

# CAN-bus BMS Protocol

## CONFIDENTIAL

This document is intended for manufacturers of Managed Batteries: batteries with a CAN-bus connected BMS that communicate with a Victron system. This document describes the protocol used.

## 1. General

### 1.1 The central GX Device

The BMS of the battery is connected to a VE.Can or BMS-Can port on the [GX-device](#).

The GX-device is the central master in a Victron system.

Victron features different models of GX devices, and some inverter/chargers also include a built-in GX-Card. For details see the public [GX product range](#) page.

For examples of detailed installer documentation, refer to the [BYD + Victron](#) or the [Pylontech + Victron](#) manual.

All GX devices run the same software, called Venus OS. A changelog of Venus OS, including changes to its CAN-bus BMS protocol implementation, is available at <https://professional.victronenergy.com>. Create a login there, go to Firmware → Venus OS, and then see the word document `Venus OS - firmware changelog.docx`.

### 1.2 CAN-bus specification

- Standard frame format, 11-bit identifier, also known as CAN 2.0A.
- The preferred and common bitrate is 500kbps. Some of our GX Devices support only that bitrate and configuration. Some other GX devices have an option to select between 250kbps and 500kbps bit rates. In that case, the installer needs to select the correct profile when setting up the system, in the Settings → Services → VE.Can port menu. For details on what GX device supports what, see BMS-Can port on our [GX Product range page](#).
- There is no built-in termination in a GX Device: at least one termination plug ("VE.Can RJ45 Terminator") must be installed.
- Some Victron CAN-bus ports are isolated, others are not. All Victron CAN-bus ports on GX devices do not need any externally supplied voltage to function. All ports, including isolated ports, feature an (isolated) power supply that powers the CAN-bus transceiver and surrounded circuitry. Therefore it's always sufficient to connect Minus, CAN-H and CAN-L.
- Victron stocks and sells two types of cables to convert between Victron CAN-port pinning and typical battery CAN-port pinning - both RJ45 but different pinning. These are called the VE.Can to CAN-bus BMS cables; pin-out and more information on the [product page](#).
- For pin-out of the Victron RJ45 CAN connection, see manual on above linked page. Note: do not use any of the unconnected pins, they are reserved for future use for the Victron VE.Can protocol; a different protocol.

## 1.3 VE.Can vs. CAN-bus BMS

VE.Can is a proprietary Victron protocol, also based on CAN-bus, but not the same as the CAN-bus BMS protocol.

VE.Can is 29-bit, 250kbps, and has a completely different message structure.

On most GX Products, the CAN ports can be configured to be either one.

Note that, while technically possible to combine both in one CAN-bus network (by setting the BMS to 250kbit and using the fact that one protocol is 11-bit and the other 29-bit and thus can be mixed without software conflicts), such installation is not allowed for new integrations.

## 1.4 Communication timeouts

### Timeout mechanism implemented in the BMS

In case the BMS / Battery has a safety feature that disconnects the pack with an internal contactor in case it no longer receives the 0x305 keep alive messages, make sure that the time out is not too short. The minimum advised time out is 10 minutes, in order to cater for firmware updates and reboots of the Victron GX Device.

### Timeout mechanism implemented in the Victron system

When the inverter does not receive the 0x351 message from the battery every 3 seconds, the system will stop charging and discharging. Our advice is to make the BMS send this message every second.

## 1.5 CVL & CCL implementation

### 1.5.1 Introduction

NOTE: This is probably the most important part of this document for BMS developers. Read it carefully.

At a fundamental level, Victron's approach to interacting with BMSes is in the form of 3 setpoint parameters:

- Charge Current Limit (CCL)
- Charge Voltage Limit (CVL)
- Discharge Current Limit (DCL)

Once the BMS is communicating with the GX device over CAN-bus, these parameters are listed in the "Parameters" menu in the battery device on the Remote Console.

All three parameter, but at least the CVL (Charge Voltage Limit) and CCL (Charge Current Limit), are sent by the BMS to the Victron system. These are the only parameters that control the charging

behaviour.

Victron's approach to BMSes and chargers is focused DC-Coupled solar systems where there can be multiple charging sources, usually a combination of MPPT solar charger and charger/inverters.

### 1.5.2 Typical AC-Coupled solar systems (non-Victron)

Typically, BMSes designed for the common AC-Coupled Energy Storage Type systems hardcode a CVL, which remains the same in all situations. And will reduce CCL to 0, or near-zero, once the battery is fully charged, and/or one of the cells is fully charged.

That strategy **does not** work with a Victron system.

Below two chapters explain why  $CCL=0$  will not work for a Victron system, as well as recommendations for the alternative implementation: controlling on voltage.

### 1.5.3 DC-Coupled solar systems (Victron)

As (briefly) explained in the specification of the CCL/CVL fields in chapter 2, a typical Victron system with a MPPT solar charger (ie. a DC-Coupled system), cannot function properly with a  $CCL=0$  instruction.

Here is why: in a DC-Coupled system, the interaction between the MPPT solar charger and the inverter/charger works on voltage. The aim of the system is to have the inverter/charger sell any excess PV power back into the grid. This excess PV power is coming from the MPPT solar charger: a separate device, with its own controller.

By way of example, if the BMS sends a CVL of 58V, the MPPT solar charger will add 0.4V to the CVL for a operating charging voltage of 58.4V. At the same time, the inverter/charger will try and hold the voltage down to the CVL value of 58V, feed-in limits and maximum inverter power permitting. As a result, the MPPT solar charger will keep operating at its MPP point, pushing power onto the DC-bus, while the inverter/charger works to remove that power from the DC-bus by inverting it and feeding it to the AC side.

