by Thunderstruck as part of preparing the order.

The EVCC supports the following TSM2500 charger types:

- `tsm2500` - default
- `tsm2500_41`
- `tsm2500_42`
- `tsm2500_43`

The default value for tsm2500 chargers is "40". (The EVCC uses the CAN address 0x18e5**40**24 for messages TO the charger and 0x18eb24**40** FROM the charger to the EVCC).

The TSM2500 can report the following errors:

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

These strings are printed in "**trace charger**" output.

ELCON

ELCON chargers must programmed with the CAN option. In addition, an external ELCON-provided CAN module is needed that terminates the CAN. Only two pins are provided for the CAN connection: CANH and

CANL. The ELCON CAN module does not contain an integrated termination resistor.

The CAN addresses of the ELCON chargers are determined by the outboard serial to CAN converter. In order to change the CAN address, a different serial to CAN module is needed.

The EVCC supports the following ELCON charger types:

- `elcon` - default
- `elcon_e7`
- `elcon_e8`
- `elcon_e9`

The default value for ELCON chargers is "E5". (The EVCC uses the CAN address 1806**e5**f4 for messages TO the charger and 18ff50**e5** FROM the charger to the EVCC).

The ELCON charger can generate the following errors:

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

These strings are printed in "trace charger" output.

LEAR

The EVCC has support for some Lear chargers. This support is limited to the "control message" to the charger with CAN ID "0x0050" and the status message from the charger with CAN ID "0x0617". It has been found that not all Lear chargers use these messages due to different firmware and may not support this message set.

The EVCC only defines a single Lear charger type, named "`lear`". When a Lear charger is configured, the CAN datarate is changed from the default of 250kbps to the Lear required 500kbps.

Lear charger support has a number of limitations:
.
- Only one Lear charger may be configured. Having more than one Lear charger is not supported. Having one Lear charger and another non-Lear charger is not supported.
- If a CAN enabled BMS is being used with the EVCC, it must be configured for a 500kbps datarate.

Charger States and Termination Reasons

The EVCC-basic will be in one of the following states: WARMUP, CHARGE, CHARGE/TOPBALANCE, FINISH CHARGE, FLOAT CHARGE, and STANDBY.

The WARMUP state is a transitory state that starts a charge cycle. The next state is the CHARGE state (bulk charge). When the CHARGE state completes, the next state, if defined, is FINISH CHARGE, and the one after is FLOAT CHARGE, if defined. The CHARGE/TOPBALANCE state is used if the **topbalance** option is enabled.

The STANDBY state is used when the EVCC is powered up but not charging.

A number of conditions prevent or stop charging, including: cell loop error, BMS reported HVC, and charger message timeout. Each stage has a “normal” termination condition as well as a configured timeout for the state.

