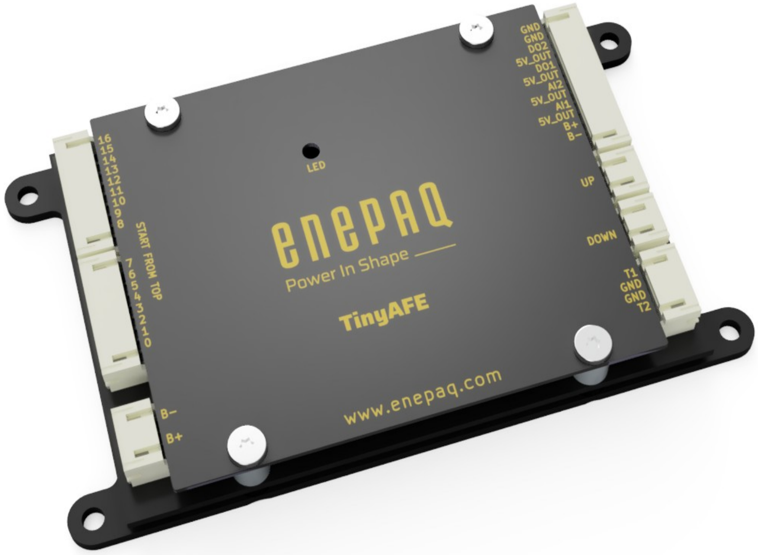
Datasheet

Tiny AFE v1.1



Revision A, 2024-05-30

INTRODUCTION

The TinyAFE device serves as the analog front-end for high-voltage battery system data acquisition. It acts solely as a slave device and does not replace the necessity for a user-side master device for controlling and safeguarding the battery. Featuring an isolated communication interface, TinyAFE interconnects all devices within a stacked high-voltage battery network along with the user-side master device. This internal isolated communication interface is proprietary and not directly accessible to the user. A dedicated ENEPAQ UART-to-isolated communication adapter is required to access the data on the isolated stacked network. *Enepaq Tiny AFE* supports lithium batteries of any chemistry and up to *60V* nominal per unit. Up to 20 unit can be connected in series. No limited battery capacity. *Tiny AFE* measures individual voltages of parallel cell groups and during charging or activated by the user, cells are balanced by bleeding-off higher cells to accomplish full balance and maintain good health of battery pack.

SAFETY

The TinyAFE device monitors cell over-voltage, under-voltage, over-temperature, under-temperature, internal PCB over-temperature, internal PCB under-temperature, and cells mismatch protections. Users can read the protection status via communication commands. While the TinyAFE device itself does not directly protect the battery, users can monitor its protection status and take appropriate actions to safeguard the battery. This feature can offload calculations from the user-side master device or serve as secondary protection monitoring.

FEATURES

- Supports *up to 1000V system*
- All lithium chemistry
- Stackability (up to 20 unit)
- Adjustable: voltage, temperature
- Supporting up to two external temperature sensors or Enepaq multipoint temperature sensor
- Dissipative balancing up to *150mA*
- Connectivity: *UART*, *MODBUS*
- Usage Statistics
- Free firmware upgrades
- Programmable Inputs & Outputs
- Compact design: 113*x65x14 mm*

APPLICATIONS

- High voltage battery
- Industrial equipment, robotics
- Stationary solar & wind storage
- Personal transportation

Contents

1. Electrical characteristics

Table 1: Product characteristics (all parameters rated at 25 °C if not specified otherwise)

Parameter	Comment	Min.	Typ.	Max.	Unit
Cell voltage	Measurement range	*0.8*	-	*4.5*	*V*
Cell voltage resolution		-	*1*	-	*mV*
Battery voltage*[1]	Operation range	*9.0*	-	*75*	*V*
Battery stack voltage*[2]	Range	*9.0*	-	*1000*	*V*
Power consumption	Idel	*2.5*	-	*6.5*	*mA*
	Sleep	-	*80*	-	*uA*
Balancing current	Per cell, U_{cell}=*4.2V*	-	*100*	*150*	*mA*
Expansion *I/O* current	Long term	-	-	*250*	*mA*
External temperature	Measurement range	*-40*	-	*120*	*°C*
External temperature resolution		-	*0.1*	-	*°C*
Operating temperature	Operation range	*-40*	-	*85*	*°C*
Dimensions	With cooling plate	-	*113x65x14*	-	*mm*

*[1]- per one AFE unit.
*[2]- max. 20 unit in series.