The solution is for the battery to lower its CVL setpoint as soon as it wants to reduce charge into the battery. This reduction must be enough to stop the battery voltage rising, but not so much that it causes the system to discharge the battery: a setpoint that is too low will cause the Victron system to discharge the battery to that setpoint.

The other option, setting  $CCL=0$  when the battery is fully charged, will result in the Solar chargers being curtailed. And in laymen's terms: once the battery is fully charged will stop solar production.

For most batteries, a hardcoded **lower** CVL that is put in effect when the BMS wants to stop charge (and would have normally sent a  $CCL=0$ ) works sufficiently. A control loop (in the battery) that dynamically determines the CVL works even better; especially in case of high imbalance, which we see regularly in new installed systems.

Don't under estimate the amount of time needed to test and tweak such dynamic algorithm: it needs to be very stable and slow to prevent oscillations. It is probably not helpful to update the parameters more than once every 20 seconds, but do test intensively to determine your own timing.

Implementations that set the CVL to the currently measured battery voltages minus some constant should be avoided. As mentioned previously, this will cause the system to discharge the battery. Also that won't work due to voltage drops and calibration differences between the BMS and the Victron system.

#### **1.5.3.1 Testing your BMS with with Feeding Back to the Grid**

1. Install a system with inverter/charger, solar charger, GX Device and your battery
2. Connect a (HV) power supply to the PV input of the solar charger
3. Configure the Inverter/charger for ESS (install the ESS Assistant and set a grid code with VEConfigure 3)
4. Make sure the inverter/charger is connected to mains via AC input
5. Allow for more than enough simulated solar power available on the solar charger (ie. in excess of any load on AC output)

In this situation, wait for the battery to fully charge and ensure that:

1. the battery does not switch off in an over-voltage error
2. solar power is fed back into the grid, in a stable manner
3. system can handle switching on and off of sizeable AC loads

#### **1.5.3.2 Testing your BMS with Cell Imbalance**

1. First create an imbalance by charging one cell more than the others
2. Install a system with inverter/charger, solar charger, GX Device and your battery
3. Connect a (HV) power supply to the PV input of the solar charger
4. Configure the Inverter/charger for ESS (install the ESS Assistant and set a grid code with VEConfigure 3)
5. Make sure the inverter/charger is connected to mains via AC input
6. Allow for more than enough simulated solar power available on the solar charger (ie. in excess of any load on AC output)

In this situation, wait for the battery to fully charge and ensure that:

1. the battery does not switch off in an over-voltage error
2. solar power is fed back into the grid, in a stable manner
3. system can handle switching on and off of sizeable AC loads

#### **1.5.3.3 Testing your BMS with Excessive PV Power**

1. Make more power available on the solar charger than the inverter/charger can feed-back
  1. Eg. Set a low input current limit on the inverter/charger (VictronConnect or VEConfigure 3)
2. The DC voltage on the battery terminals of the inverter/charger is expected to rise to CVL + 0.4V
3. Ensure that the battery/BMS can handle this situation

### 1.5.4 Victron & AC Coupled solar systems

An AC Coupled Solar system (read [here for more info](#)), has similar but slightly different challenges. With an Offgrid (no connection to mains) AC Coupled solar systems its simple fact that DC over voltages can (and will) occur in case battery is full, high power AC load is enabled, PV is fully producing, and then this AC-load is switched off at once. The system will take some time to react (inverter/charger needs to increase Ac output frequency on which the PV Inverter needs to react by reducing generated AC power, during which the PV power will continue to be pushed onto the DC bus ie. battery.

Depending on make and model of the PV Inverter, and its country settings, this period of continuous charge while the battery was already full can vary.

The solution for a stable system is to make sure to not too quickly switch the battery off on cell over voltage. Off-grid systems need, by design and requirements, more leniency than typical grid-connected self-consumption type systems.

## 2. Messages sent from BMS to the Inverter

The minimum CAN-IDs required for the core functionality are 0x351, 0x355, 0x356 and 0x35A. All other fields are optional and the system will operate without it. For officially supported batteries, we also need 0x35E and 0x35F.

CAN ID	Offset (bytes)	Name	Data type	Scaling	Unit
0x351	0	Charge voltage limit (CVL)	un16	0.1	V
	2	Max charge current (CCL)	sn16	0.1	A
	4	Max discharge current (DCL)	sn16	0.1	A
	6	Discharge voltage	un16	0.1	V
0x355	0	SOC value	un16	1	%
	2	SOH value	un16	1	%
	4	High res SOC (optional)	un16	0.01	%
0x356	0	Battery voltage	sn16	0.01	V
	2	Battery current	sn16	0.1	A
	4	Battery temperature	sn16	0.1	C