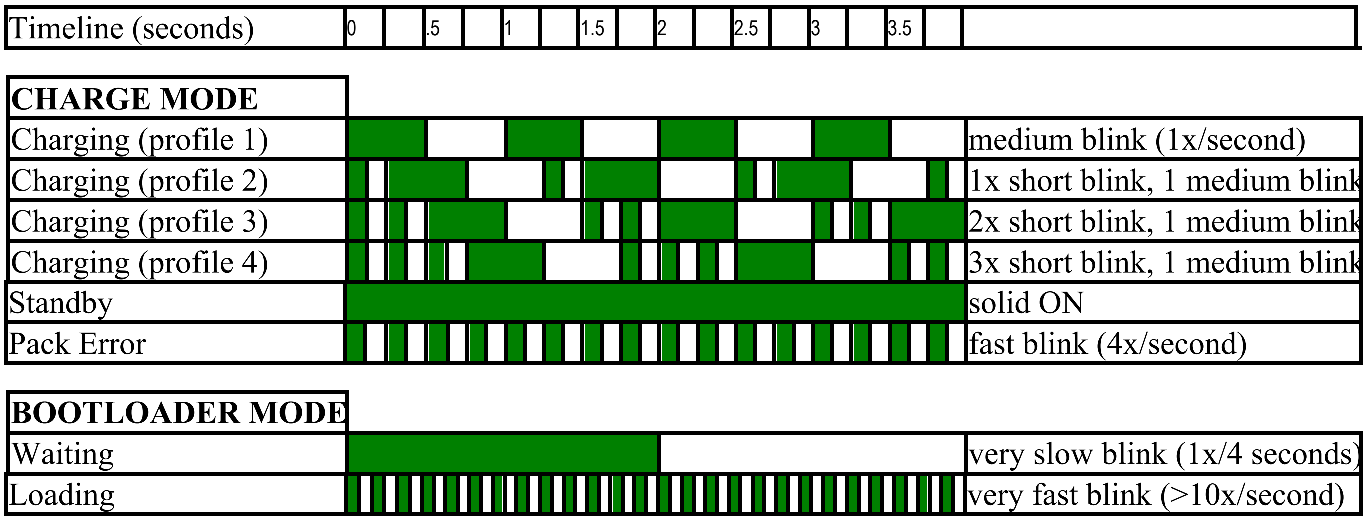
A charge history record is written after each charge attempt and is available with the **show history** command. The record is written at the end of the CHARGE (or FINISH CHARGE state, if defined), and will contain the charge “termination reason”. Charge termination reasons are one of: CELL LOOP, HVC, CHARGER RX TIMEOUT, PACK NOT CONNECTED, CHARGE TIMEOUT, FINISH CHARGE TIMEOUT, NORMAL, or FINISH CHARGE COMPLETE.

When the charge history record is written, the EVCC will either enter the FLOAT CHARGE state, if defined, and there are no conditions preventing it, or the STANDBY state. The FLOAT CHARGE state can be configured to run indefinitely, or a time limit can be set.

Normally, once the EVCC is in STANDBY, charging is considered complete and the EVCC will remain in the STANDBY state until power is cycled to the EVCC. The first exception is if the charger is power cycled (and the EVCC is not). In this case, the EVCC will first detect an RX TIMEOUT from the charger, which later clears. In this case, the EVCC will assume that a new charge cycle is needed and the EVCC will exit the STANDY state and start a new charge cycle from WARMUP. The second case is if a FLOAT phase is defined, and that there are no conditions that would otherwise prevent charging, including CELL LOOP and HVC. In this case, the EVCC will exit STANDBY and return to FLOAT.

LED Operation

The EVCC-basic LED has the following blink patterns:



CHARGE MODE	
Charging (profile 1)	medium blink (1x/second)
Charging (profile 2)	1x short blink, 1 medium blink
Charging (profile 3)	2x short blink, 1 medium blink
Charging (profile 4)	3x short blink, 1 medium blink
Standby	solid ON
Pack Error	fast blink (4x/second)

BOOTLOADER MODE	
Waiting	very slow blink (1x/4 seconds)
Loading	very fast blink (>10x/second)

Figure 5 – EVCC-basic LED Blink Patterns

Firmware Upgrade

The EVCC firmware may be upgraded by the customer using the serial port. Firmware upgrade instructions are provided in the document *Driver Installation*. Contact Thunderstruck if a firmware upgrade is needed.

Command Line Interface

Startup Banner

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

```
*******************************************************
*                   EVCC-basic v3.0                   *
*      Thunderstruck Motors / Dilithium Design        *
*******************************************************
evcc>
```

help

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

```
evcc> help
  SHow [<>|Version|Config|History]
        <>        - status
        version   - firmware version
        config    - configuration
        history   - charge history
  SEt [ <>
        |BMS
        |CHARGER|CHARGER2|CHARGER3|CHARGER4
        |PROfile|MAP
        |LINEV|LINEC
        |MAXV|MAXC|MAXBC|TERMC|TERMT
        |FIN_MAXV|FIN_MAXC|FIN_TERMT
        |FLT_MAXV|FLT_MAXC|FLT_TERMT
       ]
  REset [History|PROFILE]
        history      - reset charge history
        profile <n>  - deletes a charge profile
  ENable  [CANTERM|TOPBALANCE|LOOPGROUND]
        canterm   - enable CAN termination resistor
        topbalance- cell HVC stops current but not charging
        loopground- loop OK if loop2 is grounded
  DISable [CANTERM|TOPBALANCE|LOOPGROUND]
  TRace [<>|CHarger|CANbus|STate|OFF]
        <>        - trace toggle ON/OFF
        charger   - trace charger messages
        canbus    - trace canbus messages
        state     - trace EVCC state changes
        off       - disable all tracing
  MEasure [<>|LOOP|PSELect]
        <>        - 'measure' help
        loop      - measure Cell Loop A/D
        pselect   - measure Profile Selection A/D
  UPGRADE         - performs a firmware upgrade
```

