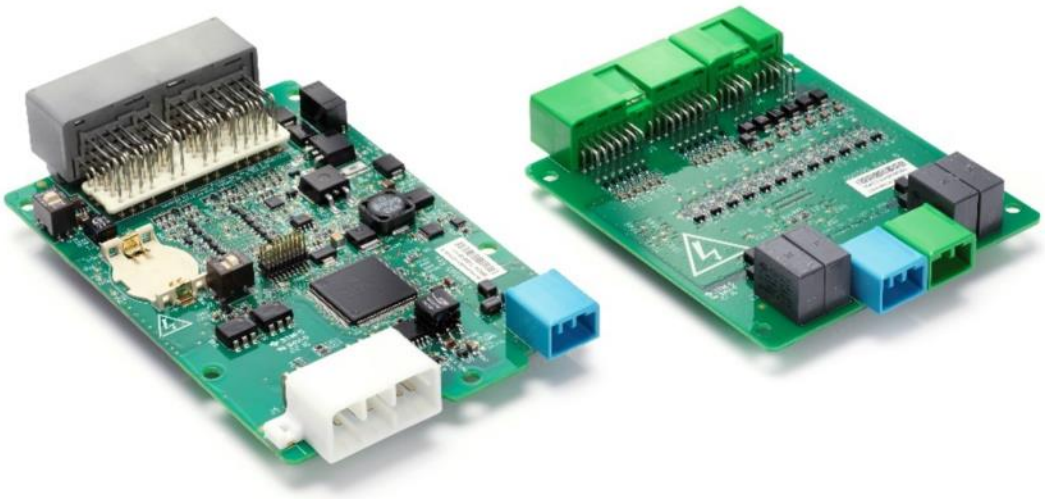
USER MANUAL
n-BMS

Network Battery Management System for automotive battery pack application



LiTHIUM BALANCE
BATTERY MANAGEMENT SYSTEMS

www.lithiumbalance.com
contact@lithiumbalance.com
Tel: +4558515104

LiTHIUM BALANCE A/S
Hassellunden 13
2765 Smørum, Denmark

Table of Contents

1 Introduction

1.1 Document purpose and structure

The purpose of this manual is to provide the user of the Lithium Balance n-BMS systems with an overview of the BMS system and to allow the user to configure the system hardware and set-up basic parameters in the software.

1.1.1 Disclaimer

We have taken every precaution to ensure that all information provided in this manual is correct and up to date. However, Lithium Balance assumes no responsibility for damage to persons and property arising as a result of following recommendations and/or procedures described in this manual. Furthermore, Lithium Balance assumes no responsibility for any infringements of rights of third parties which may result from the use of this manual.

1.1.2 General safety warning

Please, carefully read all sections describing safety issues and precautions before setting up, operating, or performing service work on any BMS controlled system! Failure to do so could result in reduced system performance, damage to the system, personal injury or even casualties.

1.1.3 Copyright and patent information

Patents owned or applied for by Lithium Balance are listed below.

Area	Number	Status	Area
ZT/CN	200780048774	Granted	Master/Slave
US/US	8,350,529	Granted	Master/Slave
EPO/EU	07817888.6	Application	Master/Slave
US/US	13/735,946	Application	Passive Balancing
PCT/WO	IB2015/055441	Application	EIS in BMS
PCT/WO	EP2016/066287	Application	EIS in BMS
PA/DK	2016 00228	Granted	Active Balancing

1.1.4 Abbreviations

n-BMS	Networked Battery Management System
BPU	Battery Protection Unit – a unit containing e.g. switches and fuses for battery protection
DoD	Depth of Discharge
BMS	Battery Management System
SoC	State-of-Charge
SoH	State-of-Health

VCU	Vehicle Control Unit
DLC	Data Length Code (of CAN Frame)
MCU	Main Control Unit (n-BMS)
CMU	Cell Monitoring Unit (n-BMS)
NTC	Negative Temperature Coefficient, thermistors

1.2 BMS overview

The Lithium Balance Battery Management Systems are multi-functional and highly configurable systems which can manage all types of Li-ion cell chemistries and packages. The system addresses several aspects of battery management such as:

- **Safety –** the BMS monitors the essential parameters (cell voltage, current and temperature) and adjusts charge and discharge current levels to ensure safe operation. If for some reason the BMS can not bring the system to operate within safe limits it will abort the operation by opening contactors/relays to the battery.
- **Performance** – the BMS monitors the battery performance, e.g. State of Charge and optimises operation, e.g. by balancing the battery to ensure that all cells can be utilized to provide the maximum energy.
- **Communication and control** – the BMS communicates with users and connected equipment and **controls** safety related system elements like switches and chargers.

Figure 1-1 below shows a block diagram of a typical n-BMS system.

1. The n-BMS printed circuit board in connected to auxiliary equipment by dedicated cables.
2. The BMS communicates with other equipment over UDS CAN. To interface computers a (PEAK) CAN adaptor is required
3. Configuration parameters **to** the BMS is configured via the Lithium Balance BMS Creator operating on a Windows PC. A correct configuration of interfaces to auxillary equipment, safety critical parameter threshold and system response in case of errors is critical to the safe operation of the system.
4. Operating parameters (e.g current and voltage) are read **from** the BMS via a standard CAN bus interface software. Here Lithium Balance recommends the use of the Busmaster program.

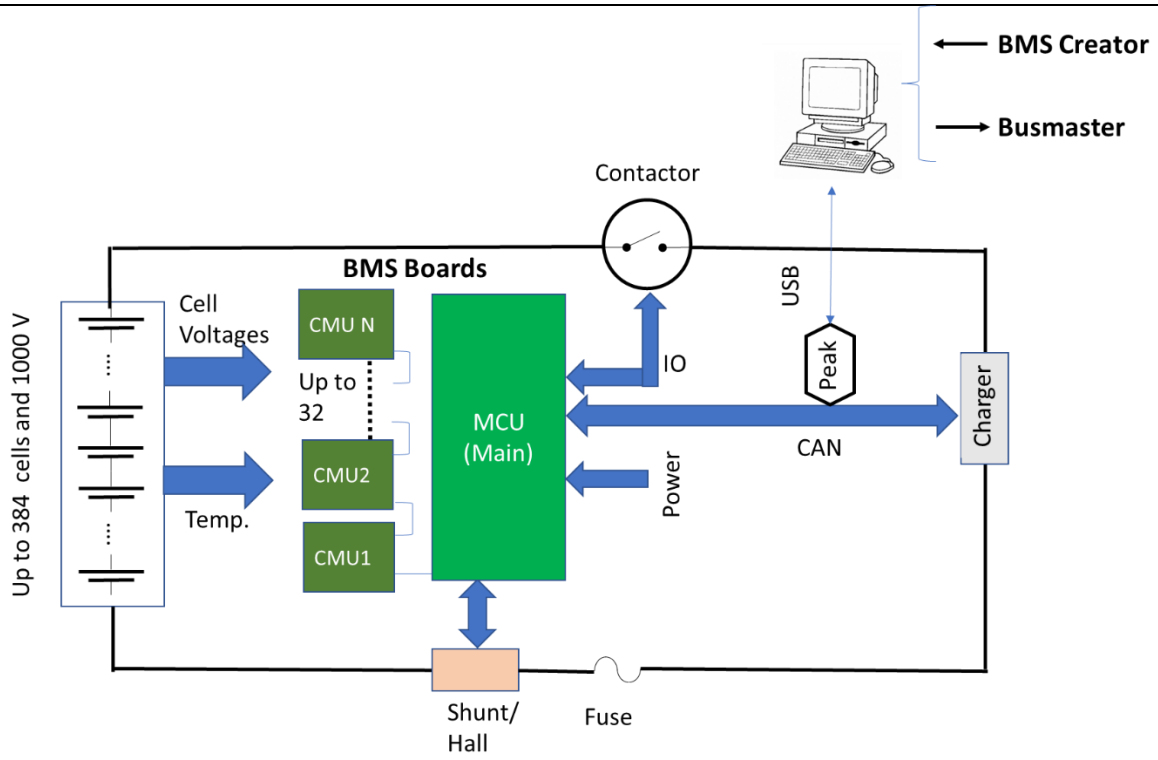


Figure 1-1 n-BMS system block diagram

1.2.1 Hardware

Physically, the n-BMS system is based on two board types, the Main Control Unit (MCU) and the Cell Monitoring Unit (CMU).

- The MCU is the main board controlling the slave boards and communicating with external devices.
- The CMUs are the cell measurement boards, which are connected to the battery cells. Each CMU can measure the voltage and temperature of up to 12 cells

Up to 32 CMU boards can be connected, serving up to 384 cells for a maximum battery voltage of 1000 V. The power to the Main (MCU) functions are provided from an external 12V power source (e.g a lead acid battery), while the power to the Cell Monitoring Units are provided directly from the main battery.

1.2.2 Firmware

The advanced functions of the BMS systems are enabled by the firmware running in the advanced microprocessor. Firmware is already present on the c-BMS boards shipped from Lithium Balance. They are configured without a configuration, which means the boards will start up in bootload mode. Firmware versions and configurations can be updated using the Lithium Balance “BMS Creator” PC tool.

1.2.3 BMS Creator

A PC configuration and interface tool called ‘BMS Creator’ is provided to configure the BMS. These functions include for example:

- configuring the BMS including setting up the safety limits for battery operation and defining associated actions if these thresholds are exceeded
- setting up CAN communication with the auxiliary equipment
- Optimising system operation, e.g. controlling charger regulations

1.3 Safety

1.3.1 Battery safety

To achieve safe operation of lithium-ion batteries, these must operate within the safe limits of:

- cell voltage
- cell/battery current
- cell/battery temperature

Exceeding these limits can trigger chemical processes that may lead to rapid degradation of battery performance. In more extreme cases, exceeding the limits can result in out-gassing from the batteries and in battery fire. In the worst cases, this may have fatal consequences.

Toxic Gas

Risk of fire

Risk of explosion

One of the main functions of any BMS is to ensure that the battery operates within the safe limits for cell voltage, temperature and current under all circumstances. To perform this safety critical function, the BMS must be correctly installed, configured and operated as described in this manual.

1.3.2 Electrical safety

Apart from the chemical related safety aspects of working with batteries and BMSs, batteries are powerful electrical devices with potential voltage levels exceeding 1000 and maximum current levels typically exceeding several hundreds of Amperes.

All personnel working with BMS systems and batteries should therefore be properly trained in handling high voltage/current installations. The legally required training is regulated by national/regional standards such as EN 50110-1 (EU) and IS 5216 (India).

As a general precaution when working with battery systems, Lithium Balance recommends that all personnel:

- wear electrically insulating gloves
- use electrically insulated tools
- disconnect the BMS from high voltage sources such as the charger and the main battery whenever possible

1.3.3 Safe design

The battery system must be adequately configured and tested in order for the BMS to ensure safe and proper operation in all operating conditions. The system configurations needed for a safe battery system may vary depending on the application. However, for all systems Lithium Balance strongly recommends that the three guidelines below are ***always*** followed:

- The safe operating cell limits (voltage, current and temperatures) of the battery cells must be configured correctly in the BMS before the first operation of the battery. The operating cell limits must be supplied directly by the cell manufacturer, e.g. taken from the **cell datasheet**.
- The system must be able to stop charging when the maximum cell voltage has been reached. This requires:
    a. robust and well-tested **communication between the BMS and the charger**
    b. a **switch** which can disconnect the charger from the battery in case the direct communication between the BMS and the charger fails.

    Such ***redundant charger control*** is required as communication failures could happen in case of e.g. inadequately mounted connectors, defective or broken communication cables or excessive electrical noise.

- It must be possible to block excessive current from the battery in order to ensure safe operation and handling of batteries. For this purpose, a fast switch or a quick blow **fuse** should be placed in the main current path. This will typically be next to the battery or inside the battery as a 'mid pack' switch or fuse. Even if a switch is used, fuses are in general recommended as they can prevent hazardous situations such as the following:
    a. The load draws an excessive current from the battery which could lead to rapid overheating of the battery cells.
    b. The battery is accidentally short-circuited.

1.3.4 Testing for safe operation

Before the BMS is put into operation, proper operation of the BMS with the battery system must be validated under supervision. Specifically, it is recommended to actively verify the correct function all safety features listed in the previous section (1.3.3) during a supervised full charging and balancing of the battery.

It is strongly recommended that the first charge cycle is always supervised to verify that charging stops when the maximum cell voltage level has been reached. The anticipated maximum duration of the first charge cycle can be calculated as the battery capacity (Ah) divided by the charger current.

1.4 Handling

This section covers handling, storage and shipping for the BMS boards only. Storage, handling and shipping of lithium batteries typically requires special precautions. Please refer to the documentation from the battery supplier for such precautions.

1.4.1 Electrostatic discharge

Electrostatic discharge (ESD) can damage or destroy sensitive electronic components (typically semiconductors) on the BMS boards. When handling the boards, it is therefore critical to comply with good ESD prevention workmanship standards like the following:

- For storage and shipping, please keep the BMS boards in the antistatic bags in which they are shipped from Lithium Balance.
- Always wear a grounded antistatic wrist-strap in contact with the skin when handling the boards outside the antistatic bags.
- Ensure that all personnel wear antistatic smocks (clothes).
- Ensure that all personnel receive education and training on ESD preventive measures in general.

A detailed guideline for safe handling of ESD sensitive devices can be provided by Lithium Balance

1.4.2 Storage

The BMS boards must be stored at temperature and humidity levels that do not exceed specified operating conditions. See data sheet for further details.

If the BMS boards are to be stored for longer periods of time, it is recommended to store them at temperatures below 25 °C and humidity levels below 70% RH.

1.4.3 Shipping

When shipping the BMS boards, it is important that the boards are:

- kept in the antistatic bags in which they are supplied
- packaged individually to prevent boards from touching physically, as this could cause damage to the boards during transportation. If possible, please use cardboard boxes with individual slots for each board, similar to those used for shipping from Lithium Balance.

Figure 1-2 Example of BMS board packaging in individual slots

1.4.4 Disposal

At the end of their service life, BMS boards must be disposed of as waste electrical and electronic equipment (WEEE). Follow appropriate guidelines in force locally (e.g. Directive 2012/19 in the EU) regarding collection and disposal.

1.5 Document structure

The manual is structured to support easy installation of BMS's in battery systems:

- Chapter 2 describes the BMS hardware and how to physically connect the BMS to the battery pack and other auxiliary units
- Chapter 3 describes the BMS functions and the key parameters used for configuring the BMS
- Chapter 4 describes the key aspects of CAN bus communication
- Chapter 5 describes the BMS Creator used for configuring the BMS
- Chapter 6 describes auxiliary components and how they interact with the BMS
- Chapter 7 summarises the main steps of the BMS installation in a Quick Installation Guide

- A number of appendices are found in Chapter 9

1.6 Key Specifications

1.6.1 Environmental conditions and tests

Standard operating conditions which will not reduce system lifetime are Temperature: 15 – 35 °C, Humidity: 25 – 75% RH, air pressure 860 to 1060 mBar.

The BMS system has been environmentally tested as listed below:

EMC Emission	CISPR 12, 16, 22, 25: EMC susceptibility per UN Addendum 9: regulation No.10 rev. 5 ISO 11452: EMC susceptibility 20 MHz– 2 GHz per UN Addendum 9: regulation No.10 rev. 5
EMC, Reversed polarity protection on 12 / 24 V.	ISO 16750-2: Electrical Loads (Reversed, protection only).
EMC, Immunity	ISO 11452-4, Immunity Bulk current injection, 60 mA from 20 MHz to 400 MHz. per UN Addendum 9: regulation No.10 rev. 5 ISO 11452-2, Absorber chamber test, 30 V/m from 200 MHz to 2 GHz. per UN Addendum 9: regulation No.10 rev. 5 ISO 7637-2: Immunity to Electrical disturbances from conduction and coupling. Pulse: 1, 2a, 2b, 3a, 3b, 4. For 12 V & 24 V. per UN Addendum 9: regulation No.10 rev. 5 IEC 61000-4-4: Immunity of ESA to electrical fast transient/burst disturbances conducted along AC and DC power lines. +- 2 kV.
Temperature/Humidity	ISO 16750-4: Climatic Loads (Code G: -40 to 85°C). Tested up to 93% RH at 55 C
Vibration	ISO 16750-3: Mechanical Loads (Sinusoidal and random vibration)

1.6.2 Balancing

Balancing scheme	Passive balancing (bleeding)
Balancing current	200 mA @ 4.2 VDC max

1.6.3 Voltage sensing

Cell voltage range	0-5 V. High accuracy (see next line) for 1 – 4.5V
Cell voltage accuracy	<±1.5 mV over the entire temp range and from 1 – 4.5V
Cell voltage sampling frequency	10 Hz typical, down to 100 Hz is possible

1.6.4 Current sensing

Measurement schemes	1 Shunt and 2-channel Hall sensor

Current measurement by shunt[1]	±450 mV Resolution is 16 bit so 0.005 mV / bit. If e.g 150 mV / 300 A (2 A/mV) shunt is used the current resolution is 0.01 A
Shunt measurement accuracy	±0.5 mV
Hall measurement range	0 – 2.5 V Current in (Charging) 2.5 – 5V Current Out (Discharging)
Hall measurement accuracy	±1.5 mV

1.6.5 BMS Creator operating system

Supported operating systems	Windows 10, 8, 7 or Vista, please see details in section 5.1

1.7 n-BMS specifications

1.7.1 Dimensions

MCU Dimensions	166.1 x 92.0 x 24.2 mm
MCU Weight	App. 80 g
CMU Dimensions	104.0 x 92.0 x 18.0 mm
CMU Weight	App. 80 g

3D models of the boards are available upon request.

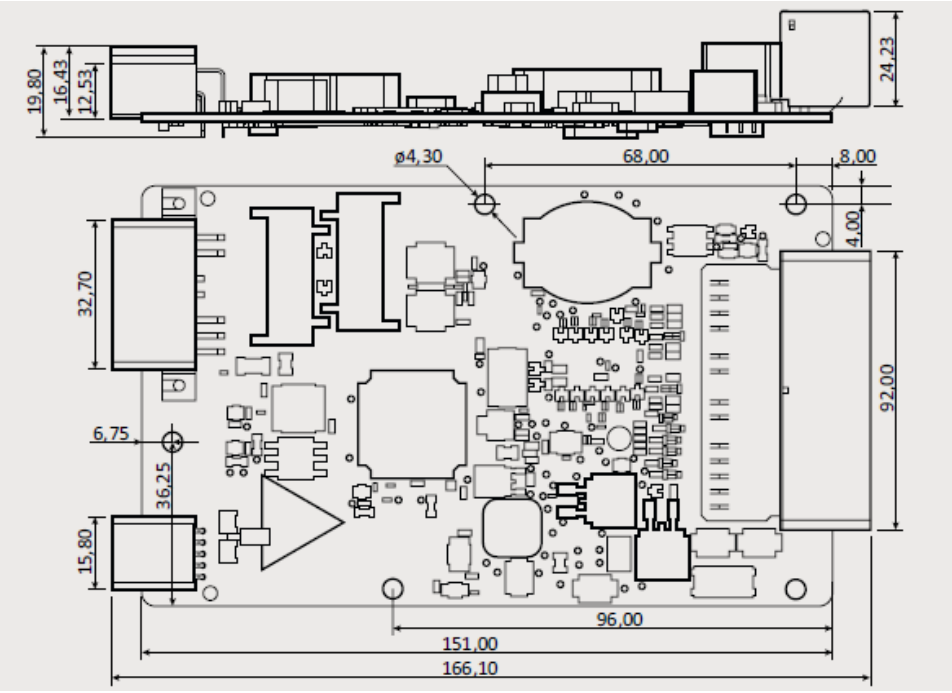


Figure 1-3 MCU dimensions in mm

[1] 100-1000 µΩ shunt

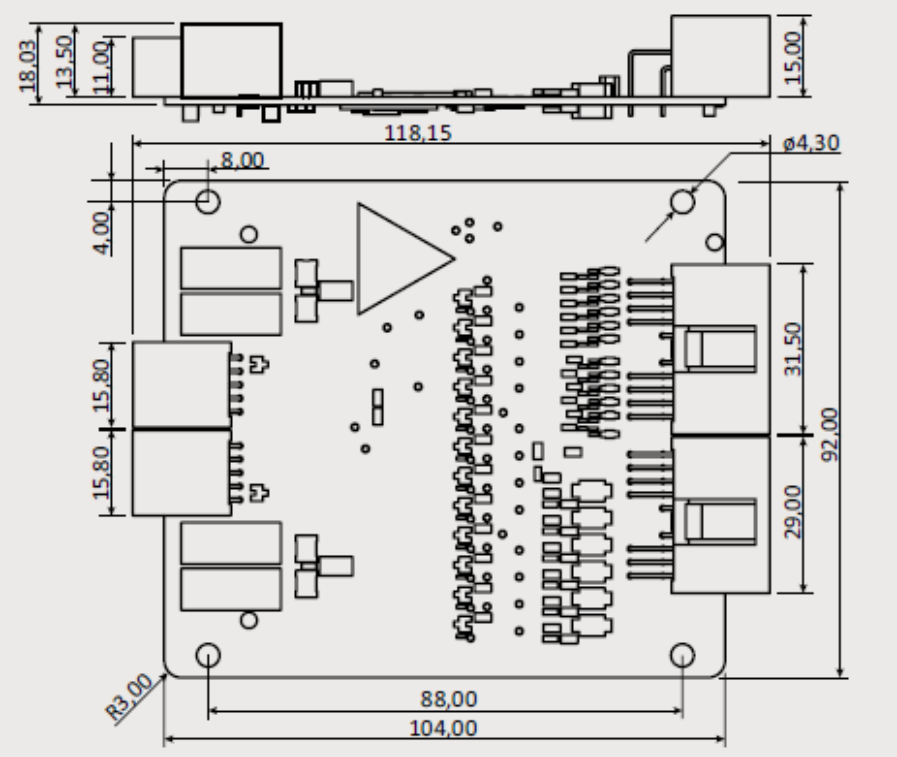


Figure 1-4 CMU dimensions in mm

1.7.2 Battery limits

Parameter	Interval
Number of Cells per CMU	Up to 12
Minimum number of cells	The CMU requires at least 11 V from the main battery to power the cell monitoring part. For most chemistries 4-5 cells are sufficient
Number of CMU's supported	1 - 32
Max battery capacity	2000 Ah[2]
Max battery voltage	1000 V

1.7.3 High Voltage Measurements

Parameter	Interval
High voltage measurement	Up to 1000V ±1 V

1.7.4 Voltage isolation

High Voltage isolation	Up to 1000 V according to ISO 6469-3-2011

1.7.5 Power supply and consumption

Supply voltage[3]	12 VDC (6-35 VDC)

[2] Software setting. Can be changed if required
[3] Up to 150 mA required

MCU Power consumption from supply voltage (12V) – Active mode	<3.2 W @ 12V
MCU Power consumption from Main battery – Active mode	<0.6 W for 12 cells. Measured with 56V across BCU
MCU Power consumption from supply voltage (12V) – Sleep mode	<20 mW @ 12V
MCU Power consumption from Main battery – Sleep mode	<0.3 mW @ 56V

1.7.6 Temperature sensing

No of temp sensors	CMU: 2 on-board + 12 inputs MCU: 11 inputs
Temperature sensor type	NTC, 10 kΩ @ 25 °C, β = 3900[4]
Temperature Range	-40 – 85 °C
Temperature measurement tolerance	BMS tolerance < ±1.0 °C, for the entire temp range. To this the temperature tolerance of the NTC has to be added.

1.7.7 Communications & control

CAN	2 x CAN (sCAN and iCAN), bus 2.0 A/B 11 bit and 29 bit ID. sCAN: Isolated from battery pack, referenced to BMS Supply voltage GND. iCAN: Isolated from battery pack and from BMS Supply voltage GND. Protocol UDSonCAN according to ISO 14229-1 125, 250, 500, 1000 kBit/sec
For BMS Creator	UDSonCAN as described above (sCAN)
GPIO Configurable input/output ports	16

[4] Sensor characteristics are user-configurable for improved sensor support.

2 Printed Circuit Boards

This chapter explains the basic functions of the two n-BMS printed circuit boards and how to properly connect them to the other components of the battery pack.

Please note that n-BMS can operate at high voltages and care must be taken when installing or performing maintenance tasks on the BMS and the battery system.

2.1 Mechanical Overview

A mechanical overview of the n-BMS boards is shown below indicating connectors. LEDs and labels. These are described in more details below

2.1.1 Connectors

The connectors of the MCU and CMU boards are indicated in Figure 2-1, showing that each board contain four connectors for customer interfaces. Furthermore, a JTAG connector is available on the MCU. This is used for flashing software during production and will in general not be used by the customer.

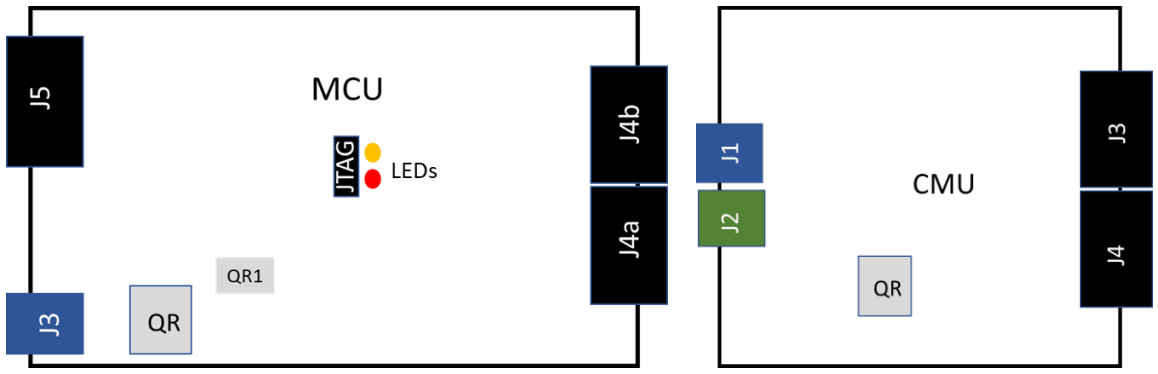


Figure 2-1 Connectors, labels and LEDs on the MCU and CMU boards

2.1.2 LEDs

The MCU board also contain light emitting diodes (LED).

1. The red LED is indicating the hardware state of the board:
   - A SINGLE 'BLINK* indicates that power has been connected to the board
   - A SOLID RED LIGHT indicates that hardware fault has occurred.
2. The orange LED is indicating operating mode of the BMS,
   - A FLASHING ORANGE LIGHT means it is in bootload mode and ready to be programmed with either a new configuration or a new application SW
   - A SOLID ORANGE LIGHT indicates that the application could not be started and the BMS is not operating.

2.1.3 Labels

The QR labels (market QR on Figure 2-1) contain information about:

- Part number and revision version of the board
- Serial number including the production date

To read the QR-labels, Lithium Balance can recommend to use dedicated QR readers. If smart phones are used the “QR Reader from TapMedia Ltd” have been tested successfully.

2.2 Connections and cables

To simplify installation, Lithium Balance provides cables which can be ordered together with the BMS boards.

Connector	**Cable assembly Order code**	**Cable dimensions**
MCU J4a IO cable	000807	100 cm, 20AWG = 0,5 mm²
MCU J4b IO cable	000808	100 cm, 20AWG = 0,5 mm²
MCU J5 Shunt & HV	000847	50 cm, 20 AWG = 0.5 mm²
ISOSPI Connection MCU J3 CMU J1, J2	000802, 803, 804, 805	Shielded cable: 12, 30, 100, 500 cm
CMU J4 Cell Voltage	000809	150 cm, 20AWG = 0,5 mm²
CMU J3 Cell Temp	000801	100 cm, 22AWG = 0,35 mm²

Table 2.1 Overview of n-BMS connectors and matching cables

2.2.1 MCU J4a Connector IO Connector1 – 35 pin

The J4a connector provides interfaces for:

- Board Power and Ground
- Digital Input/Output signals
- Ignition (Wake-up)
- Hall sensor signals
- HVIL signal
- General/Aux temperature sensors

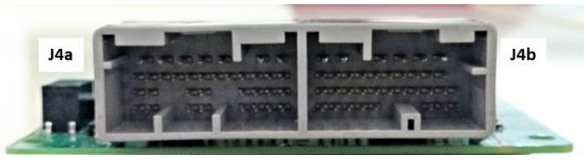


Figure 2-2 MCU J4a and J4b IO Connectors

Please note that the MCU J4a and J4b connectors are keyed differently, thereby assuring that only the correct cables fit into J4a (000807) and J4b(000808)

The J4a connector PIN numbers and associated 000807 cable wire numbers are summarised in Table 2.2

J4a Pin No.:	**PIN Name**	**000807 wire number**	**IN/OUT**	**Description**
1	GPIO10	1	In/Out	Configurable IO signal
2	GPIO9	2	In/Out	Configurable IO signal
3	HVIL OUT	3	Out	HV INTERLOCK OUT – For future SW-releases
4	HVIL IN	4	In	HV INTERLOCK IN – For future SW-releases
5	GND Power/In	5	In	Ground for board
6	GND Power/in	6	In	Ground for board

7	6V-35V Power	7	In	(12 V) Power Supply input
8	No Connection			
9	No Connection			
10	No Connection			
11	GPIO1	8	In/Out	General Purpose IO
12	IGNITION	9		Wake-up from Sleep mode
13	T-AUX-GND-4	10	In	Aux Temp Ch. 4 GND
14	T-AUX-4	11	In	Aux Temp Ch. 4
15	T-AUX-GND-3	12	In	Aux Temp Ch. 3 GND
16	T-AUX-3	13	In	Aux Temp Ch. 3
17	T-AUX-GND-1	14	In	Aux Temp Ch. 1 GND
18	T-AUX-1	15	In	Aux Temp Ch. 1
19	No Connection			
20	CSENSEHI	16	In	Hall effect sensor High input
21	CSENSEHI GND	17	Out	Hall effect sensor High Ground
22	CSENSEHI 5V	18	Out	Hall effect sensor High 5V
23	GPIO2	19	In/Out	General Purpose IO
24	No Connection			
25	No Connection			
26	T-AUX-2	20	In	Aux Temp Ch. 2 GND
27	T-AUX-GND-2	21	In	Aux Temp Ch. 2
28	CSENSELO	22	In	Hall effect sensor Low input
29	CSENSELO GND	23	Out	Hall effect sensor Low Ground
30	CSENSELO-5V	24	Out	Hall effect sensor Low 5V
31	No Connection			
32	No Connection			
33	No Connection			
34	No Connection			
35	No Connection			

Table 2.2 MCU J4a pin-out and 000807 wire overview

Please note that the J4a and J4b connectors and connectors are keyed differently to avoid that the connector J4a by mistake is inserted into the connector for J4b and vica versa

2.2.2 MCU J4b IO Connector2 – 35 Pin

The J4a connector provides interfaces for:

- Digital Input/Output signals
- CAN Communication
- General/Aux temperature sensors

The J4b connector PIN numbers and associated 000808 cable wire numbers are summarised in Table 2.3

J4b Pin No.:	**PIN Name**	**000808 wire number**	**IN/OUT**	**Description**
1	GPIO16	1	In/Out	General Purpose IO
2	GPIO15	2	In/Out	General Purpose IO

3	GPIO14	3	In/Out	General Purpose IO
4	GPIO13	4	In/Out	General Purpose IO
5	GPIO12	5	In/Out	General Purpose IO
6	GPIO11	6	In/Out	General Purpose IO
7	sCAN HI	Shielded cable	In/Out	Non-ISOLATED CAN High Signal
8	sCAN LO	Shielded cable	In/Out	Non-ISOLATED CAN Low Signal
9	GPIO8	12	In/Out	General Purpose IO
10	GPIO7	11	In/Out	General Purpose IO
11	GPIO6	10	In/Out	General Purpose IO
12	GPIO5	9	In/Out	General Purpose IO
13	GPIO4	8	In/Out	General Purpose IO
14	GPIO3	7	In/Out	General Purpose IO
15	No Connection			
16	No Connection			
17	No Connection			
18	T-AUX-GND-9	22	In	Aux Temp Ch. 9 GND
19	T-AUX-9	21	In	Aux Temp Ch. 9
20	T-AUX-GND-8	20	In	Aux Temp Ch. 8 GND
21	T-AUX-8	19	In	Aux Temp Ch. 8
22	T-AUX-GND-7	18	In	Aux Temp Ch. 7 GND
23	T-AUX-7	17	In	Aux Temp Ch. 7
24	T-AUX-GND-6	16	In	Aux Temp Ch. 6 GND
25	T-AUX-6	15	In	Aux Temp Ch. 6
26	T-AUX-GND-5	14	In	Aux Temp Ch. 5 GND
27	T-AUX-5	13	In	Aux Temp Ch. 5
28	iCAN LO	Shielded cable	In/Out	ISOLATED CAN Low Signal
29	iCAN HI	Shielded cable	In/Out	ISOLATED CAN High Signal
30	iCAN GND	Shielded cable	In	ISOLATED CAN GROUND
31	No Connection			
32	T-AUX-GND-11	26	In	Aux Temp Ch. 11 GND
33	T-AUX-11	25	In	Aux Temp Ch. 11
34	T-AUX-GND-10	24	In	Aux Temp Ch. 10 GND
35	T-AUX-10	23	In	Aux Temp Ch. 10

Table 2.3 MCU J4b pin-out and 000808 wire overview

2.2.3 MCU J5 – High Voltage

The MCU J5 connector is used for connecting the BMS with the electrical voltage signals which are galvanically connected to the battery, see Figure 3-12.

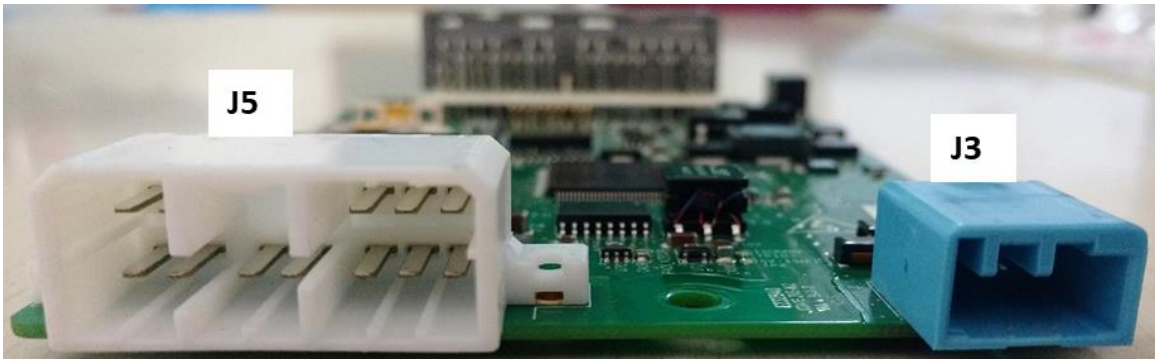


Figure 2-3 MCU J5 High voltage and J3 isoSPI Connectors

The J5 connector PIN numbers and associated 000847 cable colour coding are summarised in Table 2.4

PIN NO	**PIN Name**	**000847 Colour coding**	**IN/OUT**	**Description,** please refer to Figure 3-12
1	SH+	White	In	SHUNT POSITIVE
2	SH-	**Dark Blue**	In	SHUNT NEGATIVE
5	HV+	Red*	In	HV POSITIVE
6	HV-	**Black**	In	HV NEGATIVE
9	HV+ Load	**Brown***	In	HV POSITIVE on Load side - For future SW release
11	HV+ Charge	**Blue***	In	HV POSITIVE on Charge side - For future SW release

Table 2.4 MCU J5 and 000847 connections. Please note that the HV+ connections (Red, Brown and Blue wires) are supplied with an additional black heat-shrink. When this is fixed/heated to the wires, isolation levels up to 1000 V are provided. Otherwise, the isolation levels of the 000847 cable are only specified to 600V

2.2.4 MCU J3 ISOSPI (Up)

The MCU J3 connector provides interfaces for the isoSPI communication between the boards. Shielded cables 000802, 803, 804, 805 are available for these interfaces. J3 Pin-out is summarised below

PIN NO	**PIN Name**	**DESCRIPTION**
1	CHASSIS GROUND	GROUND
2	CHASSIS GROUND	GROUND
3	CHASSIS GROUND	GROUND
4	IM OUT	
5	IP OUT	

Table 2.5 MCU J3 pin-out overview

2.2.5 CMU J4 - Cell Voltage and Power

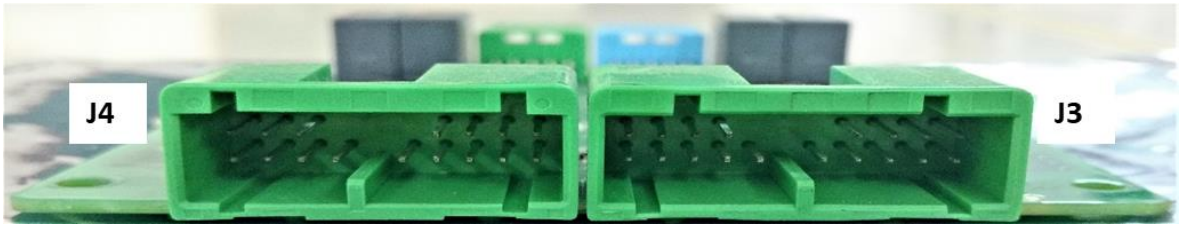


Figure 2-4 CMU J4 Cell Voltage sense wire and J3 Cell temperature probe connectors

The CMU J4 connector provides interfaces for up to 12 cell voltages and for the power needed for the cell-monitoring ASIC. The CMU J4 connector PIN numbers and associated 000809 cable labels are summarised in Table 2.6

PIN NO	PIN Name	000809 wire number	DESCRIPTION
1	CELL 12+ (CMU POWER)	CELL 12+	PACK POSITIVE
2	CELL 11+	CELL 11+	CELL 11TH POSITIVE
3	CELL 9+	CELL 9+	CELL 9TH POSITIVE
4	CELL 7+	CELL 7+	CELL 8TH POSITIVE
5	CELL 3+	CELL 3+	CELL 3RD POSITIVE
6	CELL 1+	CELL 1+	CELL 1ST POSITIVE
7	CELL 1- (CMU POWER)	CELL 1-	PACK NEGATIVE
8	CELL 12+	CELL 12+	CELL 12TH POSITIVE
9	CELL 10+	CELL 10+	CELL 10TH POSITIVE
10	CELL 8+	CELL 8+	CELL 8TH POSITIVE
11	CELL 6+	CELL 6+	CELL 6TH POSITIVE
12	CELL 5+	CELL 5+	CELL 5TH POSITIVE
13	CELL 4+	CELL 4+	CELL 4TH POSITIVE
14	CELL 2+	CELL 2+	CELL 2ND POSITIVE
15	CELL 1-	CELL 1-	CELL 1ST NEGATIVE
16			NOT USED

Table 2.6 CMU J4 pin-out and 000809 wire overview

2.2.6 CMU Temperature – CMU J3

The CMU J3 connector provides interfaces for up to 12 thermistors. The CMU J3 connector PIN numbers and associated 000801 colour codings are summarised in Table 2.7

PIN NO	PIN Name	000801 wire Colour	DESCRIPTION
1	NTC 1 SIGNAL	Orange	1st temp sensor
2	NTC 2 SIGNAL	Orange	2nd temp sensor
3	NTC 3 SIGNAL	Orange	3rd temp sensor
4	NTC 4 SIGNAL	Brown	4th temp sensor
5	NTC 9 SIGNAL	Grey	9th temp sensor
6	NTC 10 SIGNAL	Red	10th temp sensor
7	NTC 11 SIGNAL	Red	11th temp sensor
8	NTC 12 SIGNAL	Red	12th temp sensor
9	GND (NTC 1 & 2)	**Black**	1st& 2nd temp ground
10	GND (NTC 3 & 4)	**Black**	3rd& 4th temp ground
11	GND (NTC 5 & 6)	**Black**	5th& 6th temp ground

12	NTC 5 SIGNAL	Yellow	5th temp sensor
13	NTC 6 SIGNAL	Green	6th temp sensor
14	NTC 7 SIGNAL	Blue	7th temp sensor
15	NTC 8 SIGNAL	White	8th temp sensor
16	GND (NTC 7 & 8)	**Black**	7th& 8th temp ground
17	GND (NTC 9 & 10)	**Black**	9th& 10th temp ground
18	GND (NTC 11 & 12)	**Black**	11th& 12th temp ground

Table 2.7 CMU J3 pin-out and 000801 wire overview

2.2.7 CMU J1 and J2 - ISOSPI

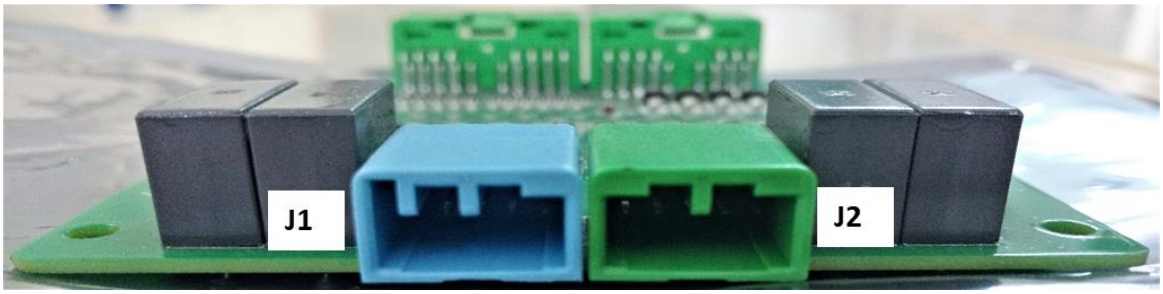


Figure 2-5 CMU J1 and J2 connectors for isoSPI communication

The CMU J1 and J2 connector provides interfaces for the isoSPI communication between the n-BMS boards. Shielded cables 000802, 803, 804, 805 are available for these interfaces please note that

- The blue connector on the cable should be plugged into the blue connector on the boards (CMU J1 and MCU J3)
- The green connector on the cable should be plugged into the green connector on the boards (CMU J2)
- The two connectors are keyed and can not be interchanged

J1 PIN NO	PIN Name	Color coding
1	CHASSIS GROUND, SHIELDING	
2	CHASSIS GROUND, SHIELDING	
3	CHASSIS GROUND, SHIELDING	
4	IMB_IN	Brown
5	IPB_IN	White

Table 2.8 CMU J1 Uplink pin-out overview

J2 PIN NO	PIN Name	DESCRIPTION
1	IPA_IN	White
2	IMA_IN	Brown
3	CHASSIS GROUND, SHIELDING	
4	CHASSIS GROUND, SHIELDING	
5	CHASSIS GROUND, SHIELDING	

Table 2.9 CMU J2 Downlink pin-out overview

2.3 MCU Features and connections

The MCU consists of two main parts, which are galvanically isolated from each other, please see Figure 2-6

- The low voltage part performing the overall control functions of the BMS
- The high voltage part interfacing the battery pack signals

The individual functions are described in more details in the following sections.

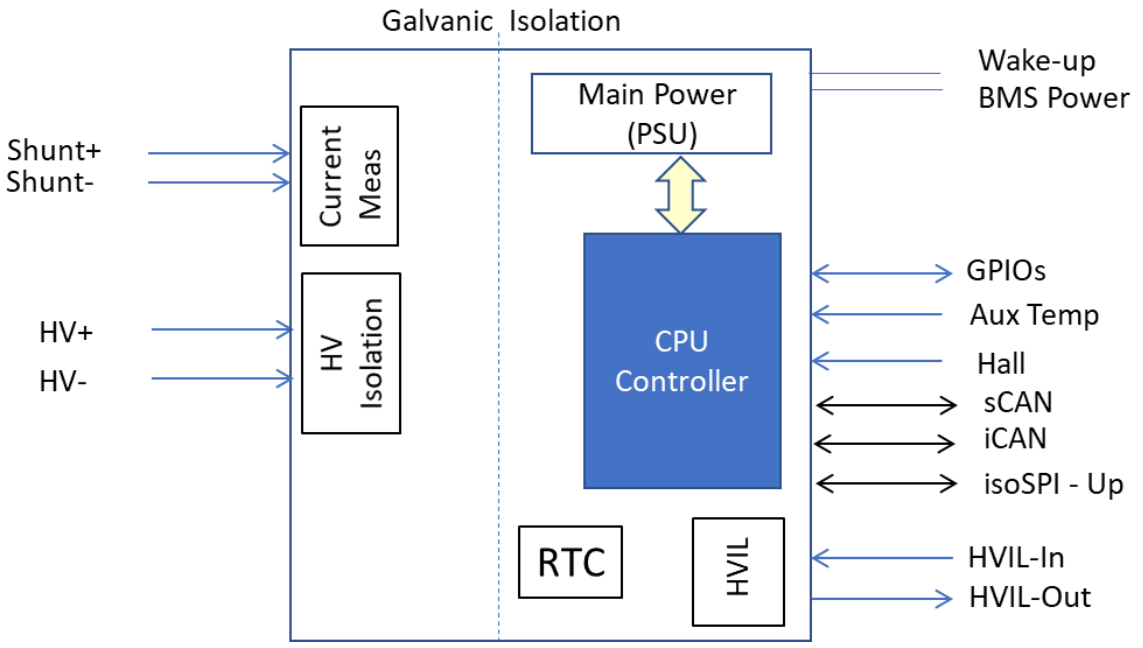


Figure 2-6 CMU block diagram. RTC indicates 'Real-Time Clock, please see section 8.9.2 . NTC indicates internal (Negative Temperature Coefficient) temperature sensors. HVIL is described briefly in section 8.9.1

2.3.1 Main Power supply (PSU)

The BMS Power (nominally 12 or 24V) and ground needs to be provided to the Main Power Supply Unit (PSU) on the MCU.

In many applications, this power is provided from a battery – typically a lead acid battery already used in the system for other purposes. If independent power supplies are not available where the BMS is used, the main power can also be provided from the main battery via a DC/DC converter.

Please note that two ground inputs are provided on the MCU J4a connector. Please connect both to ground as they are both used to sink the relative high in-rush current flow into the BMS, which can be experienced when contactors are switched.

The PSU includes an ignition/wake-up pin which can be used to bring the BMS out of sleep mode, when a voltage >3 V is applied.

2.3.2 GPIO (General Purpose Input/Output) connections

The BMS supports up to 16 digital GPIOs for example for control of relays.

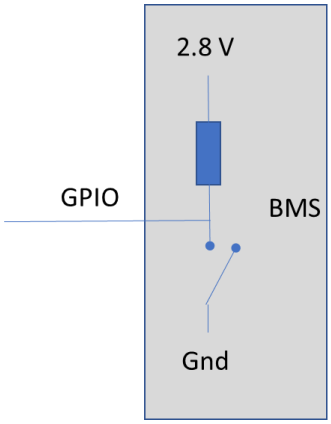


Figure 2-7 BMS IO operating as an actively low contactor driver. In the OFF state, the IO is connected to the V Supply, in the ON state it is connected to ground

These I/Os are low side switched (active low), meaning that they short to the BMS ground (0 V) when active and are pulled up to app 2.8V when not active as indicated on Figure 2-7 .