CAN ID	Offset (bytes)	Name	Data type	Scaling	Unit
0x35A	0 (bit 0+1)	General alarm	(not implemented)		
	0 (bit 2+3)	Battery high voltage alarm			
	0 (bit 4+5)	Battery low voltage alarm			
	0 (bit 6+7)	Battery high temperature alarm			
	1 (bit 0+1)	Battery low temperature alarm			
	1 (bit 2+3)	Battery high temperature charge alarm			
	1 (bit 4+5)	Battery low temperature charge alarm			
	1 (bit 6+7)	Battery high current alarm			
	2 (bit 0+1)	Battery high charge current alarm			
	2 (bit 2+3)	Contactor Alarm	(not implemented)		
	2 (bit 4+5)	Short circuit Alarm	(not implemented)		
	2 (bit 6+7)	BMS internal alarm			
	3 (bit 0+1)	Cell imbalance alarm			
	3 (bit 2+3)	Reserved			
	3 (bit 4+5)	Reserved			
	3 (bit 6+7)	Reserved			
	4 (bit 0+1)	General warning	(not implemented)		
	4 (bit 2+3)	Battery high voltage warning			
	4 (bit 4+5)	Battery low voltage warning			
	4 (bit 6+7)	Battery high temperature warning			
	5 (bit 0+1)	Battery low temperature warning			
	5 (bit 2+3)	Battery high temperature charge warning			
	5 (bit 4+5)	Battery low temperature charge warning			
	5 (bit 6+7)	Battery high current warning			
	6 (bit 0+1)	Battery high charge current warning			
	6 (bit 2+3)	Contactor warning	(not implemented)		
	6 (bit 4+5)	Short circuit warning	(not implemented)		
	6 (bit 6+7)	BMS internal warning			
	7 (bit 0+1)	Cell imbalance warning			
	7 (bit 2+3)	System status (online/offline) [1]			
	7 (rest)	Reserved			
0x35E	0	Manufacturer name	string (8 chars)		
0x35F	0	Battery Model	un16		
	2	Firmware version	un16		
	4	Online capacity in Ah	un16		
0x370	0	Battery / BMS name part 1	string (8 chars)		
0x371	0	Battery / BMS name part 2	string (8 chars)		
0x372	0	Number of modules OK	un16		
	2	Number of modules blocking charge	un16		
	4	Number of modules blocking discharge	un16		
	6	Number of modules offline	un16		
0x373	0	Min. cell voltage [1]	un16	mV	
	2	Max. cell voltage [1]	un16	mV	
	4	Lowest cell temperature [1]	un16	Kelvin	
	6	Highest cell temperature [1]	un16	Kelvin	

CAN ID	Offset (bytes)	Name	Data type	Scaling	Unit
0x374		Min. cell voltage id string [1]	string (8 chars)		
0x375		Max. cell voltage id string [1]	string (8 chars)		
0x376		Min. cell temperature id string [1]	string (8 chars)		
0x377		Max. cell temperature id string [1]	string (8 chars)		
0x378	0	Energy in [1]	un32	100	Wh
	4	Energy out [1]	un32	100	Wh
0x379	0	Installed capacity [1]	un16		Ah
0x380	0	Serial number, first 8 characters [1]	string (8 chars)	7-bit ascii	
0x381	0	Serial number, last 8 characters [1]	string (8 chars)	7-bit ascii	
0x382	0	Battery family name, 8 characters	string (8 chars)	7-bit ascii	

[1]: Victron extension, available since Venus OS v2.40.

### 0x351 - Charge voltage (CVL), discharge current limit (DCL) and charge current limit (CCL) and battery discharge voltage

- Discharge voltage is not used (yet). Instead, the installer is instructed to configure the DC low disconnect voltages according to the battery specification.
- CCL: a dc-coupled ESS system (ie. a system with solar chargers and installed and configured to Victron ESS instructions) cannot accurately handle CCL=0 while also using PV to power the loads. Our advice is to avoid sending CCL=0 when the battery is full. Send a reduced CVL instead.
- Test with the Feedback excess solar setting enabled. A CVL that is too low will cause the battery to be discharged, while a higher limit could cause over charging. See also the quirks section here: <https://github.com/victronenergy/dbus-systemcalc-py/blob/master/delegates/dvcc.py>
- When the grid is disconnected, ie the system is running in inverter aka island mode, the inverter cannot control the discharge current. The discharge current depends solely on the connected AC loads in such situation (which is unlike a grid connected self-consumption system where the inverter **can** control and reduce discharge when required). In other words, the inverter will ignore DCL when in island mode, unless DCL=0:
- A DCL of 0A will disable discharge.
  - If the grid is connected, loads will be powered from the grid. Battery power will not be used.
  - If the grid is disconnected, the Multi will switch off.
  - Charging will still be allowed if max charge current exceeds 0A.
- Both max discharge current and max charge current should be communicated as positive values.

### 0x355 - SOC information

- High res SOC is not supported (yet).

### 0x356 - Batt voltage, current and temp

- By default, a negative current indicates discharge, positive current indicates charge. Note that its possible for us to work with the reverse definition as well. We can make an exception based

on the identification sent in 0x35E and 0x35F.

- Temperature is in degrees centigrade.

## 0x35A - Warnings and Alarms

In this message, each warning and alarm is implemented to consist of two bits.

Bit 0	Bit 1	
0	0	Alarm/warning not supported
1	0	Alarm/warning active
0	1	Alarm/warning inactive (status = OK)
1	1	Alarm/warning not supported

Data is shown to the user on the GX Devices, as well as logged to the [VRM Portal](#).

Use 00 for types of warnings and alarms that are not supported by the BMS, and use 10 and 01 for those that are supported. Do not use 00 or 11 to indicate that status is OK.

The reason for this bit pattern is that we are moving to an implementation that can automatically hide fields that are not supported from the user interface. This will help us showing 100% correct information to users and installers for all different batteries and their implementations, without having to hardcode a map of supported fields per battery manufacturer, model, and/or firmware version.

Currently all alarms and warnings are still visible, except for a few hardcoded battery makers for whom we know that they use only 01 and 10 for supported warnings and alarms.

Lastly, note that these fields have no effect on the charging or discharging behaviour of our inverter/chargers, chargers or other products. The only message that is used for control, is 0x351.

### Byte 7 bit 2&3 - System status (online/offline)

A simple bit. To indicate if the main relay (or MOSFETs in certain designs) are enabled or not. This information is invaluable when remotely diagnosing problems in a system. While it can be deduced from alarm flags in some cases, the purpose of this bit is to have the status in an unambiguous way without having to know the specifics of a certain brand or type of battery.

## 0x35E - Manufacturer name

- 8 char long name, encoded in 7-bit ascii
- The data will be used to identify the battery, and may be used as battery name (see below for more information)
- If this information is not provided, a generic name will be used. See 2.1 below.