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, current status will be displayed.

For the EVCC-basic, the possible values of `state` will be one of: CHARGE, CHARGE/TOPBALANCE/FINISH CHARGE, FLOAT CHARGE, or STANDBY.

A few examples are shown below:

```
evcc> show
  state    : STANDBY
  cell loop: OK
  bmsc     : receive timeout
  uptime   : 0 hour(s), 2 minute(s), 0 second(s)
```

In STANDBY mode. The cell loop and the bmsc are configured. The cell loop is OK, but the bmsc has a receive timeout.

```
evcc> show
  state    : CHARGE
  cell loop: OK
  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)
```

Bulk charging in progress.

show version

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

```
evcc> show version
  version  :    v3.0.2; Mar 15 2018 08:13:52
evcc>
```

show config

The **show config** command displays configuration parameters. At its simplest, the output of **show config** is the following:

```
evcc> show config
  bms      : loop
  charger  : tsm2500
  maxv     :  20.0V
  maxc     :   2.0A
  termc    :   0.2A
  termt    : 720.0hr
```

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, loop, bmsc, bmsc2 bmsc3, bmsc4)

- `charger` - the configured charger type
- `charger2` - (if configured) charger2 type
- `charger3` - (if configured) charger3 type
- `charger4` - (if configured) charger4 type
- `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 minutes).
- `fin_maxv` - (if configured) finishing charge voltage (for SLA charging)
- `fin_maxc` - (if configured) finishing charge current (for SLA charging)
- `fin_termt` - (if configured) finishing charge current (for SLA charging)
- `flt_maxv` - (if configured) float charge voltage
- `flt_maxc` - (if configured) float charge current
- `flt_termt` - (if configured) float charge current
- `options` - (if configured) `canterm` – can termination resistor enabled
  - (if configured) `topbalance` – cell HVC stops current but not charging
  - (if configured) `loopground` – loop OK if loop2 is grounded

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

```
evcc> show c
  bms      : loop
  charger  : tsm2500
  charger2 : tsm2500_42
  charger3 : tsm2500_43
  charger4 : elcon
  linev    : 220.0V
  linec    :  30.0A
  maxv     : 155.0V
  maxc     :  15.0A
  maxbc    :   1.2A
  termc    :   0.2A
  termt    :   6.0hr
  fin_maxv : 160.0V
  fin_maxc :   2.0A
  fin_termt:   4.0hr
  flt_maxv : 152.0V
  flt_maxc :   0.5A
  flt_termt:   0.0hr
  options  : canterm (CAN termination resistor enabled}
           : topbalance (cell HVC stops current but not charging)
           : loopground (loop OK if loop2 grounded)
```

If more than one charge profile is defined, **show config** will display the four 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
  charger   : tsm2500
  profiles   :    1         2*        3         4
    linev    : 220.0V
    linec    :  30.0A     12.0A
    maxv     : 155.0V    152.0V
    maxc     :  15.0A      2.0A
    maxbc    :   1.2A
    termc    :   0.2A      0.2A
    termt    :   6.0hr     8.0hr
    fin_maxv : 160.0V
    fin_maxc :   2.0A
    fin_termt:   4.0hr
    flt_maxv : 152.0V
    flt_maxc :   0.5A
    flt_termt:   0.0hr
  profile map:
    inf      :    x
    20K      :    x
    5K       :    x
    0        :              x
```

show history

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

The following 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, shows a normal charge completion with a charge time of 214 minutes and includes the number of watt hours delivered.