- In output mode, the I/Os are either in a high impedance mode (off mode) or directly to ground (on mode).
- In input mode, the I/Os will detect a low (active) signal for voltages < 0,5V and a high (not active) signal for voltages between 1 and 30V.

In terms of maximum ratings, the I/Os support:

- Peak currents of up to 4,5A/1sec
- Continuing current up to 1A per I/O
- Operation specified up to 30V with load dump protection up to 52V
- Absolute minimum input level: -0,3V

2.3.3 Shunt

The battery current can be measured over a 'Shunt resistor', which can be monitored by the BMS by connecting the Shunt+ and Shunt- signal, as indicated on Figure 2-8, to the BMS (J2 pin 11 and 1). Please note that:

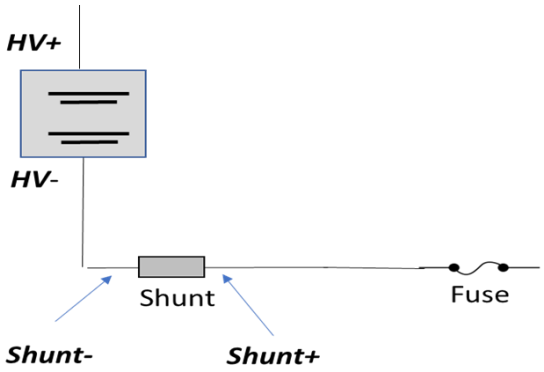


Figure 2-8 Shunt and High voltage signal definitions

The shunt circuits are internally referenced to the low voltage side of the main battery, the HV- level in Figure 2-8. Consequently, the shunt MUST always be placed on the negative side of the battery pack as close as possible to battery.

The 'shunt-' signal is defined as the voltage connected to the 'most negative terminal' of the battery pack.

2.3.4 Hall Sensor

The BMS system current can also be measured by a Hall effect sensor, where:

- Hall sensor voltages > 2.5V correspond to current In = charging
- Hall sensor voltages < 2.5V corresponds to current out = discharging
- Half of the Hall sensor supply voltage = 5 V / 2 =2.5 V corresponds to no current. Middle range will not be always 2.5 V due to small deviations on the supply voltage of the Hall sensor, however BMS reads the input voltage and adjust current readings accordingly. So, Middle range voltage will always refer to nu current, i.e 0 A.

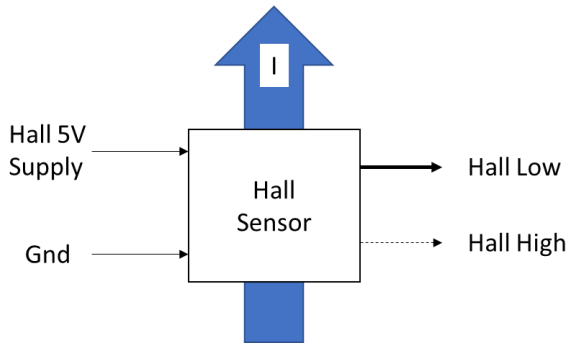


Figure 2-9 Hall sensor input and output signals

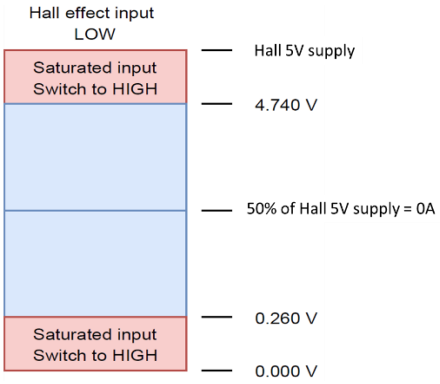


Figure 2-10 Hall sensor voltage levels

The BMS system provides a 5V Supply- and Gnd signal to the sensor as shown on Figure 2-9.

The BMS system can utilise both dual- and single channel hall sensors.

- If a dual-channel Hall sensor is used, the BMS will always use the 'low' current input until this is saturated (the limits are user-configurable). Once the 'low' current channel is saturated, the BMS will then switch to use the 'high' current input, please see Figure 2-10. As a dual channel Hall sensor, the system has been tested with the DHAB Hall effect current sensor family from LEM.
- If a single channel Hall sensor is used, the 'Hall high' input (J2 Pin 3) on the BMS must be connected to the HALL_5V pin (J2 Pin 13). This ensures that if the LOW channel is saturated in either positive or negative (please see Figure 2-10), the BMS will register a maximum input current with a positive sign.

Once a Hall sensor has been installed and configured via the BMS creator (see section 3.1.1) it is strongly recommended:

- To verify that the Hall sensor has been turned the right way, i.e. that a positive current is measured when the system is charging.
- To set the Hall sensor offset.

2.3.5 Temperature monitoring

11 external temperature sensor inputs including ground connections are available on the MCU. One end of each thermistor (e.g. thermistor #3) must be connected to a temperature input (T3) and the other end to the corresponding Gnd (Gnd3). Please note, that it is not important which end of the thermistor that is connected to the Gnd pin.

The temperature sensor detection system is designed for 10 kΩ NTC sensors with a default b-value of 3900. However, for increased support, the resistance-to-temperature characteristics are user-configurable through a set of parameters.

These temperature inputs are typically used for monitoring the temperature of system level components such as for example monitoring the shunt temperature.

2.3.6 CAN bus connection

For the n-BMS two CAN busses are available

- The sCAN bus is referenced to BMS GND

- The iCAN bus is galvanically isolated from the rest of the BMS

Both CAN bus connections include 120 Ω termination resistances on the BMS board. The user should assure correct termination in the far end as indicated on Figure 2-11, below.

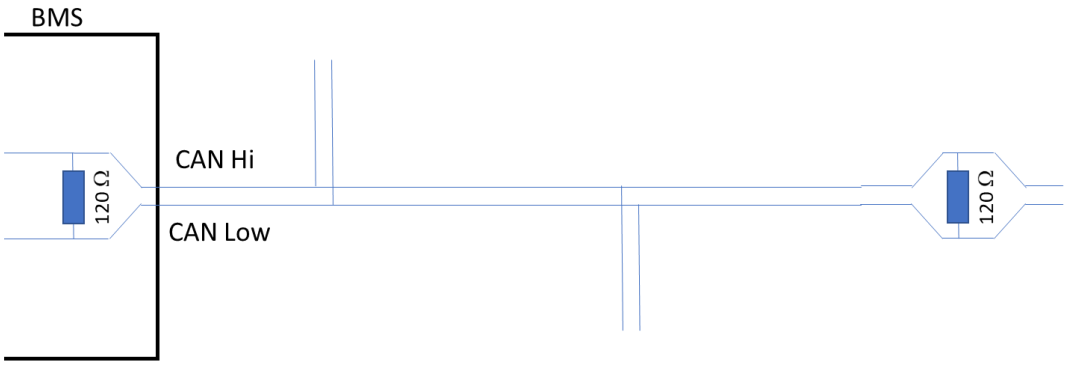


Figure 2-11 Sketch of CAN connection for more details please refer to ISO 11898

For communication between the BMS and the BMS Creator a CAN adapter is necessary.

Lithium Balance can only guarantee correct operation if the PCAN-USB adapter from Peak systems is used.

Figure 2-12 PCAN-USB adapter from Peak systems

2.4 CMU Features and connections

The CMU board is used to:

- Monitor the cell voltages and temperatures of up to 12 cells
- Bleed charges of individual cells
- Communicate with the other boards over the isoSPI databus

A block diagram of the CMU is shown in Figure 2-13.

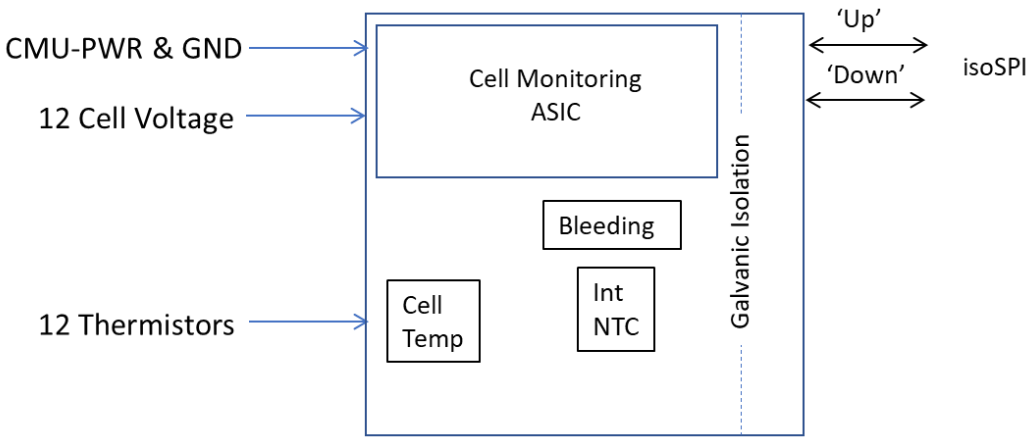


Figure 2-13 CMU block diagram

2.4.1 Cell voltage monitoring and connection

A central component on the CMU board is the 'Cell Monitoring ASIC', which measures the individual cell voltages and provides the power to the board from the main battery via the voltage sense wires. For correct function of this ASIC:

- Cells must be connected in the correct order, i.e. the lowest voltage should be connected to Cell 1- and then voltages should be stepped up the Cell 1+, Cell 2+ etc
- A minimum of 11V is required to power the CMU/ASIC. When the 00809 cable is used to connect to the cell voltages to the CMU, the highest (Cell 12+) and lowest (Cell 1-) cell voltages will automatically be connected to the power and gnd inputs of the monitoring ASIC.
- Unused cell voltage inputs should be connected to the cell with the highest voltage

This is exemplified below for a case with 9 cells, please see Figure 2-14

Start connecting cells 'from the voltage bottom', i.e. from 'Cell 1-'. Then move 'upwards' in terms of both voltage levels and cell numbers

Make sure to short unused wires on the 000809 cable to the cell with the highest voltage.

For a 9 cells battery, the 000809 wires labelled Cell 12+, Cell 11+ and Cell 10+ should all be connected to the battery Cell 9+

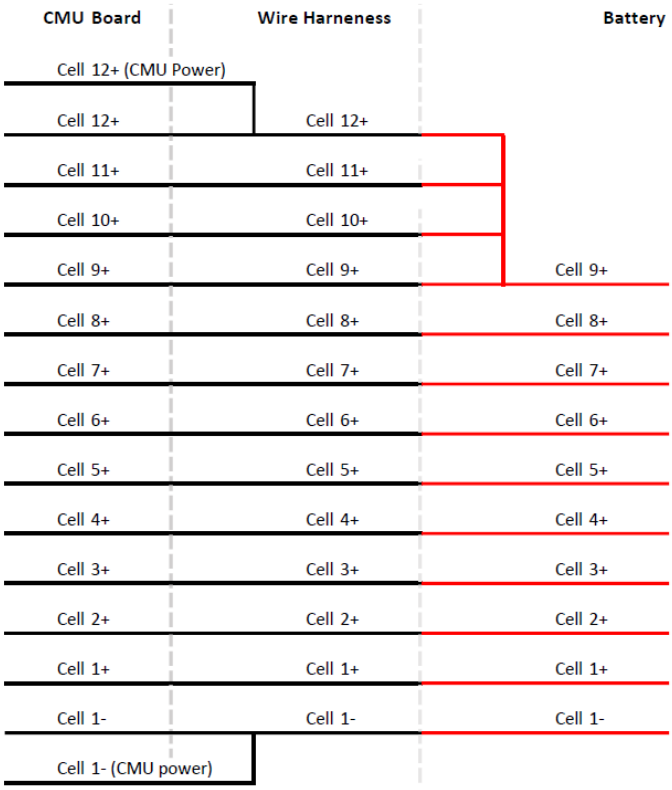
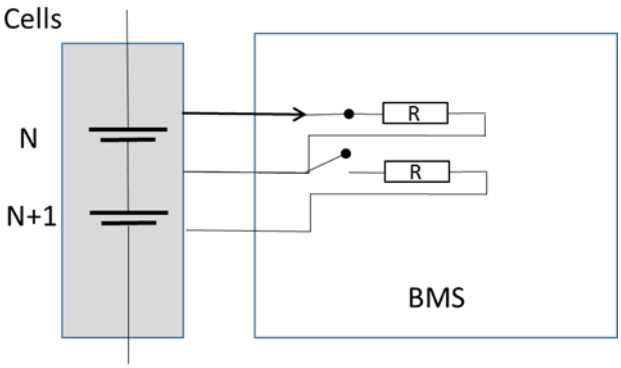


Figure 2-14 9 cell connection example

2.4.2 Bleeding circuit



Bleeding the current away from individual battery cells is a key BMS function which ensures that when a battery is charged, all cells will be charged to their maximum capacity (100% SoC).
For the n-BMS, bleeding is done by switching in a bleeding resistor between the '+' and '-' side of the individual cell. This is shown schematically in Figure 2-15 where cell N is bleeding, while cell N+1 is not.

Figure 2-15 Bleeding principle

The maximum bleeding current for each cell is 200 mA and when many cells are bleeding at the same time, the BMS board may heat up. For optimal balancing performance, it is therefore important that adequate ventilation/cooling is provided to the 'back' side of the board, where the bleeding resistors and transistors are placed.

2.4.3 Temperature monitoring

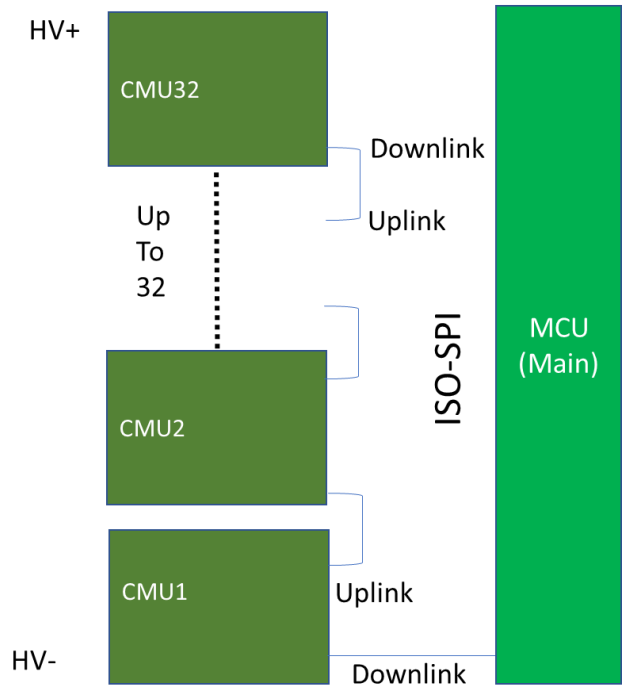
12 external temperature sensor inputs including ground connections are available on the CMU. As for the MCU, these are designed for 10 kΩ NTC sensors with a default b-value of 3900, but with user-configurability for sensor characteristics.

Internal temperature sensors measure the board temperature close to the bleeding resistors at the CMU. These resistors are used to turn bleeding off if the board exceeds the temperature limit. The bleeding temperature limit is 90°C and the bleeding will be turned on as soon as the board temperature again gets below 90°C.

2.5 Inter-board (ISOSPI) Connections

The communication between the different BMS boards is performed via a daisy-chained ISOSPI serial bus connection as shown in Figure 2-16. 'Downlink' means towards the MCU, and 'uplink' means away from the MCU.

The ISOSPI communication lines are internally terminated with 120 Ω resistors and the ISOSPI cables (cables 000802, 803, 804, 805) provided by Lithium Balance are impedance matched to 120 Ω.



The connecting principle of the ISOSPI daisy chain is as follows:

- The MCU (J3) is connected to the first CMU (CMU 1) using the 'CMU' downlink connector (J2). CMU 1 is typically connected to the low voltage (HV-) part of the battery
- The CMUs are daisy-chained using the uplink connector of the CMU (J1) closest to the MCU to connect to the downlink connector (J2) of the CMU furthest from the MCU.
- Termination of un-used uplink connector on the 'last' CMU is not needed as terminations are provided on-board

Figure 2-16 Daisy-chained ISOSPI communication between BMS boards

3 BMS functionality

This section describes the main functionality of the BMS including the key parameters necessary for the configuration of the BMS.

3.1 Activating primary measurements

At the battery cells, the BMS measures cell voltages, external temperatures and pack current.

3.1.1 Current

The pack current can be measured either by a current sensing SHUNT or a HALL effect sensor. These sensors types are described further in section 6.3.1 and section 6.3.2. Please note that by definition Current In = Charging gives positive current levels for a correctly configured system, where as discharging gives negative current levels.

Both shunt and Hall sensors have to be configured in the BMS Creator (System Configuration pane) with respect to measured-Voltage-to-displayed-current conversion, please see 'Shunt resistance' and 'Hall sensor scaling' in Figure 3-1.

The BMS allows the use of hall sensors with a single or two ('high ' and 'low' current outputs, please see section 2.3.4). If a single output Hall sensor is used, the two Hall sensor scalings MUST BE configured with the same gain.

Furthermore, the possible (0-input) offset of the shunt and Hall sensors can be corrected. This is often required for Hall sensors. In this case the offset can be adjusted after the Scaling factors have been written from the BMS creator to the BMS board:

- Make sure there is no current in the system and then read the Hall sensor current (14 ID_PACK_HALL) as listed in section 8.4.
- Insert the offset value (in mA) via BMS creator to get the current level to 0. Note that offset values are 'added', meaning that If the BMS displays a negative current, write a corresponding positive offset and vice versa.
- Write the new configuration to the BMS and check that the current reading is now 0

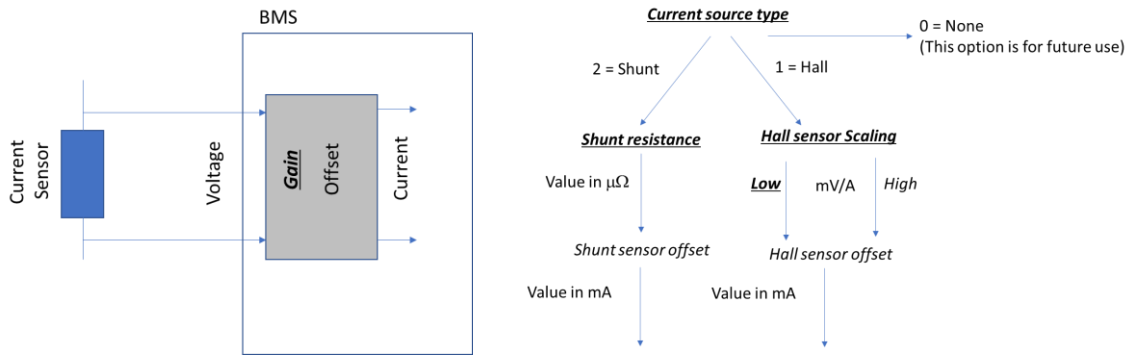


Figure 3-1 Current sensor set-up

3.1.2 Temperature

The actual temperature inputs used by the BMS needs to be activated in the BMS Creator. This is done by assigning one bit per sensor and to set the bit to 1 if the input is used and to 0 if it is not used.

To for example activate temperature sensor 6, 3 and 2, the relevant binary number is 00100110 which corresponds to the decimal number 38. This should then be inserted in the "Temp. sensors enabled" field of the "Operational limits" pane of the BMS Creator

Temp sensor		6	5	4	3	2	1
Used		Yes			Yes	Yes	
Binary	00	1	0	0	1	1	0
Decimal	38						

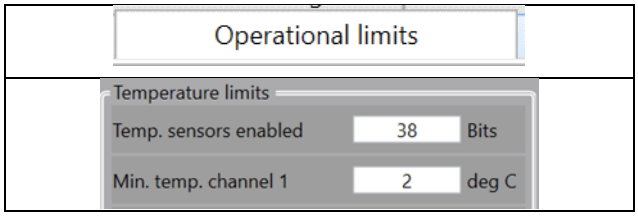


Figure 3-2 Configration example for activating temperature sensors

The n-BMS is designed to measure up to 12 cell temperatures per CMU board and up to 11 auxiliary temperatures measured by the MCU.

3.1.3 Cell Voltage

The BMS is designed to measure a configurable number of cell voltages. This is done in a similar fashion as the activation of the temperature sensor, i.e. by setting bits to 1 in binary numbers.

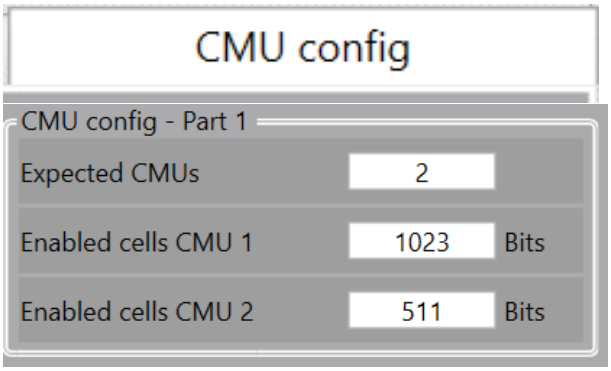


Figure 3-3 Configuration example for activating voltage sensors

This is a setting for each CMU.

3.2 Safety and thresholds of cell parameters

If a lithium ion battery is operated outside the specified limits for current, voltage or temperature, the battery may be permanently damaged or in extreme cases catch fire. The Lithium Balance BMS therefore constantly monitors battery and cell current, voltage and temperature to prevent the battery from operating outside its safe operating area. Please note that the safe operating area depends on the lithium cell chemistries and the application. Consequently, safety thresholds must be configured by the user.

If any primary parameter exceeds a safe operating limit, the BMS will do the following:

1. Raise an error signal corresponding to this specific parameter.
2. If this error signal persist, the BMS can open the contactor to block the battery current and get the system into a safe operating domain.

For the BMS, the voltage, current and temperature safety thresholds can be configured freely to allow the use of the BMS for all lithium cell chemistries and for all applications.

Please note that setting these thresholds correctly is essential to the safety, lifetime and energy storage capacity of the battery. Parameter settings must be based on the requirements of the specific application and the battery information provided by the battery manufacturer, e.g. in datasheets.

3.2.1 Operational Current limitation

The BMS measures the current running through the battery to ensure that the current is kept within user specified limits. This is required to protect both battery and auxiliary components like switches, fuses and cables.

If the BMS detects that the actual current is beyond the current limits it will immediately raise an error and, depending on configuration also open contactors.

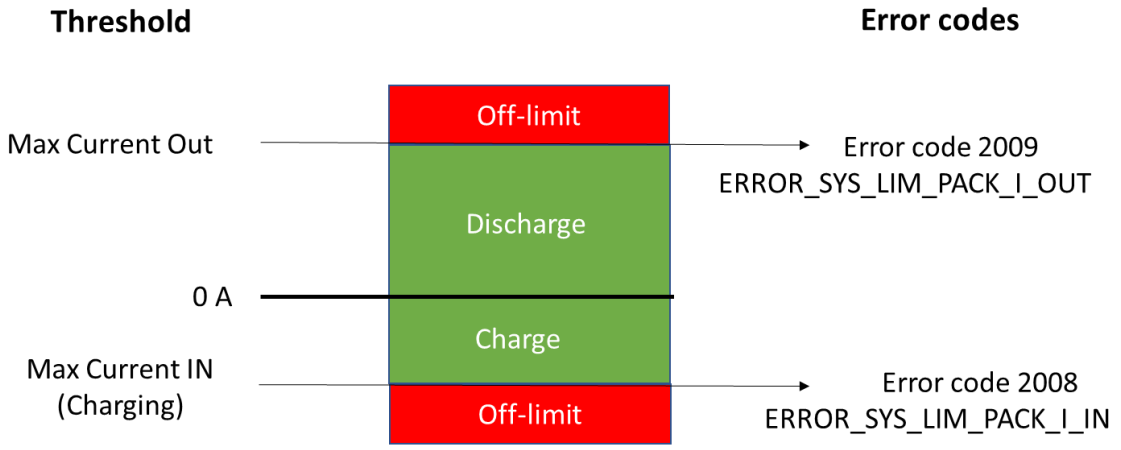


Figure 3-4:Overview of safety thresholds relating to the battery current

The BMS operates with three current limits that are used for thresholds:

- Maximum current in to the battery.
- Maximum current out of the battery.
- Maximum i2t sum for peak currents.

The first two current limits, “Max. current in” and “Max. current out”, are continuous current limits which means they are used in most situations as limits while BMS is operating. Further explanation for Max current out and in are explained in subsection 3.2.1.1 and 3.2.1.2 respectively.

The third current limit called “Max. i2t” is used to allow short periods of peak currents beyond the continuous current limits, this is explained in an individual subsection 0.

When performing checks for current limits against the thresholds the BMS will allow for small variations before triggering an error:

- If the BMS is **not** in active state (see section 3.4), the deadbands are:
  - Current in: 0.5 A
  - Current out: 0.1 A
- In active state the deadbands are:
  - Current in, continuous mode: none.
  - Current in, current regulation mode: 0.5 A + minimum charge current parameter.
  - Current out: none.

3.2.1.1 Maximum current out - discharging current

In all operating modes for the BMS, the “*Max. current out*” determines the maximum continuous current allowed to flow out from the battery. This limit is static, and does not change unless another configuration value is set.

Thus, it is important that the maximum discharging current setting does not exceed the maximum allowed continuous discharging current of the cell (specified by the battery manufacturer) to avoid potential damage of the battery cells. For short peak currents the current limit “Max. i2t” can be used.

3.2.1.2 Maximum current in

The maximum continuous current in is limited by either the static parameter “Max. current in” or the dynamically “calculated charge current request”. This is due to a need for a lower current in to the battery when cell voltages are close to their target voltages for charging.

The BMS determines which parameter to use for the current limitation by comparing the current request calculated by the BMS with the “Max. charge current” and the parameter “Dynamic current in, deadband”, as shown in Figure 3-5:

- If the charge current request is higher than “Max. charge current” minus “Dynamic current in, deadband” value, then the BMS will use the static limit, i.e “Max. current in” for current limit checking ($i_{limit} = k_{max,limit}, \{i_{ref} > k_{max,charge} - k_{dyn,deadband}\}$).
- If the charge current request is lower than “Max. charge current” minus “Dynamic current in, deadband” value, the BMS will use “charge current request” value for current limit checking ($i_{limit} = i_{ref}, \{i_{ref} \leq k_{max,charge} - k_{dyn,deadband}\}$).

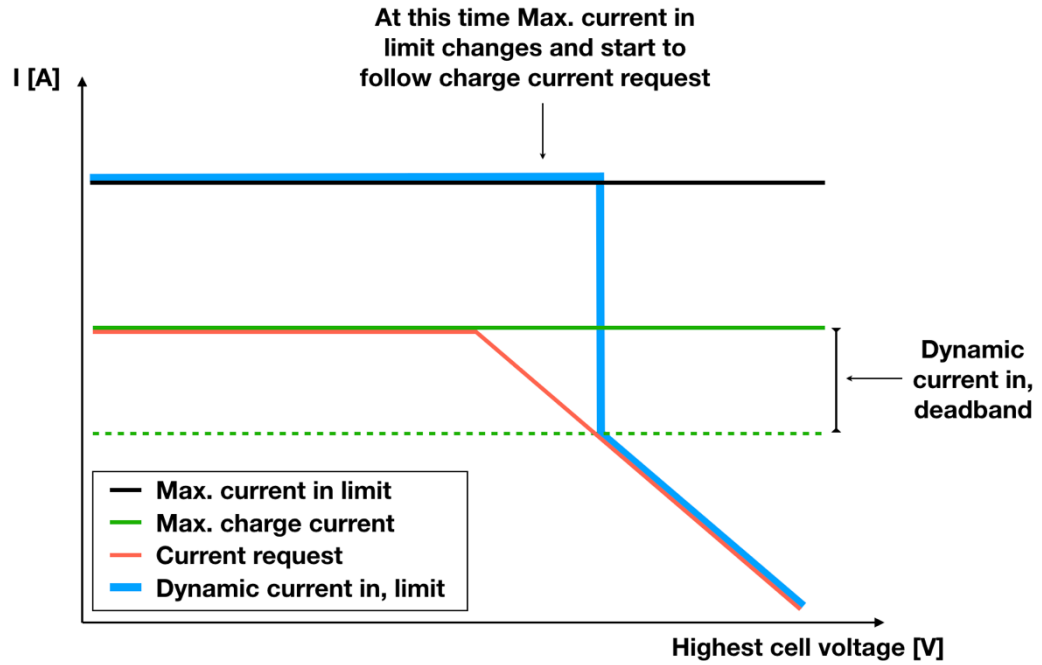


Figure 3-5 Continuous current in limit and its relation with the dynamic current in deadband parameter.

In Figure 3-5, the Dynamic current in limit initially follows the static parameter called Max current in. After a point, when charge current request gets lower than dynamic current in deadband limit, the dynamic current in parameter starts to follow charge current request value.

In the BMS creator, maximum current in parameter, maximum charge current parameter, and dynamic current in deadband parameters are all configurable.

3.2.1.3 Advanced current limitation, i2t

The BMS calculates the "i2t-parameter', which can be considered as a digital implementation of a melting fuse. The intention is to assure that the BMS can break the battery current if it significantly exceeds the specified maximum current in or out levels for extended periods of time.

The I^2t parameter reflects the 'accumulated' excess power provided by the battery. By excess power is here understood that:

- The system is specified for a max. current out and a max. current in, any current exceeding these limits are considered 'excess' current.
- The electrical power is proportional to the square of the current. Hence, the BMS constantly calculates the accumulated excess power $(I - I_{max})^2$
    - If the actual current is **larger** than the current limit, the excess power is **added** to I^2t
    - If the actual current is **smaller** than the current limit, the excess power is **subtracted** from I^2t, until 0 is reached

The I^2t parameter is probably best understood through an example. Consider a forklift where the maximum currents are:

- Max. Current In (Charging) = 75A

- Max. Current Out (Discharging) = -100A

According to the system design an excess current of 200 A is considered acceptable for 2.5 seconds. Consequently, the I^2t threshold is set to $(200\ A)^2 \times 2s = 100.000\ A^2s$

The forklift is now performing the following actions, please also refer to Figure 3-6

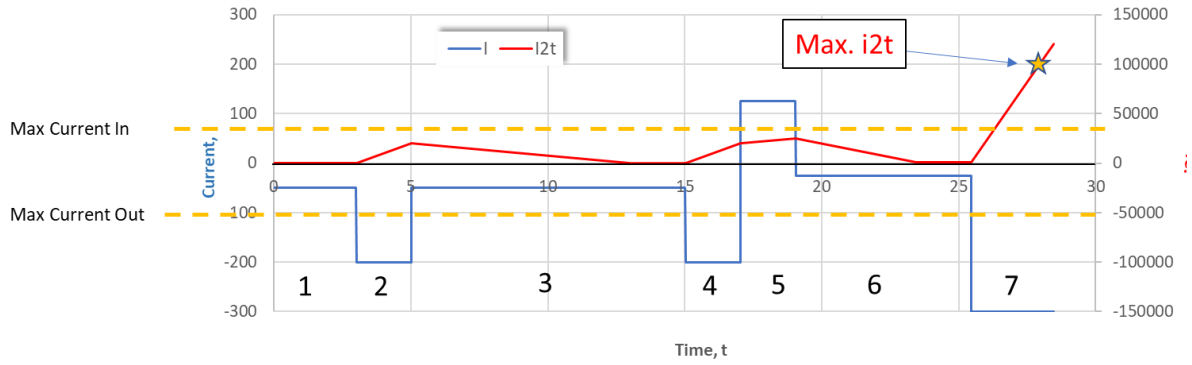


Figure 3-6 Forklift example for the i2t parameter

1. Initially the forklift drives around picking up a load. This requires only 50A which is smaller than the Max Current Out of 100A. As the I^2t parameter is already 0, it stays there.
2. A load is now lifted to a shelf which require 200A. This exceeds the Max Current Out and the I^2t parameter therefore increases
3. The forklift picks up a new load. This requires only 50A which is smaller than the Max Current Out and the I^2t parameter therefore decreases to 0 and then stays there.
4. A load is now lifted to a shelf as under point 2
5. The driver regrets and lowers the load again. In this process 125A of regenerative power I generated. This exceeds the Max Current In of 75A and the I^2t parameter therefore continues to increase, though at a lower slope.
6. The forklift picks up a new load and the I^2t parameter decreases to 0
7. A **heavy** load is now lifted to a **high** shelf which require 300A for several seconds. The I^2t parameter therefore increases significantly. In this case the I^2t threshold of 100.000 A^2s is passed and the "2010 ERROR_SYS_LIM_PACK_I2T" error is activated by the BMS.

3.2.2 Temperature

The external temperatures measured by the BMS can be configured for temperature maximum and minimum thresholds as shown in Figure 3-7. Here the max and min temperature of the individual temperature sensor can be configured independently in the "Operational Limits" pane of the BMS Creator.

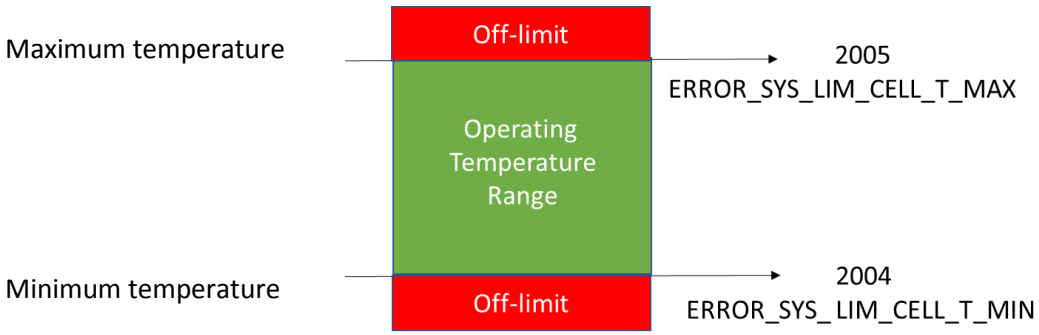


Figure 3-7 Overview of safety thresholds relating to the cell temperatures

3.2.3 Cell voltage

To achieve reliable and robust operation of a battery system, it is essential to always keep the voltages of all lithium cells within well-defined limits.

- Too low cell voltages may lead to chemical changes in the battery electrolyte which will degrade the battery and potentially produce a combustible gas.
- Too high cell voltages may cause the positive electrode to decompose which can lead to degradation and cause excessive heating of the cell.

The maximum and minimum thresholds for safe operation can be configured in the “Operational Limits” pane of BMS Creator.

Two other voltage levels to configure are the “Cell Voltage Target” and the “Charge Complete deadband”. When charging, the system adjust charger and balancing current levels to assure that at the end of charging all cell voltage are at the “Cell Target Voltage” with an accuracy defined by the “Charge Complete deadband”. These two parameters are configured in the “Charger” pane of the BMS Creator

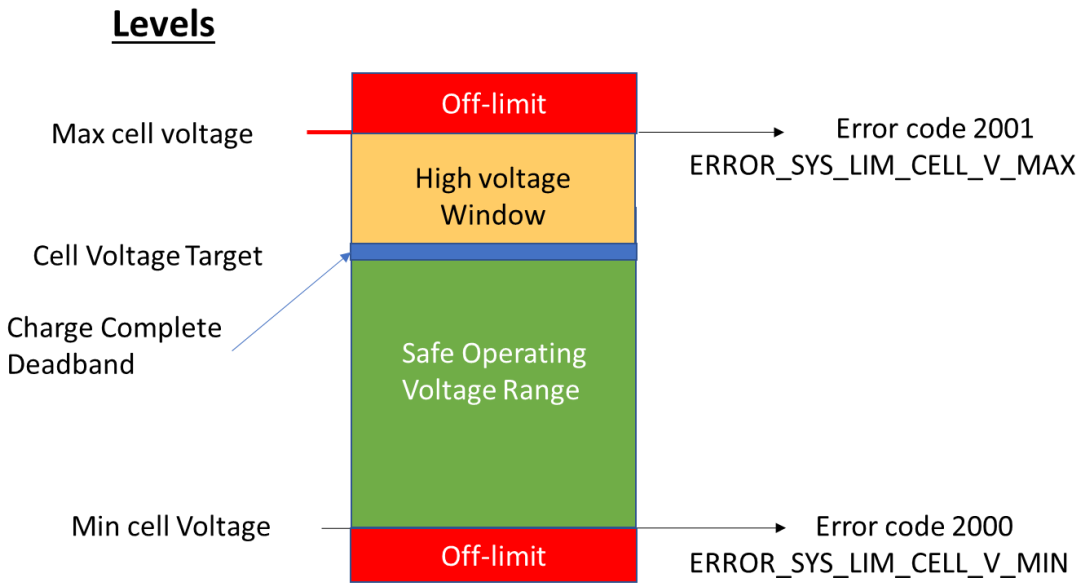


Figure 3-8 Overview of safety related cell voltage settings.

3.3 Derived metrics = Calculated parameters

The BMS uses the measured data to quantify different operational parameters of the battery.

3.3.1 Cell voltage statistics

Every time the cell voltages have been updated the BMS will calculate statistics for every measured cell, this includes (please see section 8.4):

- The minimum cell voltage (ID_CELL_V_MIN_VAL)
- The maximum cell voltage (ID_CELL_V_MAX_VAL)
- The average cell voltage (ID_CELL_V_AVG)
- The voltage sum of measured cells (ID_PACK_V_SUM_OF_CELLS)
- The number of cells measured (ID_CELL_V_NUM_AVALIABLE)

3.3.2 Capacity and SoC

State of charge (SoC) is a key battery parameter describing the electrical charge (Q = current x time) left in the battery before charging is required. SoC is the battery equivalent of a fuel gauge in a traditional vehicle. The unit of SoC is percentage points (0% = empty; 100% = full).

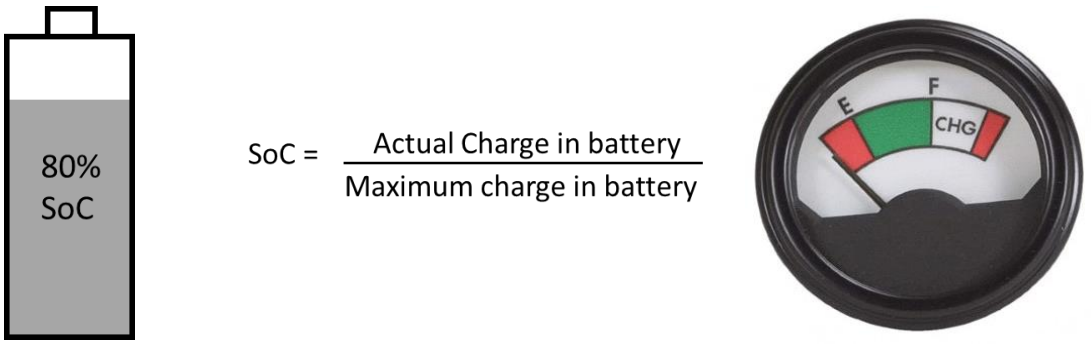


Figure 3-9 SoC describes the relative amount of energy left in the battery and can be seen as the equivalent of a fuel gauge in a car

The state of charge is measured by the BMS using a method known as Coulomb counting. Every time the pack current has been measured (either by SHUNT or HALL effect sensor) the number of ampere-seconds will be added or subtracted from a capacity counter called "remaining capacity".

When the BMS has detected a fully charged battery (see section 3.3.3) it will calibrate the "remaining capacity" so it is equal to the expected full capacity (this is typically the same as the design capacity for new batteries). The "remaining capacity" is always calculated with a high resolution of 1 ampere-second, and also available in a lower resolution of 0.1 ampere-hour.

The high resolution "remaining capacity" is used to calculate the state of charge (SoC) of the battery, this is done by dividing with the expected full capacity. Thus the SoC is a measure of how close the "remaining capacity" is to the expected full capacity.

The SoC is available in an internal version which ranges from -300 % to +300 %, but also available in a trimmed variant where the value can be a scaled version of the internal SoC. E.g. if the trimmed SoC is set up to be between 10 % (min) and 90 % (max) of the internal SoC it will display 0 % when the

internal SoC is at or below 10% and 100 % when the internal SoC is at or above 90 %, see Figure 3-10 for a visual representation.

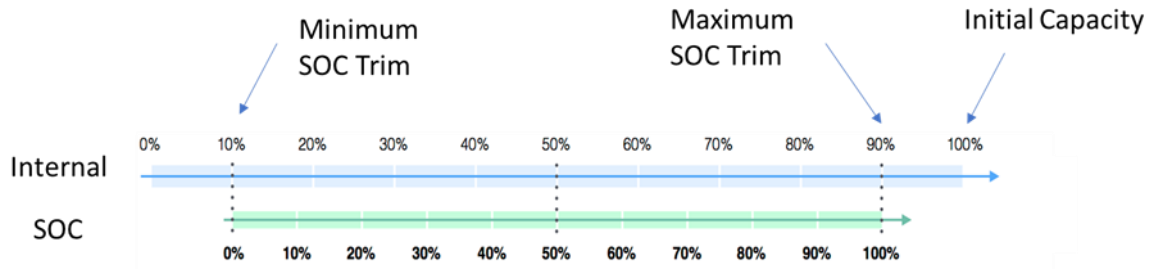


Figure 3-10 SoC mapping between internal and trimmed data.

The “Initial Capacity”, “Minimum SOC Trim” and “Maximum SOC Trim” are all configured in the “System Configuration” Pane of the BMS Creator.

3.3.3 Open circuit voltage SoC calibration

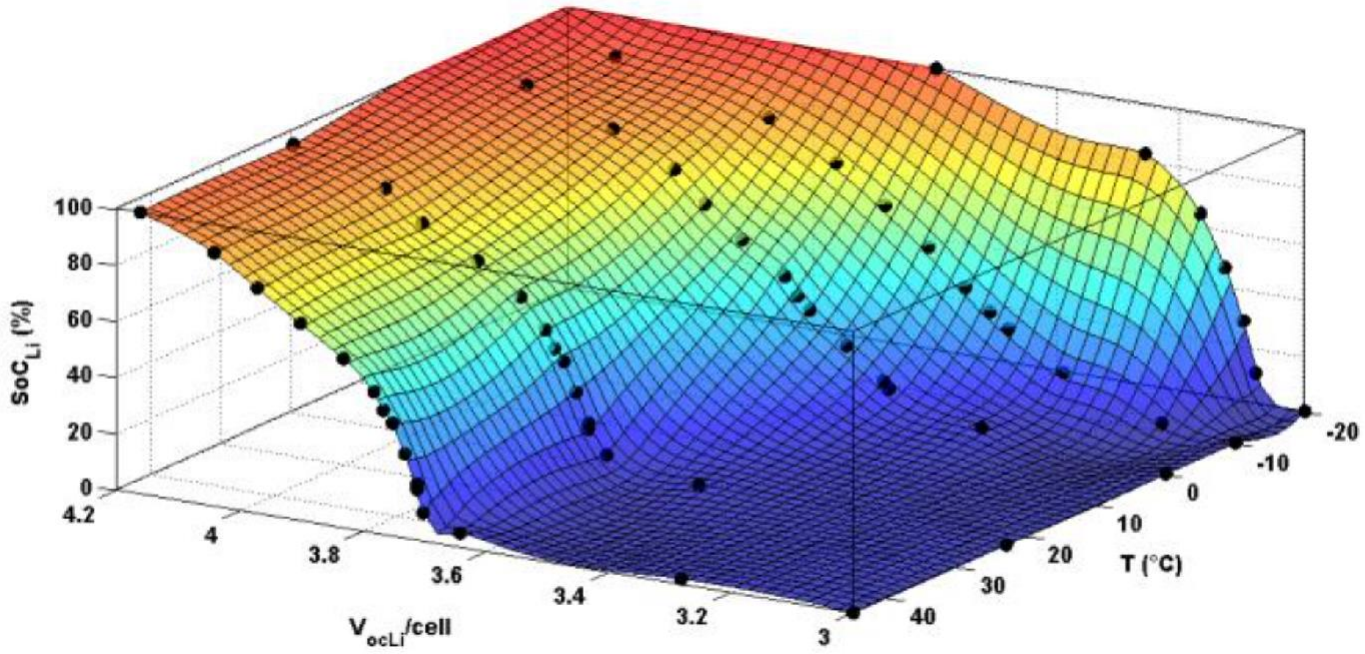
In certain scenarios, the state of charge (SoC) calculation from coulomb counting can become inaccurate and may need to be calibrated. The BMS can then calibrate the SoC in two scenarios:

1. At end of charge when all cell voltages have been sufficiently balanced.
   For end-of-charge calibration the BMS will ensure that each cell voltage is within a certain threshold of the target cell voltage, while the charge current must be below another threshold.
2. After a period of inactivity with no power being drawn from, or delivered to, the Li-ion cells. This is achieved by using the relationship between the open circuit voltage (OCV) of the Li-ion cells and their SoC. Due to the electrochemical processes the OCV will be higher when the battery is charged (100 % SoC) than it is when discharged (0 % SoC). As the temperature can influence the OCV this parameter is considered when performing the calibration. This method will be used to calibrate the SoC, when the BMS is powered up (unless functionality is disabled by user).

The OCV can be considered equal to the terminal voltage of the cell when enough time has passed and the change in voltage (dV/dt) is lower than a certain threshold. The effect of time is used as a quality factor for the SoC calibration, where “time” means the duration between last power-off and the current power-on[5]. The quality factor is then defined as a value in the range of 0-100 % depending on the time needed for the voltage at the cell terminals to reach OCV state.

The data for the OCV vs SoC vs temperature is unique for different Li-ion chemistries, manufacturers and sizes of batteries and must be obtained from the manufacturer or from experiments. The BMS can work with one to five datasets at different temperatures. The figure below shows a graphical representation of five datasets (black dots) for OCV vs SoC at the temperatures: -20 ºC, -10 ºC, 0 ºC, 23 ºC, 45 ºC.

[5] It is assumed that the Li-ion cells have been at rest while the BMS has been powered off.



The SoC selected for calibration ($SOC_{calibrated}$) is found by interpolation in the 3-dimensional data set (OCV, SoC, temperature), and the quality factor (k) determines the significance of the calibration SoC (SOC_{OCV}) with regards to the originally coulomb-counted SoC ($SOC_{original}$):

$$SOC_{calibrated} = SOC_{original} + k \cdot (SOC_{OCV} - SOC_{original})$$

The BMS can be configured to a maximum quality factor (k) of less than 100 % in case the calibration should never rely fully on the OCV factor.

See appendix 8.6 for an example of an experiment to determine the configuration values.