## 0x35F - Battery type and software versions

- Two byte type ID (not implemented, ambiguously used by some existing batteries, also see 0x382)



- Two byte firmware versions (big-endian ordering, MSB first)
- Unsigned 16-bit (little-endian) usable/online capacity of battery (also see 0x379). Set to zero if unused.

### **0x370 and 0x371 - Battery / BMS Name (Victron only fields)**

- together these two fields can contain a 16 char long name of the battery.
- encoded in 7-bit ascii
- the data is concatenated, starting with 0x370, and appending 0x371 to the right.
- If the name is shorter than 16 characters, the remainder should be padded with zeros.
- If the name is 8 characters or less, it is better to use 0x35E and omit 0x370 and 0x371.
- The contents of 0x35E and 0x370 must match if both are implemented.

### **0x372 - Module status**

This field has been designed such that it works both for battery types that have MOSFETs in each battery module as well as types that have contactors (or both).

The intention behind this field is to help installers and support engineers do initial troubleshooting on systems that are not performing correctly.

Send 0xFFFF for fields that are not supported. For example, a battery with only a Contactor will send 0xFFFF for fields 2 and 3.

### **0x373 to 0x377 - Min/max cell voltage and temperature**

0x373 contains the data.

- For any unsupported fields, send either 0x0000, 0xFFFF, or when not sending temperature, shorten the message using a lower DLC (Data Length Code).

0x374 to 0x377 is for identification:

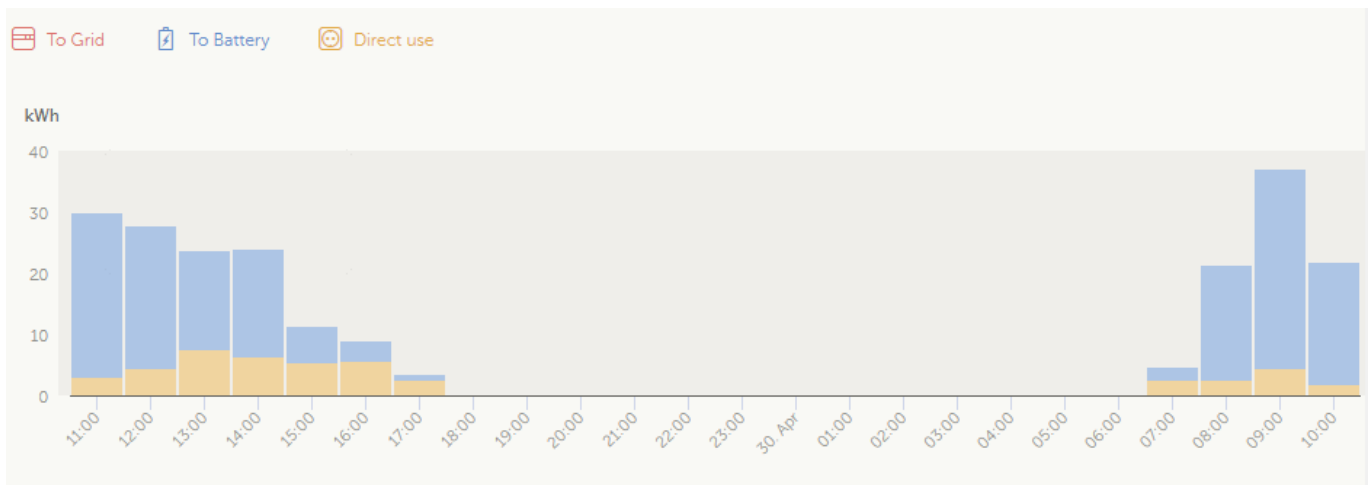
- The strings are to be encoded in 7-bit ASCII. Unused characters to be set to 0. Unprintable characters will be ignored.
- Free format: It is up to the battery manufacturer to put something in here that helps an engineer to identify which cell or unit is giving problems.
- Use the string to allow a service engineer to identify which package/module/cell is giving a low or high voltage.
- If a simple integer is to be used for identification, it must be encoded as ASCII.

### **0x378 - Total energy in and out**

- Two 32-bit values for energy in (charge) and energy out (discharge).
- These counters are used to generate the data on our VRM dashboard. See example below.
- It is recommended that these values are stored in non-volatile storage.
- If storage is not feasible, an acceptable compromise is to restart at zero. The algorithm used to

generate graphs on VRM can handle resetting counters.

- 100Wh units are used because this gives the best balance between resolution and maximum value.
  - A maximum energy value of 42MWh can be recorded.
  - If this value is exceeded, the counter should roll over to zero.



## 0x379 - Installed capacity

This is different from the similar parameter in 0x35F. In 0x35F in bytes 4 & 5 the usable capacity in Ah is given as a UN16. At 48V this translates to maximum possible capacity of just over 3MWh, which should be sufficient for the foreseeable future. It is recommended that batteries send the usable online capacity: if one or more modules have disconnected, send a lower total usable capacity.

The purpose of 0x379 is to show the installed capacity, so that the service engineer can see how much capacity is offline.

## 0x380 and 0x381 - Serial number

This field is optional and may not be usable by all battery manufacturers. Some example usages are:

- Systems with a single BMS
- For batteries that have their own cloud-function, send an ID that allows the service technician to identify the same battery on the Battery Manufacturer Cloud
- The definition is the same as for 0x370 and 0x371, and the cell voltage and temperature id fields.
- Free format ASCII.
- Both 0x380 and 0x381 must be sent. The serial number is only considered complete once 0x381 is received. If the serial number is 8 characters or less in length, then pad 0x381 entirely with zeroes.