```
evcc> show history

      |   term    |  charge |           |   watt | maximum| maximum|  ending|
  num |  reason   |   time  |  charger  |  hours | voltage| current| current|
---------------------------------------------------------------------------
 last |    normal | 214 mins|tsm2500    |  3249Wh| 152.9V |    7.9A |    0.5A |
  - 1 | comm err |   2 mins|tsm2500    |     0Wh|   0.0V |    0.0A |    0.0A |
evcc>
```

The full set of "term reason" codes is:

- `pack flt` - a cell loop fault or HVC condition was detected
- `comm err` - communications error with the charger
- `pack disc` - no pack was detected
- `timeout` - the maximum charge time was reached
- `normal` - normal completion (charge current is less than terminating charging current)
- `fintimeout` - finishing charge timeout
- `fin normal` - normal termination of finishing charge
- `flttimeout` - float charge timeout

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

```
evcc> show history
```

```
     |   term    |  charge |            |    watt | maximum| maximum|  ending|
  num |  reason   |   time  | charger    |  hours | voltage| current| current|
--------------------------------------------------------------------------
 last |   normal |   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. For voltage and current, whole numbers (145) or decimal numbers (145.2) can be entered. The EVCC supports one decimal digit of precision.

set <>

Using the **set** with no parameters will option will print help for the "set" command.

```
evcc> set
  SEt [ <>
        |BMS
        |CHARGER|CHARGER2|CHARGER3|CHARGER4
        |PROfile|MAP
        |LINEV|LINEC
        |MAXV|MAXC|MAXBC|TERMC|TERMT
        |FIN_MAXV|FIN_MAXC|FIN_TERMT
        |FLT_MAXV|FLT_MAXC|FLT_TERMT
       ]
      <>                 - 'set' help
      bms configuration
        set bms [NONE|<bmsn>[,<bmsn>]]
                  <bmsn>  - [LOOP|BMSC|BMSC2|BMSC3|BMSC4]
      charger configuration
        set <chargern> <type> - defines <chargern>
        set <chargern> NONE   - deletes <chargern>
               <chargern> - [CHARGER|CHARGER2|CHARGER3|CHARGER4]
                   <type> - [ TSM2500|TSM2500_41|TSM2500_42|TSM2500_43
                              |ELCON |ELCON_E7 |ELCON_E8 |ELCON_E9
                              |LEAR]
      profile editing
        set profile <n>    - 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>      - 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
      SLA charge parameters
        set fin_maxv <v>  - finishing charge voltage
        set fin_maxc <a>  - finishing charge current
        set fin_termt <h> - finishing charge termination timeout
        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)
```

set bms

This sets the BMS type. The EVCC can use a cell loop and/or up to four CAN BMSs. The BMS determines whether a cell in the pack has exceeded the High Voltage Cutoff, Low Voltage Cutoff, or Balance Voltage Cutoff.

The following example just sets the bms type to be the cell loop.

```
evcc> set bms loop
```

The next example sets the bms to be the Dilithium BMSC.

```
evcc> set bms bmsc
```

The next example sets the bms to use both cell loop and CAN messaging.

```
evcc> set bms loop, bmsc
```

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 command sets a single charger

```
evcc> set charger tsm2500
```

The following command sets a second charger

```
evcc> set charger2 tsm2500_42
```

set profile <n>

This command selects a profile for editing. There are four possible profiles: 1-4. Initially, only profile 1 is defined: 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:

This is the default configuration:

```
evcc> show config
  bms       : loop
  charger   : tsm2500
  maxv      :  20.0V
  maxc      :   2.0A
  termc     :   0.2A
  termt     : 720.0hr
evcc>
```

This command creates Profile 2 with default configuration:

```
evcc> set profile 2
evcc> show c
  bms       : loop
  charger  : tsm2500
  profiles   :    1         2*        3         4
    maxv     :  20.0V     20.0V
    maxc     :   2.0A      2.0A
    termc    :   0.2A      0.2A
    termt    : 720.0hr   720.0hr
profile map:
    inf      :    x
    20K      :    x
    5K       :    x
    0        :    x
```