3.4 Operating modes and state machine

The BMS is designed as a state machine which can be in a number of different modes as shown in Figure 3-11. Before entering a new state, an error checklist is verified to allow access to the new state:

- If no critical errors on the relevant checklist are present at the severity level triggering action, the BMS will enter the error mode. By critical errors are understood, errors at the severity level triggering action as described in section 3.6.2.
- If no critical errors are present, the BMS will move to the next level in the state machine

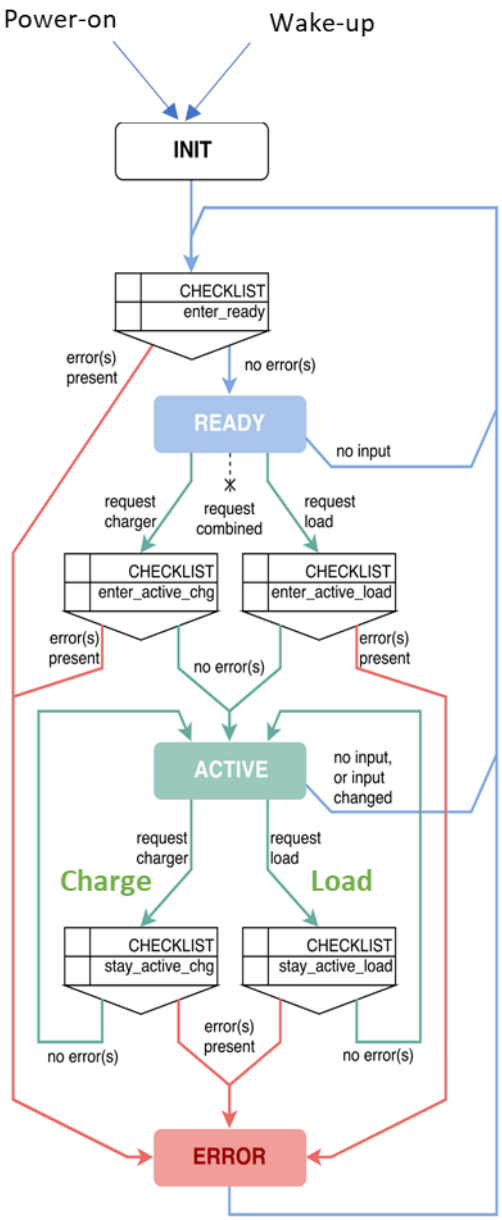


Figure 3-11 BMS state diagram.

The BMS is started up by applying power to the BMS. This requires that:

- The main supply voltage (nominal 12V) is fed to the BMS
- The BMS is not in sleep mode

After starting-up, the BMS enters the basic initialisation (INIT) mode. If no critical errors are present, it will immediately move on to the READY mode

3.4.1 Boot/Configuration mode

The BMS boot-mode indicates a mode outside of the state diagram of Figure 3-11. In this mode the system is ready to receive new firmware or new configuration (*.XLM) files. Once these have been successfully installed the system will leave boot mode and go into the so-called application mode, which corresponds to entering the state diagram of Figure 3-11

3.4.2 Sleep Mode

The BMS supports sleep mode, which is entered by activating an I/O if this function is configured, see section 3.9

Waking up the BMS can be done in two ways:

- Via the ignition-pin please see section 2.3.1.
- Via a CAN wake-up frame a described in section 4.4.1.

3.4.3 Ready mode

In READY mode, contactors stay open, but otherwise all system functions are active. When either the 'Request Charger' or the 'Request Load' IO signals are detected, the system will enter the 'Active Charge' or the 'Active Load' mode

Please note that if BOTH the 'Request Charger' or the 'Request Load' signals are active, the system will enter the 'Active Charge' mode

3.4.4 Active mode

In the ACTIVE mode, contactors will be closed as described in section 3.5 and current can flow to and from the battery pack.

The BMS will leave active mode again:

- If the system is turned off
- If both the Request Charger' and the 'Request Load' signals are in-activated in which case the BMS will go back to Ready mode
- If a critical error appears and the system enters error mode

Please note that in the active mode, no limitations are set to the allowed current flow direction in 'Active Charge' nor in 'Active Load' mode. This implies that the BMS will accept charge current in 'Active Load' mode and the BMS will accept that current is drained from the battery in 'Active Charge' mode.

3.4.5 Error Mode

In Error mode, the BMS will set the battery in a passive state by assuring that no current is drawn from or sent to the battery:

- The contactors will open
- The current requested by the BMS will be set to 0

The system will leave the error mode when the critical error which triggered the error mode has disappeared. To avoid system instabilities the system waits a period known as the 'De-bounce' time' before it can re-enter the active modes (close contactors) after a critical error has forced the contactors to open.

3.5 Contactor control

3.5.1 Contactor configuration

The BMS supports a full four contactor layout arrangement to connect and disconnect both the load and the charger galvanically, please see Figure 3-12:

- Load positive
- Load negative
- Pre-charge
- Charge negative

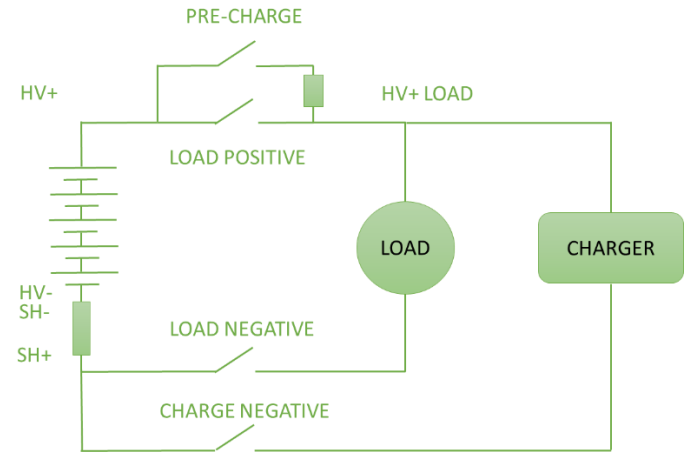


Figure 3-12 Contactor arrangement for the BMS

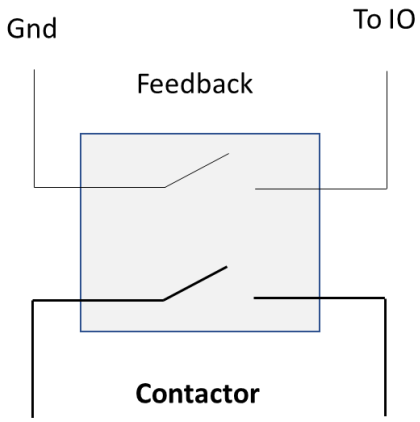


Figure 3-13 Contactor Feedback

Furthermore, each contactor may include a feedback switch/signal, which is a simple low current switch mirroring the position of the main contactor. By connecting one end of the feedback switch to Gnd and the other to a BMS IO, the actual position of the Contactor can be monitored by the BMS.

Any combination[6] of actually used contactors with or without feedback can be configured in the BMS Creator by the GPIO settings as described in section 3.9. If any of the four potential contactors is not used/configured, the standard actions (see Figure 3-15) associated with this unused contactor will simply be skipped.

3.5.2 Contactor activation

In all modes except, the ACTIVE mode, contactors are always off. In the active mode three different contactor control/enable sequences exists and are activated when the BMS in the active mode receives a mode change request, please see Figure 3-14.

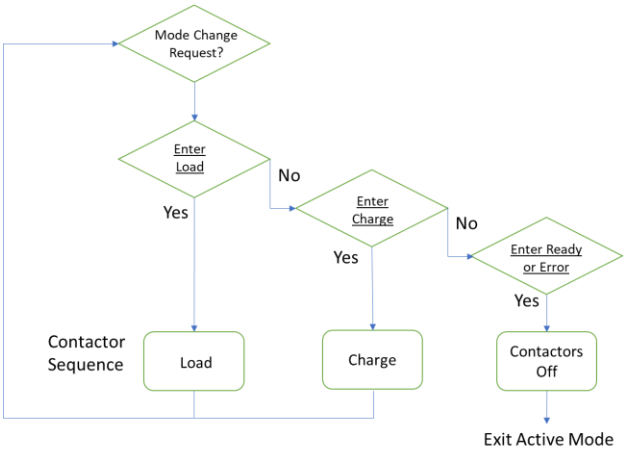


Figure 3-14 Overall contactor control scheme in ACTIVE mode

1. Entering Active Charge mode. This is entered if the system is in not in 'Charge mode' and the "Request Charger" signal becomes/is activate.
2. Entering Active Load mode. This is entered if the system is in not in 'Load mode' and the "Request Load" signal becomes/is activated
3. Contactors off. This is entered when the system leaves active mode either to go to error mode or to go to ready mode.

[6] Some combinations may be limited by the total number of available GPIOs, depending of how many GPIOs that are used for other purposes.

3.5.3 Activate Load sequence

When the BMS enters the "Activate Load" contactor sequence, the BMS will control the contactors as shown in Figure 3-15.

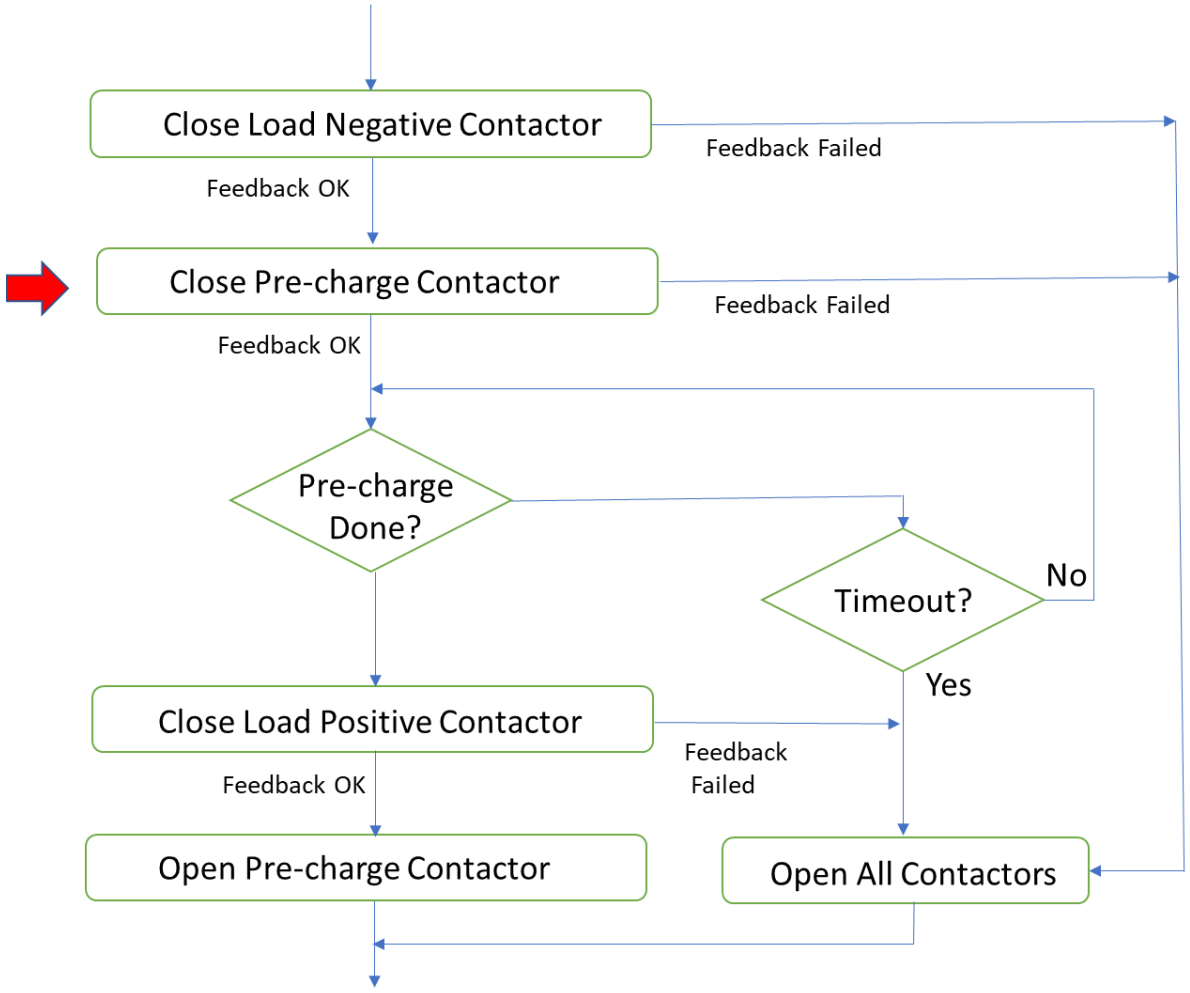


Figure 3-15 Contactor sequence for activating load. The red arrow on the 'Close Load Negative Contactor' indicates that this is the only action which is different between the 'activating load' and 'activating charge' contactor sequences

Please note that for all contactors where feedback is enabled, the BMS will after each change in contactors state:

- Test for correct feedback, i.e. if the new contactor state should be OPEN, the BMS will chech that the feedback signal matches the OPEN state
- If the feedback failed the sequence will be aborted. If this 'Feedback' error is configured in the checklist "Errors – Stay load", this will eventually send the system into error mode

Before entering the contactor activation sequence, the BMS will check if the 'de-bounce time' has expired and if not wait until this is the case. The de-bounce time is a system stabilising time, determining the time needed from opening all contactors, see section 3.5.5, until the system is again allowed to close the contactors.

3.5.4 Activate Charger sequence

The active charger contactor sequence is identical to the active load sequence except for the first contactor closing where the 'Charge Negative Contactor' is closed rather than the 'Load Negative Contactor'

3.5.5 Contactors Off sequence

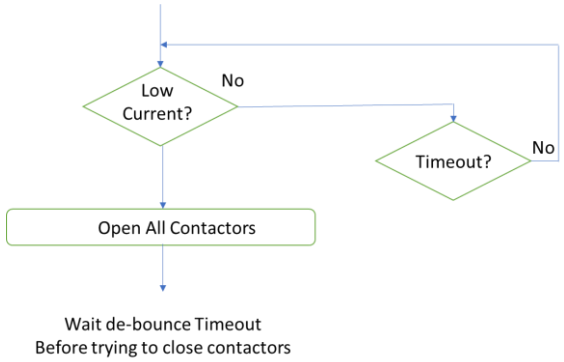


Figure 3-16 Contactor Off sequence

When requested to open contactors (Contactors Off), the BMS will, to protect the contactors, not open these until either,

- The current is below the "Max contactor break curr."
- The time passed since 'the Contactors Off' request exceeds the "Contactors off timeout"

A flow diagram is shown in Figure 3-16 and specific examples sketched in Figure 3-17

.

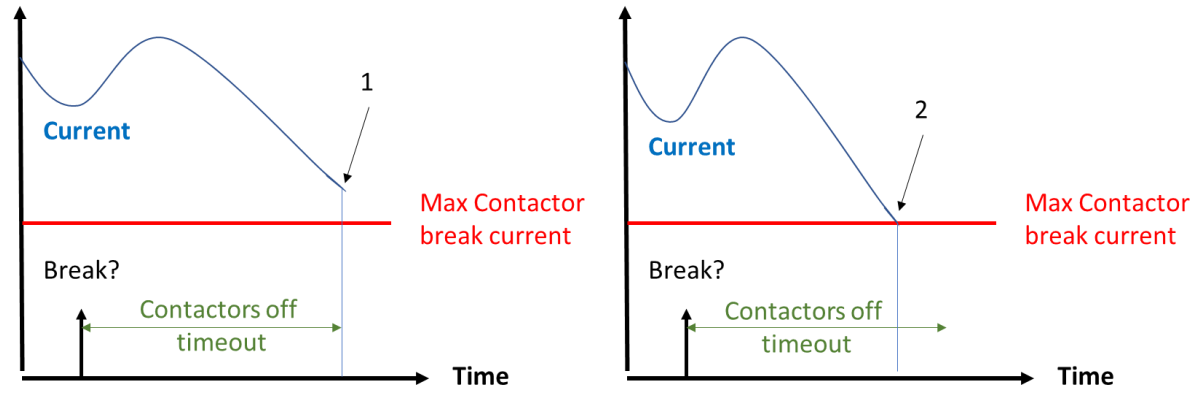


Figure 3-17 In the first example the current does not get below the desired level before the "Contactor off timeout" has experied. Consequently, the contactor opens at point '1' corresponding to the "Contactor off timeout" time. In the second example the current reaches the "Max contactor break current" before the "Contactor off timeout" Consequently, the contactor opens at point '2'.

3.5.6 Contactor timing

Configuring the timing of the contactors is done in part2 of the "Operational limits" pane of the BMS creator.

Here, the two parameters circled in Figure 3-26 below need to be configured.

Please note that all contactors in the system will operate with the same:

- Max contactor break curr.
- Contactors off timeout

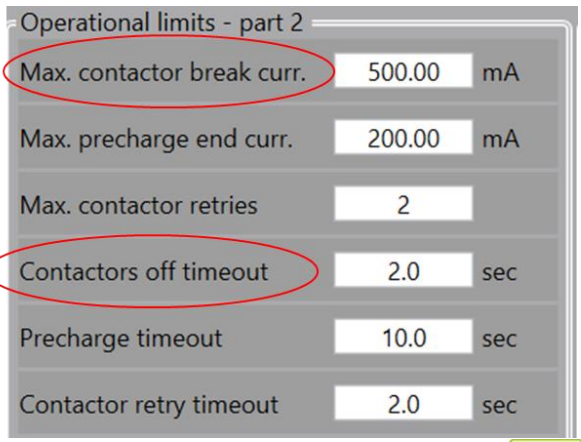


Figure 3-18 Contactor timing parameters to be configured

3.6 Error handling

The BMS monitors internal and external functionality to ensure a safe operation of the battery application. Each internal and external function is able to trigger an error that is stored in the BMS as long as the error condition is active. For each active error there is a severity level, an origin of the error, the error code, and an escalation timer. Please see section 8.3 for a list of error codes, severities, origins and escalation times.

The severity level is divided into 3 levels and 2 categories. The severity levels are: LOW, NORMAL, CRITICAL and can be either in the SYSTEM or PACK category.

An example of error handling functionality is sketched in Figure 3-19 below.

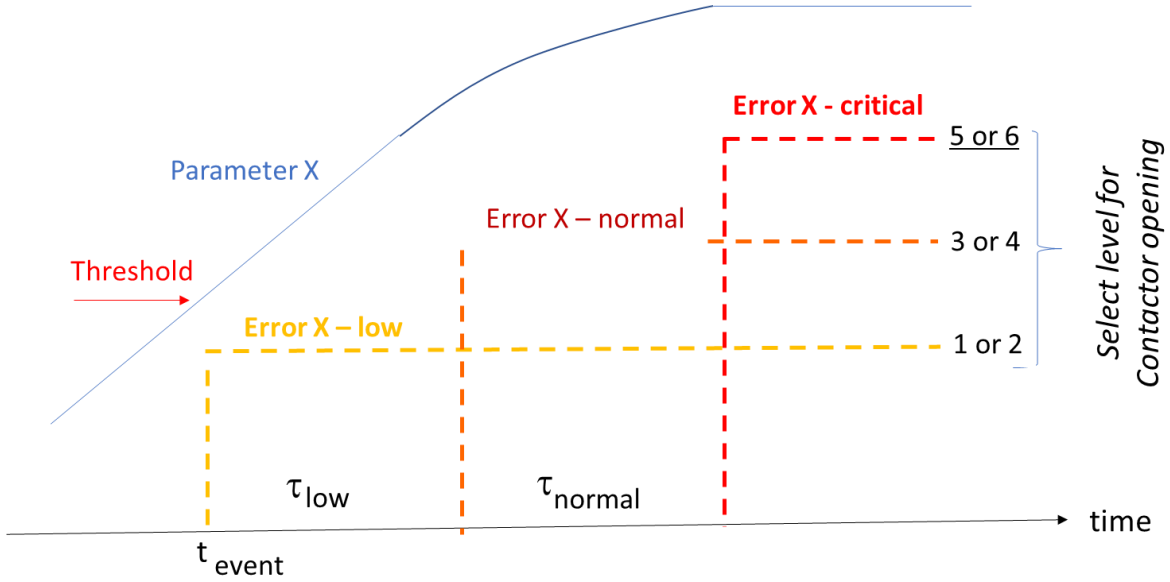


Figure 3-19 Example of error handling. "low", "normal" and "critical" refer tonames for the severity levels (1-2, 3-4 and 5-6).

When a threshold setting for a given parameter, X, is crossed, an error associated with this particular parameter is activated in the BMS (e.g. an error for cell voltage too high) with a severity level and an escalation time (that can be 0, e.g. no escalation).

If the escalation time is not 0, an internal timer will escalate the error through the severity levels. The severity level will increase when this timer reaches a configured time (τ_{low} or τ_{normal} in Figure 3-19).

In the example, initially, the error will be at the severity level 'LOW' but after a given time, τ_{low}, the system will raise the severity level to 'NORMAL'. After an additional time interval, τ_{normal}, the severity level will be raised to 'CRITICAL'.

3.6.1 Error escalation configuration

Some errors allow for user-configuration of their escalation times. These errors will always start out at severity level 'LOW' and then escalate at user-configured times.

Error timer category	Errors affected
Currents	Errors related to pack current limits. Covers: Charge current too high. Load current too high. I2t remaining is 0.
Balancing	Errors related to cell voltage balancing.
CMU communication	CMU communication errors.
Cell voltages	Errors related to pack voltage limits. Covers: Cell voltage below minimum. Cell voltage above maximum.
AUX temperatures	Errors related to auxiliary temperature sensors. Covers: No value from sensor. Open circuit (no sensor connected). Shorted sensor input. Below minimum temperature. Above maximum temperature.
Cell temperature sensors	Errors related to cell temperature sensors. Covers: No value from sensor. Open circuit (no sensor connected). Shorted sensor input.
Cell temperatures	Errors related to the measurements of cell temperature sensors. Covers: Below minimum temperature. Above maximum temperature.

3.6.2 Error checklists

An active error in the BMS does not by itself cause an action, such as opening contactors. Errors can be configured to control the BMS state machine (see section 3.3.3), which in turn controls the contactors.

In the BMS Creator, this is accomplished by filling out the error check lists as shown in Figure 3-20.

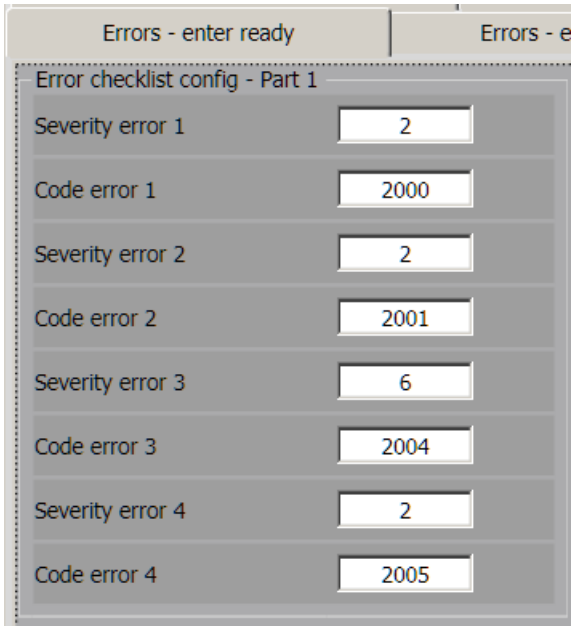


Figure 3-20: Part of the error checklist configuration for entering the Ready state.

These checklists control the BMS state machine state transitions and are thus one of the most important – and necessary – configuration steps to take, when setting up a system.

To ensure correct system operation it is important to configure the error checklists to match the application. Some settings might vary from application to application and opening of contactors might not be a desirable in all applications and situations.

First step is to consider what battery related errors are to be acted on in terms of relay protection, when a mode is active. In load mode, the "enter" and "stay" related checklists should be populated with errors and fault conditions, that if not handled by contactor protection, would damage the battery or cause a dangerous situation. For active load mode, this might be a current overload or a temperature overload. For active charge mode, it might be cell over voltage, temperature overload or sensor errors.

The differentiation between "enter" and "stay" is useful to define entering a mode with a perfect system without any problems, and then defining when not to stay in that mode by the use of the "stay" checklist. An example would be to define that entering load mode is not possible with a bad temperature sensor, but while in the mode, one bad temperature sensor does not exit the mode.

The "error codes" in the checklists are tightly connected to the "severity" of the errors and thereforeit is important to evaluate at what severity the error code in the checklist should take effect. One example would be to stop the request for current at a low severity (i.e. when it is first registered) and then safeguard it with an opening of contactors ("stay" checklist) if the error is persistent and increases to a higher severity.

3.6.3 Assigning actions to errors

To prevent entering or staying in a specific state under some specific condition, the relevant error code and severity should be inserted.

As an example, to prevent entering the Ready state when the “Max Cell Voltage” is exceeded, the error code 2001 has to be inserted in the “Code error” field of the error pane. This is shown for the Code error 2 field in Figure 3-20

The severity level at which the contactors are opened is configured in the associated “Severity error” field.

In Figure 3-20, severity 2 has been chosen for error 2, corresponding to contactor opening at severity level ‘low.

In the example, this means that the Ready state cannot be entered whenever error 2001 is active with severity level “low” or higher.

For comparison, severity level 6 has been chosen for error code 2004, which means that this error is only acted upon if its severity level is “Critical”.

If it desired NEVER to not act on a given error, this is done by NOT listing this error in the error pane.

Please note that error actions are configured individually for the different system states. These states are described further in section 3.3.3. This implies for example, that the user can configure the system to react in one way to a given error when charging and in a different way when discharging.

3.7 Charging and Balancing

Charger control is essential to a well-functioning BMS. If charging cannot be stopped when the cells are fully charged, unsafe scenarios may occur. Consequently, the BMS is designed to support redundant ways of stopping charging:

1. The primary method for stopping or modifying charging is via a communication protocol from the BMS, as described in this section.
2. The back-up method for stopping charging is to break the current path between the charger and the battery by opening all contactors.

The BMS is designed to request charge current from an external charger when the user has activated charging using an I/O (see chapter 3.9), if no errors prevent the BMS from operating in that mode. The errors that prevent entering and staying in charge mode is set up by the user using the BMS Creator. When in charge mode, the BMS will request a current over either CAN or PWM, depending on configuration.

3.7.1 CAN Charging

For CAN Charging functionality to function properly it is necessary to configure the correct CAN frame format according to the charger equipment manufacturer specifications. The information is typically available in their data sheet or upon request. The BMS has a number of built-in CAN charger formats that may be used instead of a user-configured CAN frame.

3.7.2 PWM

When PWM functionality is enabled the BMS will output a PWM signal at 1 kHz with a variable duty cycle. The active period of the duty cycle is when it is pulled to ground. A configurable setting is

available to invert the signal if needed. It is also possible to configure the minimum and maximum duty cycle for the PWM output. The BMS will then adjust the duty cycle until the measured current into the batteries matches the requested current for charging.

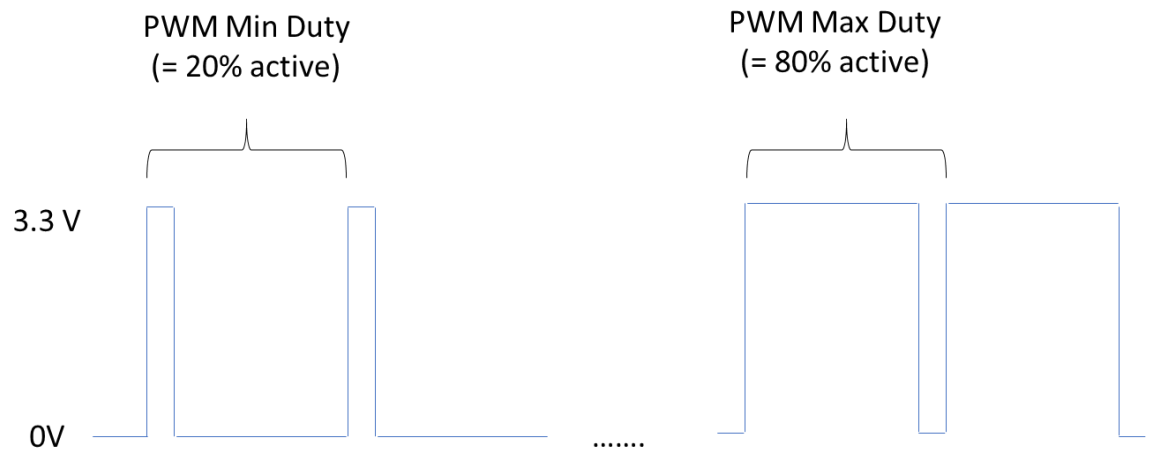


Figure 3-21 Examples of PWM signal with 20% and 80% Max Dutycycle. In this case the PWM operation is non-inverted (PWM Signal Inverted = 0 in the BMS Creator). If inverted operation had been selected, PWM signals would have been 0V when 'active' and 3.3 V when 'passive'

If it is required to control a charger by an analog voltage it si possible to convert the PWM signal to an analogue control voltage by using for example the set-up shown in

3.7.3 Requesting charge current and possible errors

The requested charge current is controlled by a software PID controller with configurable coefficients to allow the user to optimise the charger control for the specific battery system.

The PID controller determines the requested charge current (I_{Charge}) as function of the difference between the desired target cell voltage (V_{target}) and the actual maximum cell voltage (V_{max}), ie.

$$I_{Charge} = PID(V_{target} - V_{Max})$$

Please note that special error settings are for the 'request current' function, this is found in the BMS Pane 'Errors – request current'. These errors are set in the same way as the other errors panes, but in case of a critical error here the actions are different from the other errors settings. In case of a 'Request current' error:

- The BMS does not go into error mode, but stays in the present mode and does not change contactors
- The requested current is set to 0

3.7.4 Cell Balancing

Cell balancing is a key BMS function which ensures that when a battery is charged, all cells will be charged to their maximum capacity (100% SoC). This is essential for getting the full capacity out of a battery, as the maximum capacity of a battery is determined by the cell with the least capacity.

Cell balancing is controlled by measuring the cell voltages. When balancing is activated, the voltages of the cells with the highest voltages exceed a certain threshold, their charging is reduced or even inversed by bleeding, please see section 2.4.2

The BMS will only perform the balancing

- if an I/O has been configured to activate the balancing functionality (see section 3.9).
- This specific I/O is activated, i.e. connected to Gnd

When the configured I/O has been activated, the BMS bleeds all cells which have a voltage higher than the calculated average voltage of all measured cells within a small deadband. The balancing process will be active (but not necessarily bleeding) as long as the configured I/O is active. Actively bleeding will occur as long as all cell voltages are not approximately equal, possibly paused by high internal board temperatures, until such temperatures are no longer critical.

The balancing is performed by the combined regulation of the cells bleeding and adjusting the charge current level, as described below. Please also refer to Figure 3-22 and Figure 3-23.

- When bleeding starts, the bleeding circuits are activated for all cells with cell voltages larger than the average voltage. Referring to Figure 3-22 cells 2, 4 and 8 will be bleeding
- The charge current will be regulated/reduced as function of the difference between the maximum voltage and the target voltage.

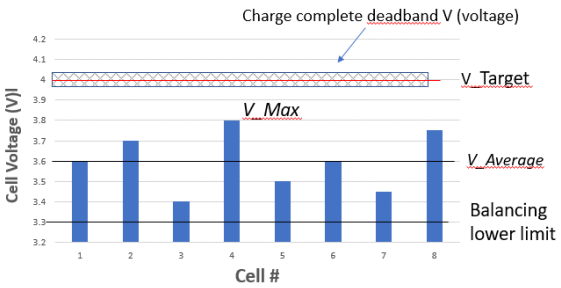


Figure 3-22 Key cell voltage parameters for bleedin

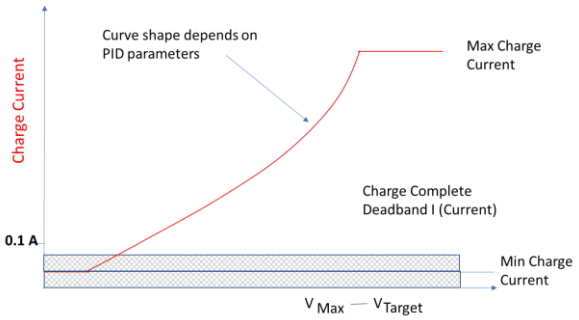


Figure 3-23 Charge current regulation

Balancing is initiated when the input pin configured for 'Activate Balancing' is pulled to ground. This can either be done actively by the user or automatically whenever charging starts by hardware connecting the 'Request Charge Active' input pin to the 'Activate Balancing'' pin.

Charging and balancing is ended when all cell voltages are within the charge deadband voltage around the target voltage AND the charging current is below the charge current deadband level, please refer to section 3.7.5

3.7.5 Charge complete

The charging process will end when two criteria are met at the same time:

- 1: the min. and max. cell voltage must be within the "charge complete deadband for voltage" from the target cell voltage, e.g. +/- 10.0 mV from 3600.0 mV.

- 2: the measured current must be within the "charge complete deadband for current" from 0 A.

Once these criteria are met the BMS will stop the charge process and calibrate the remaining capacity to be equal to the expected full capacity. At the same time the flags "fully charged" and "fully charged latched" will be set active. When the two criteria are no longer met the "fully charged" flag will be deactivated, but the "fully charged latched" flag will be active until the BMS has registered at least 0.1 Ah being discharged from the battery pack.

3.7.6 Selecting PID parameters

When optimising the PID parameters for a given system it is important to notice, that setting any of the coefficients to zero will ensure that the term is not used, e.g. if the D coefficient is set to zero the controller will become a PI controller.

Furthermore, PID regulation can be characterised as ‘over damped’, ‘critically damped’ or ‘under damped’ as sketched in Figure 3-24.

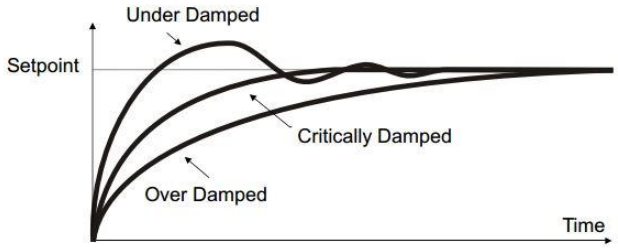


Figure 3-24 PID Damping

- By BMS Creator default settings, the PID controller is overdamped
- To ensure the fastest charge response it is advisable to tune the PID coefficients to achieve a critically damped response.
- Be aware that using underdamped responses will cause the cell voltages to increase beyond the target cell voltage, thus creating a potential unsafe scenario.

Optimising PID parameters is not a simple task. The basic approach recommended for tuning PID parameters, is to make sure that no oscillations occur. To do this a simple procedure can be applied:

- Set: [Kp = 0.05; Ki = 0.0; Kd = 0.0], this should make the PID controller over-damped and will typically result in a reduction in charge current (lower than maximum for most of the charging time, and not only a charge current reduction at higher cell voltages).
- If charge current is still oscillating, reduce Kp further in 0.01 decrements.
- If charge current is not oscillating, set Kp = 0.1. If no oscillations occur increase again by 0.05. It is up to the user to find the correct value, depending on timings in the specific charger setup. A too low Kp will result in the cells never charging completely (unless Ki is used to correct the charge current).
- Once Kp is at the value where oscillation occurs, reduce it slightly and set Ki = 0.02. Increasing Ki further will cause the PID controller to correct the charge current quicker.

Depending on the system setup a few iterations of experiment is required to achieve the optimal PID control parameters.

Please note that:

- An error can occur if the oscillation of the current becomes so large that “the actual” current exceeds the requested current by more than 500 mA.

- Moreover, the risk of exceeding the 500 mA limit increases when the charger resolution approaches 500 mA. Consequently, it is desirable to have charger resolutions well below 500 mA, e.g. at 10 mA or less

3.8 Pre-charge

When the main contactors of a battery pack close and the battery is connected to the load or charger, an in-rush current will flow until the voltage level of the load/charger is equal to the voltage level of the battery. As the resistance in the main battery path can be very low (< <1 Ω), very high in-rush currents may flow in the main battery path.

In most battery systems, the load and chargers have high capacitance at the high-current side facing the battery. These high capacitances (C1 and C2 on Figure 3-25) imply that the high in-rush currents may be present for quite a while until the voltage in C1 or C2 matches the battery pack voltage. High current levels for extended periods of time can damage the system; for example, the connection pads of the main contactor may weld. Welded contactors cannot switch off, which is an unacceptable safety risk in a battery system.

Consequently, special measures are often needed to avoid damage to contactors when these are switched on. Such measures will typically be the use of:

- Dedicated pre-charge functions built into the charger and load
- A dedicated pre-charge circuit in the battery pack

A pre-charge circuit in the battery pack, typically consists of a pre-charge contactor and a pre-charge resistor as shown in Figure 3-25. When the system is to be started up, the following sequence is used:

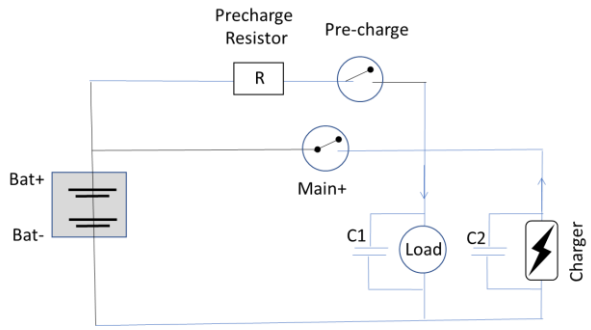


Figure 3-25 Battery system using a pre-charge circuit

1. The pre-charge contactor is switched on while the Main+ contactor is off.
2. The in-rush current flows through the pre-charge circuit, and the current is limited by the pre-charge resistor.
3. After a while, the voltages have equalised, and the main contactor can be switched on safely.
4. Once the main switch is on, the pre-charge switch can be switched off again.

If the load and charger of a given battery system do not provide special pre-charge measures, it is fair to assume that a pre-charge circuit is needed in the battery pack. Selecting the pre-charge resistor is not a trivial task and is beyond the scope of this manual. If you need assistance in designing a pre-charge circuit for a specific battery system, you are very welcome to contact Lithium Balance.

3.8.1 Pre-charge parameter configuration

Configuring the operating parameters of a pre-charge circuit is done in part2 of the “Operational limits” pane of the BMS creator. Here, the four parameters circled in Figure 3-26 need to be configured.

- ”Max precharge end curr.” has to be reached before the main contactor opens
- ”Precharge timeout” is the time allowed for the current to reach the ”Max precharge end curr.” level. If the is not achieved the pre-charge try will be aborted
- If the pre-charge sequence has been aborted it will be ‘re-tried’ the number of times specified by “Max contactor retries”
- The interval between each retry is specified as “Contactor retry timeout”

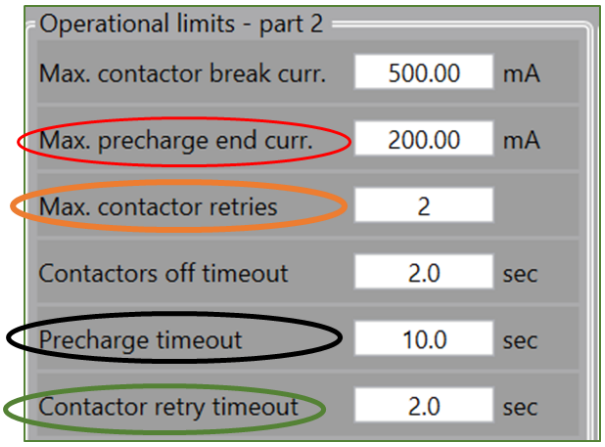


Figure 3-26 Precharge parameters to be configured

These parameters are exemplified in Figure 3-27 below

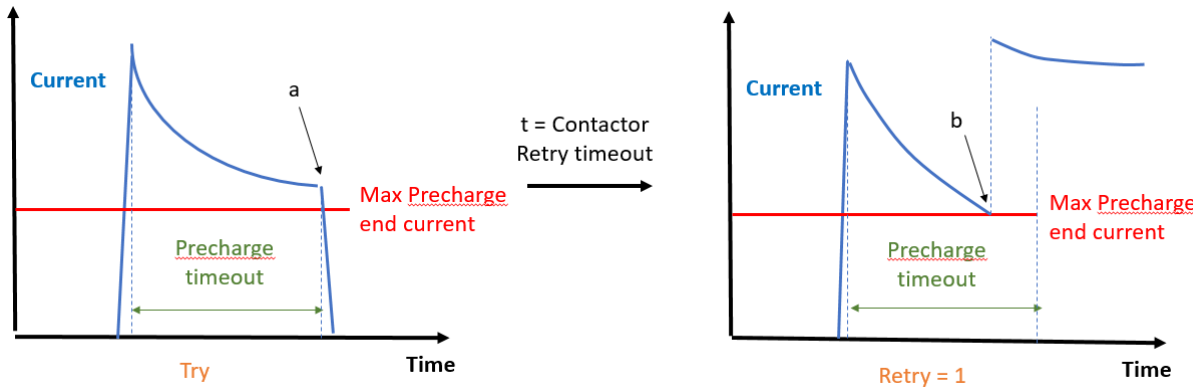


Figure 3-27 In the first pre-charge try, the actual current is above the max end current limit after the pre-charge timeout has expired. Consequently, the pre-charge contactor opens again at point 'a' and the pre-charge is aborted. After the timeout period and new pre-charge attempt (retry 1) starts. Here, the actual current gets below the max end current limit in due time and at point ‘b’ the Main+ contactor is opened.

3.9 I/O settings for state control and contactor configuration

The BMS has a number of I/Os which are essential to the function and state control of the BMS.

Up to 16 I/Os can be used for either input or output. Please note that:

- A special case exists for I/O 4, which is the only I/O that can support PWM output.
- It is advised to use I/Os 9-16 for contactors, as they can supply the most current.”

Configuration of the IOs is performed with the BMS Creator “GPIO Mapping” pane (see section 5.5), where the I/Os are assigned to different functionality. In this pane any functionality that has been configured with the I/O as 0 will not be enabled.

4 CAN communication and set-up

The BMS supports Controller Area Network (CAN) communication in accordance with ISO 11898-2:2003 and ISO 11898-5:2007.

The implementation supports

- CAN 2.0 A (11-bit identifiers)
- CAN 2.0 B (29-bit identifiers).

The BMS is able to run at 4 different CAN speeds: 125 kbps, 250 kbps, 500 kbps, 1000 kbps. Detailed information of how to set up CAN frames (e.g. 11 or 29 bit identifiers) are found in section 4.2.

4.1 Configurable CAN output

The CAN output from the BMS is structured in CAN frames each containing an identifier and up to 8 bytes. The bytes in a CAN frame are specified from left to right, meaning the left most byte is denoted byte 0, while the right most byte is denoted byte 7.

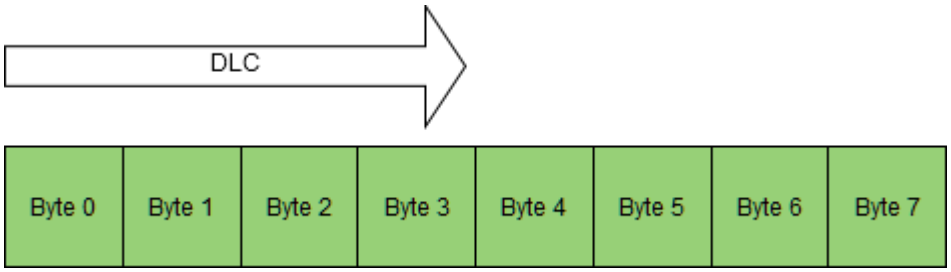


Figure 4-1:CAN frame byte ordering.

The Data Length Code (DLC), denotes the byte wise length of a given frame. The DLC N frame counts from CAN frame byte 0. A DLC of 4, thus means CAN frame byte 0, 1, 2 and 3, as indicated on Figure 4-2

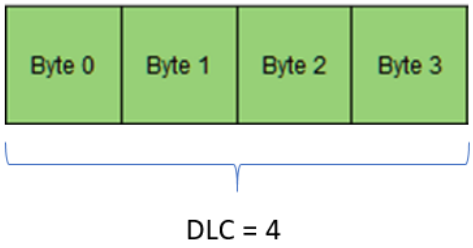


Figure 4-2 CAN data byte order, sketching a Data Length Code (DLC) of 4 bytes.

When desiring to transmit a given parameter from the BMS over the CAN bus, the BMS needs to know where in the CAN frame to place this parameter (e.g. in Byte 2 and 3), the transmit frequency and a number of other settings. This is done in the BMS Creator and is described further in section 4.2.

CAN data is structured in one of two formats known as “Big Endian” or “Little Endian”. Which of these formats to use for a given parameter is typically determined by the format requirements of the other CAN-communicating equipment in the system.

4.1.1 Big endian bit (Motorola) format

In big endian bit ordering mode, or Motorola format, the bit ordering starts from the right most part of the CAN frame, with bit 0 being the right most bit, while bit 63 is the left-most bit.

Figure 4-3 Big endian bit order in CAN frame.

A full big endian CAN frame can be seen in Figure 4-4

Byte 0	63	62	61	60	59	58	57	56
Byte 1	55	54	53	52	51	50	49	48
Byte 2	47	46	45	44	43	42	41	40
Byte 3	39	38	37	36	35	34	33	32
Byte 4	31	30	29	28	27	26	25	24
Byte 5	23	22	21	20	19	18	17	16
Byte 6	15	14	13	12	11	10	9	8
Byte 7	7	6	5	4	3	2	1	0

Figure 4-4 Big endian bit order for all CAN frame bits.

A caveat of using big endian bit ordering mode is that the DLC counts from byte 0 (left most byte), but bits count from byte 7 (right most byte). Thus, for a DLC of 4 (where bytes 0 to 3 are used as shown in Figure 4-2), only bits 32-63 are available.

4.1.2 Little endian (Intel) format

In little endian bit ordering format, or Intel format, the bit ordering is somewhat more complex. The bit ordering within each byte is from right to left, which results in the CAN frame layout seen in Figure 4-5

Figure 4-5 Little endian bit order in CAN frame.

A full little endian CAN frame can be seen in Figure 4-6

Byte 0	7	6	5	4	3	2	1	0
Byte 1	15	14	13	12	11	10	9	8
Byte 2	23	22	21	20	19	18	17	16
Byte 3	31	30	29	28	27	26	25	24
Byte 4	39	38	37	36	35	34	33	32
Byte 5	47	46	45	44	43	42	41	40
Byte 6	55	54	53	52	51	50	49	48
Byte 7	63	62	61	60	59	58	57	56

Figure 4-6 Little endian bit order for all CAN frame bits.

4.2 Setting up CAN frames and data

Each CAN frame to be transmitted by the BMS must be setup individually. Setting up a CAN frame involves

1. Setting up the frame itself
2. Setting up the data to be placed in the frame.

4.2.1 Setting up CAN frames

In the BMS Creator up to 20 CAN frames can be configured. Each available CAN frame has the following overall configurable items:

Item name	Description	Acceptable values
Enable Frame	The CAN frame is enabled if set to 1. If set to 0 it is not enabled	0 and 1.
Update Interval	CAN frames can be updated at a maximum rate of 100 ms. 1: the CAN frame is updated at the maximum rate 2: the CAN frame is updated at half the maximum rate, i.e every 200 ms 3: The 3 CAN frame is updated at one third the maximum frequency, i.e every 300 ms …and so on.	1 to 65535.
DLC	The length of the CAN frame in bytes.	0 to 8.
ID	The CAN frame id as a decimal value.	0 to 536870912 [2^(29) 29 bit ID] 0 to 2048 [2^(11) 11 bit ID]
Is ID Extended	1: the frame_id is a 29-bit extended CAN frame ID. 0: the frame_id is an 11-bit CAN frame ID.	0 and 1.
CAN channel	The CAN channel to output the frame on. 0: S-CAN, the first CAN channel 1: I-CAN, the second CAN channel	0 to 1, depending on the number of available CAN channels on the BMS hardware.