## 0x382 - Battery family name

This field is optional and supplementary to the information in 0x35E. It identifies the name of the battery family or series. The purpose of this field is to make it possible to distinguish between

different types of batteries made by the same manufacturer. The Format is the same as for 0x35E: A 7-bit ASCII string identifying the family name of the battery.

### 3. Messages sent by the Inverter to the BMS

#### 0x305 [Keepalive from inverter to BMS]

Sent to the battery every second by the GX device (data consists of 8 bytes, all zeros).

#### 0x307 [Inverter identification from inverter to BMS]

Sent to the battery every second. This message contains the ASCII string 'VIC' in bytes 4 to 6.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x12	0x34	0x56	0x78	V	I	C	0x00

Byte 7 is reserved for future use.

### 4. FAQ

#### Q1: Do you need to see the 0x359 frame (warnings and alarms) to operate?

No, that is a battery manufacturer specific CAN-ID which we implemented because that manufacturer does not support 0x35A. For all other manufacturers, we prefer 0x35A over 0x359.

#### Q2: How does the GX Device determine the battery name?

There are several sources used by the GX Device to determine the name of the battery. In order of priority:

- The data of CAN-IDs 0x370 and 0x371 if present.
- The data of CAN-ID 0x35E if present and recognised. The data may be used literally, but may also be replaced by a more readable name.
- If the GX Device identifies the battery by some other means, a hard coded name will be used.
- If all else fails, the name will be *CAN-bus BMS battery*.

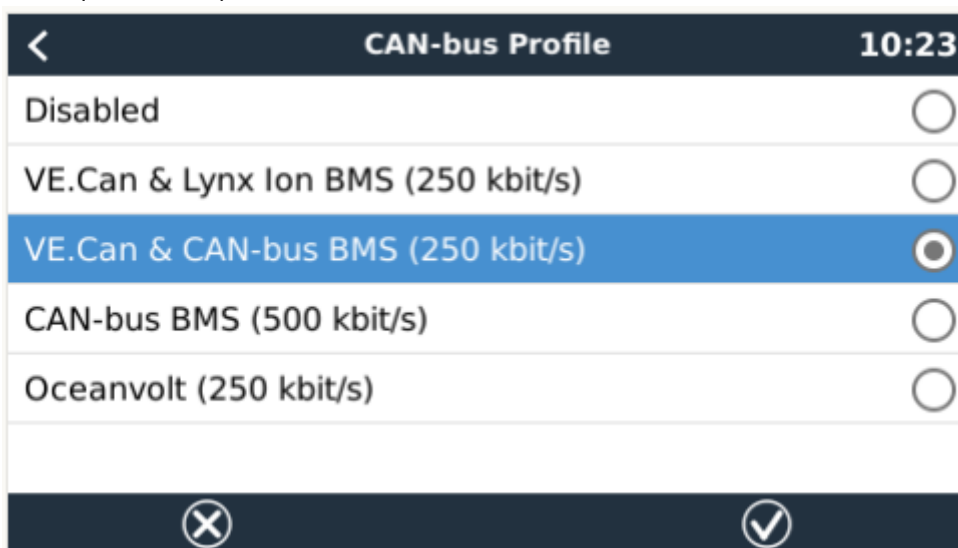
### 5. Verifying implementation

After implementing this protocol in your BMS, follow these steps to verify proper working:

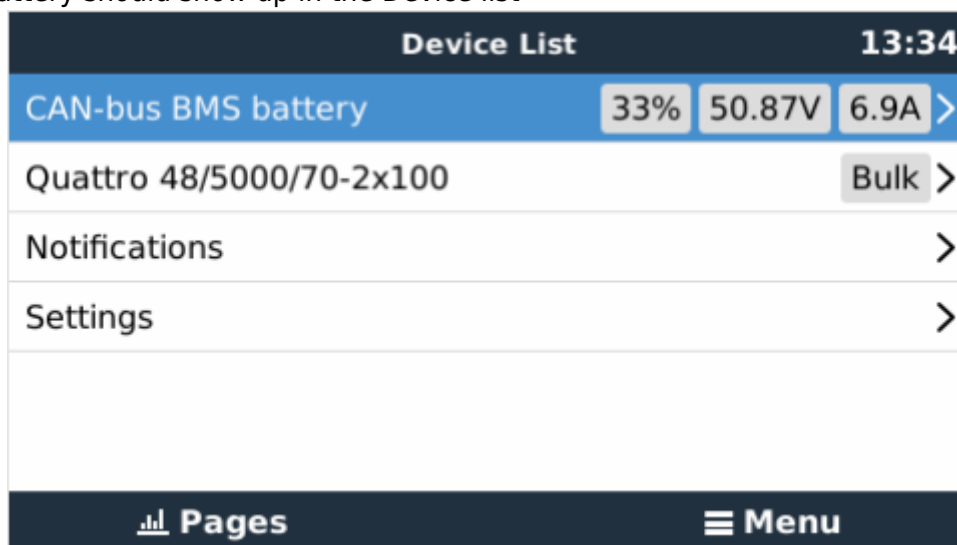
## 5.1 Connection and basic communication

Connect the CAN-bus between the GX device and the battery:

1. Connect the battery to the GX device, on the VE.Can port.
  - Pin 3 is GND
  - Pin 7 is CAN-H
  - Pin 8 is CAN-L
2. In the Menu, go to Settings → Services → CAN-bus → CAN-bus profile, and depending on the CAN baudrate of your BMS select either:
  1. VE.Can & CAN-bus BMS (250 kbit/s)
  2. CAN-bus BMS (500 kbit/s)



3. Next, the battery should show up in the Device list



In case it does not, here are some tips that help identify the problem:

First of all, login to the GX device with ssh. Instructions [here](#)

Then use `candump` to see the data being communicated on the port. Note than on a Venus GX you should replace `can0` with `can1`.