Now set some parameters in Profile 2:

```
evcc> set maxv 150
evcc> set maxc 12
evcc> show config
  bms      : loop
  charger  : tsm2500
  profiles   :    1         2*        3         4
    maxv     :  20.0V    150.0V
    maxc     :   2.0A     12.0A
    termc    :   0.2A      0.2A
    termt    : 720.0hr   720.0hr
profile map:
    inf      :    x
    20K      :    x
    5K       :    x
    0        :    x
```

Now return to Profile 1 and set some parameters in Profile 1:

```
evcc> set profile 1
evcc> set maxv 160
evcc> set maxc 15
evcc> show config
  bms      : loop
  charger  : tsm2500
  profiles   :    1*        2         3         4
    maxv     : 160.0V    150.0V
    maxc     :  15.0A     12.0A
    termc    :   0.2A      0.2A
    termt    : 720.0hr   720.0hr
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
  maxv     : 160.0V
  maxc     :  15.0A
  termc    :   0.2A
  termt    : 720.0hr
```

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 >= 30K	the result is "inf"
30K > R >= 10K	the result is "20K"
10K > R >= 2K	the result is "5K"
20K > R	the result is "0"

Each result is mapped to a profile using the profile map. Initially all choices select 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
    maxv     : 160.0V    150.0V
    maxc     :  15.0A     12.0A
    termc    :   0.2A      0.2A
    termt    : 720.0hr   720.0hr
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 command to enable the mapping from a 5K resistor to Profile 3 would be:

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

set linev, set linec

This sets the maximum line voltage and line current available.

```
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.

```
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" indication to the EVCC.

```
evcc> set maxbc .7
```

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, Finishing Charge for examples of use.

set flt_maxv, set flt_maxc, set flt_termt
These commands are used to define the "float charge" phase voltage, current, and charge time. If flt_termt is set to 0, then that is treated as "infinite" time.

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
```

enable

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

```
evcc> enable canterm
```

If this option is set, this will be indicated in the **show config** output.

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

enable 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. Enabling this option allows this

behavior. In this case, the cell loop / HVC condition does not stop charging, but all other termination reasons (overall time, charge plug disconnected, message timeout, etc) will still apply.

```
evcc> enable topbalance
```

If this option is set, this will be indicated in the **show config** output.

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

enable 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 is left unconnected.

```
evcc> enable loopground
```

If this option is set, this will be indicated in the **show config** output.

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

disable

disable canterm

The **disable canterm** command disconnects the CAN termination resistor.

disable topbalance

The **disable topbalance** command sets the charging behavior back to default (e.g., terminate a charging cycle when there is a cell loop or HVC condition).

disable loopground

The **disable loopground** command sets the cell loop supervision behavior back to default (e.g., measure continuity between Loop1 and Loop2).

trace

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

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

trace <>

Trace with no parameters toggles state trace on and off.

trace charger

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

```
evcc> trace charger
charger tracing is now ON
```

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

trace canbus

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

```
evcc> trace can
canbus tracing is now ON
evcc> 00:02:20.9        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 state transitions. It shows whether the EVCC is in DRIVE, CHARGE, or CHARGE/WARMDOWN, as well as the state of the J1772 charge plug.

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

```
evcc> trace state
state tracing is now ON
evcc> 00:06:53.4  old state=DRIVE, new state=CHARGE, term rsn=0
00:07:16.9  old state=CHARGE, new state=STANDBY, term rsn=EVSE UNLOCKED
```

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
    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 pselect

The **measure pselect** command gives a real time measurement of the **pselect** 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 mode, waiting for the load to begin. The EVCC must be power cycled in order to leave this mode.

```
evcc> upgrade
```

Configuring the EVCC with Multiple Chargers

Up to four chargers can be used in parallel for faster charging. See diagram below.