Table 4.1 Overall CAN frame configurable parameters.

4.2.2 Setting up data within a CAN frame

Within each CAN frame up to 10 individual data sets can be configured, implying that each frame can contain up to 10 individual parameters. Each 'data' to be contained in the CAN frame has the following overall configurable items:

Item name	Description	Acceptable values
Config X enabled	The configuration entry number 'X' (for the CAN frame) is enabled if set to 1.	0 and 1.
Config X entry type	The type of the entry data. 0 means constant, 1 means BMS variable.	0 or 1.
Config X start_bit	The bit in the CAN frame, where the first data bit is to be placed	0 to 63.
Config X length	The amount of bits to write data to, starting from start_bit	0 to 32.

Config X is_little_endian	If set to 1, bits are treated as little endian (Intel format). If set to 0, bits are treated as big endian (Motorola format).	0 or 1.
Config X data	Either the constant data or an ID to the BMS variable to place in the CAN frame. Id numbers are found in section 8.4.	0 to 4294967295 (2^(32) – 1)

Table 4.2 CAN frame data configuration parameters.

Note that if data configurations for a CAN frame are set up such that they place data in the same bits of the CAN frame, the result is implementation defined. What this means is that it may work as a user expects (e.g. the top or last data configuration has priority) in a given version, but be different in other versions. ***Do not depend on any given overlap functionality.***

4.3 Custom data processing for CAN

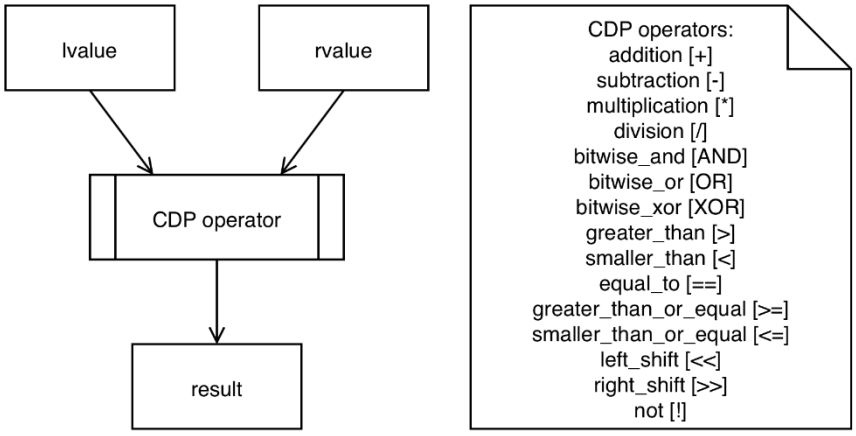
The BMS supports Custom Data Processing (CDP) for up to 25 operations. The CDP is set up using the BMS Creator configuration tool and the processed data is available in the ID MAP with an ID per CDP entry.

The CDP functionality requires 3 inputs:

1. The first value, referred to as the *lvalue* (left value).
2. The second value, referred to as the *rvalue* (right value).
3. The operator to use between *lvalue* and *rvalue*.

The processed output is then calculated as[7]:

CDP_result = lvalue (operator) rvalue



The *lvalue* and *rvalue* can either be a constant value (direct data) or data from the ID MAP (addressed data). Typically the *lvalue* will be data from the ID MAP (addressed) and the *rvalue* will be a constant (direct data). The CDP result from processing is available in the ID MAP using IDs

[7] With the exception of the not [!] operator, this operator applies directly to the lvalue.

ID_CDP_OUTPUT1 to *ID_CDP_OUTPUT25*. Using the configurable CAN functionality it is then possible to broadcast the value on the CAN bus.

It is possible to have multiple calculations with intermediate results, e.g. if scaling and offset is needed for a value. The intermediate results are then used as input for the next CDP calculation[8].

CDP results are processed successively (one after the other) meaning that CDP entry *n* will always be processed before *n+1*. Therefore, if using intermediate results, it would be advisable to configure them in the order needed for the calculations.

For examples on how to use custom data processing, see appendix

4.4 CAN Applications

4.4.1 Wake-up on CAN

The BMS supports remote-wakeup according to ISO 11898-5:2007 which consist of the following pattern:

- a dominant phase of at least 0.5 us, followed by
- a recessive phase of at least 0.5 us, followed by
- a dominant phase of at least 0.5 us.

The complete dominant-recessive-dominant pattern must be received within 0.5 ms to be recognized as a valid wake-up pattern.

As an example, a CAN frame with a Data Length Code (DLC) of 2 bytes with 11-bit ID 0x000 and both data bytes at 0x00, send at 125 kbps to 1000 kbps will wake up the BMS. Here 'x' indicates hexadecimal format.

NOTE: Make sure that the I/O used to enter sleep mode is NOT active when trying to wake the BMS. If the I/O is active the BMS will be stuck in a reset loop until the I/O is no longer active.

4.4.2 CAN charger support

It is possible to enable a built-in CAN charger frame for certain CAN charger types. If one of the built-in CAN frames is used for controlling the external charger, make sure that the format matches the charging equipment. The built-in CAN charger frames have fixed timings and this is not configurable. If other timings are required a custom CAN frame must be configured.

4.4.2.1 EA-PS8200-70 from Elektro-Automatik,

This format includes a special state machine that is only supported on the specified model.

4.4.2.2 Custom big endian format 1

The frame format is as specified in the figure below.

[8] The intermediate results are stored as regular CDP entries and can be accessed using IDs *ID_CDP_OUTPUT1* to *ID_CDP_OUTPUT25.*

CAN Frame	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Content	Charge enable	100	0	Maximum charge voltage		Requested charge current		0
Description	*1 = charge allowed 0 = charge forbidden*	*Constant*	*Constant*	*0.1 V resolution*		*0.1 A resolution*		*Constant*

4.4.2.3 Custom big endian format 2

The frame format is as specified in the figure below.

CAN Frame	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Content	Charge enable	100	0	Maximum charge voltage		Requested charge current		0
Description	*See byte 0 specification figure*	*Constant*	*Constant*	*0.1 V resolution*		*0.1 A resolution*		*Constant*

Byte 0 uses multiple bits for different purposes as shown in the figure below.

Byte 0	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Content	0	0	0	0	0	0	Clear error flag	Charge enable
Description	*Constant*	*Constant*	*Constant*	*Constant*	*Constant*	*Constant*	*Flag that starts out as 0, becomes 1 after 12 CAN frame transmissions, becomes 0 again after 12 more CAN frame transmissions (24 in total), then remains 0 until BMS is rebooted.*	*1 = charge allowed 0 = charge forbidden*

4.4.2.4 Custom big endian format 3

The frame format is as specified in the figures below. The output in the frame switches between two formats continuously throughout frame broadcasting. The first format outputs the requested charge voltage and the second the requested charge current.

Format 1:

Can Frame	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Content	0x23	0x01	0x24	0x1	Requested charge voltage			
Description	*Constant*	*Constant*	*Constant*	*Constant*	*0.001 V resolution*			

Format 2:

Can Frame	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Content	0x23	0x01	0x24	0x2	Requested charge current			
Description	*Constant*	*Constant*	*Constant*	*Constant*	*0.001 A resolution*			

5 BMS Creator

To configure and service the BMS, the Lithium Balance PC tool, BMS CREATOR (TM) is provided [9].

Note that the BMS Creator only supports CAN adapters from PEAK SYSTEMS available from www.peak-system.com. Recommended models are

- PCAN-USB IPEH-002022 or
- PCAN-USB IPEH-002021

5.1 Requirements

- Windows 10, Windows 8, Windows 7 or Windows Vista
- .NET 4.5 framework
- PEAK Systems PCAN USB driver
- Full read/write rights for the installation directory of the BMS Creator to use the 'Use default save location' when generating a configuration, and the 'Use latest generated' feature when uploading BMS configurations.
- 120 Ω termination at the 'far-end' of the CAN bus, please see hardware setup section for details.

5.2 Installation of the BMS Creator Software

1. Run the installation file supplied by Lithium Balance.
2. The installation Wizard will guide the user through the installation process.
3. At the end of the installing process, await confirmation that the installation was completed (this normally takes several minutes).
4. Run the installed software, e.g. by clicking on the newly installed icon on your desktop.
5. A Site Key needs to be entered to active the software. Copy the Site Code and send it to activation@lithiumbalance.com
6. A Site Key will be generated for the software by Lithium Balance.
7. The user will receive an e-mail including a Site Key specifically generated for the user PC.
8. When the user receives the Site Key, it should be entered into the site key field, in order to validate the software license.
9. The software is now unlocked and ready to use.

5.3 Connecting the PC to the BMS

The CAN Adapter dropdown box will display available CAN adapters to use for communication with the BMS. If the list is empty, the device(s) is in use by another program or not available. After making the CAN adapter available, click the refresh button next to the CAN Hardware dropdown box to refresh the CAN Adapter list.

[9] The BMS configuration interface is based on the ISO 14229-1 Unified Diagnostic Services (UDS) protocol so other tools or devices that support this standard can in principle be used to interact with the BMS.

As default, the Auto baud rate detection feature is enabled. This feature will automatically detect the BMS baud rate and use this when communicating with the BMS. If the baud rates are known, this feature can be switched off for speed optimization.

The BMS UDS services is only available on the s-CAN bus and hence the CAN adapter should be connected to this bus only.

When CAN speeds and CAN Channel have been configured correctly, click the Connect button to initiate a connection sequence. If no BMS is available the BMS Creator will indicate this with a pop-up box indicating that connection failed.

When connected, the BMS firmware version text field (FW) will contain the firmware version of the connected BMS. The firmware version displayed will be that of the connected state. In Boot load state the version of the Boot loader firmware will be displayed and in Application state the version of the Application firmware.

The BMS Mode status textbox will display the state of the BMS at the point of connection. Note that the BMS state only updates when the connect button, 'Upload BMS configuration', 'Reset BMS', 'Boot load BMS' or 'Read error log' button is pressed.

The BMS connection indication will display a small white check on green background when a connection to the BMS has been established. Like Mode status, the indication is only updated when a button is pressed. If there is no connection, a X on a red background will be displayed. When actively reading data from the BMS, a green process ring is visible around the connection indication.

5.4 Boot loading the BMS

The service/UPGRADE page contains the interface to the BMS Boot load feature. Click the Boot load button to activate a boot load sequence to update the application firmware. The BMS creator will automatically switch the BMS into boot load mode and back after finished upload process.

The BMS Creator will prompt for a file to upload to the BMS if the 'Use built-in' checkbox is not checked. The file format is *.bin and is supplied by Lithium Balance. In a normal Upgrade situation, the built-in should be used by checking the 'Use built-in' checkbox. A progress bar will show the upload progress below the buttons.

When the boot loading sequence is done there will be a small white check on green background indication next to the progress bar. If not, the boot loading sequence has failed and a pop up box with the error code will be displayed. Please note, that the tool will indicate success even though the connected Mode is Boot, and the expectation would be App. This could be because there is no correct and matching configuration on the BMS. In this case continue to the next step of writing configurations.

5.5 Configuration of the BMS

The configuration/MCU page contains the interface to create a configuration for the BMS. The configuration file (*.bin) is always created from a *.XML file and there for the first step

is to open a XML file. If no file exists, one can be created by pressing the 'New' button. If one already exists, press the 'Open' button, and a file dialog will allow selection of already existing file. When done changing the XML file, the 'Save' button can be used to open a dialog of where to save the file.

Change all relevant configuration parameters to match the application the BMS is to be installed. All configuration options are displayed scaled and with a unit next to the value textbox. When all necessary changes are done, press the 'Generate BMS configuration' button. This will generate a *.bin BMS configuration file. This file can be used to send to a BMS service users, who should not change anything in the configuration, but only upload it onto a system. When using the tool to change configurations and uploading them in one workflow, it is very useful to have the 'Use default save location' checked. In this way the user will not be prompted for a file location every time a small configuration change is made. After generation of the configuration file, a small indication is given on the right of the 'Generate BMS configuration' button.

Please refer to the Manual on configuration parameters for an overview and short explanation of the different parameters to be configured.

Please note that this only provides checks of input data which cannot be read by the BMS creator tool. At the moment there is no check on input data which might be meaningless from a BMS point of view, such as configuring a given function for the non-existing IO number 17 or configuring to functions to the same IO

Please, always evaluate the Errors tabs for correct BMS action in case of any application critical error conditions.

5.5.1 IO Configuration

Configuration of the IOs is performed with the BMS Creator “GPIO Mapping” configuration section, where the I/O’s are assigned to different functionality. In this section any functionality that has been configured with the I/O as 0 will not be enabled. One example of configuration could be:

The 4 digital inputs assigned to:

- Activating load state (IO1)
- Activating charge state (IO2)
- Activating balancing modes (IO3)
- Feedback from a contactor (IO4)

The digital input/outputs are used for:

- Load positive contactor control (IO5)
- Load negative contactor control (IO6)
- Charge negative contactor control (IO7)
- PWM output (IO 8) for charger control

In this example, there is only feedback from the load-positive contactor. Consequently, all other feedback mappings are set to 0

5.5.2 Upload configuration to BMS

The last step of the BMS configuration process is to navigate to the service/UPGRADE page and press the 'Upload BMS configuration'. To use the last generated configuration, be sure to keep the 'Use latest generated' checkbox checked. This will then not prompt for a file location, but directly use the default location which is in the tool install folder. Note that this will only be valuable if the 'Use default save location' was checked during the last configuration generation.

The BMS Creator will automatically switch the BMS from Application mode to Configuration mode, upload the configurations, and switch the mode back to Application.

If the upload succeeded but the BMS is still in Boot mode, it could be because the configuration does not match the firmware, or the configuration values are not valid. In this case the BMS will stay in configuration (Boot) mode.

While the tool is uploading the configurations, a small process ring next the progress bar will indicate activity. Most of the elapsed time associated with this process ring being active but no activity on the progress bar, is while the tool is switching the BMS between App mode and Boot mode or waiting for the hardware to restart.

5.6 Error Log Readout

The service/ERROR LOG page contains functionality to read out a history of logged error in the system. The feature is only available in Application mode and is activated by pressing the 'Read Error Log' button. The tool will read out the log structure and display them in the page. If the user requires the log structure to be saved, the 'Save Error Log' button should be pressed. This will prompt the user for a file location for the CSV file to be saved for later analysis. A successful readout/save of the error log is indicated by a check. During operation a process ring will be visible.

5.7 Live view

The live view pages contain functionality to monitor BMS data. The DASHBOARD page gives an overall indication of system status and displays the system current in a graph view. The MCU DATA page contains information regarding the master controller like IO states, alarms, internal measurements and control data. The CMU DATA page contains data regarding the cell measurement front-end like cell voltages and temperatures. The LOGGING page allows the user to start and stop a log of all live view data to a csv file. The update rate is fixed to 1 [s] interval. When pressing the 'start log' button, the user is prompted for a file save location and name.

5.8 Troubleshooting

- If a communication error occurs, it is likely because the Peak CAN adaptor was not found by the BMS creator tool or is in use by another application. If the "CAN Adapter" field is empty, this is usually the reason. Close any programs that is connected to the CAN adapter or disconnect from the CAN adapter and click the refresh button to refresh the CAN adapter list.

- The LED on the BMS board will flash while in boot loading mode. This information can be useful to help debug communication problems.

5.9 CAN error frames

In the “CAN Settings” section one the configuration page of the BMS Creator, it is possible to enable a setting (“CAN ID start error frames”).

- A setting of 0 will disable CAN error frames
- Any other setting will start CAN error frames output from the specified CAN frame onwards.

The output of error frames from the firmware onto CAN will start at the “CAN ID start error frames”. Note that the specified CAN frame is in decimal, not hexadecimal notation.

It is possible to denote the CAN frame ID as extended (“CAN ID error extended?”) and to specify which CAN channel to output to (“CAN channel error frames”).

5.9.1 CAN output

If enabled, the CAN output will consist of the contents displayed in Figure 5-1 and explained in tables Table 5.1 and Table 5.2. The data is in big endian data format, please see the section discussing configurable CAN for more information.

Frame	Byte 7	Byte 6	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0
X	# 1	# 2	# 3	<Unused>	# 4		<Unused>	<Unused>
X +1	# 5				<Unused>			
X + 2	#7		# 8		# 9	# 10	# 11	
X + 3	#7		# 8		# 9	# 10	# 11	
...					...			
X + 255	#7		# 8		# 9	# 10	# 11	
x + 256	#7		# 8		# 9	# 10	# 11	

Figure 5-1: CAN frame output

CAN frame	Description
X (the value chosen in the BMS creator field “CAN ID start error frames”)	Overview of the number of active errors and their severity
X + 1	Error statistics (sum of errors since boot)
X + 2 to X + 255	Information about error locations 1 to 254[10]
X + 256	Most recently reported error[11]

Table 5.1: Description of error CAN frames

ID	Description
1	Total number of active errors with critical severity
2	Total number of active errors with normal severity
3	Total number of active errors with low severity
4	Total number of all active errors (across all the severities)
5	Total number of errors (active and otherwise) since boot
7	Internal Lithium Balance A/S variable used for verification.
8	Internal Lithium Balance A/S variable used for verification.
9	Severity of the error, please see error handling section for more information.
10	Internal Lithium Balance A/S variable used for verification.
11	The error code, please see section 8.3.3

Table 5.2: Description of CAN error frame content

[10] Errors are generally reported in the first “free” slot of the available error locations. There is no fixed position for a specific error.

[11] This error will also exist in one of the error locations. This frame is meant as an easy way to see the latest activated error.

5.9.2 CAN Configuration set-up example

To exemplify how to set-up the CAN configuration of the BMS, the following example shows the procedure for how to add **cell voltage 1 to 4** measured on the battery pack to the CAN frames transmitted by the BMS. For information, this example matches the Busmaster set-up example of section 8.8

5.9.2.1 Step 1: Connect BMS creator

5.9.2.2 Step 2: Set up the CAN frame

- Enable frame 1
- Set update interval to 100 ms
- 8 bytes in the frame
- Frame ID set to 200, could be any other unused number
- 11 bit frame

5.9.2.3 Step3 Configure CAN data for cell voltage 1

- Enable 1st parameter in a given frame
- Entry type 1 for 'BMS variable' data
- Start bit 48 (equals end bit in byte 1 in big endian format, see section 4.1.1)
- 16 bits (UINT16) for the given parameter, cell voltage 1, (see ID 540 in section 8.4 for a list of parameter formats.)
- 0 for big endian
- Data ID = 540 for Cell Voltage1 (CMU0 and V0)

5.9.2.4 Step4 Configure data for cell voltage 2 to 4

Repeat step 3 for cell voltage 2 to 4 (Data ID 541 to 543 as listed in section 8.4) moving two bytes = 16 bits 'down' for each new entry

Byte 0	63	62	61	60	59	58	57	56
Byte 1	55	54	53	52	51	50	49	48
Byte 2	47	46	45	44	43	42	41	40
Byte 3	39	38	37	36	35	34	33	32
Byte 4	31	30	29	28	27	26	25	24
Byte 5	23	22	21	20	19	18	17	16
Byte 6	15	14	13	12	11	10	9	8
Byte 7	7	6	5	4	3	2	1	0

ID 540 Cell Voltage 1
ID 541 Cell Voltage 2
ID 542 Cell Voltage 3
ID 540 Cell Voltage 4

Figure 5-2 Allocation of start bits for 2-byte data when using the big endian bit order

Figure 5-3 CAN setting for cell voltage 2 to 4

5.9.2.5 Step 5: Upload configuration

Once the new CAN set-up has been completed, it should be uploaded to the BMS. Press 'Upload BMS configuration' on the service

1. The program will jump to 'boot mode', upload the configurations to the BMS in Configuration mode and progress bar will indicate upload progress during the operation.
2. When writing is completed, the system will jump back to application mode automatically.

6 Battery system and auxiliary components

To provide the desired functions and ensure safe battery system operation, the BMS must be well integrated with the other components in the battery system. An example of a typical battery system is shown in Figure 6-1 below.

1. Cell voltages and temperatures are monitored directly by the BMS together with the battery current. Please note that if a shunt is used for current monitoring, the shunt must be placed at the negative side of the battery as shown in Figure 6-1.
2. The BMS communicates via the CAN bus with the charger to reduce or stop charging when the desired cell voltages are approached.
3. The BMS communicates via the CAN bus with the load (typically via a Vehicle Control Unit – VCU) to reduce the discharge current if the discharge current exceeds pre-defined limits.
4. Switches are included in the main battery power circuit, between the battery terminals and the load/charger. For safety reasons, at least one switch is MANDATORY to assure that overcharging or discharging can be stopped even if BMS communication (item 2 and 3 in this list) fails.
5. A dedicated soft start or pre-charge switch and pre-charge resistor can be used to reduce potentially damaging current transients during start-up.
6. A fuse in the main battery power circuit is MANDATORY to cut excessive (typically discharge) currents caused by e.g. an external short-circuit or a load requiring too much current for too long.
7. A PC needs to be connected to the BMS during setup for writing configurations. A dedicated CAN adapter (see section 2.3.6 is necessary between the BMS and the PC to utilise the Lithium Balance configuration tool.

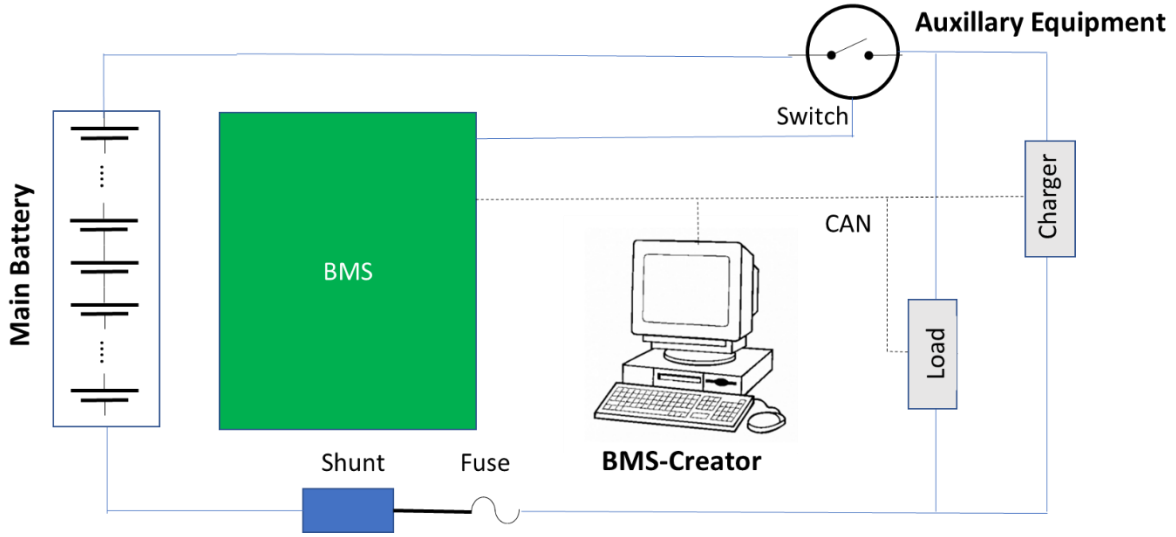


Figure 6-1 Example of a basic battery configuration and the key auxiliary components used

Choosing the right auxiliary components for a battery system can be a complicated task which is beyond the scope of this manual. If you need assistance, please contact Lithium Balance; we may be able to help you directly or can refer you to experts in different areas.

The following section describes the necessary considerations to ensure well-functioning interfaces between the BMS and auxiliary components.

6.1 Battery

Lithium ion batteries are available with different chemistries[12] such as LCO, LMO, NMC, LFP, NCA and LTO. All of these can be managed by the Lithium Balance BMS system. Different chemistries have different operating cell voltages, and these differences have to be taken into account when configuring the BMS. Furthermore, different cell packages and chemistries will vary significantly with regard to maximum sustainable currents and possibly also safe operating temperature range.

It is essential that the selected safety thresholds for cell voltage, temperature and current are based on the requirements of the specific application and on the specific cell information provided by the cell manufacturer, e.g. in datasheets.

In many cases, the battery supplier may specify the maximum value of the battery charge/discharge current in terms of a 'C-rate' rather than an ampere value. The 'C' (Coulomb) rate reflects "the 'inverse' minimum time in hours for a full charge or discharge". If for example a 200 Ah battery is specified for up to 3C discharge and 2C charge operation, then the

- minimum charge time is 1/2[C] hour = 30 min
- minimum discharge time is 1/3[C] hour = 20 min
- maximum charge current is 200 Ah/30 min = 200 A*2[C] = 400 A
- maximum discharge current is 200 Ah/20 min = 200 A*3[C] = 600 A

6.2 Load

In general, the BMS has no direct control over the load. The BMS needs to be able to communicate via CAN with the central system processor, e.g. the Vehicle Control Unit (VCU). The BMS can then ask the central processor to adjust the load, e.g. to reduce the current if it exceeds the maximum level.

The maximum discharge current (Over Current OUT) threshold can be configured in the BMS Creator. The user can then select how the switches in the system should react if this level is exceeded. Please refer to the BMS Creator manual for threshold setting and switch configuration. The BMS Creator manual also includes detailed information on setting up CAN communication with a VCU.

6.3 Sensors

While cell and battery voltages can be measured directly, sensors are needed to measure current and temperature.

6.3.1 Shunts for current measurements

The battery current (I) can be measured by the voltage drop (ΔV) over a shunt resistor with a known resistance (R), i.e. $\Delta V = R * I$.

Please note that in the BMS, the shunt detection circuits are internally referenced to the most negative battery voltage (HV-). Consequently, the shunt MUST always be placed on the negative

[12] See for example http://batteryuniversity.com/learn/article/types_of_lithium_ion

side of the battery pack as shown in Figure 6-1. Otherwise, the BMS may be irreparably damaged.

To provide an accurate measurement, the shunt resistance must be well defined and have a low temperature coefficient. High quality shunt resistors are available with resistance variations below 0.5% at room temperature. Typical temperature coefficients are 0.002%/K.

The accuracy of the shunt resistance will affect the accuracy of the SoC calculations, and it is therefore recommended to use high quality shunts. Typically, an inaccuracy of e.g. 0.5% of the shunt resistance can lead to inaccuracies of up to 0.5% for the SoC values calculated by the BMS.

Shunt resistors have very low Ohmic resistances in the μΩ range. Low resistance reduces the risk of overheating the shunt resistor and reduces the shunt related power loss (P) in the system where $P = R \times I^2$.

The BMS shunt measurement circuit can measure voltages between -250 and +250 mV. When selecting the resistance of a shunt, it is important not to exceed these limits for all specified operating currents in the system. At the same time, it is recommended to use a significant part of the voltage dynamic range in order to obtain a good current resolution.

Lithium Balance can provide shunt resistors for

- nominal currents up to 500 A
- low resistance variations of ≤ ±0.25% at 25 °C
- operating temperatures between -40 and +60 °C
- electrical isolation up to 750 V
- temperature coefficients of 0.002%/K

Please refer to section 8.1.1 for more information.

Figure 6-2 Example of shunt resistor

6.3.2 Hall effect sensors for current measurements

A Hall effect sensor is a transducer that varies its output voltage in response to a magnetic field. When this magnetic field is generated by the battery current, the Hall effect sensor can be used for measuring the main current (I).

The BMS supports Hall sensors powered 5V and providing up to two isolated outputs at voltages between 0 and 5V. Here 2.5V corresponds to 0A. Lithium Balance recommends the use of the DHAB S/34 Hall effect sensors from LEM. These can be provided by Lithium Balance (ordering code 000637)

6.3.3 Temperature sensors

The Lithium Balance BMS is designed for temperature sensing, based on negative temperature coefficient (NTC) thermistors. NTC thermistors are non-linear resistors where the resistance decreases as the temperature increases.

The NTC temperature dependence is typically characterised by the parameter 'β', where

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{\beta}\ln(\frac{R}{R_0})$$

Here, R_0 is the reference resistance (typically 25 °C = 298.15 K) and R is the measured resistance at the temperature T. All temperatures including β are measured in K.
Please refer to section 8.1.3 for more information.

6.4 Contactors

Contactors are essential safety mechanisms that provide a redundant means for shutting down the current to or from the battery, if a safety critical event is about to happen. Contactors are used for several different purposes in battery systems. Typical examples are:

- To switch the load in or out
- To switch the charger in and out
- To ensure that the battery in the off-state is galvanically disconnected from the external battery pack terminals
- For precharge/soft-start, as described in section 3.8

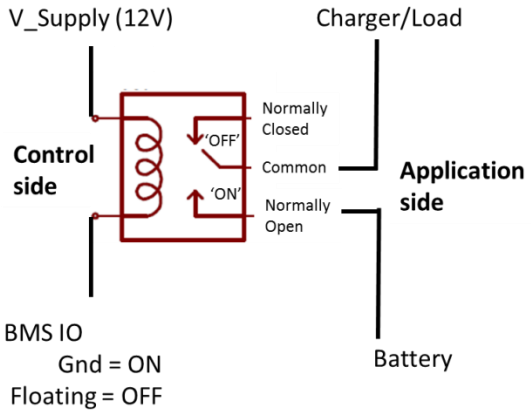


Figure 6-3 Typical configuration of a contactor

In a contactor, an electromagnet on the 'control side' is used to switch a mechanical contact on the 'application side' between an ON and an OFF position.

The current needed to drive the electromagnet in the contactor is typically provided by coupling the 12 V external power source to ground via the contactor control terminals to the BMS IO as shown in Figure 6-3

To reduce the steady state power consumption of the contactor, a so-called 'coil economiser' is often included. The coil economiser will reduce the steady state current, but will often also introduce a large capacitance which can lead to significant in-rush currents when the contactor is switched on. This in-rush current level is important to consider, if the supply voltage to the contactor is taken from current limited sources. Typically, a battery can easily provide the required in-rush current, where a DC/DC converter must be selected carefully to be able to handle the in-rush current.

When a contactor is switched off and the current in the magnetic coil on the control side suddenly drops, a voltage peak may appear, since a rapid current change in an inductor will introduce a significant voltage, i.e. V = L x dI/dt[13].
To prevent voltage (V) induced damage to the BMS IO port when the contactor is turned off, it is mandatory to use a so-called flyback diode in parallel with the contactor electromagnet as shown in Figure 6-4. Some contactors, such

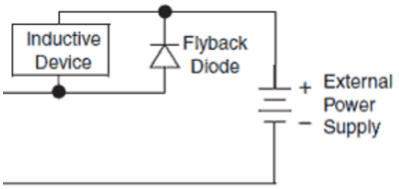


Figure 6-4 Flyback diode

[13] V is the induced voltage, L is the inductance and dI/dt is the change in current

as those supplied by Lithium Balance come with built-in flyback diodes.

6.5 Fuel gauges

In some systems, it is desired to be able to display the SoC value on a dedicated display, here called a fuel gauge. Dedicated fuel gauge displays are typically used in retrofit systems where the Li-ion battery pack replaces another power source in an already developed application. Examples could be:

- vehicles which are converted from combustion engines to Li-ion powered electrical vehicles
- electrical vehicles which are converted from using lead-acid batteries to using Li-ion batteries

The BMS can interface with SoC fuel gauges via the CAN bus. Lithium Balance recommends the use of CAN-based fuel gauges and can provide these (ordering code 100538).

Please note that the full functionality of the 100538 and possibly other CAN-based fuel gauges are not supported by the BMS rel. 2.0 software

6.6 DC/DC converter

DC/DC converters are often used in (low-voltage) battery systems where a 12 V power source such as a lead acid battery is not available.

In this case, the 12 V power to the BMS can be provided by a DC/DC converter fed from the main battery as shown in Figure 6-5.

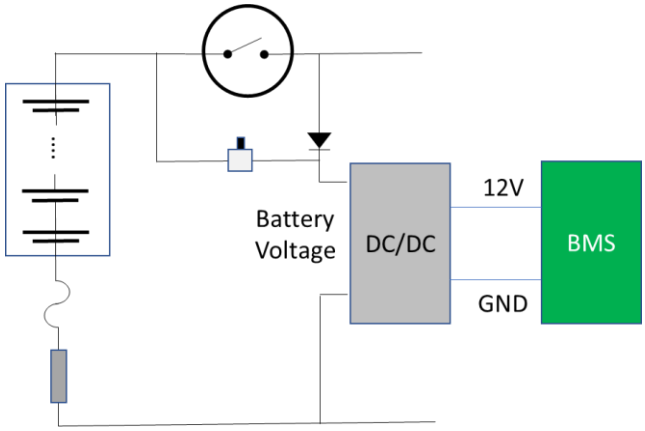


Figure 6-5 Battery pack where the BMS is powered from the main battery via a DC/DC converter

When selecting a DC/DC converter for a battery pack system, a number of key aspects are important to consider:

- The battery power to the DC/DC converter should be taken after the **contactor**. Otherwise, the DC/DC converter may drain the battery when the system is off. To start-up the system again after the contactor has been opened, a spring loaded manual push-button switch (in combination with a diode) can be included as shown in Figure 6-5
- If the BMS and the rest of the battery system needs to be galvanically isolated from the battery, a galvanically isolated DC/DC converter must be used.
- If 12 V powered switches are used in the system, the DC/DC converter must be able to provide sufficient current both for the steady state operation of the switch AND the

contactor in-rush current. Alternatively, a contactor powered directly from the battery pack can be used. However, high voltage contactors may compromise a desired galvanic isolation

7 BMS Quick Installation Guide

The purpose of the "Quick Installation Guide" is – with reference to the manual - to provide an overview of the steps recommended for the hardware installation and the use of the BMS based on configuration via the BMS Creator and monitoring the performance via the Bus-master.

Safety: Please read the safety instructions before installation as described in section 1.3 of the manual.

7.1 Preparation

The following components are ideally required for the full system installation:

- Mandatory for basic setup
  - BMS boards
  - BMS cable kits
  - BMS creator on an PC running Windows operating system
  - Peak PCAN CAN Adapter
  - 200 mA quick blow fuse (for the 12V power supply line)
- Mandatory for systems including batteries
  - Battery Pack (or simulator)
  - Shunt or Hall sensor
  - Contactor(s)
  - Main battery fuse (battery short circuit)
- Optional as per system architecture
  - Charge contactor
  - Pre-charge contactor and pre-charge resistor
- External interfaces
  - Charger
  - Load
- Support equipment and consumables
  - Termination Resistors for CAN (120 ohms)
  - Wire Stripper

7.2 Typical configuration

An example of a typical n-BMS configuration is shown in Figure 7-1 below:

- The MCU board is connected to PWR, CAN, Contactors and Aux temperatures via the two IO cables 000807 and 000808
- The MCU is connected to the High Voltage sections of the battery pack via the 000847 cable
- The CMUs measure cell voltages over the 000809 cable
- The CMUs measure cell temperatures over the 000801 cable
- The BMS communicates with the charger via CAN
- The BMS communicates with a PC on CAN which is adapted to USB in the PEAK adaptor
    - Configuration of the BMS is done via the 'BMS creator' software licensed from Lithium Balance
    - Reading actual operational parameters (e.g actual cell voltages) are done via a CAN bus monitor tool.

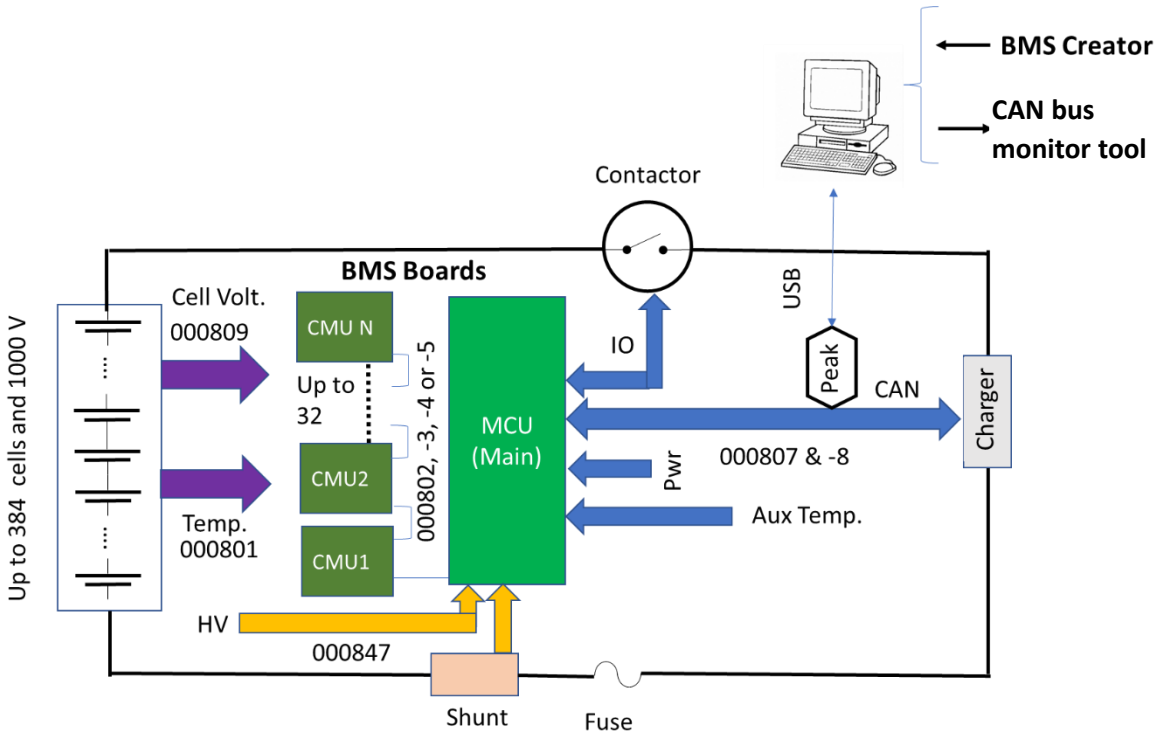


Figure 7-1 Typical n-BMS configuration

7.3 Connect Power and PC to the BMS

The first recommend steps in the BMS installation is to power-up the BMS and connect the BMS to the PC tools, i.e BMS Creator and Busmaster:

1. Connect the board main power (nominally 12V) to the BMS board (for connections see Table 2.1 / Table 2.2). ***After successful installation, the red LED will blink once***
2. Connect the s-CAN bus to the PC via the Peak CAN adaptor
3. Install the BMS Creator Software as described in section 5.2 and connect the PC to the BMS as described in section 5.3.

   a. Verify that the BMS Creator communicates with the BMS by verifying that clicking connect will change BMS Creator status to “CONNECTED”.
4. Install a CAN bus monitor tool e.g. Busmaster as described in section 8.8
   a. Once installed, load the CAN database file (*.dbc) or (*.dbf) provided by Lithium Balance and connect to the BMS.
   b. The CAN database file contains translations of data sent from the BMS, and the file provided by Lithium Balance contains some examples that matches a default configuration.

7.3.1 Handling CAN data base configuration files (*.dbc or *.dbf)

To read the data transmitted form the BMS on the CAN bus, Lithium Balance provides standard database set-up (*.dbc or *.dbf) files for data reading programs such as BusMaster. The database files contain the information of the CAN IDs, data formats, location of the useful data etc. in a way that BusMaster or other programs can use to interpret CAN messages and to convert and monitor them as physical values. This physical value can be a cell voltage in mV, or pack current information in amperes etc.

The BMS creator standard set-up *.XML file (DeviseSettingsMap_xx.XML) provided by Lithium Balance has default CAN settings enabling for broadcasting of some BMS variables via CAN. The CAN-reading data base files (*.dbc and *.dbf) provided by Lithium Balance are created to match this standard *.XML file such that all parameters transmitted on the CAN bus by the BMS using the default standard DeviseSettingsMap_xx.XML file can be read when using the default *.dbc or *.dbf files.

It is possible for the user to change CAN settings in BMS creator e.g. add to a new variable in a CAN frame, change the orientation of the variable, change the location of the variable within a CAN frame, delete a CAN frame, or change CAN ID. Examples of this is given in section 5.9.2

When any change is done in the BMS Creator CAN setting, please remember also to update the CAN reading database (*.dbc or *.dbf) files accordingly. Examples of how to do this for BusMaster files are given in section 8.8.

Please note that, the CAN (reading) data base (*.dbf) of (*.dbc) files prepared for version 2.0 are only compatible with BMS version 2.0 default settings. Do not use it for example for previous BMS software versions such as v1.1.

7.4 Connect the BMS to the battery

Step	Connect	Verify
2.1	Connect battery cell voltages to the CMUs as described in section 2.2.5. Start from the bottom and connect Cell1- on the first CMU to the lowest voltage on the battery. When adding a new CMU (e.g. CMU4), then connect this to the previous CMU (e.g CMU3) via the isoSPI conenctions as decribed in section 2.5 When all cell voltages have been connected • Configure Voltage Channels (Figure 3-3) • Configure Cell voltage thresholds (Figure 3-8)	Verify on CAN output that all cell voltages are found.

2.2	Connect CMU temperature sensors to the cells, see section 2.2.6. When all cell temperatures have been connected • Configure temperature channels (Figure 3-2) • Configure temperature thresholds (**Figure 3-7**)	Verify on CAN output that all temperatures are found.
2.3	Connect the current sensing device Shunt or Hall sensor (J2 and 100931) see section 2.3.3 or 2.3.4 • Configure Shunt/Hall settings, (Figure 3-1) • Configure Current thresholds in the BMS Creator (section 5)	When the BMS is operating for the first time verify that the current measured by the BMS corresponds to the actual current in the application.

7.5 Connect the digital Input and Output signal to contactors and control signals

3.1	Connect output signals to contactors • Configure Contactors	Pending first activation of active mode
3.2	Connect digital input control signals • Configure input signals	Pending first activation of active mode

7.6 Connect charger and load

4.1	Connect Charger • Configure Charger communication (CAN or PWM) • Optimise PID settings for 'request current'	Consult charger manufacturer for CAN bus configuration parameters like Current and Voltage request location in frame. Frame ID etc.
4.2	Connect load	Test system and verify current measurement (Hall or Shunt measurement). Remember Current In = Charging = positive current.

7.7 Test and verification

Before the BMS is put into operation, the operation of the BMS within the battery system must be validated under supervision. Specifically, it is recommended to actively verify the key safety aspects listed below during a supervised full charging and balancing of the battery:

- Safe operating limits of voltage, current and temperature.
- The ability of the system to reduce or stop charging when the maximum cell voltage has been reached for any cell.
- A robust and well-tested communication between the BMS and the charger.
- Well-functioning contactors which can disconnect the charger and load from the battery in case direct communication between the BMS and these units fail.

Furthermore, please note that the ability to block excessive current from the battery (preferably by using a fuse) is essential to the safe operation and handling of batteries.