This is what typical CAN communications look like:

```

root@ccgx:~# candump can0
interface = can0, family = 29, type = 3, proto = 1
<0x354> [8] 04 c0 00 1f 03 00 00 00
<0x35a> [8] 00 00 00 00 00 00 00 00
<0x351> [8] 45 02 4c 04 4c 04 00 00
<0x355> [8] 59 00 64 00 00 00 00 00
<0x356> [8] 31 16 ce 00 3e 01 00 00
<0x305> [8] 00 00 00 00 00 00 00 00
<0x354> [8] 04 c0 00 1f 03 00 00 00
<0x35a> [8] 00 00 00 00 00 00 00 00
<0x351> [8] 45 02 4c 04 4c 04 00 00
<0x355> [8] 59 00 64 00 00 00 00 00
<0x356> [8] 31 16 d1 00 3e 01 00 00
<0x305> [8] 00 00 00 00 00 00 00 00
<0x354> [8] 04 c0 00 1f 03 00 00 00
<0x35a> [8] 00 00 00 00 00 00 00 00
<0x351> [8] 45 02 4c 04 4c 04 00 00
<0x355> [8] 59 00 64 00 00 00 00 00
<0x356> [8] 31 16 d0 00 3e 01 00 00
<0x305> [8] 00 00 00 00 00 00 00 00
<0x354> [8] 04 c0 00 1f 03 00 00 00
<0x35a> [8] 00 00 00 00 00 00 00 00

```

The message <0x35a> is the one used by the driver to recognize that there is battery connected.

In case you see 29 bit identifiers, change the code in the BMS to use 11 bit identifiers instead. 29 bit identifiers look like this in candump:

```

<0x00000356> [8] 14 79 01 62 00 f0 02 40
<0x1510a8a0> [8] 00 00 00 00 10 05 07 01
<0x161088b9> [8] 03 01 00 00 00 00 00 04
<0x305> [8] 00 00 00 00 00 00 00 00      <--- sent by the GX device,
that's why this one is 11-bits while the others are not :)

```

You can also look at the output of the driver. But that doesn't say much:

```

root@ccgx:~# cat /log/lg-resu-interface/current | tai64nlocal
2016-06-29 02:06:27.860565500 *** CCGX booted (1) ***
2016-06-30 06:57:03.917694500 *** CCGX booted (1) ***

```

## 5.2 Parameter transmission

Go into the menu, and check that the four dynamic limits are received correctly. In this example, the battery low voltage is not available:

Parameters		10:30
Charge Voltage Limit (CVL)		56.5V
Charge Current Limit (CCL)		0.0A
Discharge Current Limit (DCL)		51.0A
Low Voltage Disconnect (always ignored)		--
Pages		Menu

### 5.3 Alarms

Generate the alarms on the BMS, and double check that they appear on the GX device.

### 5.4 Detailed battery diagnostics information

If your BMS supports the optional information CAN IDs 0x374 to 0x377, go into the *Details* menu and check that this information is correctly reflected. On each line it should show the module id containing the affected cell, along with the temperature- or voltage information. It should also show if any modules are offline and if any are blocking charge or discharge.

In the example below, the battery is blocking discharge because there is a low cell.

		06:47
Lowest cell voltage	1	2.680V
Highest cell voltage	1	3.080V
Minimum cell temperature	1	24°C
Maximum cell temperature	1	27°C
Battery modules	1 online	0 offline
Nr. of modules blocking charge / discharge	0	1
Pages		Menu

If your BMS does not support this information, this menu will show two dashes.

<	16:21
Lowest cell voltage	-- --
Highest cell voltage	-- --
Minimum cell temperature	-- --
Maximum cell temperature	-- --
Battery modules	-- --
Nr. of modules blocking charge / discharge	-- --
Pages	Menu

## 5.5 Enable DVCC and disable SVS

Now with the basic CAN-bus communication and exchange of parameters between the Battery and the GX Device working, its time to test the system.

Before doing that, enable DVCC and disable the SVS setting, in the Settings → DVCC menu.

Once completed, the menu should look like this:

<	DVCC	16:49
<b>CAUTION:</b> Read the manual before adjusting		
DVCC		<input checked="" type="checkbox"/>
Limit charge current		<input type="checkbox"/>
Limit managed battery charge voltage		<input type="checkbox"/>
SVS - Shared voltage sense		<input type="checkbox"/>
STS - Shared temperature sense		<input type="checkbox"/>
SCS - Shared current sense		<input type="checkbox"/>
Pages		Menu

For more information about the DVCC menu, read [the user manual of the Cerbo GX - DVCC chapter](#).

In short, once DVCC is enabled, the GX Device will control the connected inverter/chargers, solar chargers as well as (indirectly) any connected PV Inverters based on the CCL, DCL and CVL parameters as received from the battery.

## 5.6 Recovery from blackout

Test recovery from a black-out due to a battery empty situation. Obviously important in off-grid solar systems, that they can self-restart in the morning in case they had to shut down overnight.

The same applies for an occasional overvoltage or overcurrent event.

## 5.7 CVL & CCL implementation

See chapter 1.5.

## 6. Information visible to end-customer

### 6.1 Information on the UI and also via Remote Console






**Picture 1:** Main system page which is normally shown. Shown components adapt to the installed system.



**Picture 2:** Device-List. This page lists all components in the system and shows summary data.



**Picture 3:** Main BMS page. This page shows all basic information as well as provides entry to a few submenus.

 BSLBATT battery		 20:11	
Battery	52.92V	-18.2A	-963W
State of charge	82%		
State of health	100%		
Battery temperature	27°C		
Details	>		
Alarms	>		
History	>		
Device	>		
Parameters	>		
Redetect Battery	Press to redetect		
 Pages			 Menu

**Picture 4:** The details submenu. Shows information mostly relevant for Diagnostics. The 1 : : 9 and similar numbers are the cell id strings, see PGN 0x374 definition above, free format.