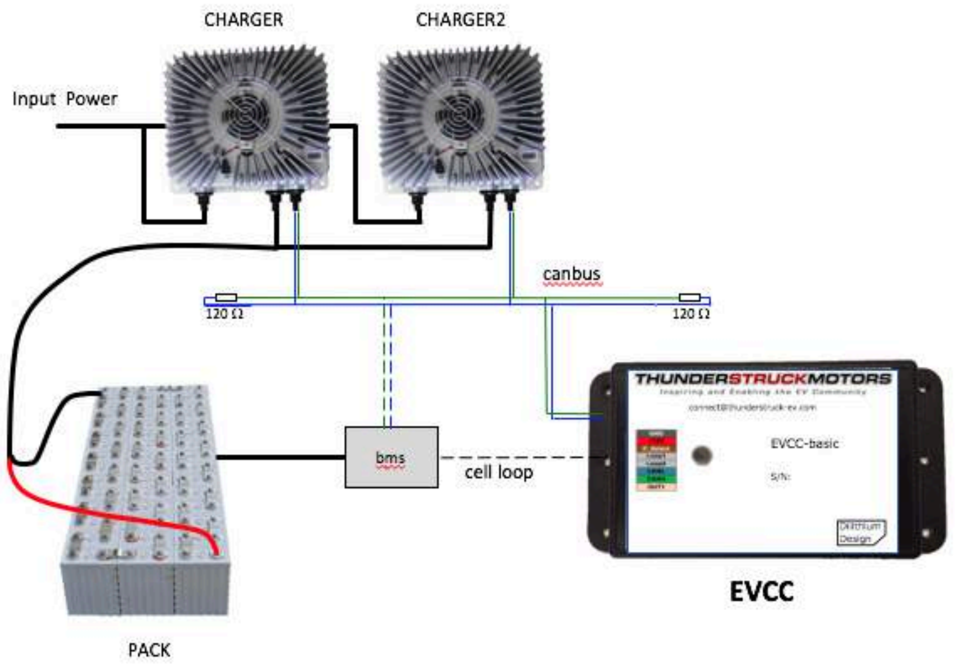


Figure 6 – Multiple Chargers - System Diagram

The chargers are in parallel and they charge a single pack. All chargers are placed on the CANBUS. There is a single EVCC and it communicates with the chargers independently.

There are several design considerations when installing multiple chargers.

- **Line power**. Two chargers require more power than a single charger. One must verify that adequate line power is available.
- **CAN wiring and addressing**. 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.
- **EVCC configuration**. Each charger must be explicitly configured in the EVCC.

Line Power

Power calculations are needed to make sure that there is sufficient power available to power all chargers. A 220V / 30A circuit has 6600Watts available. Two 2.5Kw chargers running at full power can be placed on the line, but three chargers cannot. (In contrast, a 110V / 15A circuit only has 1650Watts available).

CAN Wiring and Addressing

See the section on CANBUS, above, for general guidelines. When installing multiple chargers, care must be taken that 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.

EVCC Configuration

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) at the same time.

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

```
evcc> set charger tsm2500
evcc> show config
  bms       : loop
  charger   : tsm2500
  maxv      : 158.0V
  maxc      :  12.0A
  termc     :   0.5A
  termt     : 720.0hr
evcc>
```

This example defines a second charger, and sets its type to tsm2500_42.

```
evcc> set charger2 tsm2500_42
evcc> show config
  bms       : loop
  charger   : tsm2500
  charger2 : tsm2500_42
  maxv      : 158.0V
  maxc      :  12.0A
  termc     :   0.5A
  termt     : 720.0hr
evcc>
```

A charger can be deleted by setting the type to "none".

```
evcc> set charger2 none
```

Charging with Multiple Chargers

When charging with multiple chargers, maxc is divided by the number of chargers and requested from each charger. Below 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.

```
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
00:10:28.9  tsm2500   : V=126.3, A= 5.9, W=745, Wh= 0.09
00:10:29.3  tms2500_42: V=126.6, A= 5.7, W=721, Wh= 0.19
00:10:29.3  tsm2500   : V=126.6, A= 5.8, W=734, Wh= 0.19
00:10:29.8  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.30
00:10:29.9  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.31
00:10:29.9  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.33
00:10:30.0  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.34
00:10:30.0  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.36
00:10:30.1  tsm2500_42: V=127.2, A= 5.9, W=750, Wh= 0.37
```

Integration with CAN Enabled BMS

The EVCC can be used with a CAN enabled Battery Management System. 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 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 current cell balancers to prevent additional charging of 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.