8 Appendices

8.1 Appendix 1: Parts and ordering codes

8.1.1 Shunts

Part number	Description	Comment
100682	200 A/50 mV	Compact
100683	300 A/50 mV	Compact
100684	500 A/50 mV	Compact
100452	150 A/150 mV	
100399	300 A/100 mV	
100400	150 A/150 mV	
100402	500 A/100 mV	

8.1.2 Hall effect Sensors

Part number	Description	Comment
000637	DHAB S/34 Hall effect sensors	Compact

8.1.3 Thermistors

These thermistors can be ordered separately. One is part of the 100930 cable accessory

R_0 = 10 kΩ at 25 °C and β = 3900 K. R_0 tolerance = 5%

Part number	Description	Comment
000510.1	150 mm wire	Contains Vishay thermistor NTCLE100E3103JB0

A 5% tolerance corresponds to a potential temperature errors around 1 °C, i.e. 0.75 °C at -20 °C, 1.0 °C at 20 °C and 1.7 °C at 100 °C

8.1.4 Contactors

Part number	Description	Comment
000014	500 A	Main switch
000015	100 A	Typical for pre-charge

8.1.5 Fuse

Part number	Description	Comment
000560	500 mA quick blow	For 12 V power fuse

8.1.6 Pre-charge resistors

Part number	Description	Comment
000018	47 Ω	
000016	100 Ω	
000389	150 Ω	

8.1.7 Fuel gauge

Part number	Description	Comment
100538	CAN based	

Please note that the full functionality of these CAN based fuel gauges is not supported by the present BMS software version

8.1.8 DC/DC converters

Part number	Description	Comment
000433	9-36 V input	12 V and 2.5 A output
000489	18-75 V input	12 V and 2.5 A output

8.1.9 Cables and accessories

Part number	Description	Comment
000807	MCU J4a IO cable	100 cm long
000808	MCU J4b IO cable	100 cm long
000847	MCU J5 Shunt & HV	50 cm long
000802	ISOSPI, 12 cm long	Shielded cable
000803	ISOSPI, 30 cm long	Shielded cable
000804	ISOSPI, 100 cm long	Shielded cable
000805	ISOSPI, 500 cm long	Shielded cable
000809,	CMU J4 Cell Voltage	
000801	CMU J3 Cell Temp	

100802: Consists of:

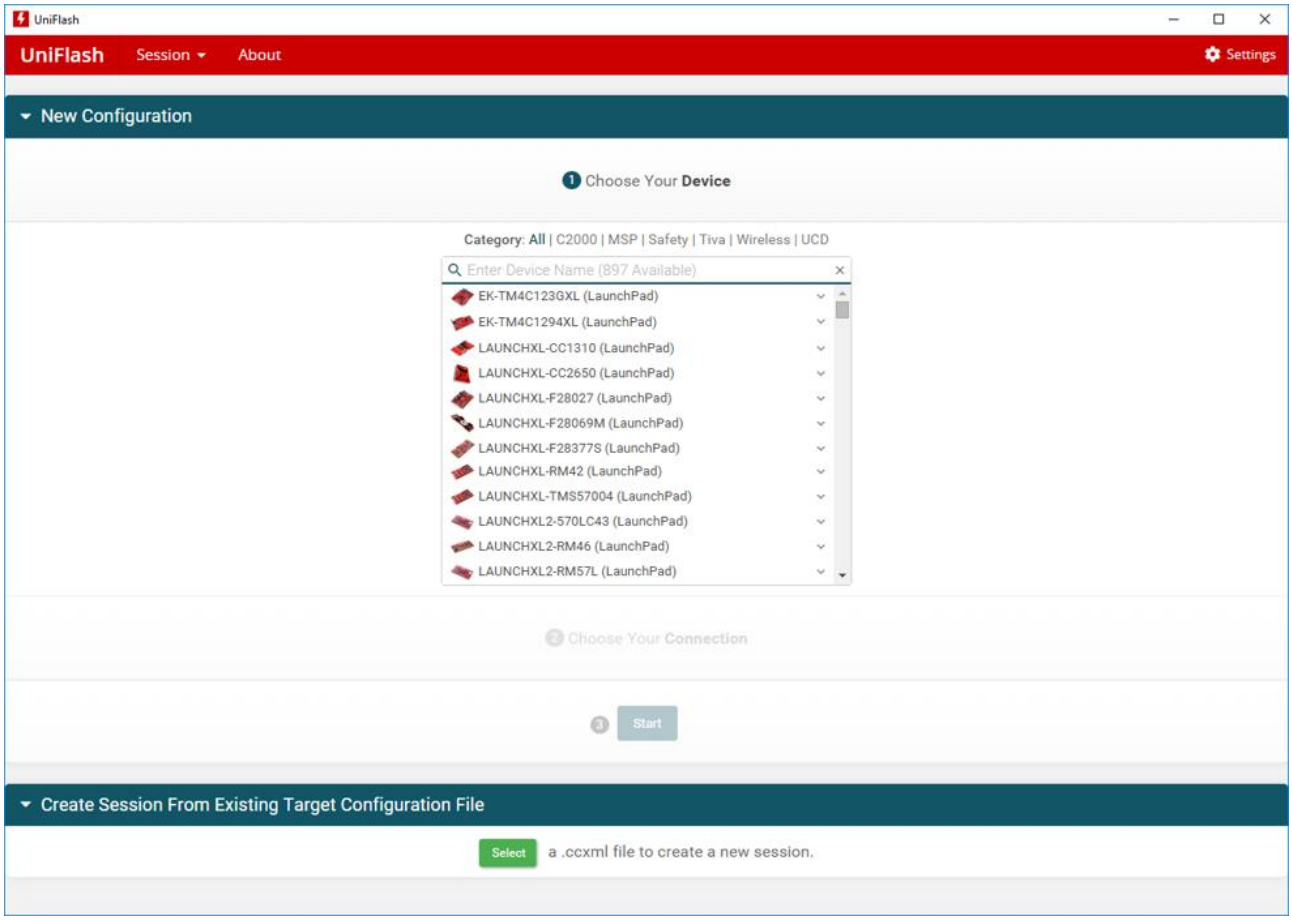
000807
000808
000847

100803.300

- Pack monitoring temp sensore 000510.1 - Termistor
- Temp sensor wire 1000 m
- Serial sensor – ISOSPI cable 300 mm
- Voltage sensor cable

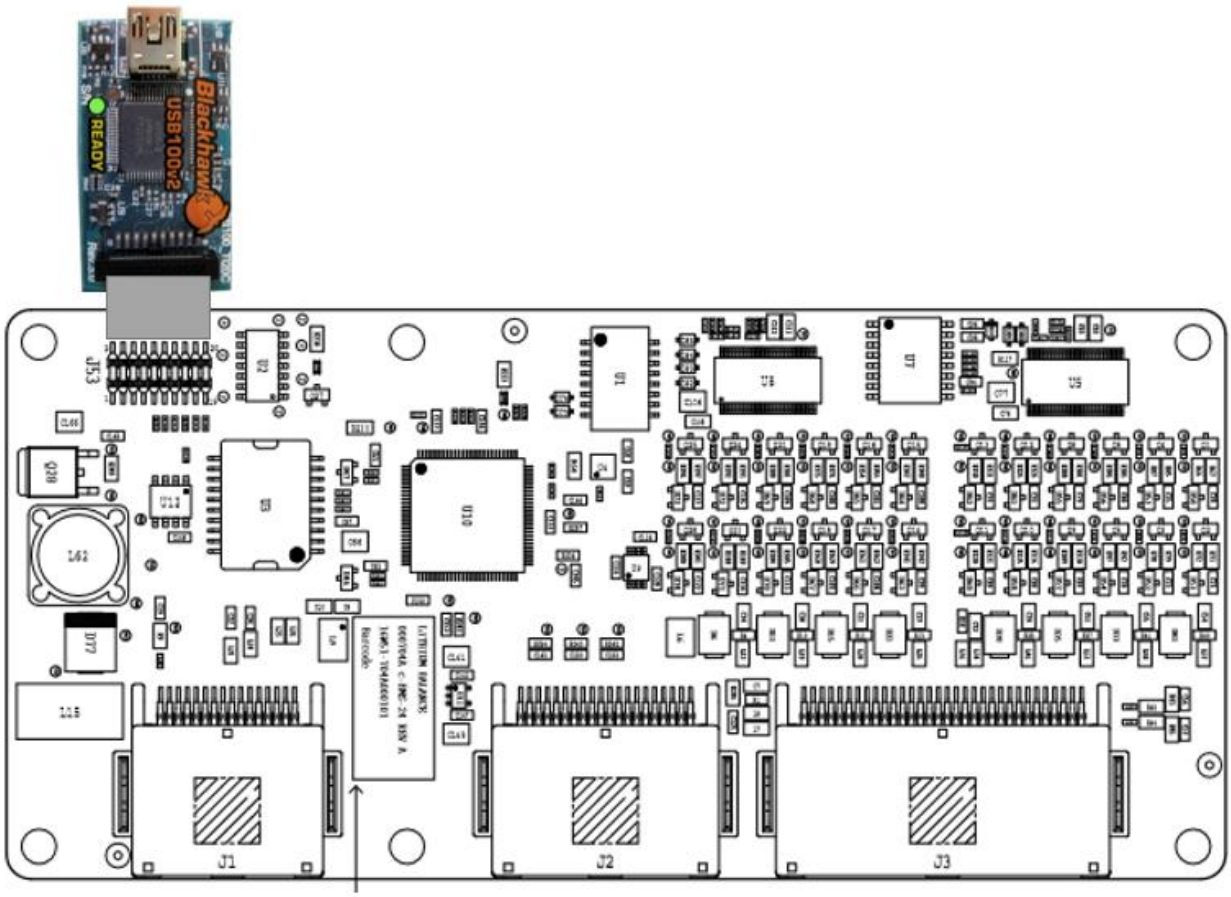
8.2 Appendix 2: Flashing (JTAG)

Step 1: Install UniFlash v4.1 from Texas Instruments (http://www.ti.com/tool/uniflash)

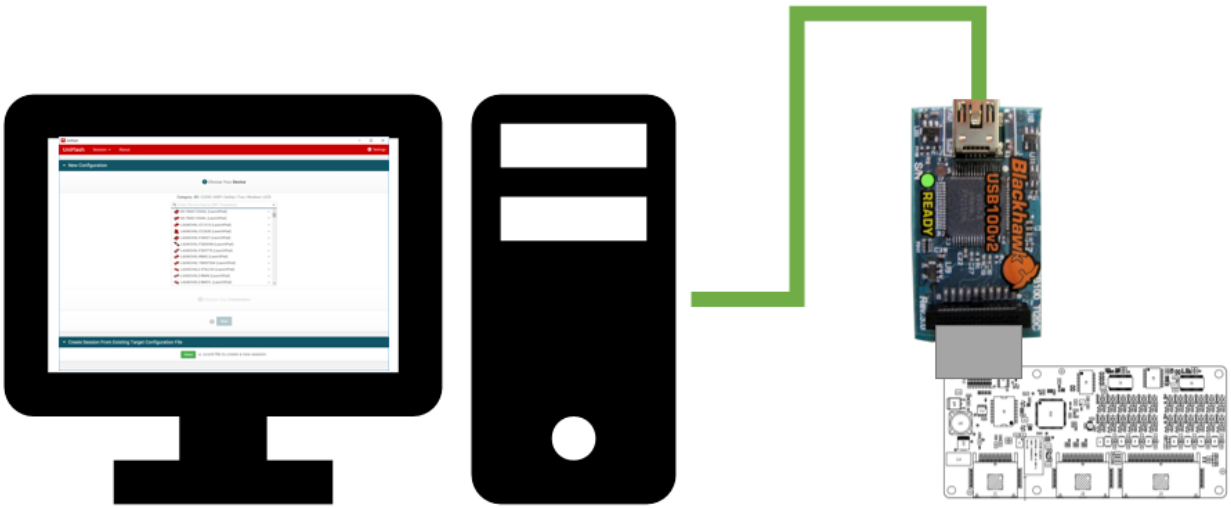


Step 2: Connect Blackhawk USB100v2 to BMS (J53)

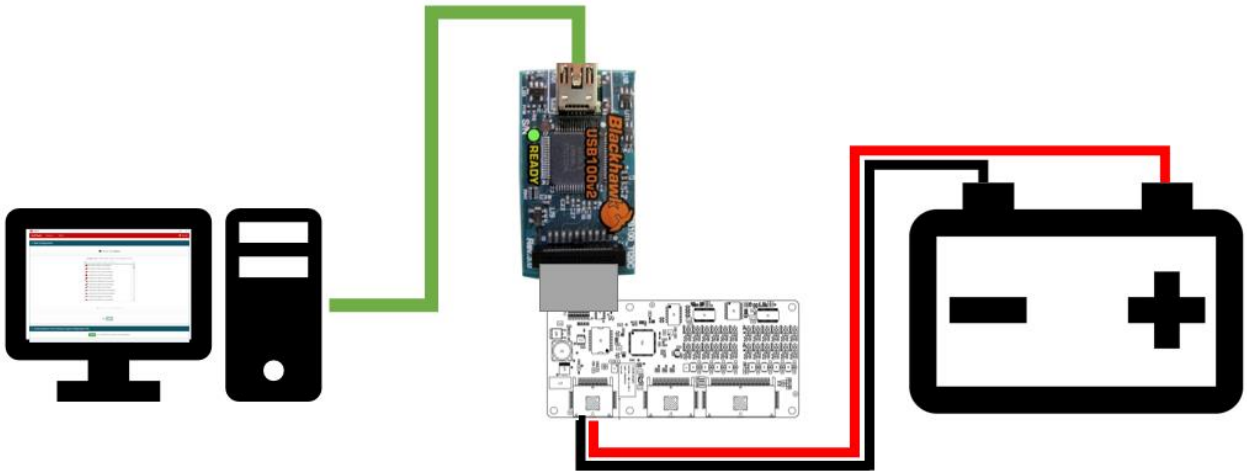
The JTAG connector on the Blackhawk flat cable can only fit in one orientation. Locate the blocked pin hole and align it with the missing pin in the BMS JTAG connector (J53) for correct connection.



Step 3: Connect Blackhawk USB100v2 to a PC

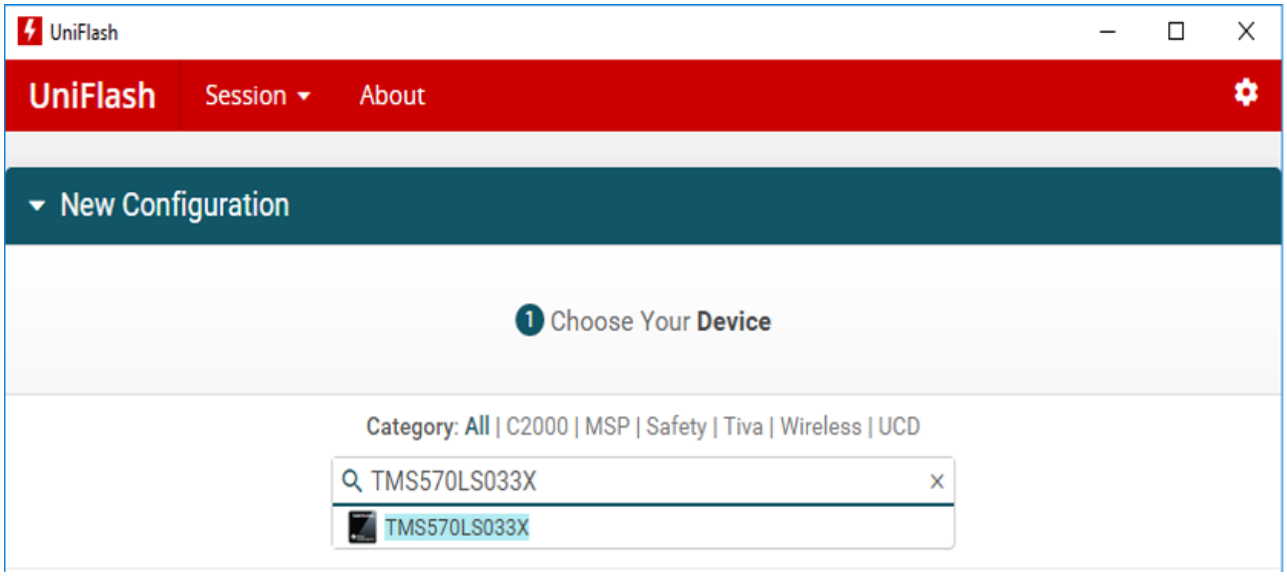


Step 4: Connect BMS to power (J1)



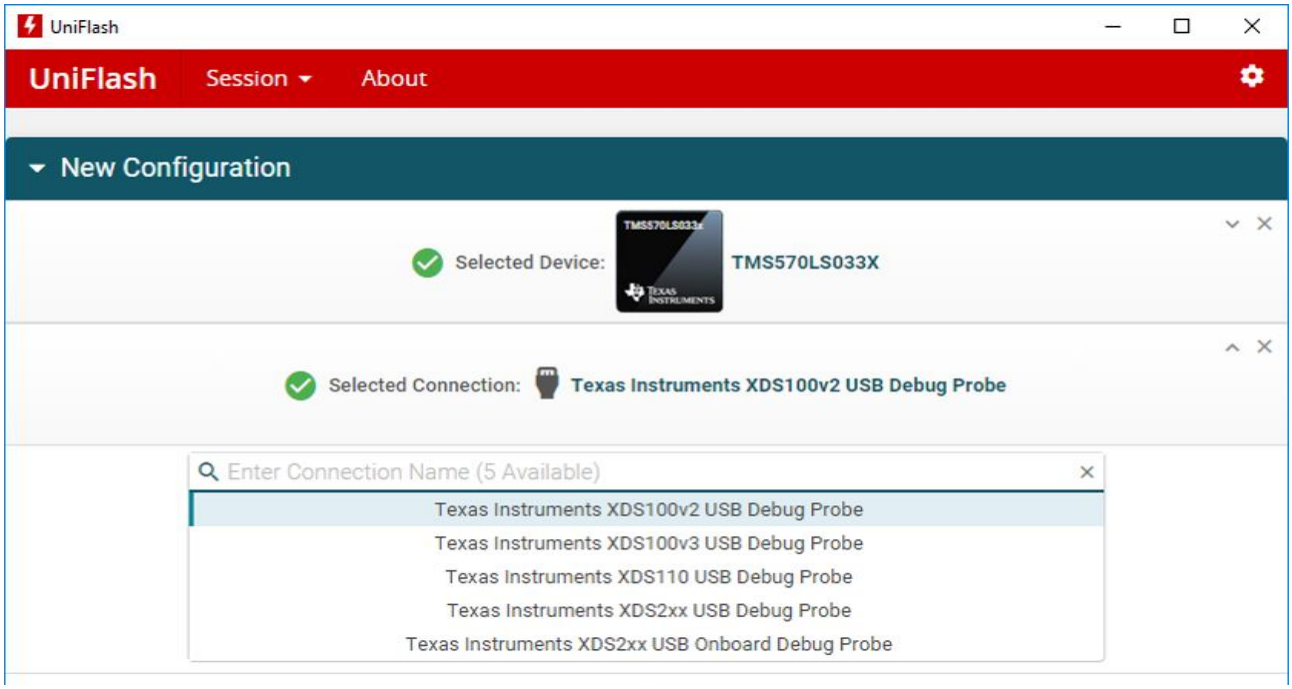
Step 5: Start UniFlash and select “New Configuration”.

Type in “TMS570LS033X” and click on the highlighted Device in blue.



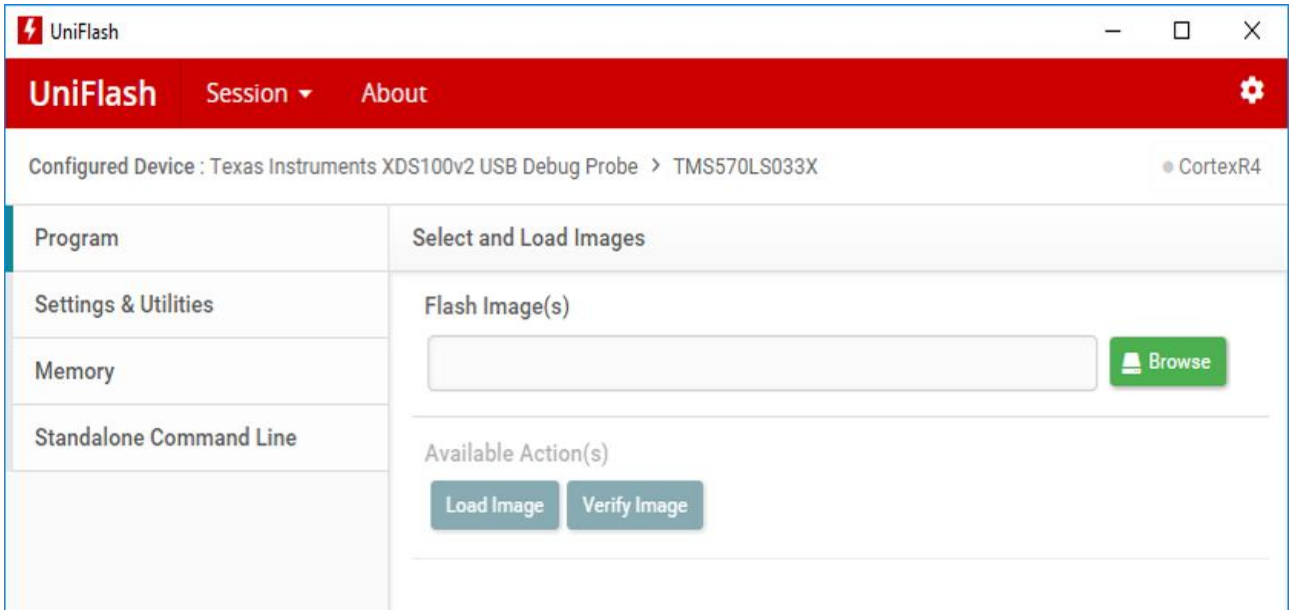
Step 6: In UniFlash, choose connection.

Select “Texas Instruments XDS100v2 USB Debug Probe” by clicking on it.



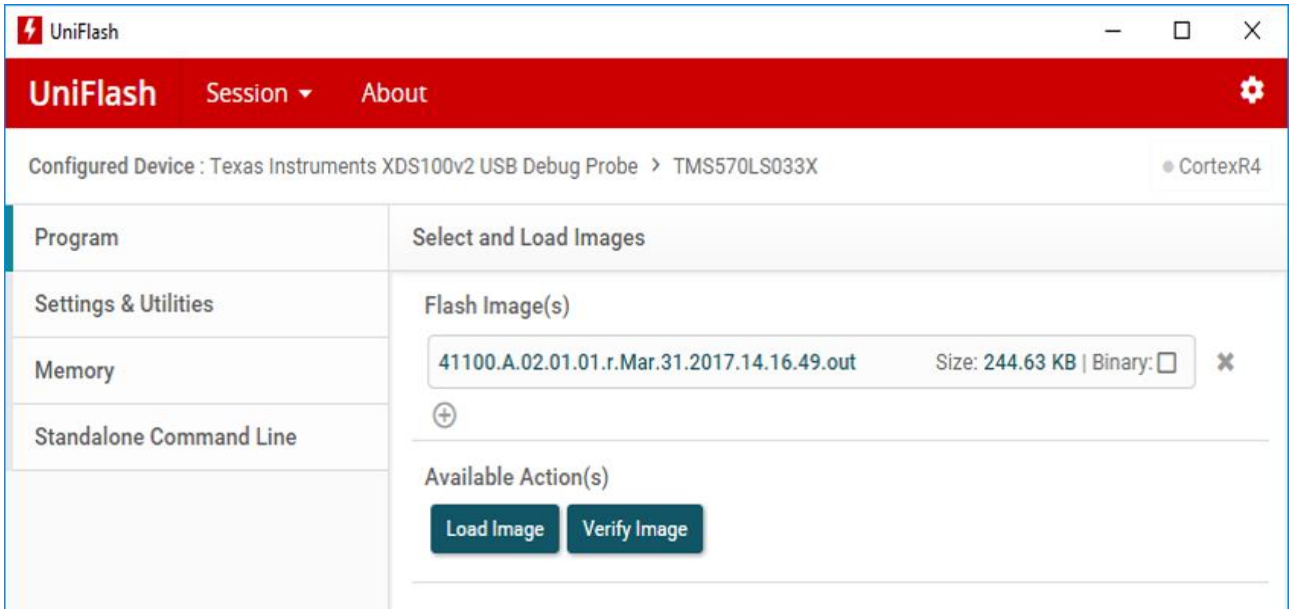
Step 7: In UniFlash, select Flash Image

Click the green “Browse” button and select the Lithium Balance supplied *.out file



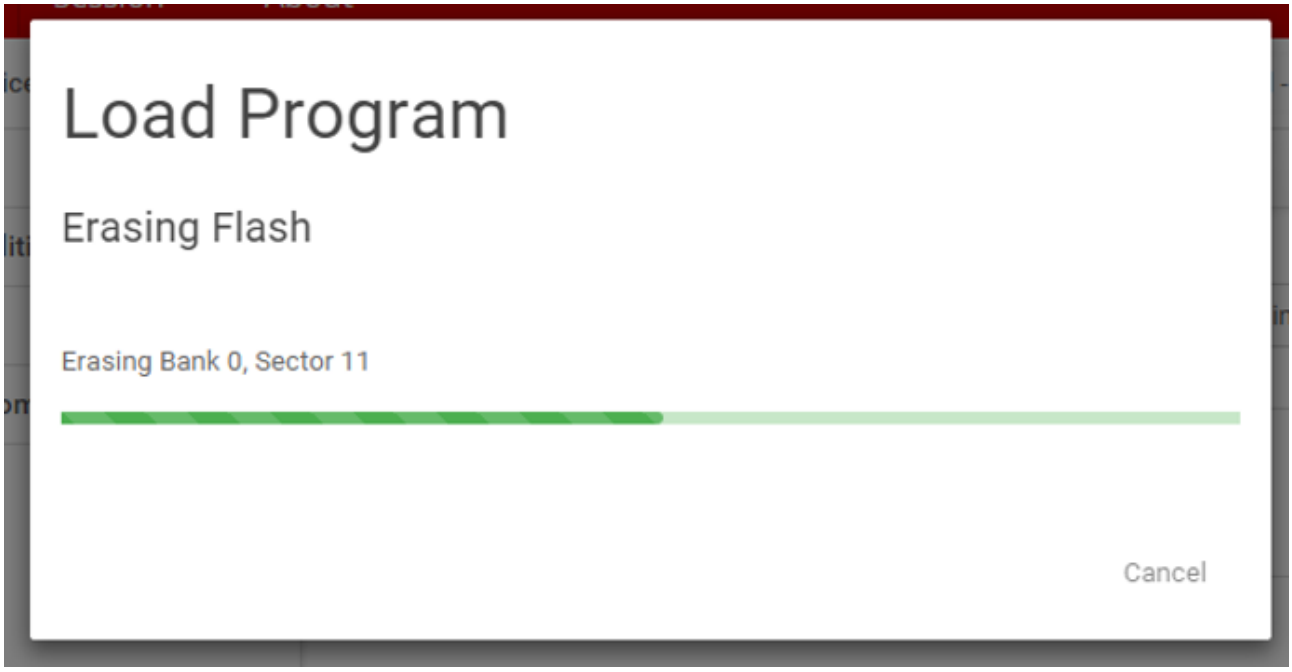
Step 8: In UniFlash, Load Image

Click the blue “Load Image” button and wait for the sequence to finish.



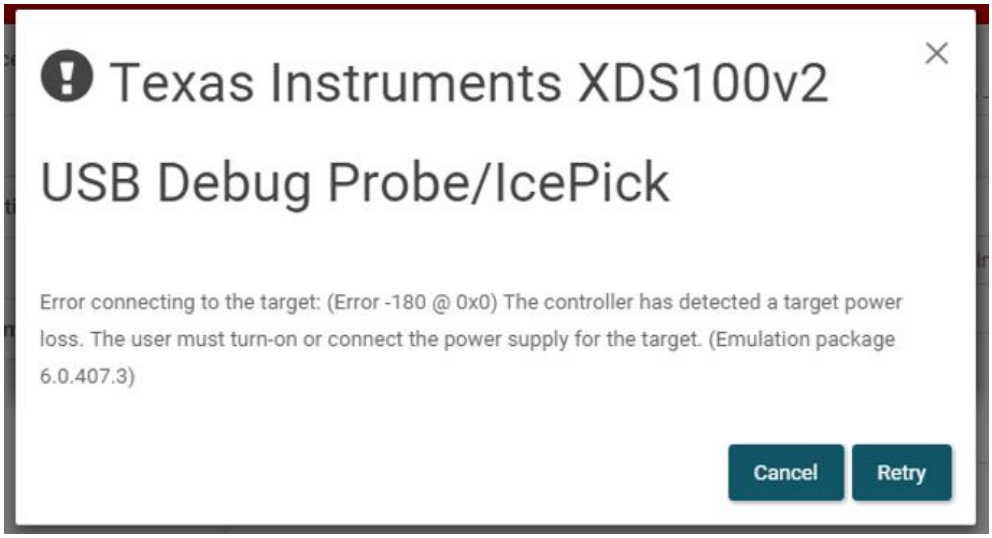
Step 9: In UniFlash, this is normal operation

This is UniFlash if flashing works as intended:



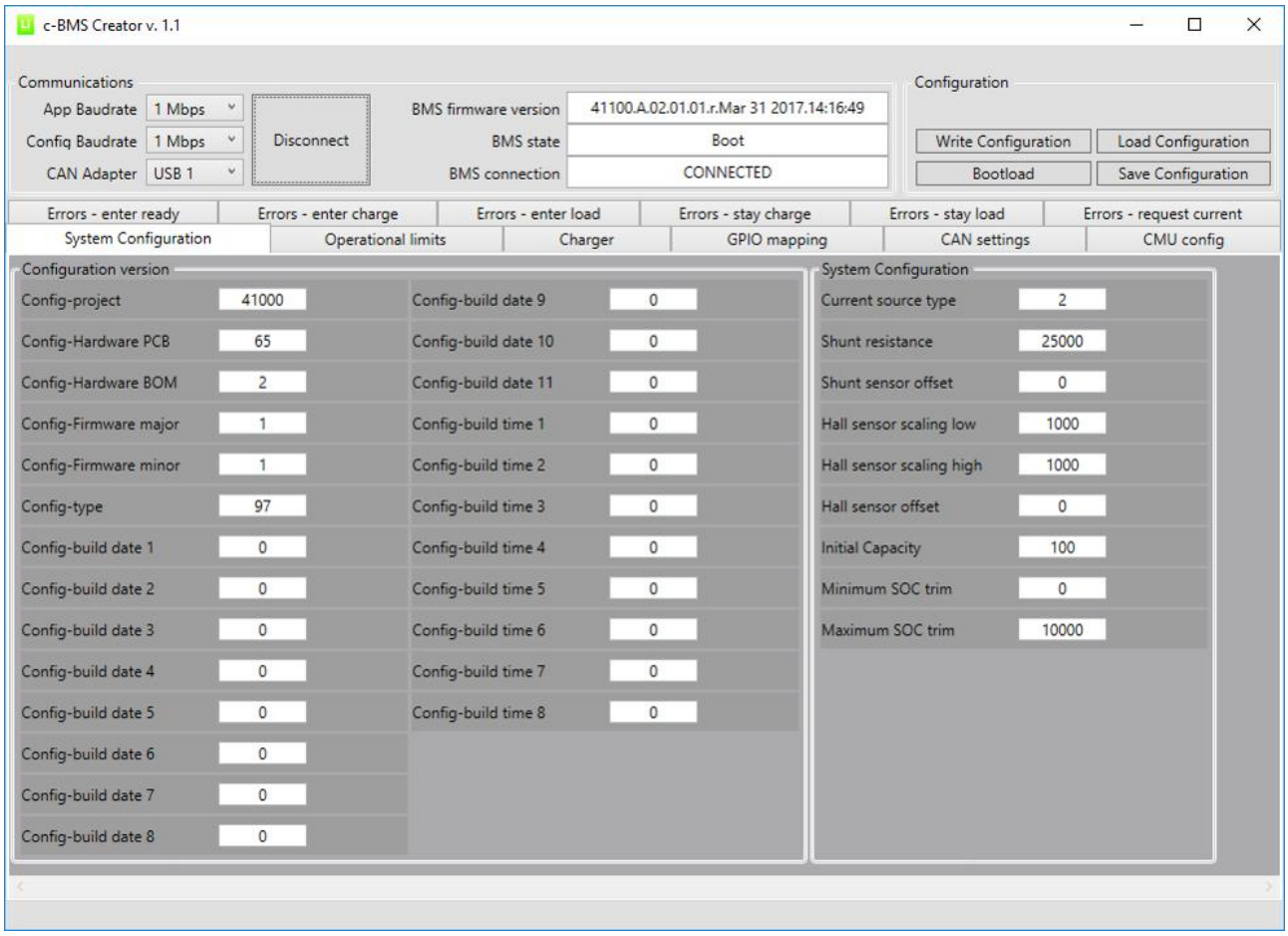
After successful flash sequence the D76 LED (Hardware rev. B-01) on the BMS should now be flashing.

If Uniflash outputs the following error message, it could be because the BMS is not correctly connected to the PC, or it is not powered on. Ensure the first steps are done correctly before contacting Lithium Balance for support.



Step 10: After boot load firmware flash

The BMS now has the latest version of the boot loader firmware. Next step is to update the BMS with application firmware through the CAN interface using the BMS Creator. Consult the BMS Creator section on how to boot load new application firmware and configuration values to the BMS.



8.3 Appendix 3: Firmware error codes

8.3.1 Error severity codes

Code	Severity
0	NO_SEVERITY
1	LOW_SEVERITY_SYSTEM
2	LOW_SEVERITY_PACK
3	NORMAL_SEVERITY_SYSTEM
4	NORMAL_SEVERITY_PACK
5	CRITICAL_SEVERITY_SYSTEM
6	CRITICAL_SEVERITY_PACK

8.3.2 Error code origins

Origin code	Origin name
0	ERROR_ORIGIN_NONE
1	ERROR_ORIGIN_CPU
2	ERROR_ORIGIN_SPI_HAL
3	ERROR_ORIGIN_ADC_HAL
4	ERROR_ORIGIN_SPI_SYS
5	ERROR_ORIGIN_CAN_SYS
6	ERROR_ORIGIN_ADC_EXT
7	ERROR_ORIGIN_MAX14661
8	ERROR_ORIGIN_LTC68XX
9	ERROR_ORIGIN_PACK_AND_CELL
10	ERROR_ORIGIN_CAN
11	ERROR_ORIGIN_STATE_MANAGER
12	ERROR_ORIGIN_PSU_MANAGER
13	ERROR_ORIGIN_ERROR_HANDLING
14	ERROR_ORIGIN_BAT_DIAG
15	ERROR_ORIGIN_CHG_CTRL
16	ERROR_ORIGIN_CMU
17	ERROR_ORIGIN_RTC
18	ERROR_ORIGIN_UDS
19	ERROR_ORIGIN_ID_MAP
20	ERROR_ORIGIN_MAIN
21	ERROR_ORIGIN_M95640
22	ERROR_ORIGIN_BOOT
23	ERROR_ORIGIN_FLASH_SYS
24	ERROR_ORIGIN_CRC
25	ERROR_ORIGIN_GPIO
26	ERROR_ORIGIN_INTERPOLATION
27	ERROR_ORIGIN_QUEUE_HANDLER
28	ERROR_ORIGIN_SRAM_INTERFACE

29	ERROR_ORIGIN_PCF2127
30	ERROR_ORIGIN_TASK_MONITOR
31	ERROR_ORIGIN_MEASURE_AND_MANAGE
32	ERROR_ORIGIN_CDP
33	ERROR_ORIGIN_ERROR_LOG
34	ERROR_ORIGIN_MAIN_APP

8.3.3 Error codes

Error code (dec)	Error code (hex)	Error name
0	0000	ERROR_NOERROR
1	0001	ERROR_GENERAL_ERROR
2	0002	ERROR_GENERAL_IOB
3	0003	ERROR_GENERAL_ADDRESS
4	0004	ERROR_GENERAL_RESACC
5	0005	ERROR_GENERAL_BUSY
6	0006	ERROR_GENERAL_NULLPOINT
7	0007	ERROR_GENERAL_PAOB
8	0008	ERROR_GENERAL_READFAIL
9	0009	ERROR_GENERAL_WRITEFAIL
10	000A	ERROR_GENERAL_BUFEMPTY
11	000B	ERROR_GENERAL_DATAERROR
12	000C	ERROR_GENERAL_RTOS_INIT
13	000D	ERROR_GENERAL_QUEUE_OVERRUN
14	000E	ERROR_GENERAL_ACCESS_LOCKED
15	000F	ERROR_GENERAL_COM
16	0010	ERROR_GENERAL_CHKSUM_MISMATCH
17	0011	ERROR_GENERAL_TIMEOUT
18	0012	ERROR_GENERAL_MUTEX
19	0013	ERROR_GENERAL_NOT_IMPLEMENTED
20	0014	ERROR_GENERAL_NOT_INITIALIZED
100	0064	ERROR_QUEUE_ACCESS_PACK_AND_CELL
101	0065	ERROR_QUEUE_ACCESS_STATE_MANAGER
102	0066	ERROR_QUEUE_ERROR_HANDLING
103	0067	ERROR_QUEUE_PSU_STATUS
104	0068	ERROR_QUEUE_WRONG_NUMBER_OF_QUEUES
105	0069	ERROR_QUEUE_ALREADY_CREATED
106	006A	ERROR_QUEUE_RTOS_ERROR
107	006B	ERROR_QUEUE_NOT_CREATED
108	006C	ERROR_QUEUE_SIZE_NOT_1
109	006D	ERROR_QUEUE_EMPTY
110	006E	ERROR_QUEUE_SIZE_MISMATCH
201	00C9	ERROR_EEPROM_FSMBUSY
202	00CA	ERROR_EEPROM_FSMREADY

203	00CB	ERROR_EEPROM_FAIL
204	00CC	ERROR_EEPROM_NULLPTR
205	00CD	ERROR_EEPROM_INVCMD
206	00CE	ERROR_EEPROM_INVECCADDR
207	00CF	ERROR_EEPROM_OTPCHKSM
208	00D0	ERROR_EEPROM_INVHCLK
209	00D1	ERROR_EEPROM_INVBNK
210	00D2	ERROR_EEPROM_INVADDR
211	00D3	ERROR_EEPROM_INVREADMODE
212	00D4	ERROR_EEPROM_BUFLEN
213	00D5	ERROR_EEPROM_ECCBUFLEN
214	00D6	ERROR_EEPROM_BUFFMISMATCH
215	00D7	ERROR_EEPROM_INVFEATURE
216	00D8	ERROR_EEPROM_WRITE_PROTECTED
217	00D9	ERROR_EEPROM_SN_INVALID
250	00FA	ERROR_FLASH_FAIL
275	0113	ERROR_SRAM_CRC
276	0114	ERROR_SRAM_WRITE_ERROR
300	012C	LB_ERROR_CAN_BUSOFF
301	012D	LB_ERROR_CAN_BUSWARN
302	012E	LB_ERROR_CAN_BUSPER
303	012F	LB_ERROR_CAN_INTERNAL_ERROR
400	0190	ERROR_IO_INPUT_UPDATE
401	0191	ERROR_TASK_MON_EXECUTION_TIME_VIOLATION
402	0192	ERROR_TASK_MON_TIME_BETWEEN_EXECUTIONS_VIOLATION
403	0193	ERROR_TASK_MON_TASK_DEAD
700	02BC	LB_ERROR_RTC_LOWBAT
701	02BD	ERROR_INVALID_TIME_FROM_RTC
600	0258	ERROR_ADC_EXT_GENERAL_ERROR
601	0259	ERROR_ADC_EXT_INIT
610	0262	ERROR_ADC_EXT_START_SHUNT
611	0263	ERROR_ADC_EXT_READ_SHUNT
620	026C	ERROR_ADC_EXT_START_HVPLUS
621	026D	ERROR_ADC_EXT_READ_HVPLUS
630	0276	ERROR_ADC_EXT_START_HALL
631	0277	ERROR_ADC_EXT_READ_HALL
900	0384	ERROR_ADC_UNHANDLED_PIN
901	0385	ERROR_ADC_INTERNAL_LOGIC
902	0386	ERROR_ADC_MCU_FIFO_SIZE
903	0387	ERROR_ADC_MCU_TIMING
904	0388	ERROR_ADC_INT_INIT
905	0389	ERROR_ADC_INVALID_ID
1000	03E8	ERROR_LTC68XX_CHECKSUM
1001	03E9	ERROR_COMPARISON_FAILED

1002	03EA	ERROR_I2C_SLAVE_NACK
1003	03EB	ERROR_I2C_WRONG_CONTROL_BITS
1004	03EC	ERROR_CELL_VOLTAGE_SELF_TEST_FAILED
1005	03ED	ERROR_OPEN_WIRE_DETECTED
1006	03EE	ERROR_BALANCING_OPERATION
1100	044C	ERROR_TEMP_AUX_NO_VALUE_CH01
1101	044D	ERROR_TEMP_AUX_NO_VALUE_CH02
1102	044E	ERROR_TEMP_AUX_NO_VALUE_CH03
1103	044F	ERROR_TEMP_AUX_NO_VALUE_CH04
1104	0450	ERROR_TEMP_AUX_NO_VALUE_CH05
1105	0451	ERROR_TEMP_AUX_NO_VALUE_CH06
1106	0452	ERROR_TEMP_AUX_NO_VALUE_CH07
1107	0453	ERROR_TEMP_AUX_NO_VALUE_CH08
1108	0454	ERROR_TEMP_AUX_NO_VALUE_CH09
1109	0455	ERROR_TEMP_AUX_NO_VALUE_CH10
1110	0456	ERROR_TEMP_AUX_NO_VALUE_CH11
1116	045C	ERROR_TEMP_AUX_SHORTED_CH01
1117	045D	ERROR_TEMP_AUX_SHORTED_CH02
1118	045E	ERROR_TEMP_AUX_SHORTED_CH03
1119	045F	ERROR_TEMP_AUX_SHORTED_CH04
1120	0460	ERROR_TEMP_AUX_SHORTED_CH05
1121	0461	ERROR_TEMP_AUX_SHORTED_CH06
1122	0462	ERROR_TEMP_AUX_SHORTED_CH07
1123	0463	ERROR_TEMP_AUX_SHORTED_CH08
1124	0464	ERROR_TEMP_AUX_SHORTED_CH09
1125	0465	ERROR_TEMP_AUX_SHORTED_CH10
1126	0466	ERROR_TEMP_AUX_SHORTED_CH11
1132	046C	ERROR_TEMP_AUX_OPEN_CH01
1133	046D	ERROR_TEMP_AUX_OPEN_CH02
1134	046E	ERROR_TEMP_AUX_OPEN_CH03
1135	046F	ERROR_TEMP_AUX_OPEN_CH04
1136	0470	ERROR_TEMP_AUX_OPEN_CH05
1137	0471	ERROR_TEMP_AUX_OPEN_CH06
1138	0472	ERROR_TEMP_AUX_OPEN_CH07
1139	0473	ERROR_TEMP_AUX_OPEN_CH08
1140	0474	ERROR_TEMP_AUX_OPEN_CH09
1141	0475	ERROR_TEMP_AUX_OPEN_CH10
1142	0476	ERROR_TEMP_AUX_OPEN_CH11
1148	047C	ERROR_TEMP_AUX_MIN_CH01
1149	047D	ERROR_TEMP_AUX_MIN_CH02
1150	047E	ERROR_TEMP_AUX_MIN_CH03
1151	047F	ERROR_TEMP_AUX_MIN_CH04
1152	0480	ERROR_TEMP_AUX_MIN_CH05
1153	0481	ERROR_TEMP_AUX_MIN_CH06

1154	0482	ERROR_TEMP_AUX_MIN_CH07
1155	0483	ERROR_TEMP_AUX_MIN_CH08
1156	0484	ERROR_TEMP_AUX_MIN_CH09
1157	0485	ERROR_TEMP_AUX_MIN_CH10
1158	0486	ERROR_TEMP_AUX_MIN_CH11
1164	048C	ERROR_TEMP_AUX_MAX_CH01
1165	048D	ERROR_TEMP_AUX_MAX_CH02
1166	048E	ERROR_TEMP_AUX_MAX_CH03
1167	048F	ERROR_TEMP_AUX_MAX_CH04
1168	0490	ERROR_TEMP_AUX_MAX_CH05
1169	0491	ERROR_TEMP_AUX_MAX_CH06
1170	0492	ERROR_TEMP_AUX_MAX_CH07
1171	0493	ERROR_TEMP_AUX_MAX_CH08
1172	0494	ERROR_TEMP_AUX_MAX_CH09
1173	0495	ERROR_TEMP_AUX_MAX_CH10
1174	0496	ERROR_TEMP_AUX_MAX_CH11
2000	07D0	ERROR_SYS_LIM_CELL_V_MIN
2001	07D1	ERROR_SYS_LIM_CELL_V_MAX
2002	07D2	ERROR_SYS_LIM_CELL_DV_NEG
2003	07D3	ERROR_SYS_LIM_CELL_DV_POS
2004	07D4	ERROR_SYS_LIM_CELL_T_MIN
2005	07D5	ERROR_SYS_LIM_CELL_T_MAX
2006	07D6	ERROR_SYS_LIM_CELL_DT_NEG
2007	07D7	ERROR_SYS_LIM_CELL_DT_POS
2008	07D8	ERROR_SYS_LIM_PACK_I_IN
2009	07D9	ERROR_SYS_LIM_PACK_I_OUT
2010	07DA	ERROR_SYS_LIM_PACK_I2T
2011	07DB	ERROR_SYS_CELL_V_NO_VALUE
2012	07DC	ERROR_SYS_CELL_T_NO_VALUE
2013	07DD	ERROR_SYS_CELL_T_SHORTED
2014	07DE	ERROR_SYS_CELL_T_OPEN
2015	07DF	ERROR_SYS_CMU_PCB_T_NO_VALUE
2016	07E0	ERROR_SYS_CMU_PCB_T_SHORTED
2017	07E1	ERROR_SYS_CMU_PCB_T_OPEN
2018	07E2	ERROR_SYS_PACK_I_MASTER_SELECTION
2019	07E3	ERROR_SYS_COMM_CMU
2020	07E4	ERROR_SYS_NUM_CMU_MISMATCH
2021	07E5	ERROR_SYS_FB_LOAD_NEG_MISSING
2022	07E6	ERROR_SYS_FB_LOAD_POS_MISSING
2023	07E7	ERROR_SYS_FB_PRECHARGE_MISSING
2024	07E8	ERROR_SYS_FB_CHG_NEG_MISSING
2025	07E9	ERROR_SYS_FB_LOAD_NEG_WELDED
2026	07EA	ERROR_SYS_FB_LOAD_POS_WELDED
2027	07EB	ERROR_SYS_FB_PRECHARGE_WELDED

2028	07EC	ERROR_SYS_FB_CHG_NEG_WELDED
2029	07ED	ERROR_SYS_PACK_V_EXT_LOW
2030	07EE	ERROR_SYS_PACK_V_EXT_HIGH
2031	07EF	ERROR_SYS_CONTACTOR_RETRIES
2032	07F0	ERROR_SYS_PROCESS_CMU_DATA
2033	07F1	ERROR_SYS_PACK_AND_CELL_INIT
2034	07F2	ERROR_SYS_PACK_AND_CELL_ADC_HV
2035	07F3	ERROR_SYS_PACK_Q_CALC
2036	07F4	ERROR_SYS_PACK_SOC_CALC
2037	07F5	ERROR_SYS_CONTACTOR_FEEDBACK
2038	07F6	ERROR_SYS_EXEC_STATE_LOGIC
2039	07F7	ERROR_SYS_VSUPPLY_MIN
2040	07F8	ERROR_SYS_VSUPPLY_MAX
2041	07F9	ERROR_SYS_PSU_NOT_IN_DIAGNOSTIC_MODE

8.4 Appendix 4: Data ID map

For configurable CAN setup and UDS request by ID service reference.