< 20:14		
Lowest cell voltage	1 ::9	3.303V
Highest cell voltage	1 ::2	3.310V
Minimum cell temperature	3 ::3	25°C
Maximum cell temperature	2 ::1	28°C
Battery modules	3 online	0 offline
Nr. of modules blocking charge / discharge	0	0
Installed / Available capacity	300Ah	--
Pages ^ Menu		

**Picture 5:** Alarm submenu. To have this best working, use the 00 = not supported feature and let us know your BMS has implemented that properly so our UI can take the advantage of that.

Alarms		20:16
Low battery voltage	Ok	
High battery voltage	Ok	
High charge current	Ok	
High discharge current	Ok	
Low temperature	Ok	
High temperature	Ok	
Low charge temperature	Ok	
High charge temperature	Ok	
Internal failure	Ok	
Pages		Menu

**Picture 6:** History submenu. This menu is empty, and should actually be hidden.

(picture is missing).

**Picture 7:** Device submenu.

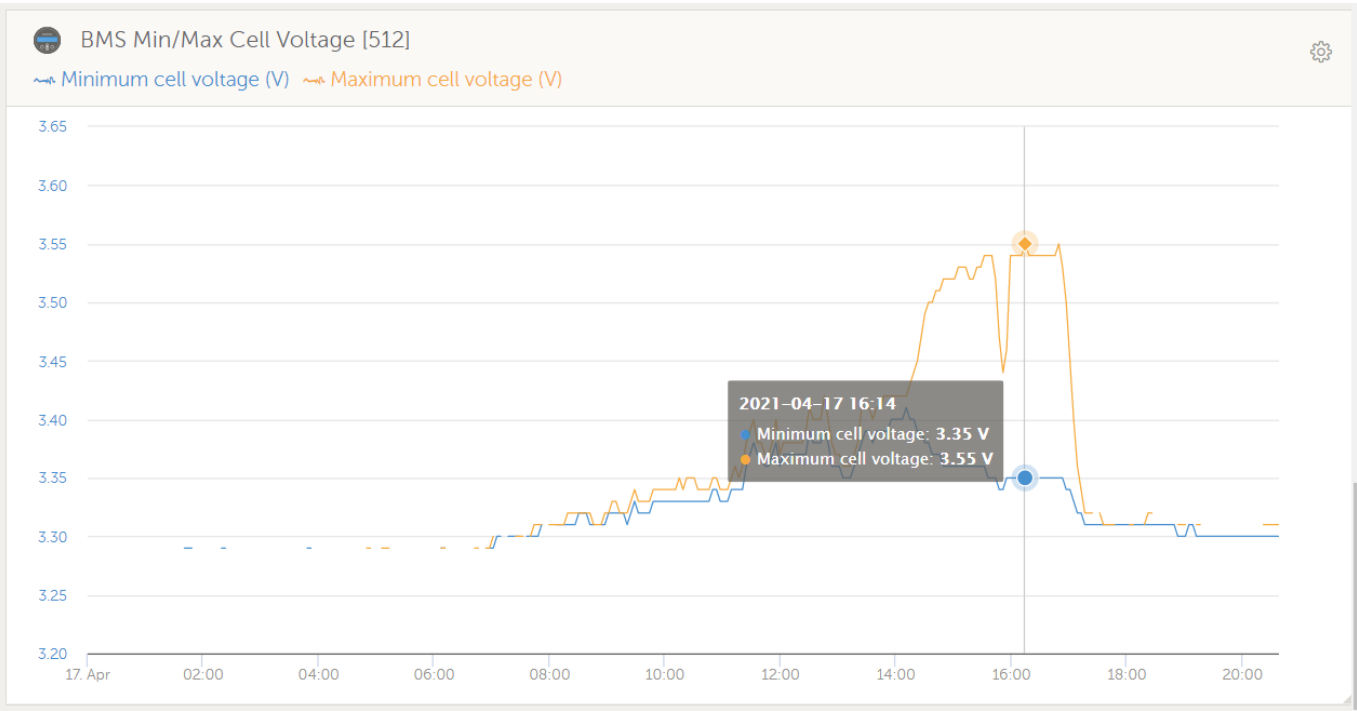


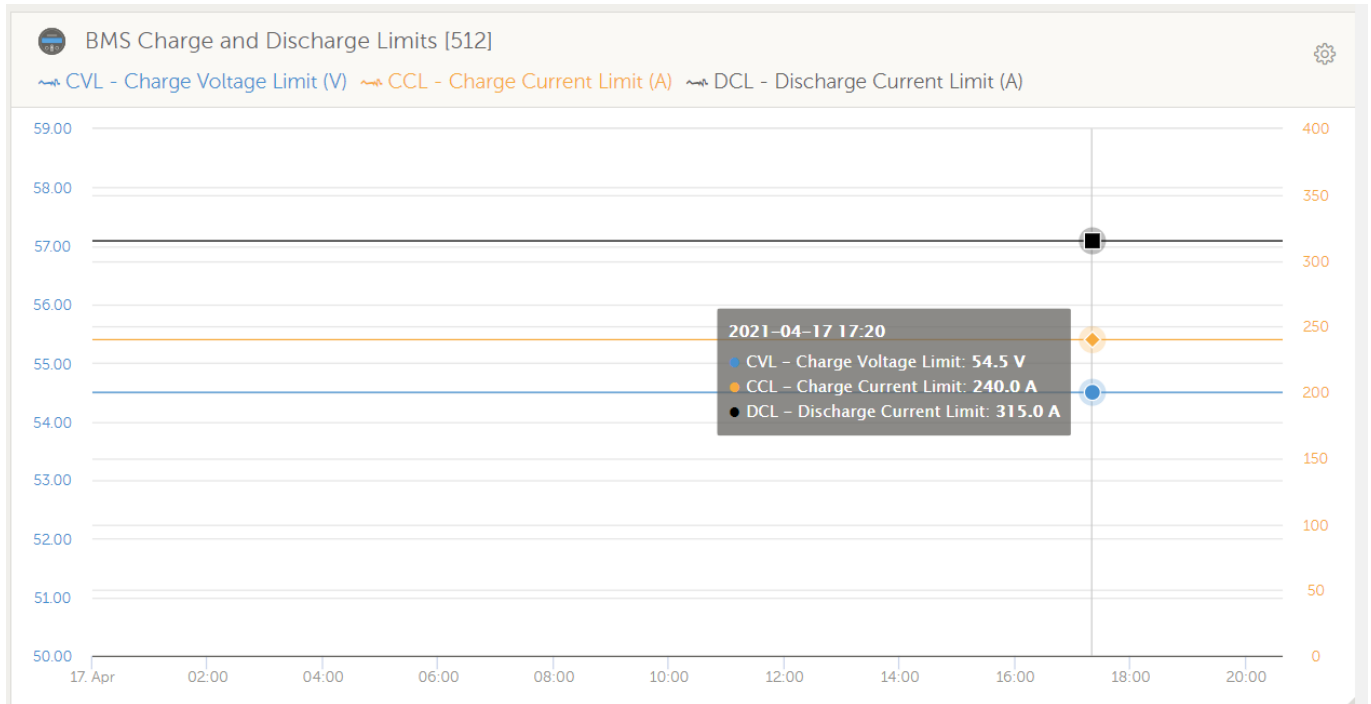
**Picture 8:** The parameters submenu. Shows actual CVL, CCL and DCL. Note that the Lower voltage limit is currently not shown, since the Victron system doesn't use it.

Parameters		10:30
Charge Voltage Limit (CVL)		56.5V
Charge Current Limit (CCL)		0.0A
Discharge Current Limit (DCL)		51.0A
Low Voltage Disconnect (always ignored)		--
Pages		Menu

6.2 Information on VRM Portal

All below screenshots are from our VRM Portal, <https://vrn.victronenergy.com>.





## 7. Changes

2021-04-17

- Added chapter 6, screenshots

2021-04-09

- Further changes to chapter 1.5, specifically the sections about testing.

2021-03-26

- Document more clearly how available/online capacity of battery is communicated in 0x35F.
- Document some pitfalls for 0x370 and 0x371, 0x380 and 0x381.

2021-03-19

- Improved introduction chapters but moreso the readability of chapter 1.5

2021-03-02

- Inserted chapter 5.5 Enable DVCC and disable SVS

2021-03-01

- Removed chapter 1.6 more information. It had a link to a PDF with details on the canbus messages, which is a duplicate of information in this document; and a link to GX Firmware update instructions. Which are nowadays simple where one would expect them: in the manual of each GX product.

2021-01-07

- Renamed What to test to chapter "1.5 CVL & CCL implementation".

- Extended the test chapter with blackout as well as CVL & CCL implementation.

2021-01-05

- Added first version of “What to test” section to chapter 1.

2020-03-19