2. Mechanical data

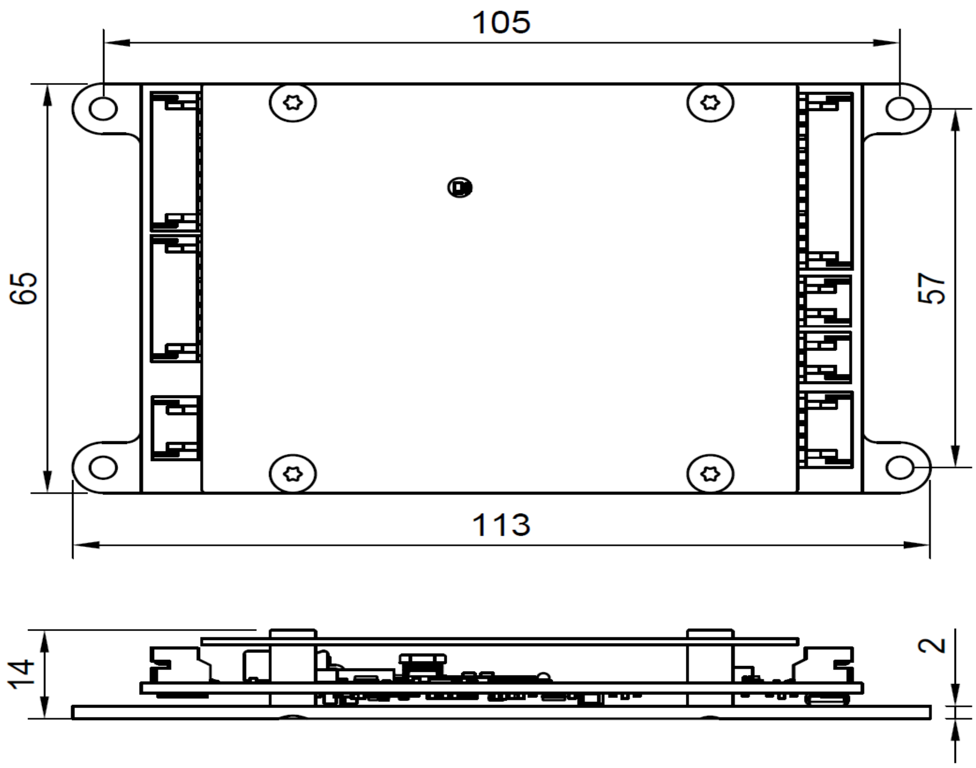


Figure 1: Mechanical dimensions (in mm)