ID	Name	Type	Scaling	Unit	Description
1	CELL_V_MIN_VAL	UINT16	0.1	mV	Lowest cell voltage
2	CELL_V_MIN_ID_CMU	UINT8	1	-	CMU of lowest cell voltage
3	CELL_V_MIN_ID_CELL	UINT8	1	-	Cell number of cell w. lowest voltage
4	CELL_V_MAX_VAL	UINT16	0.1	mV	Highest cell voltage
5	CELL_V_MAX_ID_CMU	UINT8	1	-	CMU of highest cell voltage
6	CELL_V_MAX_ID_CELL	UINT8	1	-	Cell number of cell w. highest voltage
7	CELL_V_AVG	UINT16	0.1	mV	Average cell voltage
8	CELL_V_NUM_AVAILABLE	UINT16	1	-	Number of cell voltages configured
9	PACK_V_EXT_LOAD	UINT16	0.1	V	Voltage measurement at positive terminal of load
10	PACK_V_EXT_CHG	UINT16	0.1	V	Voltage measurement at positive terminal of charger
11	PACK_V_SUM_OF_CELLS	UINT16	0.1	V	Sum of all cells
12	HALL_V_HALL_V_LO	UINT16	0.1	mV	Voltage level from HALL low channel, master
13	HALL_V_HALL_V_HI	UINT16	0.1	mV	Voltage level from HALL high channel, master

14	PACK_I_HALL	INT32	0.001	mA	HALL current
15	PACK_I_SHUNT	INT32	0.001	mA	Shunt current
16	PACK_I_MASTER	INT32	0.001	mA	Selected current source (shunt or hall) will be the system reference current.
17	PACK_Q_REMAINING_HI_RES	INT32	1	As	Capacity remaining
18	PACK_Q_SOC_INTERNAL	INT16	0.001	%	SoC
19	PACK_Q_SOC_TRIMMED	UINT16	0.001	%	SoC
20	PACK_Q_REMAINING_NOMINAL	UINT16	0.1	Ah	Capacity remaining
21	PACK_Q_DESIGN	UINT16	0.1	Ah	Capacity as designed (nominal capacity from cell manufacturer)
22	PACK_Q_FULL	UINT16	0.1	Ah	Capacity at fully charged state
24	IO_STATE_INPUT	UINT8	1	-	Bitmask (b0 = I-1...)
25	CHARGER_OUTPUT_ENABLED	UINT8	1	-	b0 is Boolean
26	CHARGER_OUTPUT_CURRENT	UINT16	0.1	A	Current requested from charger
27	CHARGER_OUTPUT_VOLTAGE	UINT16	0.1	V	Voltage requested from charger
28	CHARGER_CAN_ACTIVE	UINT8	1	-	b0 is Boolean
29	CHARGER_PWM_ACTIVE	UINT8	1	-	b0 is Boolean
30	CHARGER_PWM_ACTIVE_DUTY	UINT8	1	%	PWM duty cycle for charger
31	DYN_LIM_I2T_REMAIN	UINT32	1	A2s	Remaining i2t sum before error is triggered
32	DYN_LIM_I_IN	UINT16	0.1	A	Current limit for incomming current (typically charge or regen)
33	DYN_LIM_I_OUT	UINT16	0.1	A	Current limit for outgoing current (typically discharge)
34	STATUS	UINT8	1	-	BMS state
35	CONTACTORS_ENABLED	UINT8	1	-	b0 is Boolean
36	CONTACTORS_ACTIVATE_CHARGE	UINT8	1	-	b0 is Boolean
37	CONTACTORS_ACTIVATE_LOAD	UINT8	1	-	b0 is Boolean
38	CONTACTORS_ACTIVATE_COMBINED	UINT8	1	-	b0 is Boolean

39	CONTACTORS_CHARGER_ACTIVATED	UINT8	1	-	b0 is Boolean
40	CONTACTORS_LOAD_ACTIVATED	UINT8	1	-	b0 is Boolean
41	CONTACTORS_COMBINED_ACTIVATED	UINT8	1	-	b0 is Boolean
42	CONTACTORS_ACTIVATION_ALLOWED	UINT8	1	-	b0 is Boolean
43	CONTACTORS_EMERGENCY_OFF	UINT8	1	-	b0 is Boolean
44	CONTACTORS_CONTACTOR_RETRIES	UINT8	1	-	Count of contactor closing retries
45	BALANCING_ALLOWED	UINT8	1	-	b0 is Boolean
46	BALANCING_LIMIT	UINT16	0.1	mV	Lower voltage limit for balancing to be active
47	FLAGS_FULLY_CHARGED	UINT8	1	-	b0 is Boolean
48	FLAGS_FULLY_CHARGED_LATCHED	UINT8	1	-	b0 is Boolean
49	FLAGS_LOAD_ACTIVE	UINT8	1	-	b0 is Boolean
50	FLAGS_CHARGER_ACTIVE	UINT8	1	-	b0 is Boolean
51	FLAGS_PRECHARGE_ACTIVE	UINT8	1	-	b0 is Boolean
52	FLAGS_BALANCING_ACTIVE	UINT8	1	-	b0 is Boolean
53	FLAGS_CHARGE_REG_ACTIVE	UINT8	1	-	b0 is Boolean
54	B_WU_IGN	UINT8	1	-	b0 is Boolean
55	B_WU_CAN	UINT8	1	-	b0 is Boolean
60	CRITICAL_SEVERITY_COUNT	UINT8	1	-	Count of critical severity errors
61	NORMAL_SEVERITY_COUNT	UINT8	1	-	Count of normal severity errors
62	LOW_SEVERITY_COUNT	UINT8	1	-	Count of low severity errors
63	ACTIVE_COUNT	UINT16	1	-	Count of total amount of errors
64	PROJECT	UINT16	1	-	Project version
65	HW_PCB	UINT8	1	-	Hardware PCB version (Interpret as Ascii)
66	HW_BOM	UINT8	1	-	Hardware BOM version
67	FW_MAJOR	UINT8	1	-	Firmware major version

68	FW_MINOR	UINT 8	1	-	Firmware minor version
69	TYPE	UINT 8	1	-	Firmware type (Interpret as Ascii)
70	BUILD_DATE	UINT 8 * 11	1	-	Firmware build date (Ascii, Mmm.dd.yyyy)
71	BUILD_TIME	UINT 8 * 8	1	-	Firmware build time (Ascii, hh:mm:ss)
72	CMUS_CMU1_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
73	CMUS_CMU1_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
74	CMUS_CMU2_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
75	CMUS_CMU2_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
76	CMUS_CMU3_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
77	CMUS_CMU3_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
78	CMUS_CMU4_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
79	CMUS_CMU4_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
80	CMUS_CMU5_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
81	CMUS_CMU5_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
82	CMUS_CMU6_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
83	CMUS_CMU6_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
84	CMUS_CMU7_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
85	CMUS_CMU7_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
86	CMUS_CMU8_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
87	CMUS_CMU8_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
88	CMUS_CMU9_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
89	CMUS_CMU9_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
90	CMUS_CMU10_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
91	CMUS_CMU10_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC

92	CMUS_CMU11_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
93	CMUS_CMU11_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
94	CMUS_CMU12_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
95	CMUS_CMU12_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
96	CMUS_CMU13_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
97	CMUS_CMU13_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
98	CMUS_CMU14_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
99	CMUS_CMU14_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
100	CMUS_CMU15_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
101	CMUS_CMU15_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
102	CMUS_CMU16_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
103	CMUS_CMU16_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
104	CMUS_CMU17_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
105	CMUS_CMU17_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
106	CMUS_CMU18_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
107	CMUS_CMU18_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
108	CMUS_CMU19_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
109	CMUS_CMU19_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
110	CMUS_CMU20_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
111	CMUS_CMU20_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
112	CMUS_CMU21_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
113	CMUS_CMU21_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
114	CMUS_CMU22_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
115	CMUS_CMU22_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
116	CMUS_CMU23_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC

117	CMUS_CMU23_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
118	CMUS_CMU24_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
119	CMUS_CMU24_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
120	CMUS_CMU25_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
121	CMUS_CMU25_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
122	CMUS_CMU26_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
123	CMUS_CMU26_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
124	CMUS_CMU27_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
125	CMUS_CMU27_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
126	CMUS_CMU28_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
127	CMUS_CMU28_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
128	CMUS_CMU29_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
129	CMUS_CMU29_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
130	CMUS_CMU30_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
131	CMUS_CMU30_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
132	CMUS_CMU31_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
133	CMUS_CMU31_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
134	CMUS_CMU32_BALANCE_T1	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
135	CMUS_CMU32_BALANCE_T2	INT8	1	°C	CMU balance circuit temperature. 121 = OC, -41 = SC
136	PCB_T1	INT8	1	°C	MCU PCB temperature 0
137	PCB_T2	INT8	1	°C	MCU PCB temperature 1
138	AUX_T1	INT8	1	°C	AUX temperature 0
139	AUX_T2	INT8	1	°C	AUX temperature 1
140	AUX_T3	INT8	1	°C	AUX temperature 2
141	AUX_T4	INT8	1	°C	AUX temperature 3
142	AUX_T5	INT8	1	°C	AUX temperature 4
143	AUX_T6	INT8	1	°C	AUX temperature 5
144	AUX_T7	INT8	1	°C	AUX temperature 6
145	AUX_T8	INT8	1	°C	AUX temperature 7

146	AUX_T9	INT8	1	°C	AUX temperature 8
147	AUX_T10	INT8	1	°C	AUX temperature 9
148	AUX_T11	INT8	1	°C	AUX temperature 10
149	CMUS_CMU1_CELL_T1	INT8	1	°C	Cell temperature for cmu[0], cell_t[0]
150	CMUS_CMU1_CELL_T2	INT8	1	°C	Cell temperature for cmu[0], cell_t[1]
151	CMUS_CMU1_CELL_T3	INT8	1	°C	Cell temperature for cmu[0], cell_t[2]
152	CMUS_CMU1_CELL_T4	INT8	1	°C	Cell temperature for cmu[0], cell_t[3]
153	CMUS_CMU1_CELL_T5	INT8	1	°C	Cell temperature for cmu[0], cell_t[4]
154	CMUS_CMU1_CELL_T6	INT8	1	°C	Cell temperature for cmu[0], cell_t[5]
155	CMUS_CMU1_CELL_T7	INT8	1	°C	Cell temperature for cmu[0], cell_t[6]
156	CMUS_CMU1_CELL_T8	INT8	1	°C	Cell temperature for cmu[0], cell_t[7]
157	CMUS_CMU1_CELL_T9	INT8	1	°C	Cell temperature for cmu[0], cell_t[8]
158	CMUS_CMU1_CELL_T10	INT8	1	°C	Cell temperature for cmu[0], cell_t[9]
159	CMUS_CMU1_CELL_T11	INT8	1	°C	Cell temperature for cmu[0], cell_t[10]
160	CMUS_CMU1_CELL_T12	INT8	1	°C	Cell temperature for cmu[0], cell_t[11]
161	CMUS_CMU2_CELL_T1	INT8	1	°C	Cell temperature for cmu[1], cell_t[0]
162	CMUS_CMU2_CELL_T2	INT8	1	°C	Cell temperature for cmu[1], cell_t[1]
163	CMUS_CMU2_CELL_T3	INT8	1	°C	Cell temperature for cmu[1], cell_t[2]
164	CMUS_CMU2_CELL_T4	INT8	1	°C	Cell temperature for cmu[1], cell_t[3]
165	CMUS_CMU2_CELL_T5	INT8	1	°C	Cell temperature for cmu[1], cell_t[4]
166	CMUS_CMU2_CELL_T6	INT8	1	°C	Cell temperature for cmu[1], cell_t[5]
167	CMUS_CMU2_CELL_T7	INT8	1	°C	Cell temperature for cmu[1], cell_t[6]
168	CMUS_CMU2_CELL_T8	INT8	1	°C	Cell temperature for cmu[1], cell_t[7]
169	CMUS_CMU2_CELL_T9	INT8	1	°C	Cell temperature for cmu[1], cell_t[8]
170	CMUS_CMU2_CELL_T10	INT8	1	°C	Cell temperature for cmu[1], cell_t[9]
171	CMUS_CMU2_CELL_T11	INT8	1	°C	Cell temperature for cmu[1], cell_t[10]

172	CMUS_CMU2_CELL_T12	INT8	1	°C	Cell temperature for cmu[1], cell_t[11]
173	CMUS_CMU3_CELL_T1	INT8	1	°C	Cell temperature for cmu[2], cell_t[0]
174	CMUS_CMU3_CELL_T2	INT8	1	°C	Cell temperature for cmu[2], cell_t[1]
175	CMUS_CMU3_CELL_T3	INT8	1	°C	Cell temperature for cmu[2], cell_t[2]
176	CMUS_CMU3_CELL_T4	INT8	1	°C	Cell temperature for cmu[2], cell_t[3]
177	CMUS_CMU3_CELL_T5	INT8	1	°C	Cell temperature for cmu[2], cell_t[4]
178	CMUS_CMU3_CELL_T6	INT8	1	°C	Cell temperature for cmu[2], cell_t[5]
179	CMUS_CMU3_CELL_T7	INT8	1	°C	Cell temperature for cmu[2], cell_t[6]
180	CMUS_CMU3_CELL_T8	INT8	1	°C	Cell temperature for cmu[2], cell_t[7]
181	CMUS_CMU3_CELL_T9	INT8	1	°C	Cell temperature for cmu[2], cell_t[8]
182	CMUS_CMU3_CELL_T10	INT8	1	°C	Cell temperature for cmu[2], cell_t[9]
183	CMUS_CMU3_CELL_T11	INT8	1	°C	Cell temperature for cmu[2], cell_t[10]
184	CMUS_CMU3_CELL_T12	INT8	1	°C	Cell temperature for cmu[2], cell_t[11]
185	CMUS_CMU4_CELL_T1	INT8	1	°C	Cell temperature for cmu[3], cell_t[0]
186	CMUS_CMU4_CELL_T2	INT8	1	°C	Cell temperature for cmu[3], cell_t[1]
187	CMUS_CMU4_CELL_T3	INT8	1	°C	Cell temperature for cmu[3], cell_t[2]
188	CMUS_CMU4_CELL_T4	INT8	1	°C	Cell temperature for cmu[3], cell_t[3]
189	CMUS_CMU4_CELL_T5	INT8	1	°C	Cell temperature for cmu[3], cell_t[4]
190	CMUS_CMU4_CELL_T6	INT8	1	°C	Cell temperature for cmu[3], cell_t[5]
191	CMUS_CMU4_CELL_T7	INT8	1	°C	Cell temperature for cmu[3], cell_t[6]
192	CMUS_CMU4_CELL_T8	INT8	1	°C	Cell temperature for cmu[3], cell_t[7]
193	CMUS_CMU4_CELL_T9	INT8	1	°C	Cell temperature for cmu[3], cell_t[8]
194	CMUS_CMU4_CELL_T10	INT8	1	°C	Cell temperature for cmu[3], cell_t[9]
195	CMUS_CMU4_CELL_T11	INT8	1	°C	Cell temperature for cmu[3], cell_t[10]
196	CMUS_CMU4_CELL_T12	INT8	1	°C	Cell temperature for cmu[3], cell_t[11]

197	CMUS_CMU5_CELL_T1	INT8	1	°C	Cell temperature for cmu[4], cell_t[0]
198	CMUS_CMU5_CELL_T2	INT8	1	°C	Cell temperature for cmu[4], cell_t[1]
199	CMUS_CMU5_CELL_T3	INT8	1	°C	Cell temperature for cmu[4], cell_t[2]
200	CMUS_CMU5_CELL_T4	INT8	1	°C	Cell temperature for cmu[4], cell_t[3]
201	CMUS_CMU5_CELL_T5	INT8	1	°C	Cell temperature for cmu[4], cell_t[4]
202	CMUS_CMU5_CELL_T6	INT8	1	°C	Cell temperature for cmu[4], cell_t[5]
203	CMUS_CMU5_CELL_T7	INT8	1	°C	Cell temperature for cmu[4], cell_t[6]
204	CMUS_CMU5_CELL_T8	INT8	1	°C	Cell temperature for cmu[4], cell_t[7]
205	CMUS_CMU5_CELL_T9	INT8	1	°C	Cell temperature for cmu[4], cell_t[8]
206	CMUS_CMU5_CELL_T10	INT8	1	°C	Cell temperature for cmu[4], cell_t[9]
207	CMUS_CMU5_CELL_T11	INT8	1	°C	Cell temperature for cmu[4], cell_t[10]
208	CMUS_CMU5_CELL_T12	INT8	1	°C	Cell temperature for cmu[4], cell_t[11]
209	CMUS_CMU6_CELL_T1	INT8	1	°C	Cell temperature for cmu[5], cell_t[0]
210	CMUS_CMU6_CELL_T2	INT8	1	°C	Cell temperature for cmu[5], cell_t[1]
211	CMUS_CMU6_CELL_T3	INT8	1	°C	Cell temperature for cmu[5], cell_t[2]
212	CMUS_CMU6_CELL_T4	INT8	1	°C	Cell temperature for cmu[5], cell_t[3]
213	CMUS_CMU6_CELL_T5	INT8	1	°C	Cell temperature for cmu[5], cell_t[4]
214	CMUS_CMU6_CELL_T6	INT8	1	°C	Cell temperature for cmu[5], cell_t[5]
215	CMUS_CMU6_CELL_T7	INT8	1	°C	Cell temperature for cmu[5], cell_t[6]
216	CMUS_CMU6_CELL_T8	INT8	1	°C	Cell temperature for cmu[5], cell_t[7]
217	CMUS_CMU6_CELL_T9	INT8	1	°C	Cell temperature for cmu[5], cell_t[8]
218	CMUS_CMU6_CELL_T10	INT8	1	°C	Cell temperature for cmu[5], cell_t[9]
219	CMUS_CMU6_CELL_T11	INT8	1	°C	Cell temperature for cmu[5], cell_t[10]
220	CMUS_CMU6_CELL_T12	INT8	1	°C	Cell temperature for cmu[5], cell_t[11]
221	CMUS_CMU7_CELL_T1	INT8	1	°C	Cell temperature for cmu[6], cell_t[0]

222	CMUS_CMU7_CELL_T2	INT8	1	°C	Cell temperature for cmu[6], cell_t[1]
223	CMUS_CMU7_CELL_T3	INT8	1	°C	Cell temperature for cmu[6], cell_t[2]
224	CMUS_CMU7_CELL_T4	INT8	1	°C	Cell temperature for cmu[6], cell_t[3]
225	CMUS_CMU7_CELL_T5	INT8	1	°C	Cell temperature for cmu[6], cell_t[4]
226	CMUS_CMU7_CELL_T6	INT8	1	°C	Cell temperature for cmu[6], cell_t[5]
227	CMUS_CMU7_CELL_T7	INT8	1	°C	Cell temperature for cmu[6], cell_t[6]
228	CMUS_CMU7_CELL_T8	INT8	1	°C	Cell temperature for cmu[6], cell_t[7]
229	CMUS_CMU7_CELL_T9	INT8	1	°C	Cell temperature for cmu[6], cell_t[8]
230	CMUS_CMU7_CELL_T10	INT8	1	°C	Cell temperature for cmu[6], cell_t[9]
231	CMUS_CMU7_CELL_T11	INT8	1	°C	Cell temperature for cmu[6], cell_t[10]
232	CMUS_CMU7_CELL_T12	INT8	1	°C	Cell temperature for cmu[6], cell_t[11]
233	CMUS_CMU8_CELL_T1	INT8	1	°C	Cell temperature for cmu[7], cell_t[0]
234	CMUS_CMU8_CELL_T2	INT8	1	°C	Cell temperature for cmu[7], cell_t[1]
235	CMUS_CMU8_CELL_T3	INT8	1	°C	Cell temperature for cmu[7], cell_t[2]
236	CMUS_CMU8_CELL_T4	INT8	1	°C	Cell temperature for cmu[7], cell_t[3]
237	CMUS_CMU8_CELL_T5	INT8	1	°C	Cell temperature for cmu[7], cell_t[4]
238	CMUS_CMU8_CELL_T6	INT8	1	°C	Cell temperature for cmu[7], cell_t[5]
239	CMUS_CMU8_CELL_T7	INT8	1	°C	Cell temperature for cmu[7], cell_t[6]
240	CMUS_CMU8_CELL_T8	INT8	1	°C	Cell temperature for cmu[7], cell_t[7]
241	CMUS_CMU8_CELL_T9	INT8	1	°C	Cell temperature for cmu[7], cell_t[8]
242	CMUS_CMU8_CELL_T10	INT8	1	°C	Cell temperature for cmu[7], cell_t[9]
243	CMUS_CMU8_CELL_T11	INT8	1	°C	Cell temperature for cmu[7], cell_t[10]
244	CMUS_CMU8_CELL_T12	INT8	1	°C	Cell temperature for cmu[7], cell_t[11]
245	CMUS_CMU9_CELL_T1	INT8	1	°C	Cell temperature for cmu[8], cell_t[0]
246	CMUS_CMU9_CELL_T2	INT8	1	°C	Cell temperature for cmu[8], cell_t[1]

247	CMUS_CMU9_CELL_T3	INT8	1	°C	Cell temperature for cmu[8], cell_t[2]
248	CMUS_CMU9_CELL_T4	INT8	1	°C	Cell temperature for cmu[8], cell_t[3]
249	CMUS_CMU9_CELL_T5	INT8	1	°C	Cell temperature for cmu[8], cell_t[4]
250	CMUS_CMU9_CELL_T6	INT8	1	°C	Cell temperature for cmu[8], cell_t[5]
251	CMUS_CMU9_CELL_T7	INT8	1	°C	Cell temperature for cmu[8], cell_t[6]
252	CMUS_CMU9_CELL_T8	INT8	1	°C	Cell temperature for cmu[8], cell_t[7]
253	CMUS_CMU9_CELL_T9	INT8	1	°C	Cell temperature for cmu[8], cell_t[8]
254	CMUS_CMU9_CELL_T10	INT8	1	°C	Cell temperature for cmu[8], cell_t[9]
255	CMUS_CMU9_CELL_T11	INT8	1	°C	Cell temperature for cmu[8], cell_t[10]
256	CMUS_CMU9_CELL_T12	INT8	1	°C	Cell temperature for cmu[8], cell_t[11]
257	CMUS_CMU10_CELL_T1	INT8	1	°C	Cell temperature for cmu[9], cell_t[0]
258	CMUS_CMU10_CELL_T2	INT8	1	°C	Cell temperature for cmu[9], cell_t[1]
259	CMUS_CMU10_CELL_T3	INT8	1	°C	Cell temperature for cmu[9], cell_t[2]
260	CMUS_CMU10_CELL_T4	INT8	1	°C	Cell temperature for cmu[9], cell_t[3]
261	CMUS_CMU10_CELL_T5	INT8	1	°C	Cell temperature for cmu[9], cell_t[4]
262	CMUS_CMU10_CELL_T6	INT8	1	°C	Cell temperature for cmu[9], cell_t[5]
263	CMUS_CMU10_CELL_T7	INT8	1	°C	Cell temperature for cmu[9], cell_t[6]
264	CMUS_CMU10_CELL_T8	INT8	1	°C	Cell temperature for cmu[9], cell_t[7]
265	CMUS_CMU10_CELL_T9	INT8	1	°C	Cell temperature for cmu[9], cell_t[8]
266	CMUS_CMU10_CELL_T10	INT8	1	°C	Cell temperature for cmu[9], cell_t[9]
267	CMUS_CMU10_CELL_T11	INT8	1	°C	Cell temperature for cmu[9], cell_t[10]
268	CMUS_CMU10_CELL_T12	INT8	1	°C	Cell temperature for cmu[9], cell_t[11]
269	CMUS_CMU11_CELL_T1	INT8	1	°C	Cell temperature for cmu[10], cell_t[0]
270	CMUS_CMU11_CELL_T2	INT8	1	°C	Cell temperature for cmu[10], cell_t[1]
271	CMUS_CMU11_CELL_T3	INT8	1	°C	Cell temperature for cmu[10], cell_t[2]

272	CMUS_CMU11_CELL_T4	INT8	1	°C	Cell temperature for cmu[10], cell_t[3]
273	CMUS_CMU11_CELL_T5	INT8	1	°C	Cell temperature for cmu[10], cell_t[4]
274	CMUS_CMU11_CELL_T6	INT8	1	°C	Cell temperature for cmu[10], cell_t[5]
275	CMUS_CMU11_CELL_T7	INT8	1	°C	Cell temperature for cmu[10], cell_t[6]
276	CMUS_CMU11_CELL_T8	INT8	1	°C	Cell temperature for cmu[10], cell_t[7]
277	CMUS_CMU11_CELL_T9	INT8	1	°C	Cell temperature for cmu[10], cell_t[8]
278	CMUS_CMU11_CELL_T10	INT8	1	°C	Cell temperature for cmu[10], cell_t[9]
279	CMUS_CMU11_CELL_T11	INT8	1	°C	Cell temperature for cmu[10], cell_t[10]
280	CMUS_CMU11_CELL_T12	INT8	1	°C	Cell temperature for cmu[10], cell_t[11]
281	CMUS_CMU12_CELL_T1	INT8	1	°C	Cell temperature for cmu[11], cell_t[0]
282	CMUS_CMU12_CELL_T2	INT8	1	°C	Cell temperature for cmu[11], cell_t[1]
283	CMUS_CMU12_CELL_T3	INT8	1	°C	Cell temperature for cmu[11], cell_t[2]
284	CMUS_CMU12_CELL_T4	INT8	1	°C	Cell temperature for cmu[11], cell_t[3]
285	CMUS_CMU12_CELL_T5	INT8	1	°C	Cell temperature for cmu[11], cell_t[4]
286	CMUS_CMU12_CELL_T6	INT8	1	°C	Cell temperature for cmu[11], cell_t[5]
287	CMUS_CMU12_CELL_T7	INT8	1	°C	Cell temperature for cmu[11], cell_t[6]
288	CMUS_CMU12_CELL_T8	INT8	1	°C	Cell temperature for cmu[11], cell_t[7]
289	CMUS_CMU12_CELL_T9	INT8	1	°C	Cell temperature for cmu[11], cell_t[8]
290	CMUS_CMU12_CELL_T10	INT8	1	°C	Cell temperature for cmu[11], cell_t[9]
291	CMUS_CMU12_CELL_T11	INT8	1	°C	Cell temperature for cmu[11], cell_t[10]
292	CMUS_CMU12_CELL_T12	INT8	1	°C	Cell temperature for cmu[11], cell_t[11]
293	CMUS_CMU13_CELL_T1	INT8	1	°C	Cell temperature for cmu[12], cell_t[0]
294	CMUS_CMU13_CELL_T2	INT8	1	°C	Cell temperature for cmu[12], cell_t[1]
295	CMUS_CMU13_CELL_T3	INT8	1	°C	Cell temperature for cmu[12], cell_t[2]
296	CMUS_CMU13_CELL_T4	INT8	1	°C	Cell temperature for cmu[12], cell_t[3]

297	CMUS_CMU13_CELL_T5	INT8	1	°C	Cell temperature for cmu[12], cell_t[4]
298	CMUS_CMU13_CELL_T6	INT8	1	°C	Cell temperature for cmu[12], cell_t[5]
299	CMUS_CMU13_CELL_T7	INT8	1	°C	Cell temperature for cmu[12], cell_t[6]
300	CMUS_CMU13_CELL_T8	INT8	1	°C	Cell temperature for cmu[12], cell_t[7]
301	CMUS_CMU13_CELL_T9	INT8	1	°C	Cell temperature for cmu[12], cell_t[8]
302	CMUS_CMU13_CELL_T10	INT8	1	°C	Cell temperature for cmu[12], cell_t[9]
303	CMUS_CMU13_CELL_T11	INT8	1	°C	Cell temperature for cmu[12], cell_t[10]
304	CMUS_CMU13_CELL_T12	INT8	1	°C	Cell temperature for cmu[12], cell_t[11]
305	CMUS_CMU14_CELL_T1	INT8	1	°C	Cell temperature for cmu[13], cell_t[0]
306	CMUS_CMU14_CELL_T2	INT8	1	°C	Cell temperature for cmu[13], cell_t[1]
307	CMUS_CMU14_CELL_T3	INT8	1	°C	Cell temperature for cmu[13], cell_t[2]
308	CMUS_CMU14_CELL_T4	INT8	1	°C	Cell temperature for cmu[13], cell_t[3]
309	CMUS_CMU14_CELL_T5	INT8	1	°C	Cell temperature for cmu[13], cell_t[4]
310	CMUS_CMU14_CELL_T6	INT8	1	°C	Cell temperature for cmu[13], cell_t[5]
311	CMUS_CMU14_CELL_T7	INT8	1	°C	Cell temperature for cmu[13], cell_t[6]
312	CMUS_CMU14_CELL_T8	INT8	1	°C	Cell temperature for cmu[13], cell_t[7]
313	CMUS_CMU14_CELL_T9	INT8	1	°C	Cell temperature for cmu[13], cell_t[8]
314	CMUS_CMU14_CELL_T10	INT8	1	°C	Cell temperature for cmu[13], cell_t[9]
315	CMUS_CMU14_CELL_T11	INT8	1	°C	Cell temperature for cmu[13], cell_t[10]
316	CMUS_CMU14_CELL_T12	INT8	1	°C	Cell temperature for cmu[13], cell_t[11]
317	CMUS_CMU15_CELL_T1	INT8	1	°C	Cell temperature for cmu[14], cell_t[0]
318	CMUS_CMU15_CELL_T2	INT8	1	°C	Cell temperature for cmu[14], cell_t[1]
319	CMUS_CMU15_CELL_T3	INT8	1	°C	Cell temperature for cmu[14], cell_t[2]
320	CMUS_CMU15_CELL_T4	INT8	1	°C	Cell temperature for cmu[14], cell_t[3]
321	CMUS_CMU15_CELL_T5	INT8	1	°C	Cell temperature for cmu[14], cell_t[4]

322	CMUS_CMU15_CELL_T6	INT8	1	°C	Cell temperature for cmu[14], cell_t[5]
323	CMUS_CMU15_CELL_T7	INT8	1	°C	Cell temperature for cmu[14], cell_t[6]
324	CMUS_CMU15_CELL_T8	INT8	1	°C	Cell temperature for cmu[14], cell_t[7]
325	CMUS_CMU15_CELL_T9	INT8	1	°C	Cell temperature for cmu[14], cell_t[8]
326	CMUS_CMU15_CELL_T10	INT8	1	°C	Cell temperature for cmu[14], cell_t[9]
327	CMUS_CMU15_CELL_T11	INT8	1	°C	Cell temperature for cmu[14], cell_t[10]
328	CMUS_CMU15_CELL_T12	INT8	1	°C	Cell temperature for cmu[14], cell_t[11]
329	CMUS_CMU16_CELL_T1	INT8	1	°C	Cell temperature for cmu[15], cell_t[0]
330	CMUS_CMU16_CELL_T2	INT8	1	°C	Cell temperature for cmu[15], cell_t[1]
331	CMUS_CMU16_CELL_T3	INT8	1	°C	Cell temperature for cmu[15], cell_t[2]
332	CMUS_CMU16_CELL_T4	INT8	1	°C	Cell temperature for cmu[15], cell_t[3]
333	CMUS_CMU16_CELL_T5	INT8	1	°C	Cell temperature for cmu[15], cell_t[4]
334	CMUS_CMU16_CELL_T6	INT8	1	°C	Cell temperature for cmu[15], cell_t[5]
335	CMUS_CMU16_CELL_T7	INT8	1	°C	Cell temperature for cmu[15], cell_t[6]
336	CMUS_CMU16_CELL_T8	INT8	1	°C	Cell temperature for cmu[15], cell_t[7]
337	CMUS_CMU16_CELL_T9	INT8	1	°C	Cell temperature for cmu[15], cell_t[8]
338	CMUS_CMU16_CELL_T10	INT8	1	°C	Cell temperature for cmu[15], cell_t[9]
339	CMUS_CMU16_CELL_T11	INT8	1	°C	Cell temperature for cmu[15], cell_t[10]
340	CMUS_CMU16_CELL_T12	INT8	1	°C	Cell temperature for cmu[15], cell_t[11]
341	CMUS_CMU17_CELL_T1	INT8	1	°C	Cell temperature for cmu[16], cell_t[0]
342	CMUS_CMU17_CELL_T2	INT8	1	°C	Cell temperature for cmu[16], cell_t[1]
343	CMUS_CMU17_CELL_T3	INT8	1	°C	Cell temperature for cmu[16], cell_t[2]
344	CMUS_CMU17_CELL_T4	INT8	1	°C	Cell temperature for cmu[16], cell_t[3]
345	CMUS_CMU17_CELL_T5	INT8	1	°C	Cell temperature for cmu[16], cell_t[4]
346	CMUS_CMU17_CELL_T6	INT8	1	°C	Cell temperature for cmu[16], cell_t[5]

347	CMUS_CMU17_CELL_T7	INT8	1	°C	Cell temperature for cmu[16], cell_t[6]
348	CMUS_CMU17_CELL_T8	INT8	1	°C	Cell temperature for cmu[16], cell_t[7]
349	CMUS_CMU17_CELL_T9	INT8	1	°C	Cell temperature for cmu[16], cell_t[8]
350	CMUS_CMU17_CELL_T10	INT8	1	°C	Cell temperature for cmu[16], cell_t[9]
351	CMUS_CMU17_CELL_T11	INT8	1	°C	Cell temperature for cmu[16], cell_t[10]
352	CMUS_CMU17_CELL_T12	INT8	1	°C	Cell temperature for cmu[16], cell_t[11]
353	CMUS_CMU18_CELL_T1	INT8	1	°C	Cell temperature for cmu[17], cell_t[0]
354	CMUS_CMU18_CELL_T2	INT8	1	°C	Cell temperature for cmu[17], cell_t[1]
355	CMUS_CMU18_CELL_T3	INT8	1	°C	Cell temperature for cmu[17], cell_t[2]
356	CMUS_CMU18_CELL_T4	INT8	1	°C	Cell temperature for cmu[17], cell_t[3]
357	CMUS_CMU18_CELL_T5	INT8	1	°C	Cell temperature for cmu[17], cell_t[4]
358	CMUS_CMU18_CELL_T6	INT8	1	°C	Cell temperature for cmu[17], cell_t[5]
359	CMUS_CMU18_CELL_T7	INT8	1	°C	Cell temperature for cmu[17], cell_t[6]
360	CMUS_CMU18_CELL_T8	INT8	1	°C	Cell temperature for cmu[17], cell_t[7]
361	CMUS_CMU18_CELL_T9	INT8	1	°C	Cell temperature for cmu[17], cell_t[8]
362	CMUS_CMU18_CELL_T10	INT8	1	°C	Cell temperature for cmu[17], cell_t[9]
363	CMUS_CMU18_CELL_T11	INT8	1	°C	Cell temperature for cmu[17], cell_t[10]
364	CMUS_CMU18_CELL_T12	INT8	1	°C	Cell temperature for cmu[17], cell_t[11]
365	CMUS_CMU19_CELL_T1	INT8	1	°C	Cell temperature for cmu[18], cell_t[0]
366	CMUS_CMU19_CELL_T2	INT8	1	°C	Cell temperature for cmu[18], cell_t[1]
367	CMUS_CMU19_CELL_T3	INT8	1	°C	Cell temperature for cmu[18], cell_t[2]
368	CMUS_CMU19_CELL_T4	INT8	1	°C	Cell temperature for cmu[18], cell_t[3]
369	CMUS_CMU19_CELL_T5	INT8	1	°C	Cell temperature for cmu[18], cell_t[4]
370	CMUS_CMU19_CELL_T6	INT8	1	°C	Cell temperature for cmu[18], cell_t[5]
371	CMUS_CMU19_CELL_T7	INT8	1	°C	Cell temperature for cmu[18], cell_t[6]

372	CMUS_CMU19_CELL_T8	INT8	1	°C	Cell temperature for cmu[18], cell_t[7]
373	CMUS_CMU19_CELL_T9	INT8	1	°C	Cell temperature for cmu[18], cell_t[8]
374	CMUS_CMU19_CELL_T10	INT8	1	°C	Cell temperature for cmu[18], cell_t[9]
375	CMUS_CMU19_CELL_T11	INT8	1	°C	Cell temperature for cmu[18], cell_t[10]
376	CMUS_CMU19_CELL_T12	INT8	1	°C	Cell temperature for cmu[18], cell_t[11]
377	CMUS_CMU20_CELL_T1	INT8	1	°C	Cell temperature for cmu[19], cell_t[0]
378	CMUS_CMU20_CELL_T2	INT8	1	°C	Cell temperature for cmu[19], cell_t[1]
379	CMUS_CMU20_CELL_T3	INT8	1	°C	Cell temperature for cmu[19], cell_t[2]
380	CMUS_CMU20_CELL_T4	INT8	1	°C	Cell temperature for cmu[19], cell_t[3]
381	CMUS_CMU20_CELL_T5	INT8	1	°C	Cell temperature for cmu[19], cell_t[4]
382	CMUS_CMU20_CELL_T6	INT8	1	°C	Cell temperature for cmu[19], cell_t[5]
383	CMUS_CMU20_CELL_T7	INT8	1	°C	Cell temperature for cmu[19], cell_t[6]
384	CMUS_CMU20_CELL_T8	INT8	1	°C	Cell temperature for cmu[19], cell_t[7]
385	CMUS_CMU20_CELL_T9	INT8	1	°C	Cell temperature for cmu[19], cell_t[8]
386	CMUS_CMU20_CELL_T10	INT8	1	°C	Cell temperature for cmu[19], cell_t[9]
387	CMUS_CMU20_CELL_T11	INT8	1	°C	Cell temperature for cmu[19], cell_t[10]
388	CMUS_CMU20_CELL_T12	INT8	1	°C	Cell temperature for cmu[19], cell_t[11]
389	CMUS_CMU21_CELL_T1	INT8	1	°C	Cell temperature for cmu[20], cell_t[0]
390	CMUS_CMU21_CELL_T2	INT8	1	°C	Cell temperature for cmu[20], cell_t[1]
391	CMUS_CMU21_CELL_T3	INT8	1	°C	Cell temperature for cmu[20], cell_t[2]
392	CMUS_CMU21_CELL_T4	INT8	1	°C	Cell temperature for cmu[20], cell_t[3]
393	CMUS_CMU21_CELL_T5	INT8	1	°C	Cell temperature for cmu[20], cell_t[4]
394	CMUS_CMU21_CELL_T6	INT8	1	°C	Cell temperature for cmu[20], cell_t[5]
395	CMUS_CMU21_CELL_T7	INT8	1	°C	Cell temperature for cmu[20], cell_t[6]
396	CMUS_CMU21_CELL_T8	INT8	1	°C	Cell temperature for cmu[20], cell_t[7]

397	CMUS_CMU21_CELL_T9	INT8	1	°C	Cell temperature for cmu[20], cell_t[8]
398	CMUS_CMU21_CELL_T10	INT8	1	°C	Cell temperature for cmu[20], cell_t[9]
399	CMUS_CMU21_CELL_T11	INT8	1	°C	Cell temperature for cmu[20], cell_t[10]
400	CMUS_CMU21_CELL_T12	INT8	1	°C	Cell temperature for cmu[20], cell_t[11]
401	CMUS_CMU22_CELL_T1	INT8	1	°C	Cell temperature for cmu[21], cell_t[0]
402	CMUS_CMU22_CELL_T2	INT8	1	°C	Cell temperature for cmu[21], cell_t[1]
403	CMUS_CMU22_CELL_T3	INT8	1	°C	Cell temperature for cmu[21], cell_t[2]
404	CMUS_CMU22_CELL_T4	INT8	1	°C	Cell temperature for cmu[21], cell_t[3]
405	CMUS_CMU22_CELL_T5	INT8	1	°C	Cell temperature for cmu[21], cell_t[4]
406	CMUS_CMU22_CELL_T6	INT8	1	°C	Cell temperature for cmu[21], cell_t[5]
407	CMUS_CMU22_CELL_T7	INT8	1	°C	Cell temperature for cmu[21], cell_t[6]
408	CMUS_CMU22_CELL_T8	INT8	1	°C	Cell temperature for cmu[21], cell_t[7]
409	CMUS_CMU22_CELL_T9	INT8	1	°C	Cell temperature for cmu[21], cell_t[8]
410	CMUS_CMU22_CELL_T10	INT8	1	°C	Cell temperature for cmu[21], cell_t[9]
411	CMUS_CMU22_CELL_T11	INT8	1	°C	Cell temperature for cmu[21], cell_t[10]
412	CMUS_CMU22_CELL_T12	INT8	1	°C	Cell temperature for cmu[21], cell_t[11]
413	CMUS_CMU23_CELL_T1	INT8	1	°C	Cell temperature for cmu[22], cell_t[0]
414	CMUS_CMU23_CELL_T2	INT8	1	°C	Cell temperature for cmu[22], cell_t[1]
415	CMUS_CMU23_CELL_T3	INT8	1	°C	Cell temperature for cmu[22], cell_t[2]
416	CMUS_CMU23_CELL_T4	INT8	1	°C	Cell temperature for cmu[22], cell_t[3]
417	CMUS_CMU23_CELL_T5	INT8	1	°C	Cell temperature for cmu[22], cell_t[4]
418	CMUS_CMU23_CELL_T6	INT8	1	°C	Cell temperature for cmu[22], cell_t[5]
419	CMUS_CMU23_CELL_T7	INT8	1	°C	Cell temperature for cmu[22], cell_t[6]
420	CMUS_CMU23_CELL_T8	INT8	1	°C	Cell temperature for cmu[22], cell_t[7]
421	CMUS_CMU23_CELL_T9	INT8	1	°C	Cell temperature for cmu[22], cell_t[8]

422	CMUS_CMU23_CELL_T10	INT8	1	°C	Cell temperature for cmu[22], cell_t[9]
423	CMUS_CMU23_CELL_T11	INT8	1	°C	Cell temperature for cmu[22], cell_t[10]
424	CMUS_CMU23_CELL_T12	INT8	1	°C	Cell temperature for cmu[22], cell_t[11]
425	CMUS_CMU24_CELL_T1	INT8	1	°C	Cell temperature for cmu[23], cell_t[0]
426	CMUS_CMU24_CELL_T2	INT8	1	°C	Cell temperature for cmu[23], cell_t[1]
427	CMUS_CMU24_CELL_T3	INT8	1	°C	Cell temperature for cmu[23], cell_t[2]
428	CMUS_CMU24_CELL_T4	INT8	1	°C	Cell temperature for cmu[23], cell_t[3]
429	CMUS_CMU24_CELL_T5	INT8	1	°C	Cell temperature for cmu[23], cell_t[4]
430	CMUS_CMU24_CELL_T6	INT8	1	°C	Cell temperature for cmu[23], cell_t[5]
431	CMUS_CMU24_CELL_T7	INT8	1	°C	Cell temperature for cmu[23], cell_t[6]
432	CMUS_CMU24_CELL_T8	INT8	1	°C	Cell temperature for cmu[23], cell_t[7]
433	CMUS_CMU24_CELL_T9	INT8	1	°C	Cell temperature for cmu[23], cell_t[8]
434	CMUS_CMU24_CELL_T10	INT8	1	°C	Cell temperature for cmu[23], cell_t[9]
435	CMUS_CMU24_CELL_T11	INT8	1	°C	Cell temperature for cmu[23], cell_t[10]
436	CMUS_CMU24_CELL_T12	INT8	1	°C	Cell temperature for cmu[23], cell_t[11]
437	CMUS_CMU25_CELL_T1	INT8	1	°C	Cell temperature for cmu[24], cell_t[0]
438	CMUS_CMU25_CELL_T2	INT8	1	°C	Cell temperature for cmu[24], cell_t[1]
439	CMUS_CMU25_CELL_T3	INT8	1	°C	Cell temperature for cmu[24], cell_t[2]
440	CMUS_CMU25_CELL_T4	INT8	1	°C	Cell temperature for cmu[24], cell_t[3]
441	CMUS_CMU25_CELL_T5	INT8	1	°C	Cell temperature for cmu[24], cell_t[4]
442	CMUS_CMU25_CELL_T6	INT8	1	°C	Cell temperature for cmu[24], cell_t[5]
443	CMUS_CMU25_CELL_T7	INT8	1	°C	Cell temperature for cmu[24], cell_t[6]
444	CMUS_CMU25_CELL_T8	INT8	1	°C	Cell temperature for cmu[24], cell_t[7]
445	CMUS_CMU25_CELL_T9	INT8	1	°C	Cell temperature for cmu[24], cell_t[8]
446	CMUS_CMU25_CELL_T10	INT8	1	°C	Cell temperature for cmu[24], cell_t[9]

447	CMUS_CMU25_CELL_T11	INT8	1	°C	Cell temperature for cmu[24], cell_t[10]
448	CMUS_CMU25_CELL_T12	INT8	1	°C	Cell temperature for cmu[24], cell_t[11]
449	CMUS_CMU26_CELL_T1	INT8	1	°C	Cell temperature for cmu[25], cell_t[0]
450	CMUS_CMU26_CELL_T2	INT8	1	°C	Cell temperature for cmu[25], cell_t[1]
451	CMUS_CMU26_CELL_T3	INT8	1	°C	Cell temperature for cmu[25], cell_t[2]
452	CMUS_CMU26_CELL_T4	INT8	1	°C	Cell temperature for cmu[25], cell_t[3]
453	CMUS_CMU26_CELL_T5	INT8	1	°C	Cell temperature for cmu[25], cell_t[4]
454	CMUS_CMU26_CELL_T6	INT8	1	°C	Cell temperature for cmu[25], cell_t[5]
455	CMUS_CMU26_CELL_T7	INT8	1	°C	Cell temperature for cmu[25], cell_t[6]
456	CMUS_CMU26_CELL_T8	INT8	1	°C	Cell temperature for cmu[25], cell_t[7]
457	CMUS_CMU26_CELL_T9	INT8	1	°C	Cell temperature for cmu[25], cell_t[8]
458	CMUS_CMU26_CELL_T10	INT8	1	°C	Cell temperature for cmu[25], cell_t[9]
459	CMUS_CMU26_CELL_T11	INT8	1	°C	Cell temperature for cmu[25], cell_t[10]
460	CMUS_CMU26_CELL_T12	INT8	1	°C	Cell temperature for cmu[25], cell_t[11]
461	CMUS_CMU27_CELL_T1	INT8	1	°C	Cell temperature for cmu[26], cell_t[0]
462	CMUS_CMU27_CELL_T2	INT8	1	°C	Cell temperature for cmu[26], cell_t[1]
463	CMUS_CMU27_CELL_T3	INT8	1	°C	Cell temperature for cmu[26], cell_t[2]
464	CMUS_CMU27_CELL_T4	INT8	1	°C	Cell temperature for cmu[26], cell_t[3]
465	CMUS_CMU27_CELL_T5	INT8	1	°C	Cell temperature for cmu[26], cell_t[4]
466	CMUS_CMU27_CELL_T6	INT8	1	°C	Cell temperature for cmu[26], cell_t[5]
467	CMUS_CMU27_CELL_T7	INT8	1	°C	Cell temperature for cmu[26], cell_t[6]
468	CMUS_CMU27_CELL_T8	INT8	1	°C	Cell temperature for cmu[26], cell_t[7]
469	CMUS_CMU27_CELL_T9	INT8	1	°C	Cell temperature for cmu[26], cell_t[8]
470	CMUS_CMU27_CELL_T10	INT8	1	°C	Cell temperature for cmu[26], cell_t[9]
471	CMUS_CMU27_CELL_T11	INT8	1	°C	Cell temperature for cmu[26], cell_t[10]

472	CMUS_CMU27_CELL_T12	INT8	1	°C	Cell temperature for cmu[26], cell_t[11]
473	CMUS_CMU28_CELL_T1	INT8	1	°C	Cell temperature for cmu[27], cell_t[0]
474	CMUS_CMU28_CELL_T2	INT8	1	°C	Cell temperature for cmu[27], cell_t[1]
475	CMUS_CMU28_CELL_T3	INT8	1	°C	Cell temperature for cmu[27], cell_t[2]
476	CMUS_CMU28_CELL_T4	INT8	1	°C	Cell temperature for cmu[27], cell_t[3]
477	CMUS_CMU28_CELL_T5	INT8	1	°C	Cell temperature for cmu[27], cell_t[4]
478	CMUS_CMU28_CELL_T6	INT8	1	°C	Cell temperature for cmu[27], cell_t[5]
479	CMUS_CMU28_CELL_T7	INT8	1	°C	Cell temperature for cmu[27], cell_t[6]
480	CMUS_CMU28_CELL_T8	INT8	1	°C	Cell temperature for cmu[27], cell_t[7]
481	CMUS_CMU28_CELL_T9	INT8	1	°C	Cell temperature for cmu[27], cell_t[8]
482	CMUS_CMU28_CELL_T10	INT8	1	°C	Cell temperature for cmu[27], cell_t[9]
483	CMUS_CMU28_CELL_T11	INT8	1	°C	Cell temperature for cmu[27], cell_t[10]
484	CMUS_CMU28_CELL_T12	INT8	1	°C	Cell temperature for cmu[27], cell_t[11]
485	CMUS_CMU29_CELL_T1	INT8	1	°C	Cell temperature for cmu[28], cell_t[0]
486	CMUS_CMU29_CELL_T2	INT8	1	°C	Cell temperature for cmu[28], cell_t[1]
487	CMUS_CMU29_CELL_T3	INT8	1	°C	Cell temperature for cmu[28], cell_t[2]
488	CMUS_CMU29_CELL_T4	INT8	1	°C	Cell temperature for cmu[28], cell_t[3]
489	CMUS_CMU29_CELL_T5	INT8	1	°C	Cell temperature for cmu[28], cell_t[4]
490	CMUS_CMU29_CELL_T6	INT8	1	°C	Cell temperature for cmu[28], cell_t[5]
491	CMUS_CMU29_CELL_T7	INT8	1	°C	Cell temperature for cmu[28], cell_t[6]
492	CMUS_CMU29_CELL_T8	INT8	1	°C	Cell temperature for cmu[28], cell_t[7]
493	CMUS_CMU29_CELL_T9	INT8	1	°C	Cell temperature for cmu[28], cell_t[8]
494	CMUS_CMU29_CELL_T10	INT8	1	°C	Cell temperature for cmu[28], cell_t[9]
495	CMUS_CMU29_CELL_T11	INT8	1	°C	Cell temperature for cmu[28], cell_t[10]
496	CMUS_CMU29_CELL_T12	INT8	1	°C	Cell temperature for cmu[28], cell_t[11]