- Document menus for battery details on the GX-device
- Minor improvements to language

2020-03-04

- Add new field (0x382)

2019-11-01

- Add new fields (0x372 to 0x381)

2019-10-04

- Improve wording around 0x35A alarms and 00/01/10

2019-09-19

- Add new fields

2019-08-19

- Update warnings and alarms documentation
  - complete the list in the large table; including ones known in the industry but not implemented by us.
  - explain that we prefer 01 and 10 for sending warnings and alarms

2019-08-06

- Update the document to reflect the present situation:
  - The Multi will switch off if there is no grid and the BMS requests a zero discharge current
  - We support more alarms now.
  - The way you enable the CAN-bms driver has changed.

2019-05-06

- Updated contents today's operation (DVCC is available now) & updated introduction with more links.

2017-12-28

- Added warning about soon coming DVCC and SVS improvements.

2017-09-04

- Align canbus specs text with the new canbus profiles options that are in Venus since v2.07.

2017-02-13

- Improved wording in Chapter 1.
- Added 0x35F battery type and firmware version

2017-02-01

- Since Venus v2.02, VE.Direct Solar chargers will be disabled when the BMS sends max discharge current = 0 or when the BMS stops sending messages.

2016-11-21

- Removed 100% grid parallel restriction, as this was fixed in Venus v1.72. Note that it still only works icm the Hub-4 Assistant.

2016-10-24 - v1.72

- Fixed bug related to alarms and warnings coming from LG and Sony batteries. Alarms were interpreted as warnings, warnings as alarms.

2016-10-10

- Change battery to inverter keepalive message from 0x35A to 0x351. Will be part of the Venus v1.80 release
- Improve wording in Chapter 1.

2016-09-06

- In Venus version 1.70 and 1.71, all alarms would disable charge and discharge, except the low battery alarm, which would disable discharge only. In all other versions (older and newer), the alarms will not have a direct effect on charging or discharging. Instead, the GX firmware will always follow the maximum charge limit and maximum discharge current limit, even in alarm situations.

From:

<https://wiki.victronenergy.com/> - **Victron Energy**

Permanent link:

[https://wiki.victronenergy.com/render/third\\_party\\_lithium\\_to\\_victron](https://wiki.victronenergy.com/render/third_party_lithium_to_victron)

Last update: **2021/04/17 20:43**