3. Connection diagram example

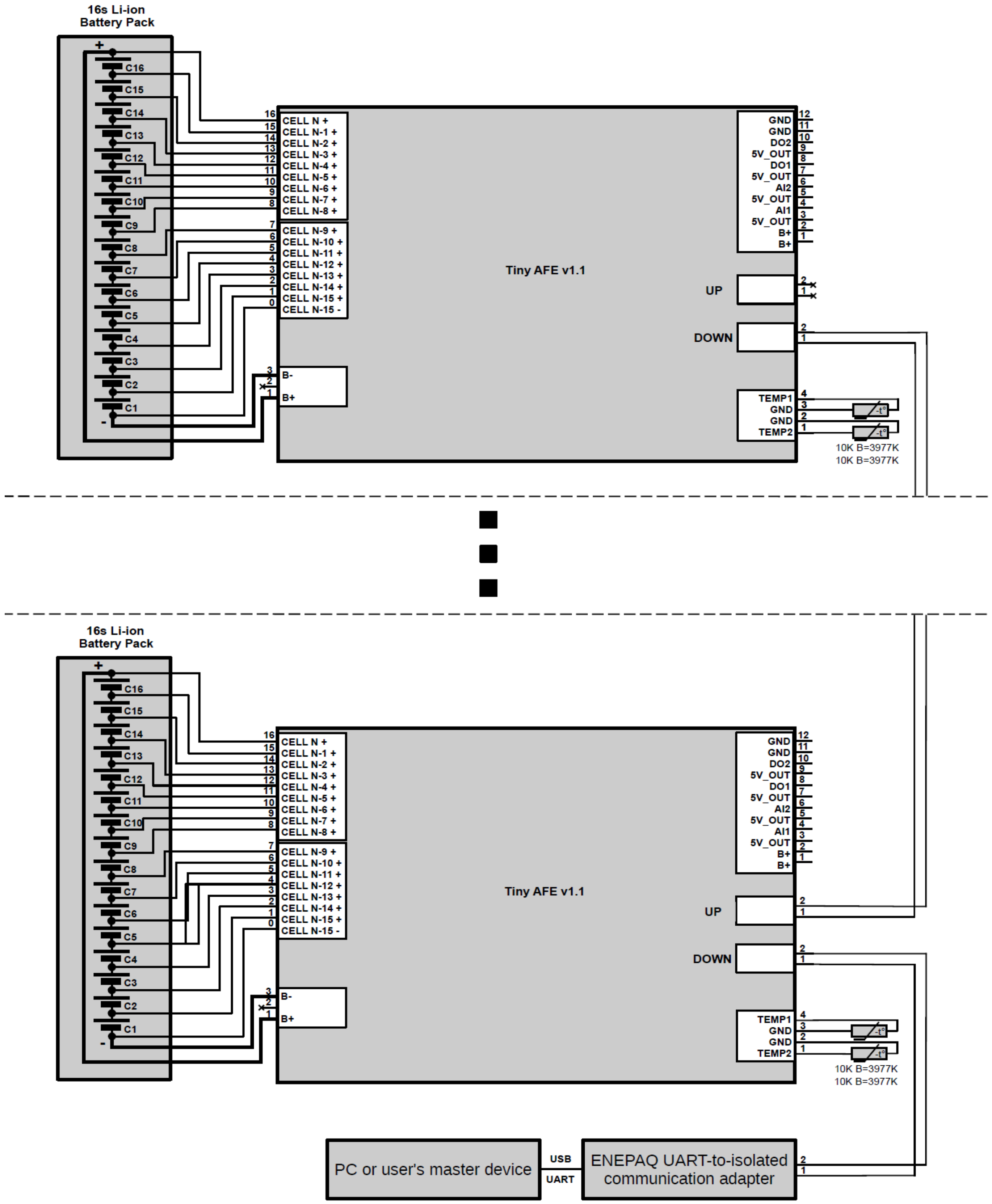


Figure 2: A typical Tiny AFE connection diagram

4. Device operation

The *TinyAFE* device serves as the analog front-end for high-voltage battery system data acquisition. It acts solely as a slave device and does not replace the necessity of a user-side master device for controlling and safeguarding the battery. Featuring an isolated communication interface, *TinyAFE* interconnects all devices within a stacked high-voltage battery network along with the user-side master device. This internal isolated communication interface is proprietary and not directly accessible to the user. A dedicated *ENEPAQ UART*-to-isolated communication adapter is required to access the data on the isolated stacked network.

4.1. Data acquisition

The *TinyAFE* device automatically and continuously measures battery module cell voltages and temperatures in active mode at a steady rate of *10 Hz* (*100 ms* interval) and determines the battery module state. Users can access all measured and calculated battery module data using a simple *UART* interface with dedicated communication commands (refer to the *TinyAFE* communication protocol).

The *TinyAFE* device operates in several distinct modes (refer to the *Live Data Register Map*):
Initialization: The device enters this state immediately after powering up. In this state, the device performs all measurements but does not check statuses, protections, or perform battery cell balancing. The device initializes and switches to *Active [Normal]* state when it detects the number of cells equal to the *Cells In Series* parameter (*Register 100* in the *Settings Register Map*).
Active [Normal]: The device is fully functional, performing all measurements, monitoring protections, and balancing battery cells if activated.
Active [Protection]: Similar to the *Active [Normal]* state, but at least one protection is triggered. Refer to the *Protections Status* register in the *Live Data Register Map*.
Sleep (ultra low power): The device enters this state if there is no communication within the *Sleep Mode Entry Timeout* interval. It wakes up from ultra-low power mode when communication resumes.