497	CMUS_CMU30_CELL_T1	INT8	1	°C	Cell temperature for cmu[29], cell_t[0]
498	CMUS_CMU30_CELL_T2	INT8	1	°C	Cell temperature for cmu[29], cell_t[1]
499	CMUS_CMU30_CELL_T3	INT8	1	°C	Cell temperature for cmu[29], cell_t[2]
500	CMUS_CMU30_CELL_T4	INT8	1	°C	Cell temperature for cmu[29], cell_t[3]
501	CMUS_CMU30_CELL_T5	INT8	1	°C	Cell temperature for cmu[29], cell_t[4]
502	CMUS_CMU30_CELL_T6	INT8	1	°C	Cell temperature for cmu[29], cell_t[5]
503	CMUS_CMU30_CELL_T7	INT8	1	°C	Cell temperature for cmu[29], cell_t[6]
504	CMUS_CMU30_CELL_T8	INT8	1	°C	Cell temperature for cmu[29], cell_t[7]
505	CMUS_CMU30_CELL_T9	INT8	1	°C	Cell temperature for cmu[29], cell_t[8]
506	CMUS_CMU30_CELL_T10	INT8	1	°C	Cell temperature for cmu[29], cell_t[9]
507	CMUS_CMU30_CELL_T11	INT8	1	°C	Cell temperature for cmu[29], cell_t[10]
508	CMUS_CMU30_CELL_T12	INT8	1	°C	Cell temperature for cmu[29], cell_t[11]
509	CMUS_CMU31_CELL_T1	INT8	1	°C	Cell temperature for cmu[30], cell_t[0]
510	CMUS_CMU31_CELL_T2	INT8	1	°C	Cell temperature for cmu[30], cell_t[1]
511	CMUS_CMU31_CELL_T3	INT8	1	°C	Cell temperature for cmu[30], cell_t[2]
512	CMUS_CMU31_CELL_T4	INT8	1	°C	Cell temperature for cmu[30], cell_t[3]
513	CMUS_CMU31_CELL_T5	INT8	1	°C	Cell temperature for cmu[30], cell_t[4]
514	CMUS_CMU31_CELL_T6	INT8	1	°C	Cell temperature for cmu[30], cell_t[5]
515	CMUS_CMU31_CELL_T7	INT8	1	°C	Cell temperature for cmu[30], cell_t[6]
516	CMUS_CMU31_CELL_T8	INT8	1	°C	Cell temperature for cmu[30], cell_t[7]
517	CMUS_CMU31_CELL_T9	INT8	1	°C	Cell temperature for cmu[30], cell_t[8]
518	CMUS_CMU31_CELL_T10	INT8	1	°C	Cell temperature for cmu[30], cell_t[9]
519	CMUS_CMU31_CELL_T11	INT8	1	°C	Cell temperature for cmu[30], cell_t[10]
520	CMUS_CMU31_CELL_T12	INT8	1	°C	Cell temperature for cmu[30], cell_t[11]
521	CMUS_CMU32_CELL_T1	INT8	1	°C	Cell temperature for cmu[31], cell_t[0]

522	CMUS_CMU32_CELL_T2	INT8	1	°C	Cell temperature for cmu[31], cell_t[1]
523	CMUS_CMU32_CELL_T3	INT8	1	°C	Cell temperature for cmu[31], cell_t[2]
524	CMUS_CMU32_CELL_T4	INT8	1	°C	Cell temperature for cmu[31], cell_t[3]
525	CMUS_CMU32_CELL_T5	INT8	1	°C	Cell temperature for cmu[31], cell_t[4]
526	CMUS_CMU32_CELL_T6	INT8	1	°C	Cell temperature for cmu[31], cell_t[5]
527	CMUS_CMU32_CELL_T7	INT8	1	°C	Cell temperature for cmu[31], cell_t[6]
528	CMUS_CMU32_CELL_T8	INT8	1	°C	Cell temperature for cmu[31], cell_t[7]
529	CMUS_CMU32_CELL_T9	INT8	1	°C	Cell temperature for cmu[31], cell_t[8]
530	CMUS_CMU32_CELL_T10	INT8	1	°C	Cell temperature for cmu[31], cell_t[9]
531	CMUS_CMU32_CELL_T11	INT8	1	°C	Cell temperature for cmu[31], cell_t[10]
532	CMUS_CMU32_CELL_T12	INT8	1	°C	Cell temperature for cmu[31], cell_t[11]
533	MCU_TEST_T1	INT8	1	°C	MCU test temperature 0
534	MCU_TEST_T2	INT8	1	°C	MCU test temperature 1
538	CMUS_ONLINE_COUNT	UINT 8	1	-	Count of online CMUs
539	CMUS_COMM_ERR	UINT 32	1	-	Counter for CMU communication errors since power on or wakeup
540	CMUS_CMU1_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[0]
541	CMUS_CMU1_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[1]
542	CMUS_CMU1_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[2]
543	CMUS_CMU1_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[3]
544	CMUS_CMU1_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[4]
545	CMUS_CMU1_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[5]
546	CMUS_CMU1_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[6]
547	CMUS_CMU1_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[7]
548	CMUS_CMU1_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[8]
549	CMUS_CMU1_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[9]

550	CMUS_CMU1_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[10]
551	CMUS_CMU1_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[0], cell_v[11]
552	CMUS_CMU2_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[0]
553	CMUS_CMU2_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[1]
554	CMUS_CMU2_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[2]
555	CMUS_CMU2_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[3]
556	CMUS_CMU2_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[4]
557	CMUS_CMU2_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[5]
558	CMUS_CMU2_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[6]
559	CMUS_CMU2_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[7]
560	CMUS_CMU2_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[8]
561	CMUS_CMU2_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[9]
562	CMUS_CMU2_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[10]
563	CMUS_CMU2_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[1], cell_v[11]
564	CMUS_CMU3_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[0]
565	CMUS_CMU3_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[1]
566	CMUS_CMU3_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[2]
567	CMUS_CMU3_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[3]
568	CMUS_CMU3_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[4]
569	CMUS_CMU3_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[5]
570	CMUS_CMU3_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[6]
571	CMUS_CMU3_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[7]
572	CMUS_CMU3_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[8]
573	CMUS_CMU3_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[9]
574	CMUS_CMU3_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[10]

575	CMUS_CMU3_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[2], cell_v[11]
576	CMUS_CMU4_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[0]
577	CMUS_CMU4_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[1]
578	CMUS_CMU4_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[2]
579	CMUS_CMU4_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[3]
580	CMUS_CMU4_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[4]
581	CMUS_CMU4_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[5]
582	CMUS_CMU4_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[6]
583	CMUS_CMU4_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[7]
584	CMUS_CMU4_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[8]
585	CMUS_CMU4_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[9]
586	CMUS_CMU4_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[10]
587	CMUS_CMU4_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[3], cell_v[11]
588	CMUS_CMU5_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[0]
589	CMUS_CMU5_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[1]
590	CMUS_CMU5_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[2]
591	CMUS_CMU5_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[3]
592	CMUS_CMU5_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[4]
593	CMUS_CMU5_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[5]
594	CMUS_CMU5_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[6]
595	CMUS_CMU5_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[7]
596	CMUS_CMU5_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[8]
597	CMUS_CMU5_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[9]
598	CMUS_CMU5_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[10]
599	CMUS_CMU5_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[4], cell_v[11]

600	CMUS_CMU6_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[0]
601	CMUS_CMU6_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[1]
602	CMUS_CMU6_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[2]
603	CMUS_CMU6_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[3]
604	CMUS_CMU6_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[4]
605	CMUS_CMU6_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[5]
606	CMUS_CMU6_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[6]
607	CMUS_CMU6_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[7]
608	CMUS_CMU6_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[8]
609	CMUS_CMU6_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[9]
610	CMUS_CMU6_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[10]
611	CMUS_CMU6_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[5], cell_v[11]
612	CMUS_CMU7_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[0]
613	CMUS_CMU7_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[1]
614	CMUS_CMU7_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[2]
615	CMUS_CMU7_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[3]
616	CMUS_CMU7_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[4]
617	CMUS_CMU7_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[5]
618	CMUS_CMU7_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[6]
619	CMUS_CMU7_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[7]
620	CMUS_CMU7_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[8]
621	CMUS_CMU7_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[9]
622	CMUS_CMU7_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[10]
623	CMUS_CMU7_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[6], cell_v[11]
624	CMUS_CMU8_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[0]

625	CMUS_CMU8_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[1]
626	CMUS_CMU8_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[2]
627	CMUS_CMU8_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[3]
628	CMUS_CMU8_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[4]
629	CMUS_CMU8_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[5]
630	CMUS_CMU8_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[6]
631	CMUS_CMU8_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[7]
632	CMUS_CMU8_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[8]
633	CMUS_CMU8_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[9]
634	CMUS_CMU8_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[10]
635	CMUS_CMU8_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[7], cell_v[11]
636	CMUS_CMU9_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[0]
637	CMUS_CMU9_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[1]
638	CMUS_CMU9_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[2]
639	CMUS_CMU9_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[3]
640	CMUS_CMU9_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[4]
641	CMUS_CMU9_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[5]
642	CMUS_CMU9_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[6]
643	CMUS_CMU9_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[7]
644	CMUS_CMU9_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[8]
645	CMUS_CMU9_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[9]
646	CMUS_CMU9_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[10]
647	CMUS_CMU9_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[8], cell_v[11]
648	CMUS_CMU10_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[0]
649	CMUS_CMU10_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[1]

650	CMUS_CMU10_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[2]
651	CMUS_CMU10_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[3]
652	CMUS_CMU10_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[4]
653	CMUS_CMU10_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[5]
654	CMUS_CMU10_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[6]
655	CMUS_CMU10_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[7]
656	CMUS_CMU10_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[8]
657	CMUS_CMU10_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[9]
658	CMUS_CMU10_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[10]
659	CMUS_CMU10_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[9], cell_v[11]
660	CMUS_CMU11_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[0]
661	CMUS_CMU11_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[1]
662	CMUS_CMU11_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[2]
663	CMUS_CMU11_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[3]
664	CMUS_CMU11_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[4]
665	CMUS_CMU11_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[5]
666	CMUS_CMU11_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[6]
667	CMUS_CMU11_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[7]
668	CMUS_CMU11_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[8]
669	CMUS_CMU11_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[9]
670	CMUS_CMU11_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[10]
671	CMUS_CMU11_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[10], cell_v[11]
672	CMUS_CMU12_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[0]
673	CMUS_CMU12_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[1]
674	CMUS_CMU12_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[2]

675	CMUS_CMU12_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[3]
676	CMUS_CMU12_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[4]
677	CMUS_CMU12_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[5]
678	CMUS_CMU12_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[6]
679	CMUS_CMU12_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[7]
680	CMUS_CMU12_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[8]
681	CMUS_CMU12_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[9]
682	CMUS_CMU12_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[10]
683	CMUS_CMU12_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[11], cell_v[11]
684	CMUS_CMU13_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[0]
685	CMUS_CMU13_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[1]
686	CMUS_CMU13_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[2]
687	CMUS_CMU13_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[3]
688	CMUS_CMU13_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[4]
689	CMUS_CMU13_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[5]
690	CMUS_CMU13_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[6]
691	CMUS_CMU13_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[7]
692	CMUS_CMU13_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[8]
693	CMUS_CMU13_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[9]
694	CMUS_CMU13_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[10]
695	CMUS_CMU13_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[12], cell_v[11]
696	CMUS_CMU14_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[0]
697	CMUS_CMU14_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[1]
698	CMUS_CMU14_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[2]
699	CMUS_CMU14_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[3]

700	CMUS_CMU14_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[4]
701	CMUS_CMU14_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[5]
702	CMUS_CMU14_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[6]
703	CMUS_CMU14_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[7]
704	CMUS_CMU14_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[8]
705	CMUS_CMU14_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[9]
706	CMUS_CMU14_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[10]
707	CMUS_CMU14_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[13], cell_v[11]
708	CMUS_CMU15_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[0]
709	CMUS_CMU15_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[1]
710	CMUS_CMU15_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[2]
711	CMUS_CMU15_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[3]
712	CMUS_CMU15_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[4]
713	CMUS_CMU15_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[5]
714	CMUS_CMU15_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[6]
715	CMUS_CMU15_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[7]
716	CMUS_CMU15_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[8]
717	CMUS_CMU15_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[9]
718	CMUS_CMU15_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[10]
719	CMUS_CMU15_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[14], cell_v[11]
720	CMUS_CMU16_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[0]
721	CMUS_CMU16_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[1]
722	CMUS_CMU16_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[2]
723	CMUS_CMU16_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[3]
724	CMUS_CMU16_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[4]

725	CMUS_CMU16_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[5]
726	CMUS_CMU16_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[6]
727	CMUS_CMU16_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[7]
728	CMUS_CMU16_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[8]
729	CMUS_CMU16_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[9]
730	CMUS_CMU16_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[10]
731	CMUS_CMU16_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[15], cell_v[11]
732	CMUS_CMU17_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[0]
733	CMUS_CMU17_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[1]
734	CMUS_CMU17_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[2]
735	CMUS_CMU17_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[3]
736	CMUS_CMU17_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[4]
737	CMUS_CMU17_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[5]
738	CMUS_CMU17_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[6]
739	CMUS_CMU17_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[7]
740	CMUS_CMU17_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[8]
741	CMUS_CMU17_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[9]
742	CMUS_CMU17_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[10]
743	CMUS_CMU17_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[16], cell_v[11]
744	CMUS_CMU18_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[0]
745	CMUS_CMU18_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[1]
746	CMUS_CMU18_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[2]
747	CMUS_CMU18_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[3]
748	CMUS_CMU18_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[4]
749	CMUS_CMU18_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[5]

750	CMUS_CMU18_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[6]
751	CMUS_CMU18_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[7]
752	CMUS_CMU18_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[8]
753	CMUS_CMU18_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[9]
754	CMUS_CMU18_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[10]
755	CMUS_CMU18_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[17], cell_v[11]
756	CMUS_CMU19_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[0]
757	CMUS_CMU19_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[1]
758	CMUS_CMU19_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[2]
759	CMUS_CMU19_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[3]
760	CMUS_CMU19_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[4]
761	CMUS_CMU19_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[5]
762	CMUS_CMU19_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[6]
763	CMUS_CMU19_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[7]
764	CMUS_CMU19_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[8]
765	CMUS_CMU19_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[9]
766	CMUS_CMU19_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[10]
767	CMUS_CMU19_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[18], cell_v[11]
768	CMUS_CMU20_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[0]
769	CMUS_CMU20_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[1]
770	CMUS_CMU20_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[2]
771	CMUS_CMU20_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[3]
772	CMUS_CMU20_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[4]
773	CMUS_CMU20_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[5]
774	CMUS_CMU20_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[6]

775	CMUS_CMU20_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[7]
776	CMUS_CMU20_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[8]
777	CMUS_CMU20_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[9]
778	CMUS_CMU20_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[10]
779	CMUS_CMU20_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[19], cell_v[11]
780	CMUS_CMU21_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[0]
781	CMUS_CMU21_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[1]
782	CMUS_CMU21_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[2]
783	CMUS_CMU21_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[3]
784	CMUS_CMU21_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[4]
785	CMUS_CMU21_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[5]
786	CMUS_CMU21_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[6]
787	CMUS_CMU21_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[7]
788	CMUS_CMU21_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[8]
789	CMUS_CMU21_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[9]
790	CMUS_CMU21_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[10]
791	CMUS_CMU21_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[20], cell_v[11]
792	CMUS_CMU22_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[0]
793	CMUS_CMU22_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[1]
794	CMUS_CMU22_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[2]
795	CMUS_CMU22_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[3]
796	CMUS_CMU22_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[4]
797	CMUS_CMU22_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[5]
798	CMUS_CMU22_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[6]
799	CMUS_CMU22_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[7]

800	CMUS_CMU22_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[8]
801	CMUS_CMU22_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[9]
802	CMUS_CMU22_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[10]
803	CMUS_CMU22_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[21], cell_v[11]
804	CMUS_CMU23_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[0]
805	CMUS_CMU23_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[1]
806	CMUS_CMU23_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[2]
807	CMUS_CMU23_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[3]
808	CMUS_CMU23_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[4]
809	CMUS_CMU23_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[5]
810	CMUS_CMU23_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[6]
811	CMUS_CMU23_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[7]
812	CMUS_CMU23_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[8]
813	CMUS_CMU23_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[9]
814	CMUS_CMU23_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[10]
815	CMUS_CMU23_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[22], cell_v[11]
816	CMUS_CMU24_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[0]
817	CMUS_CMU24_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[1]
818	CMUS_CMU24_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[2]
819	CMUS_CMU24_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[3]
820	CMUS_CMU24_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[4]
821	CMUS_CMU24_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[5]
822	CMUS_CMU24_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[6]
823	CMUS_CMU24_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[7]
824	CMUS_CMU24_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[8]

825	CMUS_CMU24_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[9]
826	CMUS_CMU24_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[10]
827	CMUS_CMU24_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[23], cell_v[11]
828	CMUS_CMU25_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[0]
829	CMUS_CMU25_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[1]
830	CMUS_CMU25_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[2]
831	CMUS_CMU25_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[3]
832	CMUS_CMU25_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[4]
833	CMUS_CMU25_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[5]
834	CMUS_CMU25_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[6]
835	CMUS_CMU25_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[7]
836	CMUS_CMU25_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[8]
837	CMUS_CMU25_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[9]
838	CMUS_CMU25_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[10]
839	CMUS_CMU25_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[24], cell_v[11]
840	CMUS_CMU26_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[0]
841	CMUS_CMU26_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[1]
842	CMUS_CMU26_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[2]
843	CMUS_CMU26_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[3]
844	CMUS_CMU26_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[4]
845	CMUS_CMU26_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[5]
846	CMUS_CMU26_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[6]
847	CMUS_CMU26_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[7]
848	CMUS_CMU26_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[8]
849	CMUS_CMU26_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[9]

850	CMUS_CMU26_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[10]
851	CMUS_CMU26_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[25], cell_v[11]
852	CMUS_CMU27_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[0]
853	CMUS_CMU27_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[1]
854	CMUS_CMU27_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[2]
855	CMUS_CMU27_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[3]
856	CMUS_CMU27_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[4]
857	CMUS_CMU27_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[5]
858	CMUS_CMU27_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[6]
859	CMUS_CMU27_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[7]
860	CMUS_CMU27_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[8]
861	CMUS_CMU27_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[9]
862	CMUS_CMU27_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[10]
863	CMUS_CMU27_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[26], cell_v[11]
864	CMUS_CMU28_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[0]
865	CMUS_CMU28_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[1]
866	CMUS_CMU28_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[2]
867	CMUS_CMU28_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[3]
868	CMUS_CMU28_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[4]
869	CMUS_CMU28_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[5]
870	CMUS_CMU28_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[6]
871	CMUS_CMU28_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[7]
872	CMUS_CMU28_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[8]
873	CMUS_CMU28_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[9]
874	CMUS_CMU28_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[10]

875	CMUS_CMU28_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[27], cell_v[11]
876	CMUS_CMU29_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[0]
877	CMUS_CMU29_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[1]
878	CMUS_CMU29_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[2]
879	CMUS_CMU29_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[3]
880	CMUS_CMU29_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[4]
881	CMUS_CMU29_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[5]
882	CMUS_CMU29_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[6]
883	CMUS_CMU29_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[7]
884	CMUS_CMU29_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[8]
885	CMUS_CMU29_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[9]
886	CMUS_CMU29_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[10]
887	CMUS_CMU29_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[28], cell_v[11]
888	CMUS_CMU30_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[0]
889	CMUS_CMU30_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[1]
890	CMUS_CMU30_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[2]
891	CMUS_CMU30_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[3]
892	CMUS_CMU30_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[4]
893	CMUS_CMU30_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[5]
894	CMUS_CMU30_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[6]
895	CMUS_CMU30_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[7]
896	CMUS_CMU30_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[8]
897	CMUS_CMU30_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[9]
898	CMUS_CMU30_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[10]
899	CMUS_CMU30_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[29], cell_v[11]

900	CMUS_CMU31_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[0]
901	CMUS_CMU31_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[1]
902	CMUS_CMU31_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[2]
903	CMUS_CMU31_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[3]
904	CMUS_CMU31_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[4]
905	CMUS_CMU31_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[5]
906	CMUS_CMU31_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[6]
907	CMUS_CMU31_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[7]
908	CMUS_CMU31_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[8]
909	CMUS_CMU31_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[9]
910	CMUS_CMU31_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[10]
911	CMUS_CMU31_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[30], cell_v[11]
912	CMUS_CMU32_CELL_V1	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[0]
913	CMUS_CMU32_CELL_V2	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[1]
914	CMUS_CMU32_CELL_V3	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[2]
915	CMUS_CMU32_CELL_V4	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[3]
916	CMUS_CMU32_CELL_V5	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[4]
917	CMUS_CMU32_CELL_V6	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[5]
918	CMUS_CMU32_CELL_V7	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[6]
919	CMUS_CMU32_CELL_V8	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[7]
920	CMUS_CMU32_CELL_V9	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[8]
921	CMUS_CMU32_CELL_V10	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[9]
922	CMUS_CMU32_CELL_V11	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[10]
923	CMUS_CMU32_CELL_V12	UINT 16	0.1	mV	Cell voltage for cmu[31], cell_v[11]
955	TIME_BETWEEN_BOOTS_S	UINT 32	1	sec	Time between this boot and the last

956	EPOCH_TIME_S	UINT 32	1	sec	System time (boot time plus uptime)
957	UPTIME_S	UINT 32	1	sec	System up time since power on or wakeup
958	HALL_EXT_V_HALL_V_LO	UINT 16	0.1	mV	Voltage measurement from HALL low channel, external ADC
959	HALL_EXT_V_HALL_V_HI	UINT 16	0.1	mV	Voltage measurement from HALL high channel, external ADC
960	HALL_EXT_V_HALL_V_SUPPLY	UINT 16	0.1	mV	Voltage measurement from HALL supply circuit, external ADC
961	HALL_SYS_V_HALL_V_LO	UINT 16	0.1	mV	Voltage measurement from HALL low channel, internal ADC
962	HALL_SYS_V_HALL_V_HI	UINT 16	0.1	mV	Voltage measurement from HALL high channel, internal ADC
963	HALL_SYS_V_HALL_V_SUPPLY	UINT 16	0.1	mV	Voltage measurement from HALL supply circuit, internal ADC
964	HALL_V_HALL_V_SUPPLY	UINT 16	0.1	mV	Voltage level from HALL supply, master
965	CELL_T_MIN_VAL	INT8	1	°C	Lowest cell temperature
966	CELL_T_MIN_ID_CMU	UINT 8	1	-	CMU of lowest cell temperature
967	CELL_T_MIN_ID_CELL	UINT 8	1	-	Cell number of cell w. lowest temperature
968	CELL_T_MAX_VAL	INT8	1	°C	Highest cell temperature
969	CELL_T_MAX_ID_CMU	UINT 8	1	-	CMU of highest cell temperature
970	CELL_T_MAX_ID_CELL	UINT 8	1	-	Cell number of cell w. highest temperature
971	CELL_T_AVG	INT8	1	°C	Average cell temperature
972	CELL_T_NUM_AVAILABLE	UINT 16	1	-	Number of cell temperature configured
973	BALANCING_BALANCE_SETTING_CMU1_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
974	BALANCING_BALANCE_SETTING_CMU2_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
975	BALANCING_BALANCE_SETTING_CMU3_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
976	BALANCING_BALANCE_SETTING_CMU4_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
977	BALANCING_BALANCE_SETTING_CMU5_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)

978	BALANCING_BALANCE_SETTING_CMU6_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
979	BALANCING_BALANCE_SETTING_CMU7_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
980	BALANCING_BALANCE_SETTING_CMU8_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
981	BALANCING_BALANCE_SETTING_CMU9_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
982	BALANCING_BALANCE_SETTING_CMU10_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
983	BALANCING_BALANCE_SETTING_CMU11_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
984	BALANCING_BALANCE_SETTING_CMU12_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
985	BALANCING_BALANCE_SETTING_CMU13_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
986	BALANCING_BALANCE_SETTING_CMU14_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
987	BALANCING_BALANCE_SETTING_CMU15_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
988	BALANCING_BALANCE_SETTING_CMU16_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
989	BALANCING_BALANCE_SETTING_CMU17_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
990	BALANCING_BALANCE_SETTING_CMU18_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
991	BALANCING_BALANCE_SETTING_CMU19_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
992	BALANCING_BALANCE_SETTING_CMU20_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
993	BALANCING_BALANCE_SETTING_CMU21_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)

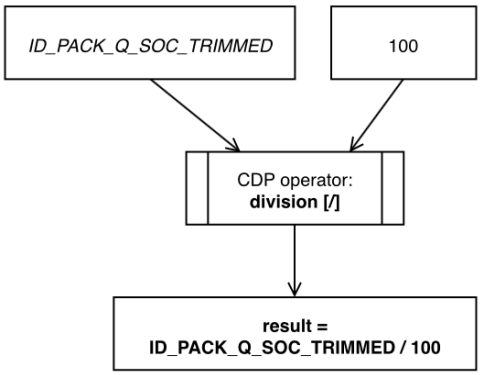
994	BALANCING_BALANCE_SETTING_CMU22_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
995	BALANCING_BALANCE_SETTING_CMU23_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
996	BALANCING_BALANCE_SETTING_CMU24_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
997	BALANCING_BALANCE_SETTING_CMU25_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
998	BALANCING_BALANCE_SETTING_CMU26_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
999	BALANCING_BALANCE_SETTING_CMU27_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1000	BALANCING_BALANCE_SETTING_CMU28_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1001	BALANCING_BALANCE_SETTING_CMU29_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1002	BALANCING_BALANCE_SETTING_CMU30_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1003	BALANCING_BALANCE_SETTING_CMU31_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1004	BALANCING_BALANCE_SETTING_CMU32_CELL_BITMASK	UINT 16	1	-	Cells being balanced as bitmask (b0 = cell 0, ... b11 = cell 11, b12-b15 = always zero)
1005	CDP_OUTPUT1	UINT 32	-	-	Custom Data Processor output number 1
1006	CDP_OUTPUT2	UINT 32	-	-	Custom Data Processor output number 2
1007	CDP_OUTPUT3	UINT 32	-	-	Custom Data Processor output number 3
1008	CDP_OUTPUT4	UINT 32	-	-	Custom Data Processor output number 4
1009	CDP_OUTPUT5	UINT 32	-	-	Custom Data Processor output number 5
1010	CDP_OUTPUT6	UINT 32	-	-	Custom Data Processor output number 6
1011	CDP_OUTPUT7	UINT 32	-	-	Custom Data Processor output number 7
1012	CDP_OUTPUT8	UINT 32	-	-	Custom Data Processor output number 8

1013	CDP_OUTPUT9	UINT 32	-	-	Custom Data Processor output number 9
1014	CDP_OUTPUT10	UINT 32	-	-	Custom Data Processor output number 10
1015	CDP_OUTPUT11	UINT 32	-	-	Custom Data Processor output number 11
1016	CDP_OUTPUT12	UINT 32	-	-	Custom Data Processor output number 12
1017	CDP_OUTPUT13	UINT 32	-	-	Custom Data Processor output number 13
1018	CDP_OUTPUT14	UINT 32	-	-	Custom Data Processor output number 14
1019	CDP_OUTPUT15	UINT 32	-	-	Custom Data Processor output number 15
1020	CDP_OUTPUT16	UINT 32	-	-	Custom Data Processor output number 16
1021	CDP_OUTPUT17	UINT 32	-	-	Custom Data Processor output number 17
1022	CDP_OUTPUT18	UINT 32	-	-	Custom Data Processor output number 18
1023	CDP_OUTPUT19	UINT 32	-	-	Custom Data Processor output number 19
1024	CDP_OUTPUT20	UINT 32	-	-	Custom Data Processor output number 20
1025	CDP_OUTPUT21	UINT 32	-	-	Custom Data Processor output number 21
1026	CDP_OUTPUT22	UINT 32	-	-	Custom Data Processor output number 22
1027	CDP_OUTPUT23	UINT 32	-	-	Custom Data Processor output number 23
1028	CDP_OUTPUT24	UINT 32	-	-	Custom Data Processor output number 24
1029	CDP_OUTPUT25	UINT 32	-	-	Custom Data Processor output number 25

8.5 Appendix 5: Custom data processing examples

8.5.1 Scaling the SOC from 0.01% resolution to 1 % resolution

1. *lvalue* is addressed data from the ID MAP using ID: *ID_PACK_Q_SOC_TRIMMED*
2. *rvalue* is constant data with the value 100
3. The *operator* is division [/]



8.5.2 Scaling and offset of pack current

Configure CDP output #1 with:

1. *lvalue* is addressed data from the ID MAP using ID: *ID_PACK_I_MASTER*
2. *rvalue* is constant data with value 48
3. *operator* is multiplication [*]

Result is stored in *ID_CDP_OUTPUT1*

Configure CDP output #2 with:

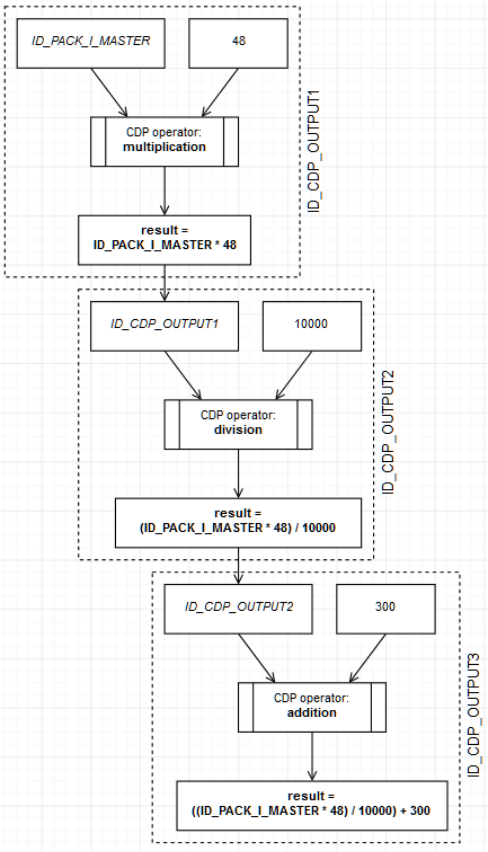
4. *lvalue* is addressed data from the ID MAP using ID: *ID_CDP_OUTPUT1* (CDP result from before)
5. *rvalue* is constant data with value 10000
6. *operator* is division [/]

Result is stored in *ID_CDP_OUTPUT2*

Configure CDP output #3 with:

7. *lvalue* is addressed data from the ID MAP using ID: *ID_CDP_OUTPUT2* (CDP result from before)
8. *rvalue* is constant data with value 300
9. *operator* is addition [+]

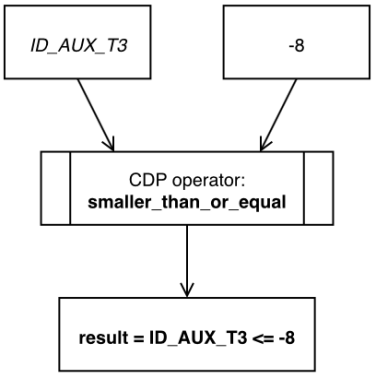
Result is stored in *ID_CDP_OUTPUT3*, where it can be configured to be broadcasted on the CAN bus.



8.5.3 Setting up custom warning bit with CDP

1. *lvalue* is addressed data from the ID MAP using ID: *ID_AUX_T3*
2. *rvalue* is constant data with the value -8
3. The *operator* is less than or equal [<=]

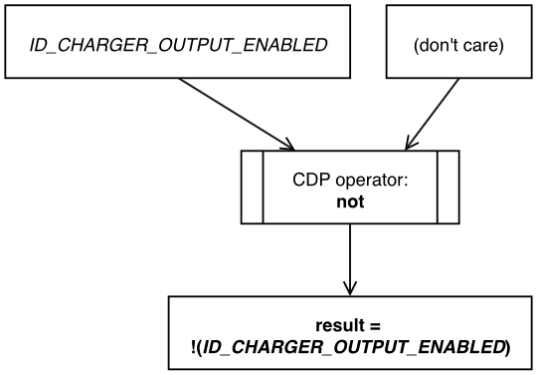
The result is then 1 if *ID_AUX_T3* is less than or equal to -8, otherwise the result is 0.



8.5.4 Inverting charger enabled bit

1. *lvalue* is addressed data from the ID MAP using ID: *ID_CHARGER_OUTPUT_ENABLED*
2. *rvalue* is not used, but a constant data with the value 0 is used
3. The *operator* is not [!]

The result is then the inverse (binary not) of the *lvalue*.



8.6 Appendix 6: OCV-SoC experimental battery data example

8.6.1 Introduction

8.6.1.1 Purpose

To use the SoC-OCV calibration functionality available in the BMS-firmware it is necessary to have data sets where OCV is measured against SoC. The purpose of implementing the SoC-OCV calibration functionality into the BMS-firmware is to supplement the BMS SOC-prediction based on coulomb counting.

Such OCV-SOC calibration curves might be obtained from the cell manufacturer or can alternatively be measured by the user. The following sections describes how such measurements have been performed at Lithium Balance.

The objective of the tests performed here has been to measure the relationship between SoC and OCV at constant operating conditions and to investigate the influence of relaxation time on the accuracy of the OCV-values measured.

8.6.1.2 OCV-SOC data requirements

The OCV-SOC input to the BMS should be data set(s) with 101 voltage readings, one for each percentage step from 0 % - 100 %. These data can be provided for up to 5 temperatures. The measurements described here were done at 5 or 10 percentage steps and then the points in between the actual measurements were calculated by linear interpolation from the two nearest data points.

8.6.1.3 The test set-up

To characterise the cell a battery cycler / potentiostat [14] was used to perform full discharge (100% SOC -> 0%) sequences at room temperature and at discharge currents of 0.5 C. The battery cycler is basically a load and a charger combined in one box with the ability to program charge, discharge and rest sequences on several cells and to record data (time, potential, current, temperature).

Four test cells[15] were used, each cell had a nominal capacity on 2450 mAh. 3 test cells were mounted on a prototype board, fused and each connected to a dedicated channel on the battery cycler from the positive and negative terminals on the cells. Further a thermo element was attached on each test cell giving the possibility of measuring the cell temperature along with the charge/discharge sequence. The battery cycler was connected to a PC via ethernet connection and controlled via the software PC-batterylab.

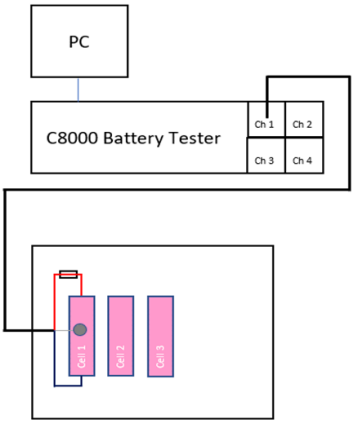


Figure 2 Sketch of the test-setup with the test cells mounted on the prototype board connected to the battery tester with cables to the terminals and a magnetic thermo element

8.6.2 Experimental results

8.6.2.1 Nominal capacity

According to the suppliers datasheet, the maximum cell voltage was 4200 mV (SOC = 100%) and the minimum cell voltage was 3000 mV (SOC = 0%).

Initially all test cells were charged to 4200 mV at minimum charge current (which in our case was 250 mA). 20 minutes later the cells were discharged to 3000 mV at 1C. By integrating the (time,

[14] Cadex C8000

[15] Panasonic CGR18650DA

current)-curve (coulomb counting) the nominal capacity of each cell (at 1C discharge rate, constant current and 22 °C) was found to be 2196, 2162 and 2217 mAh for the three cells.

8.6.2.2 OCV-SOC Test sequence

The first step of the test sequence is a full charge as described above. 20 minutes later the discharge steps were initiated. Discharging was done by a constant current at 0.5 C in steps corresponding to SOC percentages (approximately) at 95, 90, 80, 70, 60, 50, 40, 30, 20, 10, 5 and 0. The time to achieve these SOC steps were calculated from the nominal capacity measured above.

Between each step the cells rested for 20 minutes while the cell voltages were measured.

To avoid too big data files the battery cycler was programmed to log data every second the first and last 30 seconds of a charge or resting period (where the curves are typically showing the highest / lowest slopes) and every 10 seconds in between (where the curves are typically flat).

8.6.2.3 Results

Graphical representations of the measurements are shown below.

Figure 3 shows the voltage response to the current drawn from the 3 cells. The cells are discharged from being fully charged to fully discharged at a C-rate on (approximately) 0.5.

In Figure 4, the cell voltages are shown with the SOC calculated from coulomb counting based on the actual measured capacity for each cell.

Figure 5 shows the SoC-OCV graph where each cell SoC is plotted against the stabilized OCV (the cell voltages measured after 20 minutes. In Figure 5 it is noted that in some regions, 1% change of SOC-scale corresponds to less than 3 mV change of OCV. Consequently, it is essential to assure that the relaxation time when measuring OCV is sufficiently long to assure a complete stabilisation of measured cell voltage. The necessary relaxation time may vary significantly from cell type to cell types and will have to be determined experimentally for each type of cells.

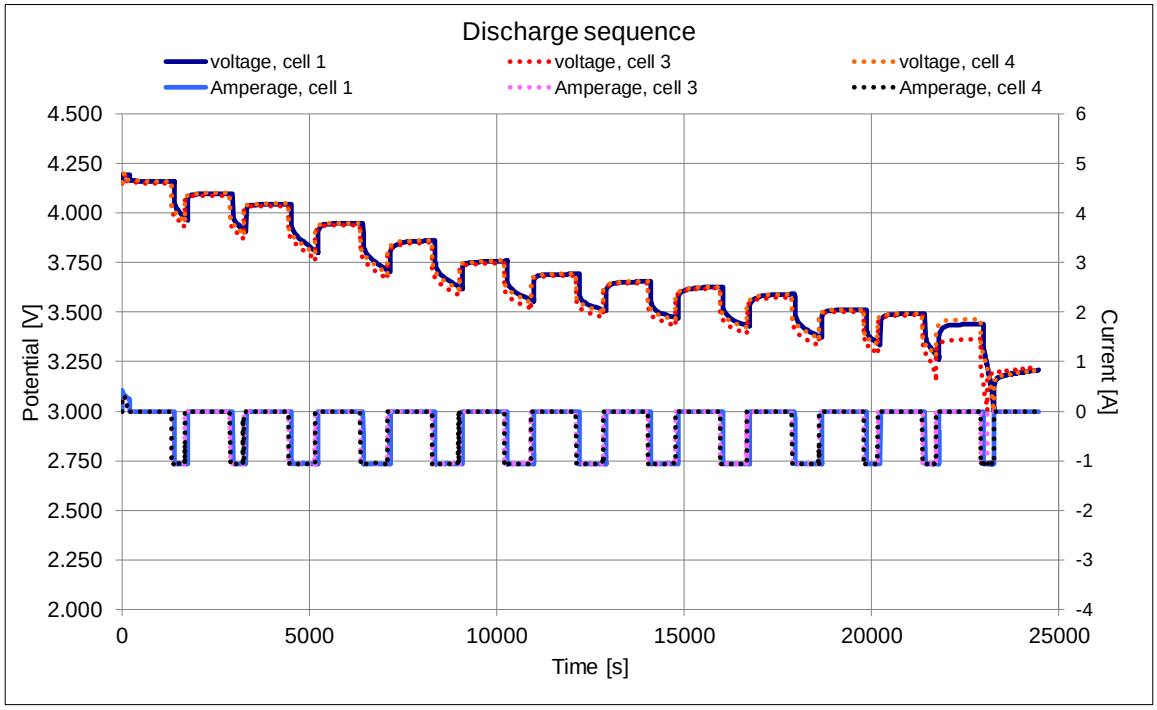


Figure 3 The discharge current and the voltage response for each cell during the full discharge sequence is shown. Note the last short discharge step where the cell voltages reach the lower voltage boundary which is made to make sure the cells are fully discharged at the measurement that corresponds to 0 % SoC.

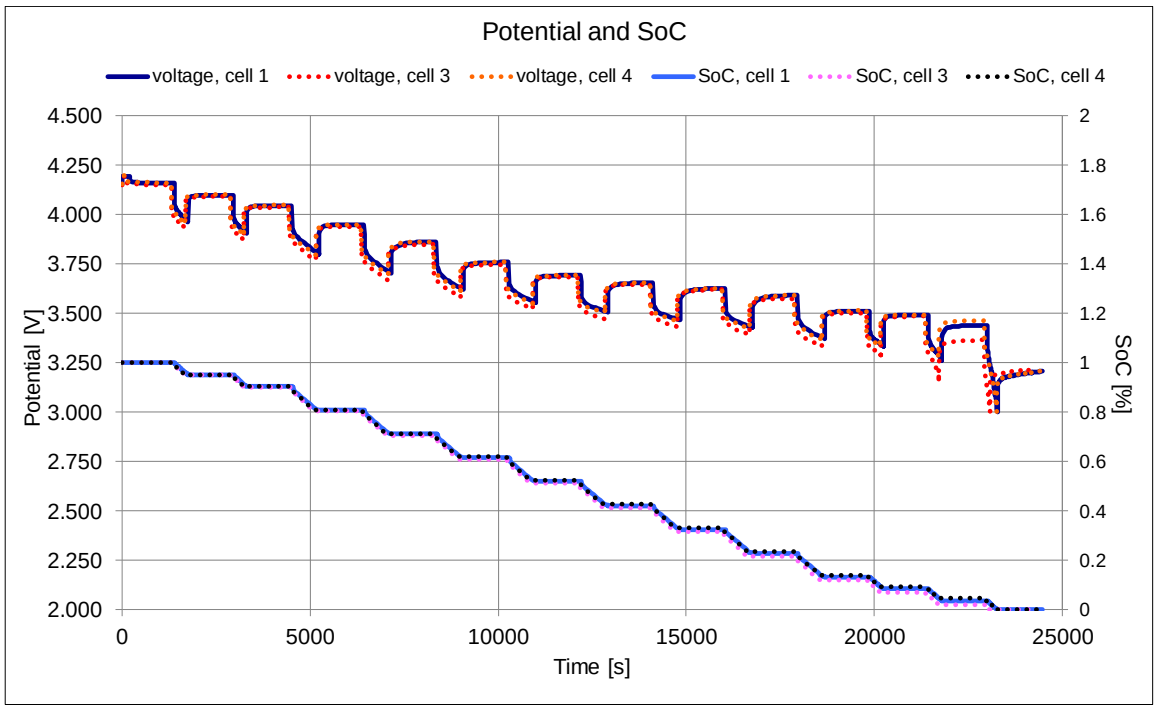


Figure 4 Cell voltages and corresponding SOC as calculated individually for each cell based on the actual capacity seen for each cell during discharge. Those are the data that is plotted against each other in the OCV-SoC curve in Figure 5.

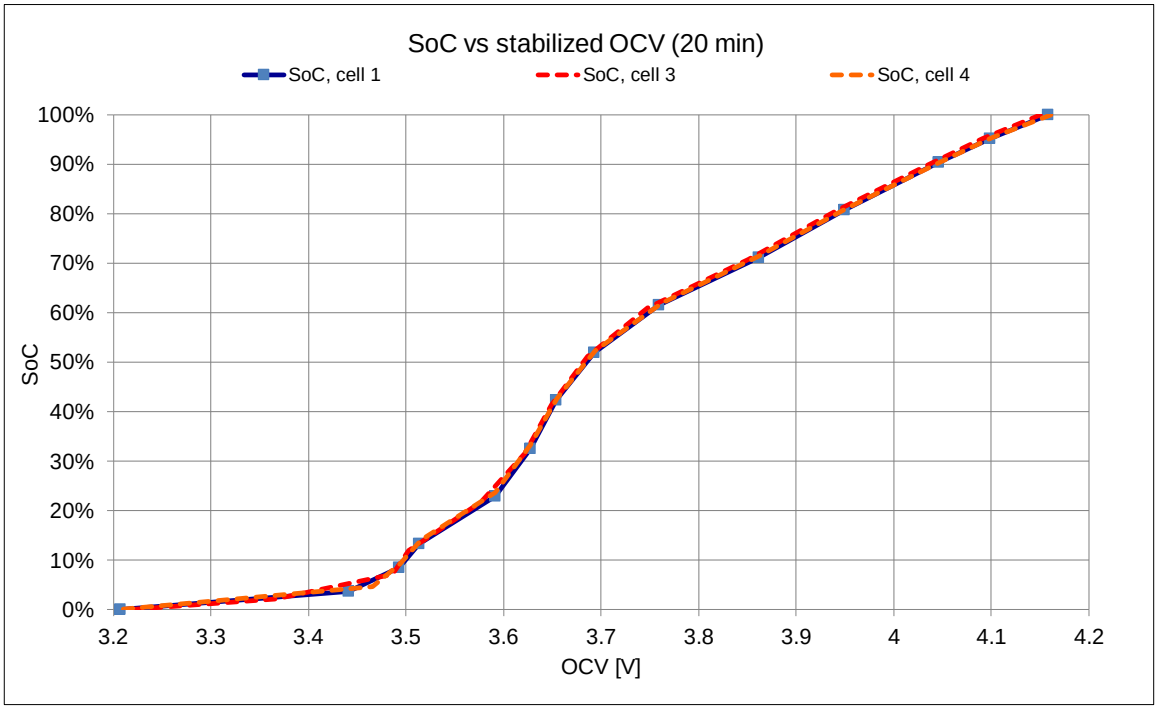


Figure 5 SoC-OCV calibration curve. The light blue squares are measurements, lines are interpolated values. The data represented here is what the BMS need for input for the SOC-OCV calibration.

8.6.3 Conclusion

Based on relatively simple cell cycling experiments it is possible to measure the OCV-SOC calibration curves needed for Lithium Balance SOC-calibration. It should however be noted that these measurements are not trivial and that special attention should be given to:

- Assuring sufficient relaxation time (at 0 current) to allow the OCV to stabilise fully. This may take more than 20 minutes and will have to be measured for each cell type.
- Having a constant temperature during the measurements. It should also be noted, that relaxation time typically will be a function of temperature

8.6.3.1 Future work

The accuracy of the SoC-OCV relationship depends on a wide range of parameters such as temperature, C-rate, battery chemistry, relaxation time, charge/discharge sequence, voltage limits, etc. as discussed for example in [1]. These needs to be understood for the relevant cells in order to evaluate the expected SOC accuracy for any specific cells.

The characterisation methods used here for the SOC-OCV calibration can also be used to evaluate the internal resistance of the battery cells.