BMS Operation

The programming of 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 250Kbps CAN data rate and 29 bit identifiers
 */

#define uint8     unsigned char
/*
 * BMS->EVCC message Identifier
 */
#define BMS_EVCC_STATUS_IND         0x01dd0001
#define BMS_EVCC_CELL_HVC_FLAG            0x01  /* set if a cell is > HVC */
#define BMS_EVCC_CELL_LVC_FLAG            0x02  /* set if a cell is < LVC */
#define BMS_EVCC_CELL_BVC_FLAG            0x04  /* set if a cell is > BVC */

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

Note that although the CAN message only has a 2 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.

EVCC Operation

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

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 mode and LVC is assumed if in DRIVE mode.

If the EVCC is configured to use only **can**, then the 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 **can** 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.

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 http://www.trojanbattery.com/pdf/TRJN0109_UsersGuide.pdf.

As an example consider a EV pack that consists of 12 Trojan 30XHS deep cycle flooded batteries, charging at 25°C (77°F). See the following from http://www.trojanbattery.com/pdf/TRJN0111_ProdSpecGuide.pdf. For reference, the C_{20} rating of 30XHS batteries is 130AH (this number comes in handy below).

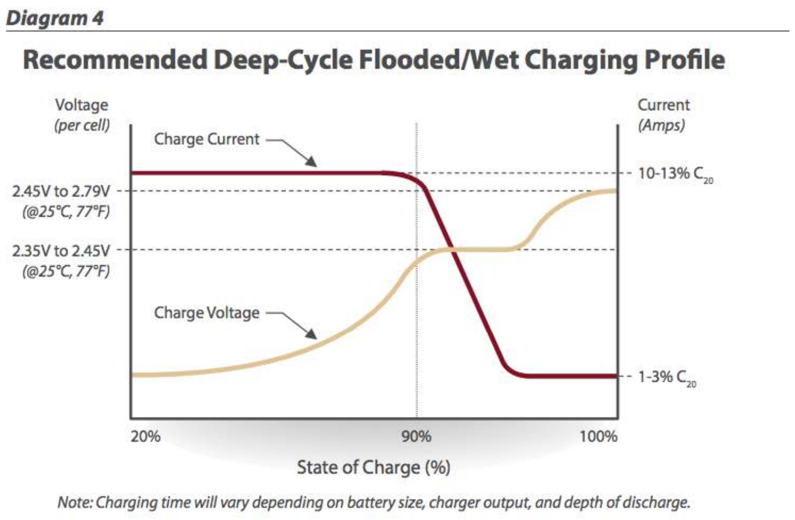


Figure 7 – Flooded Lead Acid Charging Profile (Trojan)

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 both Lithium and Lead Acid chemistries (including flooded, AGM, and Gel).

See Figure 7, 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_{20}. The bulk charge phase completes when the charging current drops to 1-3% of C_{20}.

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_{20} rating of 130AH, the charging current would be 13A.
- **termc=2.6**: The guidelines are 1-3% of C_{20}. 2.6A is 2% of the C_{20} 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. (In particular, if the bulk charge phase terminates because of a charging timeout [termt], then this is considered an abnormal termination).

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_{20}. 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 2.2v * 6 cells * 12 batteries = 158.4v.
- **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 overchargers 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 assumes charging at a constant 25°C in a well ventilated area.

DISCLAIMER: This is an example. 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.

Warrantee and Support

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

The EVCC-basic is warranted to be free from defects in components and workmanship under normal use and service for a period of 1 year.

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

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

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

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

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

Document History

Rev 3.1	March 2018	- Initial Version
Rev 3.2	May 2018	- Corrected description of OUT1