The *Synchronization Status* bit in the *Live Data* status register indicates if the measurements for all *TinyAFE* devices on the network are synchronized. To synchronize *TinyAFE* devices on the network, a special command (refer to *CMD 19* in the *TinyAFE* communication protocol) should be sent, which is recommended after the device powers up, restarts, or wakes up from sleep mode.

4.2. Battery cells balancing

The *TinyAFE* device uses a cell balancing algorithm to gradually decrease the differences in imbalanced cells in a fully charged state, preventing fully charged cells from becoming overcharged and causing excessive degradation. This process increases the overall pack energy by preventing premature charge termination. Each *TinyAFE* device in the network controls its internal cell balancing process automatically, if cell balancing is activated by the user. There are two main modes of cell balancing control:
Continuous Balancing Mode Disabled: In this mode, the *TinyAFE* device's cell balancing process is controlled (enabled/disabled) only by the user with a dedicated command (refer to *CMD 6: Control cells balancing* in the *TinyAFE* communication protocol). When enabled, the device performs cell balancing if the cell voltage is higher than the *Early Balancing Threshold* voltage level, continuing until the cells' imbalance is less than the *Allowed Disbalance* parameter or the user disables the balancing process itself.
Continuous Balancing Mode Enabled: In this mode, the *TinyAFE* device continuously monitors and performs cell balancing when the cell voltage is higher than the *Early Balancing Threshold* voltage level, continuing until the cells' imbalance is less than the *Allowed Disbalance* parameter.

In both cases, cell balancing process within individual *TinyAFE* modules is automatic and continues until the desired cells' imbalance is achieved. However, individual device modules do not have information about the lowest cell voltage of the entire battery, so a particular module may not necessarily be balanced relative to the entire battery pack. To address this issue, the *TinyAFE* device monitors all commands sent between the user-side master device and individual *TinyAFE* devices, extracting data about the minimum and maximum cell voltages of the entire battery pack from the data stream. Thus, cells are balanced with respect to the entire battery pack. In summary, if balancing is activated, but the user-side master device does not read cell data from all modules - then each module balances cells only with respect to that module, if the user-side master device reads cell data from all modules, each *TinyAFE* module balances cells relative to the whole battery pack.

Due to *TinyAFE* hardware limitations, only every fourth cell can be balanced at a time (adjacent cells cannot be balanced simultaneously). This process is controlled internally by the *TinyAFE* device itself. Before the *TinyAFE* device switches to sleep (ultra-low power) mode, the cells balancing process is always stopped automatically.

4.3. Protections

The *TinyAFE* device monitors cell *over-voltage*, *under-voltage*, *over-temperature*, *under-temperature*, internal *PCB over-temperature*, internal *PCB under-temperature*, and *cells mismatch* protections (refer to the *TinyAFE Live Data Register Map*). Users can read the protection status via communication commands. While the *TinyAFE* device itself does not directly protect the battery, users can monitor its protection status and take appropriate actions to safeguard the battery. This feature can offload calculations from the user-side master device or serve as secondary protection monitoring.

4.4. Temperature measurement

The *TinyAFE* device includes one onboard *NTC* temperature sensor and supports up to two external *NTC* temperature sensors (one per channel) or up to sixteen *ENEPAQ Multipoint Active* temperature sensors per channel available in *ENEPAQ Cell Modules*. The used temperature sensor type parameter should be configured through the *TinyAFE Insider Windows* application or communication command (refer to the *TinyAFE* communication protocol and *Settings Register Map*). Note that temperature sensor type configuration applies to both *TinyAFE* device temperature channels, and different sensor types cannot be used simultaneously on different channels. Unused temperature sensor channels should be left unconnected.