8.6.3.2 Reference documents

[1] Maitane Garmendia: “STATE-OF-CHARGE (SOC) ALGORITHM DESIGN METHODOLOGY FOR IMPLEMENTATION ON BATTERY MANAGEMENT SYSTEMS (BMS) OF INDUSTRIAL LI-ION BATTERY PACKS”

8.7 Configuration parameters

View	Name	Description	Unit	Accepted values
System Configuration	Currrent source type	Select sensor to be used for current measurement. Only one sensor can be used at a time. 0 = none, 1 = HALL, 2 = Shunt.	-	0-2
System Configuration	Shunt resistance	Resistance value of the shunt sensor. This value is usually indicated as a voltage output at an ampere level but can be converted to a resistance value.	uΩ	0.0 – 429496729.5
System Configuration	Shunt sensor offset	Offset value for the shunt sensor measurement. This value is added to the converted current measurement to correct for any offset.	mA	(-32768) – 32767
System Configuration	Hall sensor scaling low	The measured low sensor voltage will to converted to a current value using this factor.	mV/A	0 – 4294967295
System Configuration	Hall sensor scaling high	The measured low sensor voltage will to converted to a current value using this factor.	mV/A	0 – 4294967295
System Configuration	Hall sensor offset	Offset value for the Hall sensor measurement. This value is added to the converted current measurement to correct for any offset.	mA	(-32768) – 32767
System Configuration	Hall sensor low sat. Pos.	The lower saturation point, where the firmware switches to the high hall sensor. Below this value, the high sensor is used. Above, the low sensor is used.	mV	0-6563.5
System Configuration	Hall sensor high sat. Pos.	The upper saturation point, where the firmware switches to the high hall sensor. Above this value, the high sensor is used. Below, the low sensor is used.	mV	0-6563.5

System Configuration	Initial Capacity	The initial capacity is the reference point for the SoC estimation. It indicates to the BMS what the initial capacity of the connected cells was from new.	Ah	0.0-6553.5
System Configuration	Minimum SoC trim	A minimum value indicated by the trimmed SoC output can be setup. This ensures that a SoC value lower than this is never indicated. If the internal value is lower, the trimmed SoC will indicate the Minimum SoC trim value.	-	0.00-655.35
System Configuration	Maximum SoC trim	A maximum value indicated by the trimmed SoC output can be setup. This ensures that a SoC value higher than this is never indicated. If the internal value is higher, the trimmed SoC will indicate the Maximum SoC trim value.	-	0.00-655.35
System Configuration	Aux NTC mantissa A	The significant digits of the Aux A coefficient.	-	0-65535
System Configuration	Aux NTC exponent A	The exponent of the Aux A coefficient.	-	(-128)-127
System Configuration	Aux NTC mantissa B	The significant digits of the Aux B coefficient.	-	0-65535
System Configuration	Aux NTC exponent B	The exponent of the Aux B coefficient.	-	(-128)-127
System Configuration	Aux NTC mantissa D	The significant digits of the Aux D coefficient.	-	0-65535
System Configuration	Aux NTC exponent D	The exponent of the Aux D coefficient.	-	(-128)-127
System Configuration	CMUs NTC mantissa A	The significant digits of the CMUs A coefficient.	-	0-65535
System Configuration	CMUs NTC exponent A	The exponent of the Aux A coefficient.	-	(-128)-127
System Configuration	CMUs NTC mantissa B	The significant digits of the CMUs B coefficient.	-	0-65535
System Configuration	CMUs NTC exponent B	The exponent of the Aux B coefficient.	-	(-128)-127
System Configuration	CMUs NTC mantissa D	The significant digits of the CMUs D coefficient.	-	0-65535

System Configuration	CMUs NTC exponent D	The exponent of the Aux D coefficient.	-	(-128)-127
Operational Limits	Min. cell voltage	The minimum allowed cell voltage. If a cell voltage is detected to be below this threshold, an error code is indicated (2000). This error will, depending on error settings, limit allowed operation from the BMS.	mV	0.0-4999.9
Operational Limits	Max. cell voltage	The maximum allowed cell voltage. If a cell voltage is detected to be above this threshold, an error code is indicated (2001). This error will, depending on error settings, limit allowed operation from the BMS.	mV	0.0-4999.9
Operational Limits	Max. current in	The maximum allowed current in to the battery pack (charging current). If the current is detected to be above this threshold, an error code is indicated (2008). This error will depending on error settings limit allowed operation from the BMS.	A	0.0 - 6553.5
Operational Limits	Max. current out	The maximum allowed current out of the battery pack. If the current is detected to be above this threshold, an error code is indicated (2009). This error will depending on error settings limit allowed operation from the BMS.	A	0.0 - 6553.5
Operational Limits	Max. i2t	The Max. i2t is a digital implementation of a melting fuse. The value set is the limit for the calculated “i2t sum”. Low values mean only a small “i2t sum” will flag an error.	A2s	0 – 4294967295

Operational Limits	Dynamic curr. In deadband	This parameter determines when the switch from "Max. current in" to "regulated current" happens for the calculated parameter "dynamic current in limit". A typical value is 2.0 A.	A	0.0 - 6553.5
Operational Limits	Max. contactor break curr.	Before the BMS opens the battery pack main contactors it will test if the current flowing at the given time, is higher than this threshold. If the current is higher it will wait for a maximum time according to the “Contactors off timeout” and then open the contactors.	mA	0.00 - 42949672.95
Operational Limits	Max. precharge end curr.	End criteria for the precharge sequence is if the current is below this limit within the “Precharge timeout” setting. When the current is below the threshold, the precharge sequence will end successfully. If the current does not drop below this threshold within the timeframe, the precharge sequence will fail.	mA	0.00 - 42949672.95
Operational Limits	Max. contactor retries	If a contactor sequence fails, this setting will limit the number of retries to protect the contactors from damage by excessive wear from contactor cycling.	-	0 - 255
Operational Limits	Contactors off timeout	Threshold for the time the BMS is allowed to wait for the system current to drop below the accepted level for contactor opening under load. This test is bypassed if an emergency off command is issued.	sec	0.0 - 6553.5

Operational Limits	Precharge timeout	Time the BMS will wait for the current to drop below the threshold “Max. precharge end curr.” before it will consider the precharge attempt as failed.	sec	0.0 - 6553.5
Operational Limits	Contactor retry timeout	Time the BMS will wait after a failed precharge attempt before trying again.	sec	0.0 - 6553.5
Operational Limits	Temp. sensors enabled	Bitmask of which temperature sensors are enabled. All sensors are enabled by the binary value 00111111 = 63 decimal. charge complete (ignoring current direction). Once Bit 1 is sensor 1, Bit 2 is sensor 2 ect.	-	0 - 63
Operational Limits	Min. temp channel 1-6	The minimum allowed temperature on this temperature channel before the error is given.	C	(-128) - 127
Operational Limits	Max. temp channel 1-6	The maximum allowed temperature on this temperature channel before the error is given.	C	(-128) - 127
Operational Limits	Balancing lower limit	Below this cell voltage limit, no balancing will be any cell.	mV	0.0 - 4999.9
Operational Limits	Balancing deadband	Deadband added to average cell voltage. Above this limit balancing is performed.	mV	0.0 - 4999.9
Charger	Charge complete deadband I (current)	Threshold around 0 A for flagging charge complete. Once all measured current is inside the threshold charge complete may be flagged (if voltage is also within threshold)	mA	0 - 65535
Charger	Charge complete deadband V (voltage)	Threshold around target cell voltage for flagging charge complete. Once all measured cell voltages are inside the threshold charge complete may be	mV	0.0 – 6553.5

		flagged (if current is also within threshold)		
Charger	CAN Charge enabled	Selection to set CAN charger output on. 0 = Off, 1 = On.	-	0 - 1
Charger	CAN Charge Max. V	Maximum output voltage from the CAN charger.		0.0 – 6553.5
Charger	PWM Charge enabled	Selection to set PWM charge signal on. 0 = Off, 1 = On.	-	0 - 1
Charger	PWM signal inverted	Selection to invert PWM signal output. 0 = Off, 1 = On.	-	0 - 1
Charger	PWM Min. duty	Cap PWM output to never output a duty cycle below this threshold.	%	0-100
Charger	PWM Max. duty	Cap PWM output to never output a duty cycle above this threshold.	%	0-100
Charger	PWM output I deadband	If the charge output current is within this deadband, with reference to the set point charge current, the PWM output will stop regulating and stay at the previous value.	A	0.0 – 6553.5
Charger	Cell voltage target	The target voltage for all cells when charging. The BMS will use this value as the end condition for a charge sequence and the calibration point for the capacity sum used for SOC estimation.	mV	0.0 – 4999.9
Charger	Min. Charge current	This threshold is the minimum set point when in normal charge mode. It is used to force a small constant current into the battery pack when the charge cycle is almost complete.	A	0.0 – 6553.5
Charger	Max. Charge current	This threshold is the maximum current set point for the charge current into the battery pack during normal charging.	A	0.0 – 6553.5

Charger	PID constant Kp	P coefficient for PID controller	-	(-2000.001)-2000.001
Charger	PID constant Ki	I coefficient for PID controller	-	(-2000.001)-2000.001
Charger	PID constant Kd	D coefficient for PID controller	-	(-2000.001)-2000.001
GPIO mapping	Activate balancing	Input of where to map “Activate balancing” functionality. When the Input is activated the BMS will activate balancing. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Activate sleep	Input of where to map “Activate sleep” functionality. When the Input is activated the BMS will activate sleep mode. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Load positive	Output to drive Load positive contactor. 0 = Disabled. 5-8 indicates IO channel.	IO	0-16
GPIO mapping	Charge negative	Output to drive Charge Negative contactor. 0 = Disabled. 5-8 indicates IO channel.	IO	0-16
GPIO mapping	Precharge	Output to drive Precharge contactor. 0 = Disabled. 5-8 indicates Input / Output channel.	IO	0-16
GPIO mapping	Load negative	Output to drive Load negative contactor. 0 = Disabled. 5-8 indicates IO channel.	IO	0-16
GPIO mapping	Load positive feedback	Feedback for the charge negative contactor. If enabled, the BMS expects this feedback signal to follow the state of “Charge negative”. If there is a mismatch, an error is given and the contactor sequence will fail. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16

GPIO mapping	Charge negative feedback	Feedback for the charge negative contactor. If enabled, the BMS expects this feedback signal to follow the state of "Charge negative". If there is a mismatch, an error is given and the contactor sequence will fail. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Precharge feedback	Feedback for the precharge contactor. If enabled, the BMS expects this feedback signal to follow the state of "Precharge". If there is a mismatch, an error is given and the contactor sequence will fail. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Load negative feedback	Feedback for the load negative contactor. If enabled, the BMS expects this feedback signal to follow the state of "Load negative". If there is a mismatch, an error is given and the contactor sequence will fail. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Request combined active	Input of where to map "Request combined active" functionality. When the Input is activated the BMS will activate contactor combined mode. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
GPIO mapping	Request charge active	Input of where to map "Request charge active" functionality. When the Input is activated the BMS will activate contactor charge mode. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16

GPIO mapping	Request load active	Input of where to map “Request load active” functionality. When the Input is activated the BMS will activate contactor load mode. 0 = Disabled. 0-8 indicates IO channel.	IO	0-16
Custom Data Processing	Slot [N] Operator	The operation to perform. 0 = [NOP] NONE 1 = [+] ADDITION 2 = [-] SUBTRACTION 3 = [*] MULTIPLICATION 4 = [/] DIVISION 5 = [&] BITWISE_AND 6 = [|] BITWISE_OR 7 = [^] BITWISE_XOR 8 = [>] GREATER_THAN 9 = [<] SMALLER_THAN 10 = [==] EQUAL_TO 11 = [>=] GREATER_THAN_OR_EQUAL 12 = [<=] SMALLER_THAN_OR_EQUAL 13 = [<<] LEFT_SHIFT 14 = [>>] RIGHT_SHIFT 15 = [!] INVERSE	-	0-15
Custom Data Processing	Slot [N] Right side value	The value to use on the right hand side of the operator (either constant or data ID).	-	-2147483648 - 2147483647
Custom Data Processing	Slot [N] Right side value type	The data type. 0 means a constant value. 1 means data ID.	-	0-1
Custom Data Processing	Slot [N] Left side value	The value to use on the left hand side of the operator (either constant or data ID).	-	-2147483648 - 2147483647
Custom Data Processing	Slot [N] Left side value type	The data type. 0 means a constant value. 1 means data ID.	-	0-1
CAN settings	Frame [N] Config 1(-10) data	Either the constant data or an ID to the BMS variable to place in the CAN frame. Id numbers are found in section 0		0 to 4294967295 (2^(32) – 1)
CAN settings	Frame [N] Config 1(-10) is little endian	If set to 1, bits are treated as little endian (Intel format). If set to 0, bits		0 or 1

		are treated as big endian (Motorola format).		
CAN settings	Frame [N] Config 1(-10) length	The amount of bits to write data to, starting from start_bit		0 to 32
CAN settings	Frame [N] Config 1(-10) start bit	The bit in the CAN frame, where the first data bit is to be placed		0 to 63
CAN settings	Frame [N] Config 1(-10) entry type	The type of the entry data. 0 means constant, 1 means BMS variable.		0 or 1
CAN settings	Frame [N] Config 1(-10) enabled	The configuration entry number 'X' (for the CAN frame) is enabled if set to 1.		0 and 1
CAN settings	Frame [N] CAN Channel	The CAN channel to output the frame on. 0: S-CAN, the first CAN channel 1: I-CAN, the second CAN channel		0 to 1, depending on the number of available CAN channels on the BMS hardware.
CAN settings	Frame [N] Is ID Extended	1: the frame_id is a 29-bit extended CAN frame ID. 0: the frame_id is an 11-bit CAN frame ID.		0 and 1
CAN settings	Frame [N] ID	The CAN frame id as a decimal value.	-	0 to 536870912 [2^(29) 29 bit ID] 0 to 2048 [2^(11) 11 bit ID]
CAN settings	Frame [N] DLC	The length of the CAN frame in bytes.	-	0 - 8
CAN settings	Frame [N] Update interval	CAN frames can be updated at a maximum rate of 100 ms. 1: the CAN frame is updated at the maximum rate 2: the CAN frame is updated at half the maximum rate, i.e every 200 ms 3: The 3 CAN frame is updated at one third the maximum frequency, i.e	-	1 – 65535

		every 300 ms …and so on.		
CAN settings	Frame [N] Enable frame	The CAN frame is enabled if set to 1. If set to 0 it is not enabled.	BOOL	0 or 1
CAN settings	CAN channel error frames	Selector for what CAN bus to output CAN error frames. 0 = S-CAN or 1 = I-CAN.	-	0 - 1
CAN settings	CAN ID error extended?	Selector to output CAN ID error frames as extended ID. 1 = 29 bit or 0 = 11 bit.	BOOL	0 or 1
CAN settings	CAN ID start error frames	Start address for a sequential map of all relevant error codes. Select an ID where the following 200 ID's are vacant.	ID	0-2047
CAN settings	CAN Charger type	Selection of predefined legacy CAN charger types: 0 = Zeroed output. 1 = EA-PS8200-70 from Elektro-Automatik. 2 = static charge allowed flag. 3 = static charge allowed flag, with latching flag. 4 = Powerfinn charger	-	0 - 4
CAN settings	CAN Channel Frame Charger	Selector for what CAN bus to output CAN Charger Frame. 0 = S-CAN or 1 = I-CAN.	-	0 - 1
CAN settings	CAN ID Charger Extended?	Selector to output CAN ID Frame Charger as extended ID. 1 = 29 bit or 0 = 11 bit.	BOOL	0 or 1
CAN settings	CAN ID Frame Charger	ID of the fixed CAN charge frame output. Input is in decimal value.	ID	0 - 2047
CAN settings	CAN speed I-CAN	Input selection for the CAN speed of the application firmware. 0 =	-	0-3

		125 kbps, 1 = 250 kbps, 2 = 500 kbps, 3 = 1000 kbps.		
CAN settings	CAN speed S-CAN	Input selection for the CAN speed of the application firmware. 0 = 125 kbps, 1 = 250 kbps, 2 = 500 kbps, 3 = 1000 kbps.	-	0-3
CMU config	Enabled NTC inputs CMU 1-32	Bitmask for enabled NTC inputs on CMU. NTC 1 is bit 1, NTC 2 is bit 2 ect.	Bits	0 - 4095
CMU config	Enabled cells CMU 1-32	Bitmask for enabled cells on CMU. Cell 1 is bit 1 (LSB), cell 2 is bit 2 ect. Input to BMS creator is in decimal.	Bits	0 - 4095
CMU config	Expected CMUs	Setup for number of front end measuring devices.	No.	0-32
Error Timers	Cell over/under temperature alarms	Error timers relating to cell temperatures. 0 means immediate escalation	s	0-65535
Error Timers	Cell temperature sensor alarms	Error timers relating to cell temperature sensors. 0 means immediate escalation	s	0-65535
Error Timers	AUX temperature alarms	Error timers relating to auxiliary temperatures. 0 means immediate escalation	s	0-65535
Error Timers	Cell over/under voltage alarms	Error timers relating to cell voltages. 0 means immediate escalation	s	0-65535
Error Timers	CMU Communication alarm	Error timers relating to CMU communication. 0 means immediate escalation	s	0-65535
Error Timers	Balancing alarms	Error timers relating to cell balancing. 0 means immediate escalation	s	0-65535
Error Timers	Over Current IN/OUT alarms	Error timers relating to over current. 0 means immediate escalation	s	0-65535
Errors - enter ready Errors - enter charge Errors - enter load Errors - stay charge Errors - stay load Errors - request current	Code error 1-64	The specific error code associated with the error. See the full list of all error codes	-	1 – 65535

Errors - enter ready Errors - enter charge Errors - enter load Errors - stay charge Errors - stay load Errors - request current	Severity error 1-64	Severity of the error	-	0-6
SOC-OCV settings	Data set [N] voltage [X]	The data point for SoC value X (X % SoC) for data set N	mV	0-5500
SOC-OCV settings	Data set [N] temperature	The temperature at which the data points for data set N are valid	deg C	-160
SOC-OCV settings	Maximum quality	The maximum weighting to use. A value of 100 % means the OCV-SoC value completely replaces the counted SoC after "Off duration for highest quality" minutes. A value of 90 means the OCV-SoC value is weighted 90 %	%	0-100
SOC-OCV settings	Off duration for highest quality	The minimum amount of time the BMS must be turned off before considering the OCV-SoC data to be of maximum quality	min	0-65535
SOC-OCV settings	Off duration for lowest quality	The amount of time the BMS must be turned off before considering the OCV-SoC calibrated values	min	0-65535
SOC-OCV settings	Enable data sets	How many data sets to enable for OCV. 0 Means feature is not used	-	0-5

8.8 Busmaster configuration

To read and if needed store the physical values measured by the BMS, a dedicated program must be used. One such program is the freeware program Busmaster. This is available at https://rbei-etas.github.io/busmaster/.

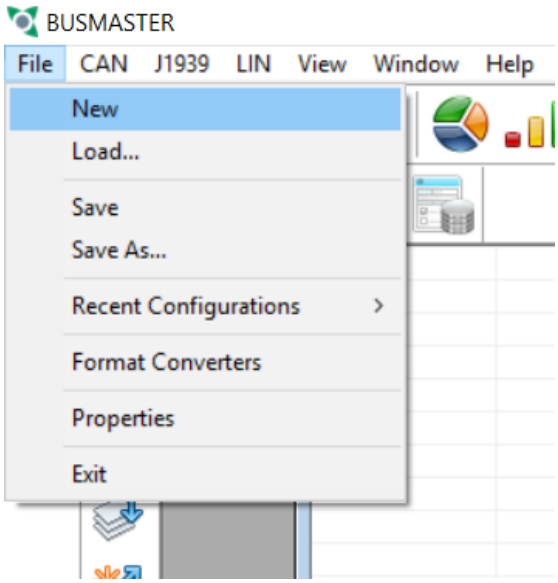
The Busmaster program operates on three levels

- Project Configuration and Database file
  - Message
    - Signals in message

The following sections describes the most important steps in setting up the Bus-master program to read data after Bus-master is installed on the PC connected to the BMS. This is exemplified by showing how to use Busmaster to read the 4 cell voltages configured on the CAN bus in section 5.9.2.

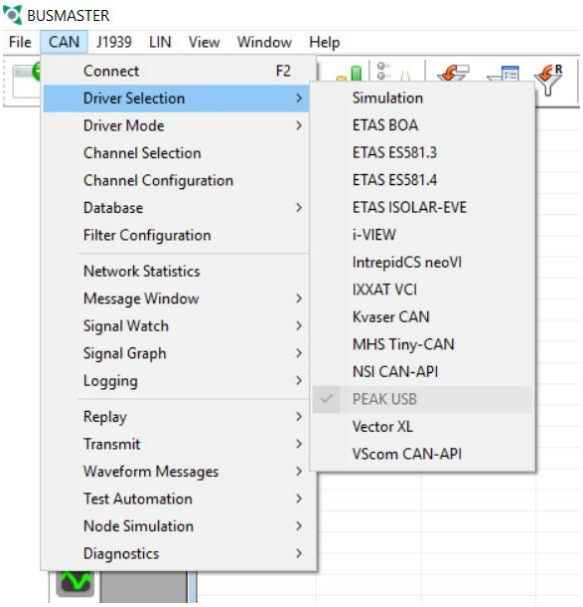
8.8.1 Create new Busmaster project

Step 1.1 Create and save file



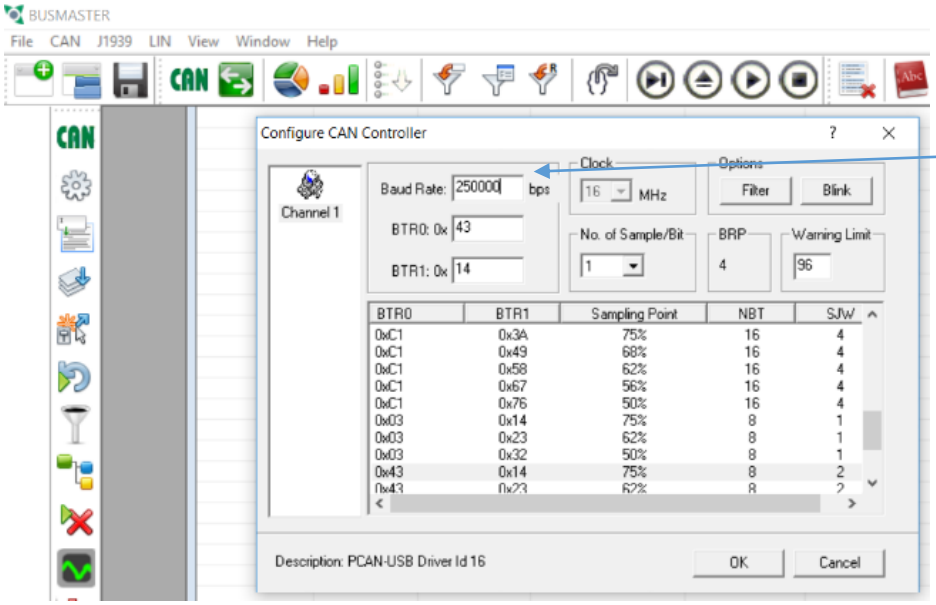
- Create a new project configuration file
- Write a project name and save the file

Step 1.2: Select the relevant adapter driver



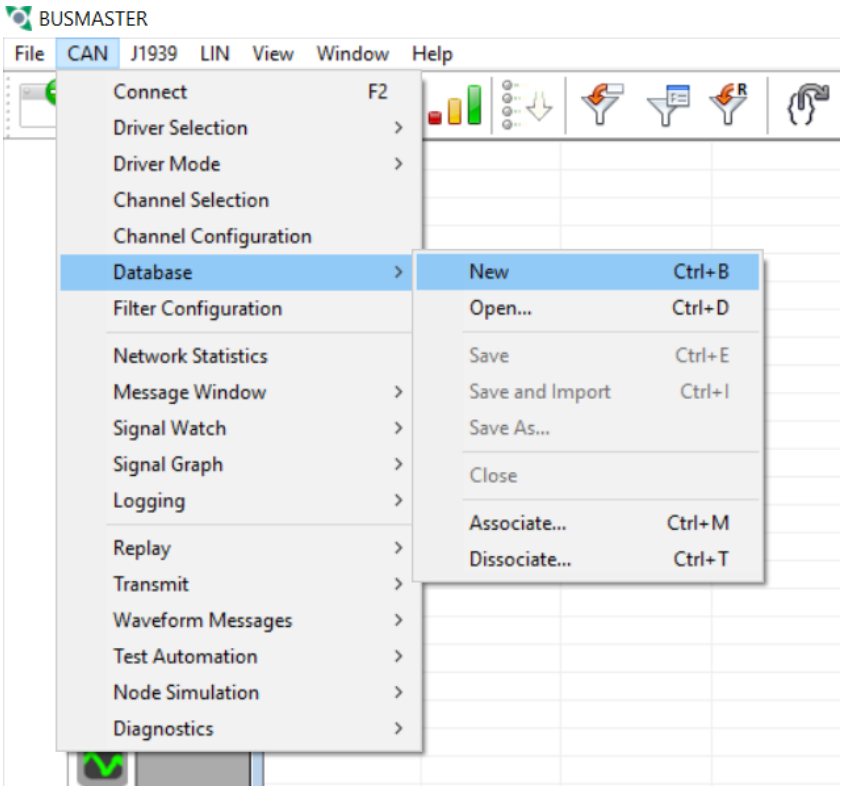
- In the CAN driver selection, select “PEAK USB” for CAN PEAK Adapter

Step 1.3: Select the correct CAN Baud rate



- Baud rate to be matched with value in BMS creator
- Refer to BMS Creator configuration Setting in section 5.3.

8.8.2 Create a new Database



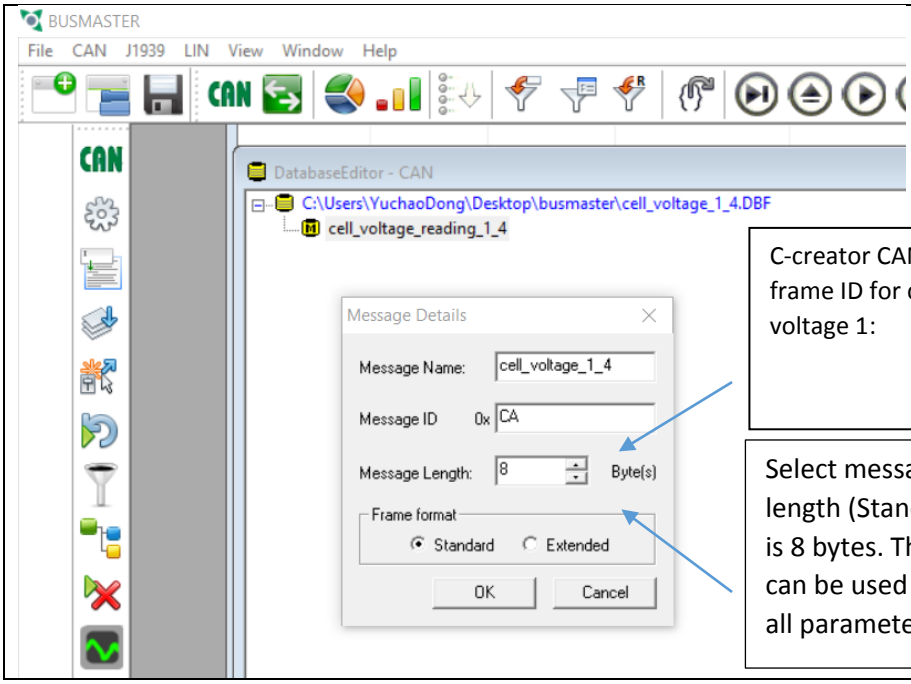
Create a new database file

Save the new file when prompted. In the example in this appendix it is the file "C:\User….\cell_voltage_1_4.DBF"

8.8.3 Create and configure a new "message"

The parameters on the CAN bus are stored in the Busmaster database in individual messages which can each contain one CAN frame corresponding to 8 bytes of information.

Step 2.1 create new message



C-creator CAN frame ID for cell voltage 1:

Select message length (Standard is 8 bytes. This can be used for all parameters)

Right-click on database, here "C:User…" to create a new message under the database

Identify the frame to be stored in the message in the Message ID.

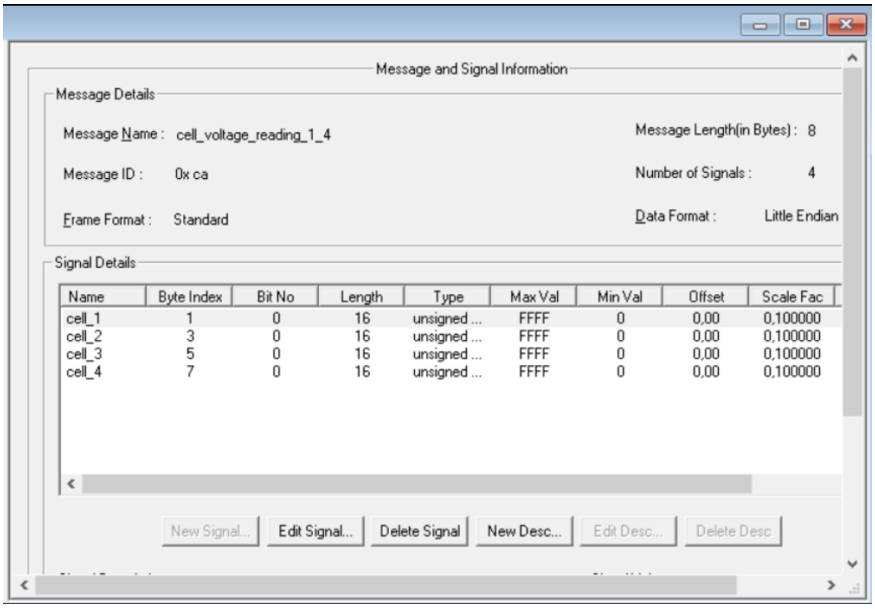
Note that Hex-numbers are to be used in the Busmaster for CAN-frame id[16].

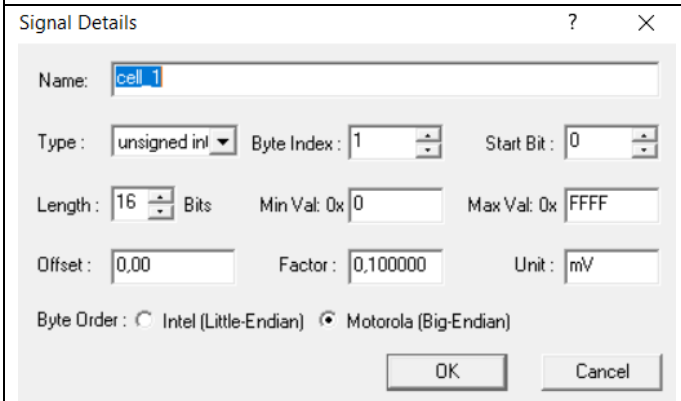
Decimal numbers are used in the BMS Creator.

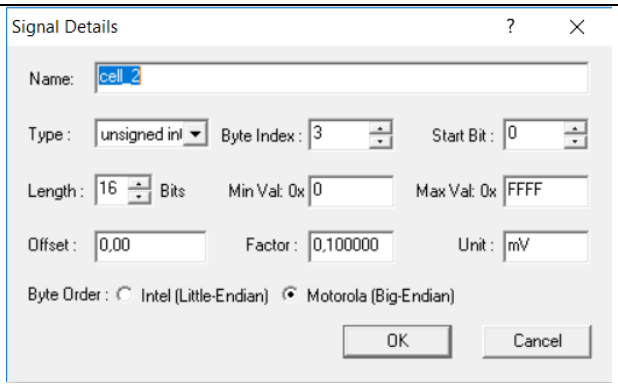
[16] Conversion between decimal and hex numbers can be done for example on http://www.binaryhexconverter.com/decimal-to-hex-converter

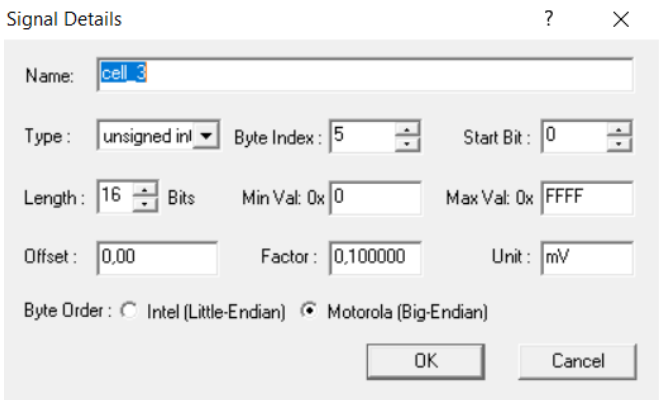
Step 2.2: Create signal contents for the message.

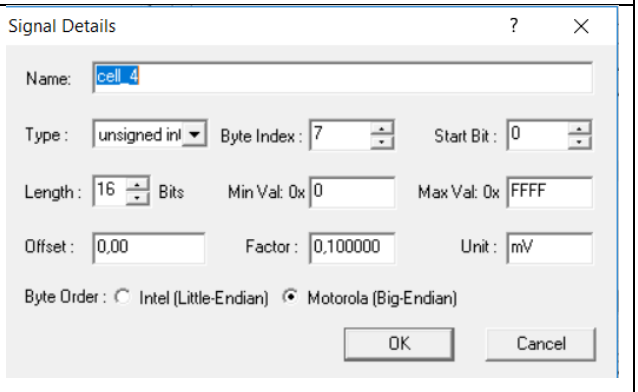
The examples below show how message contents are created for cell voltage 1 to 4 readings.











- Name: Select meaningful name to allow the user to recognise the value transmitted by the message
- Byte index: Placement in the frame, please refer to the figure below

- Bit No: Equals the start bit, which is typically 0, when all bits in a byte are used
- Length: Bit lengths as can be found in the “size” column of section 8.4
- Type: Boolean, Signed integer or unsigned Integer, please refer to the “size” column of section 8.4
- Max Val, will automatically be filled in
- Offset: to be set to 0
- Factor: please refer to the ‘Scaling” column of section 8.4.

Please note that for Busmaster, bytes are identified by the byte index, corresponding to the Byte No shown below. the CAN frame for cell voltage 1 can be seen below:

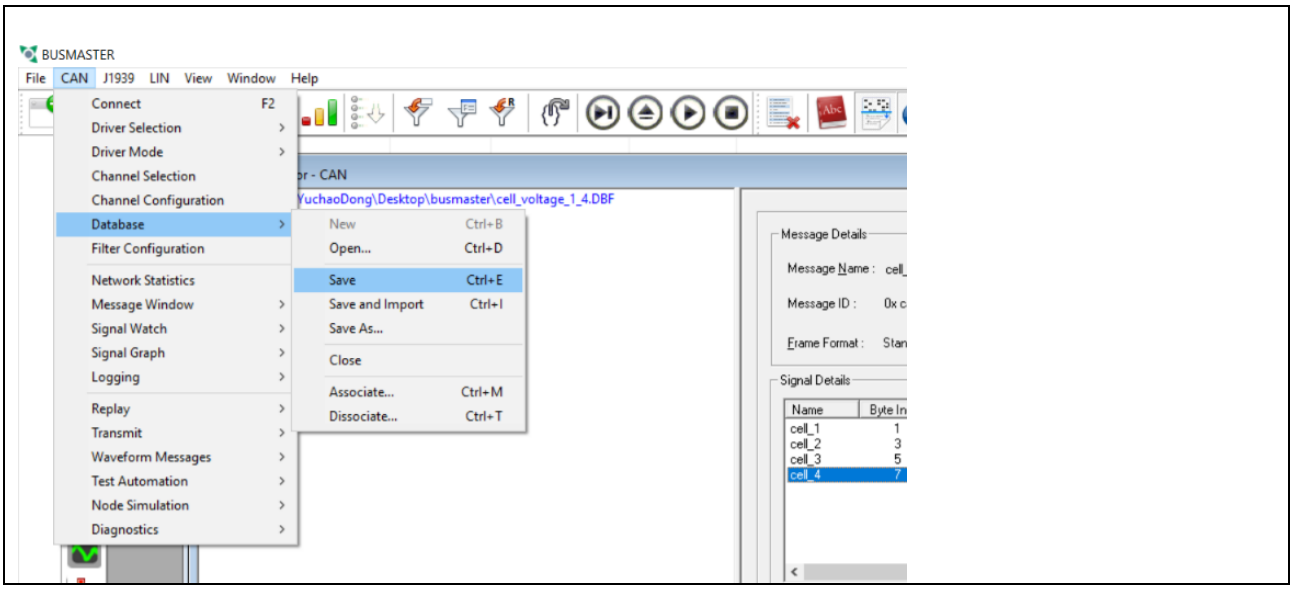
Byte No.	0								1							
Bits No.	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
Allocation	Cell voltage 1															

The total CAN frame for 4 cell voltage can be seen below:

Byte No.	0					1		2					3		4					5		6					7	
Bits No.	63					48		47					32		31					16		15					0	
Allocation	Cell voltage 1							Cell voltage 2							Cell voltage 3							Cell voltage 4						

Note that, the green boxes represent the Bits No used in setting up CAN messages in BMS creator and the blue boxes represents the byte index used in setting up the Busmaster.

Step 2.3 Save the new configuration

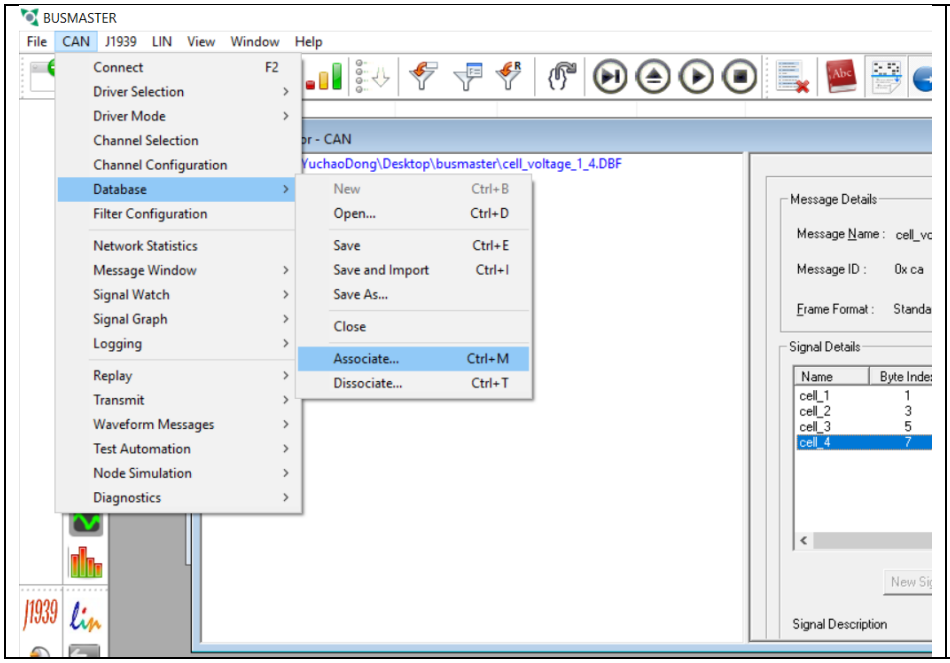


Save the database configuration (the DBF-file) including the set-up of the individual messages

8.8.4 Select Database and connect to BMS

A number of different databases may be used by the Busmater program. To select the relevant configuration for the CAN bus to be read, please use the following steps.

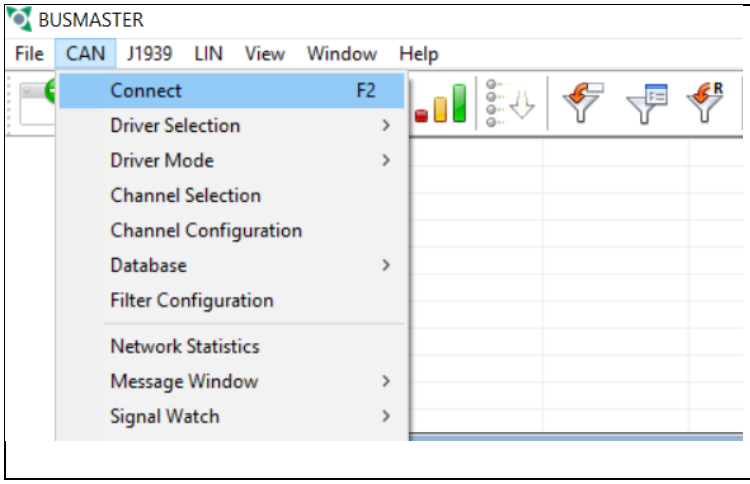
Step 3.1 Associate the relevant database with the present CAN-bus.



Associate with the database which contains the desired parameters which in this case is the cell voltage 1 to 4.

Step 3.2: Connect CAN

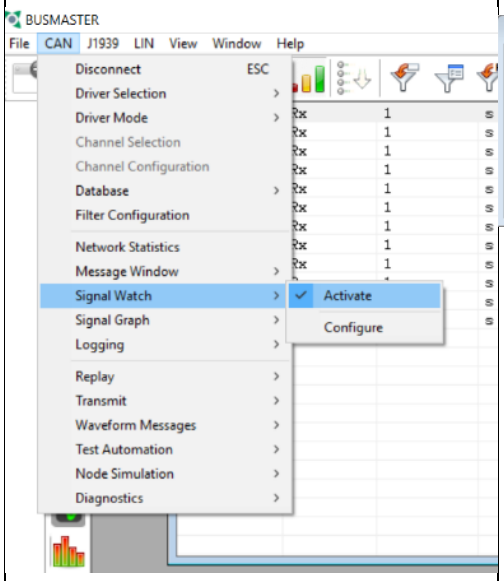
Now Busmaster is ready to read the data from the BMS via CAN and place the read values in the configured messages

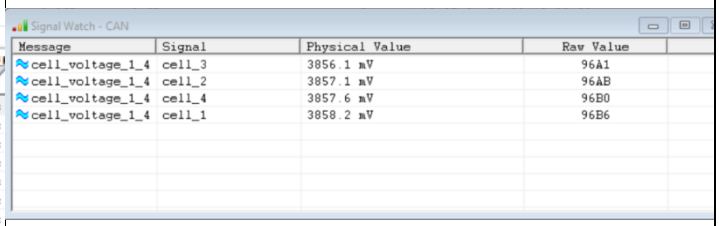


- Click on “Connect” to CAN

8.8.5 Select Signals/Parameters to be Watched/Displayed

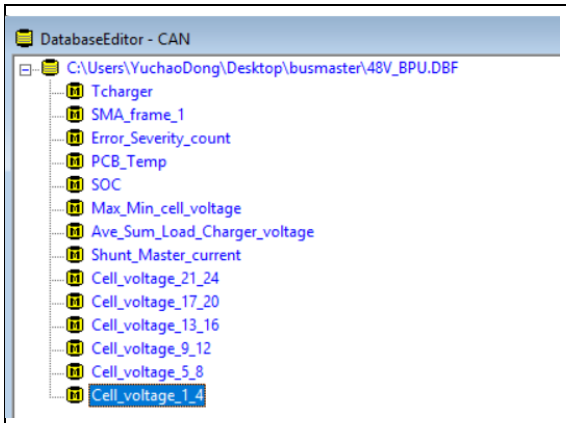
To have the actual values from the CAN-bus displayed on the Busmaster screen, please use the “Select Signal Watch” function





- Activate the Signal watch to read the physical value from the BMS
- Right click on table to select the messages to be displayed

8.8.6 Configure more values to be included in messages and displayed



To configure more values to be included in messages and displayed, please repeat

- Section 8.8.3
- Section 8.8.5

8.9 HW functions which are not yet enabled by Software

8.9.1 High Voltage Interlock (HVIL)

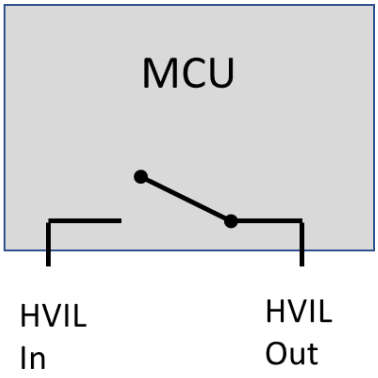


Figure 8-6 HVIL Connections

The MCU provides input and output ports for a High Voltage Interlock (HVIL) signal. This feature can be used by the BMS to communicate a critical error conditions directly to other electronic units where the battery system is used, e.g. in a Vehicle.
Under normal conditions the MCU shorts the HVIL in to HVIL out. In case of emergency conditions which might require the shut down of other electronic functions, the HVIL switch (relay) will open. This can then be immediately detected by other electronic systems connected to the HVIL system which can act accordingly.

8.9.2 Real Time Clock

A real time clock is available to assure that the BMS has a correct absolute time also when the system enters sleep mode or is powered off. The real time clock includes a dedicated on-board back-up battery. Consequently, the real time clock will operate correctly even if the MCU has been disconnected from the power supply.

RTC support is present in 2.1 and is used for time-stamping errors in error log and for determining off-time for OCV-SoC calibration. The time can not be set by the customer, however, and has no real-world meaning (it's relative time, not something that would correspond to Wednesday 27th of November 2017

8.10 Converting a PWM signal to an analogue control voltage

It is possible to create an analogue voltage signal from a PWM signal by Low Pass Filtering the PWM signal. However, almost no current can be provided from the PWM signal provided by the BMS and often chargers needs to be controlled by voltage which is isolated from the BMS ground reference.

An example of a circuit which can be used to convert a PWM signal from the BMS to an analog control voltage is shown in Figure 8-7

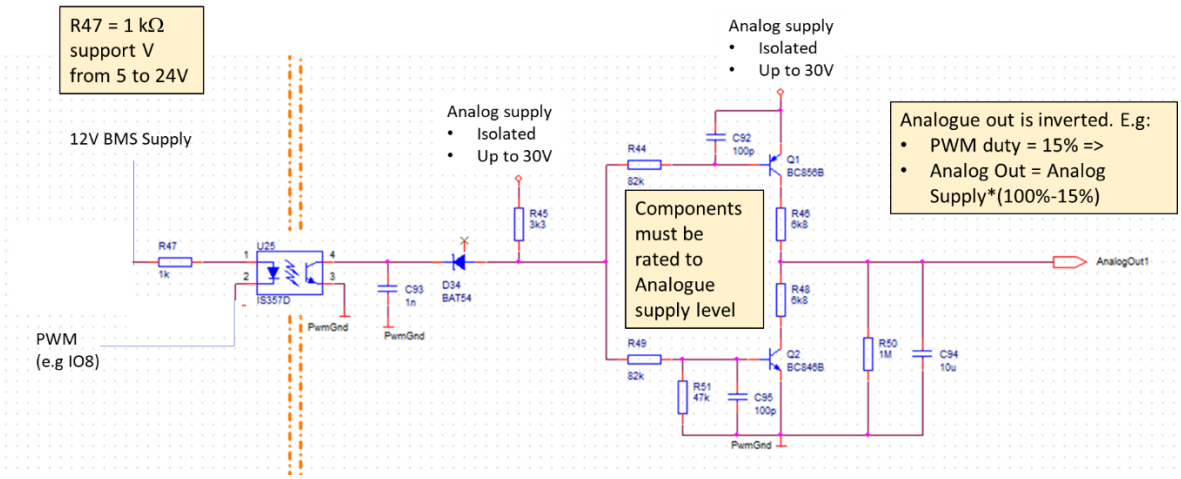


Figure 8-7 example of a circuit which can be used to convert a PWM signal from the BMS to an analogue control voltage