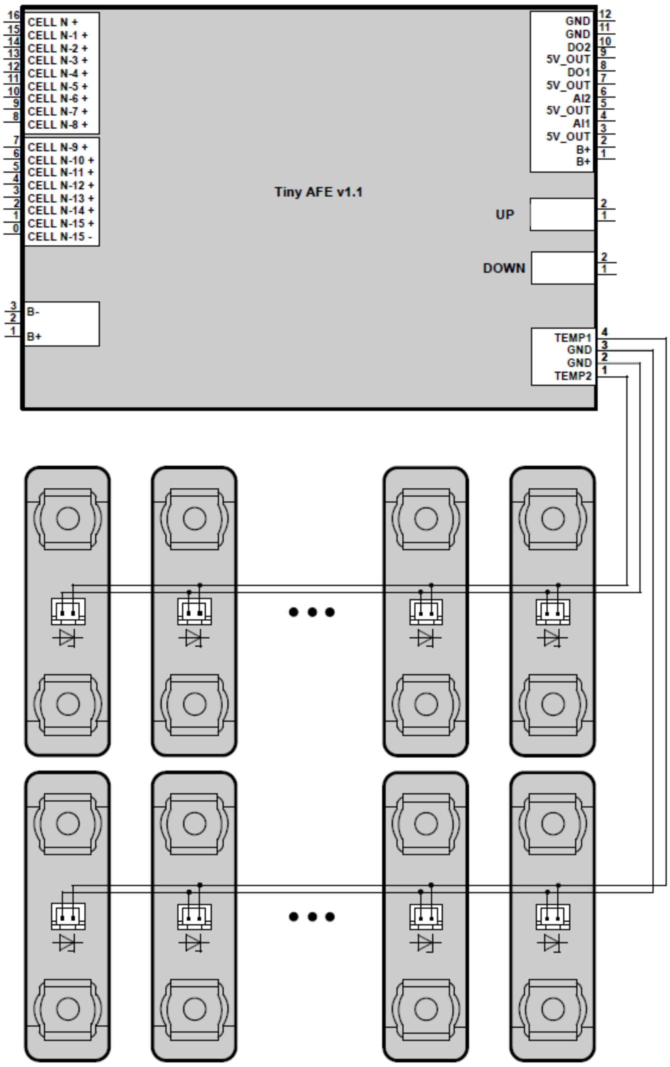


Figure 3: Enepaq multipoint active temperature sensor connection diagram

4.5. Analog Inputs / Digital outputs

The *TinyAFE* device has two general-purpose analog inputs and outputs each. These inputs and outputs are not assigned to any internal function of the device but are directly accessible to the user through dedicated communication commands. Analog inputs can measure input voltage from module voltage minus (*B*-) to module voltage plus (*B*+). The high and low logic level thresholds of the analog inputs are configurable, and their status can be monitored by the user through communication (refer to the *TinyAFE Register Map* and communication protocol). Digital outputs are open drain and can sink up to *250 mA* continuous current. Outputs can be directly controlled by the user through communication commands (refer to *CMD 7/8* in the *TinyAFE* communication protocol).

Note: External inputs / outputs are not galvanically isolated. Their voltages are related to the module's negative terminal (*B*-). *TinyAFE* devices or their accessories may be damaged if non-isolated *I/Os* are used with other devices whose ground potential differs from the module's negative terminal (*B*-).

4.6. Sleep mode

The *TinyAFE* device enters sleep (ultra-low power) mode when there is no communication within the *Sleep Mode Entry Timeout* parameter time interval and automatically wakes up from sleep mode when it detects ongoing communication. Upon receiving the first command while in sleep mode, *TinyAFE* wakes up (rejecting the first command) and responds only upon receiving the command a second time. Subsequently, *TinyAFE* does not re-enter sleep mode while communication is ongoing.
If the *Sleep Mode Entry Timeout* parameter value is set to zero, the sleep mode feature of the *TinyAFE* device is disabled.

4.7. Activity indicators

The *TinyAFE* device is equipped with an onboard *LED* indicator whose status depends on the *TinyAFE* device's operational state:
Initialization: *LED* blinks once every 100 ms.
Active [Normal]: *LED* blinks once every second.
Active [Protection]: *LED* blinks three times every second.
Sleep: *LED* is disabled.

5. TinyAFE Device Configuration

5.1. Recommended high-voltage battery network design flow for *TinyAFE* devices

Step 1: Estimate the number of *TinyAFE* devices required for the high-voltage battery design. Additionally, one dedicated *ENEPAQ UART-to-isolated* communication converter per network (battery pack design) will be needed.

Step 2: Assign an *ID* number to each *TinyAFE* device starting from *1*. Connect each powered *TinyAFE* device separately to the *PC* or the user's master device via the *UART-to-isolated* communication converter and assign the corresponding *ID* number using the *TinyAFE Insider* application or communication commands. *TinyAFE* device *IDs* must be assigned consecutively. For example, if *5* devices are used, they must be assigned *IDs* from *1* to *5*. A maximum of *20 IDs* are supported.

Step 3: Connect all enumerated *TinyAFE* devices to the battery pack stacked network and complete the configuration (cell, peripheral settings, firmware updates, maintenance) for the entire network using the *TinyAFE Insider Windows* application or, alternatively, bare communication commands on the user-side master device.

5.2. *TinyAFE* device *ID* configuration

Each *TinyAFE* device on the battery pack network must have a pre-programmed unique *ID* number. Follow these steps to achieve this:

Step 1: Connect the *ENEPAQ UART-to-isolated* communication converter to a *PC* via *USB* interface. A virtual *COM* port should appear on the *PC* side.
Step 2: Open the *TinyAFE Insider Windows* application and select the valid converter virtual *COM* port.
Step 3: Go to the *Maintenance* tab and under the *ID Configuration* section, select the *Enable ID Configuration* checkbox.

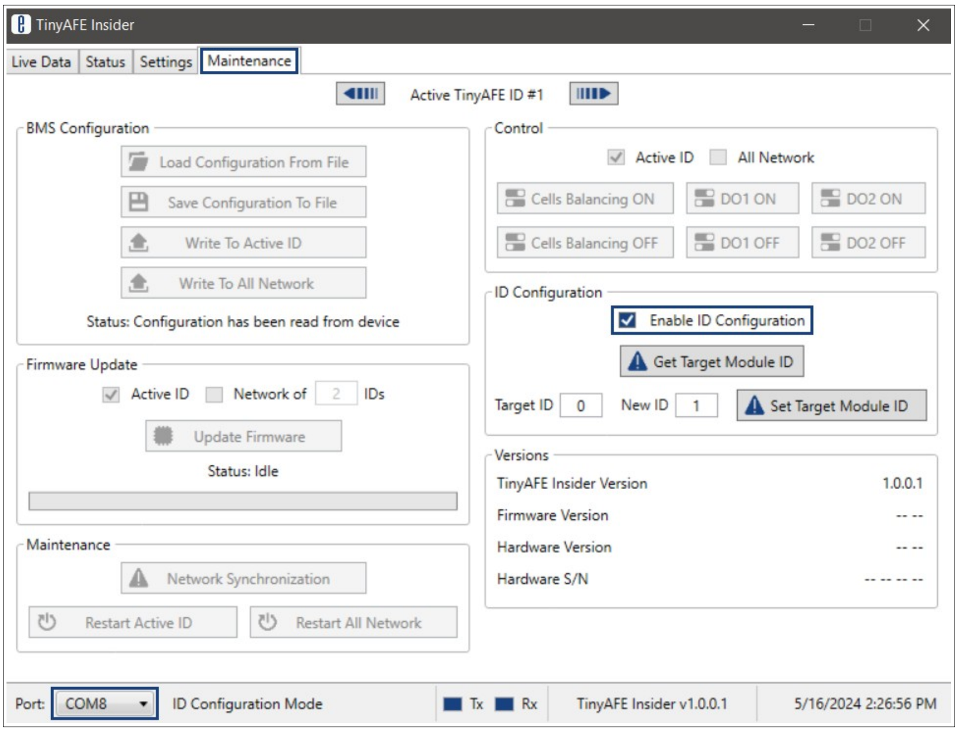


Figure 4: TinyAFE Insider enable *ID* configuration

Step 4: Connect a separate, powered *TinyAFE* device to the prepared *ENEPAQ UART-to-isolated* communication adapter according to the provided connection diagram.

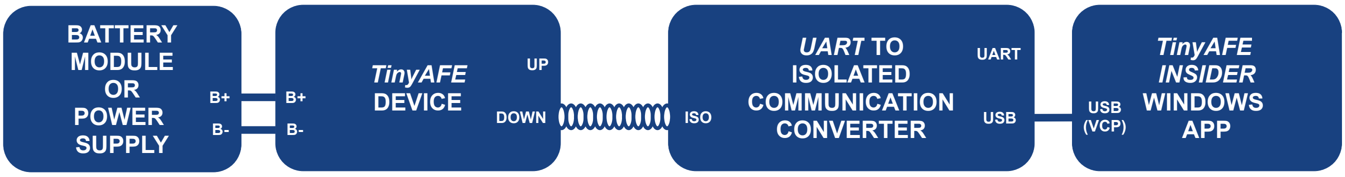


Figure 5: Simplified *TinyAFE* device *ID* configuration connection diagram

Step 5: Leave the *Target ID* parameter set to *0* (default device *ID*). Set the *New ID* parameter and click the *Set Target Module ID* button. If the configuration is successful, a *Set Module ID #n [OK]* message will appear on the status bar.

Alternatively, the new device *ID* can be configured using bare communication commands (refer to communication protocol *CMD 5 - Write TinyAFE* module *ID*) through both *UART* and *USB (Virtual COM Port*) interfaces.

Note: The *ENEPAQ UART-to-isolated* communication converter's *UART* and *USB* (*Virtual COM Port*) interfaces have the same functionality, but only one can operate at a time. When the *USB* interface is connected, the *UART* interface is internally disconnected.

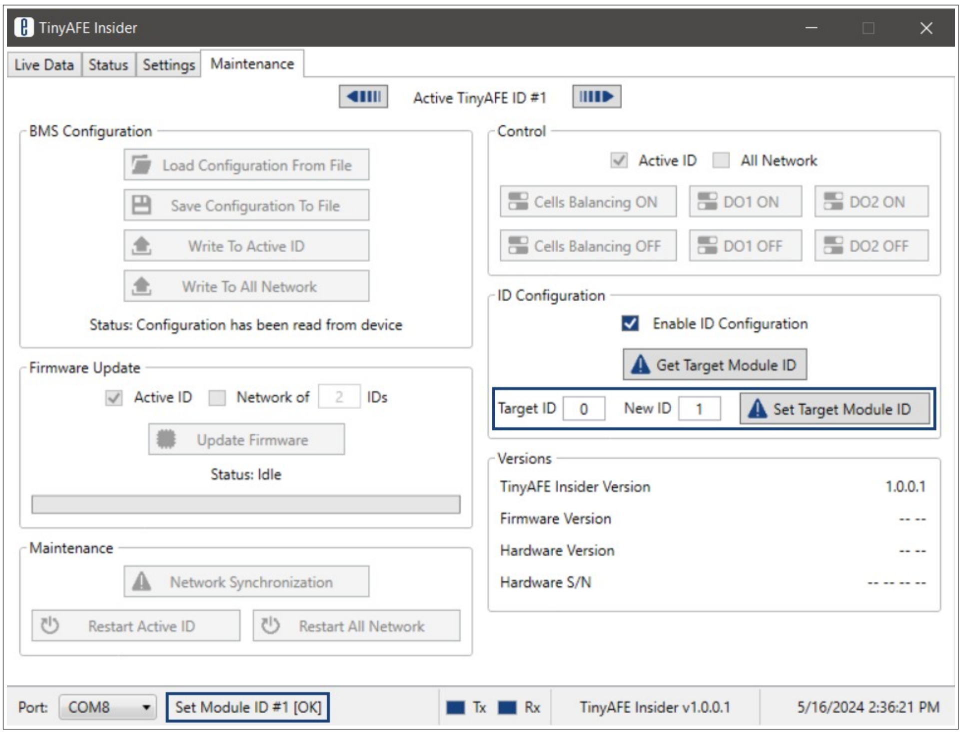


Figure 6: TinyAFE Insider *ID* configuration

Step 6: Disconnect the configured *TinyAFE* device from the *UART-to-isolated* communication converter and the power supply. Repeat steps *4* to *6* for all the required *TinyAFE* devices for the high-voltage battery network design.

6. TinyAFE Insider Application

The *TinyAFE Insider Windows* application streamlines the initial setup of the *TinyAFE* device, simplifying configuration processes. Additionally, it provides an interface for evaluating the device's features and capabilities and facilitates firmware updates.

Note: The *TinyAFE Insider* application is designed solely for configuration and testing purposes. It does not substitute the necessity of a user-side master device for controlling and safeguarding the battery. The *TinyAFE* device functions as a slave device solely for data acquisition.

6.1. TinyAFE Insider application overview

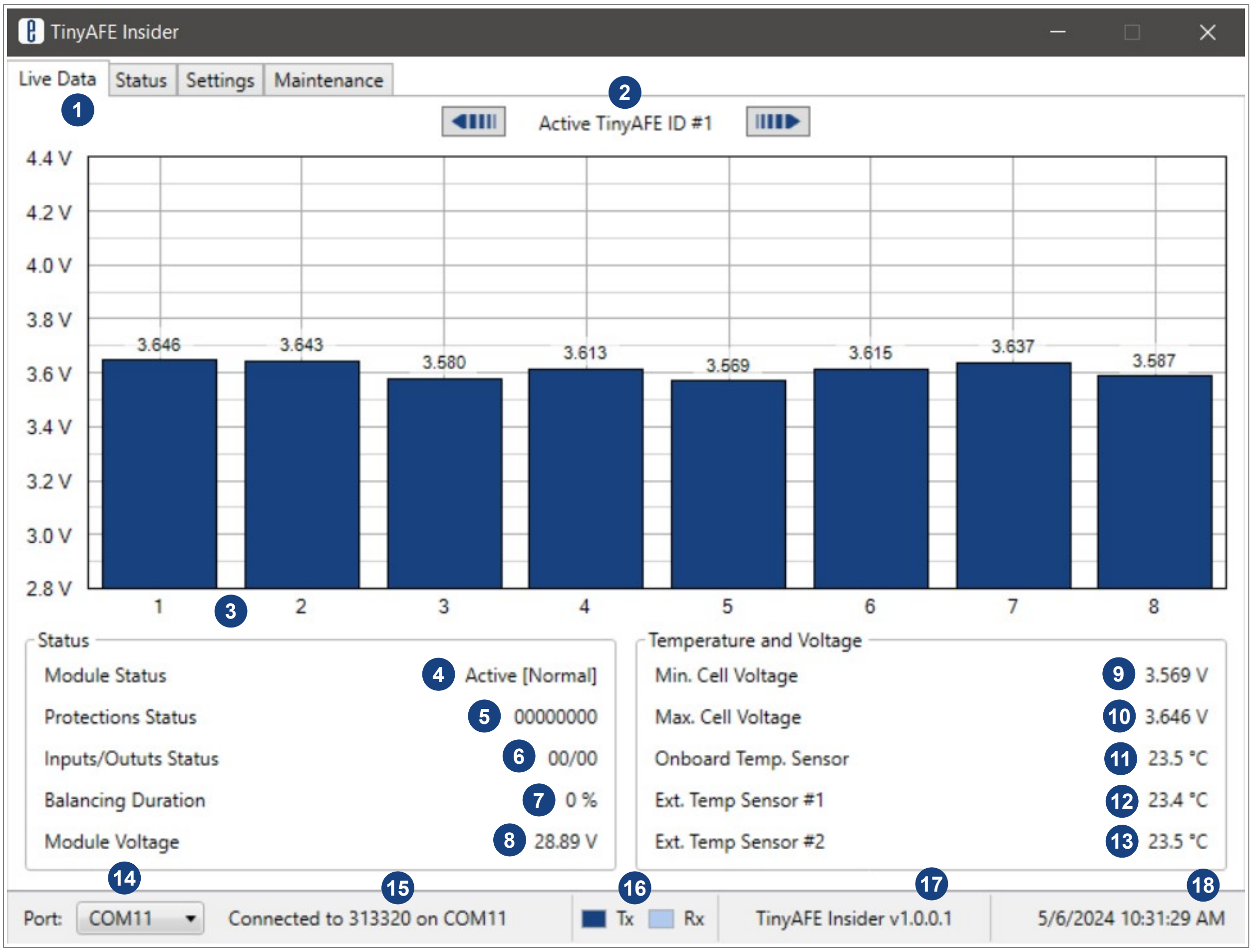


Figure 7: TinyAFE Insider Live Data tab screenshot

Table 2: TinyAFE Insider Live Data tab information

1	***Live Data* tab**
2	Buttons for selecting active *TinyAFE* device (*ID* numbers *1-20*). Only the data of the selected device in the network will be displayed.
3	Graph displaying the voltage of active *TinyAFE* module cells. The number of displayed cells equals the *Number of Series Cells* 37 parameter in the *Settings* 24 tab. When a cell is balancing, an orange bar appears on top of the blue cell voltage column.
4	Status of active *TinyAFE* device operation. Clicking links to the *Status* 19 tab.
5	Status of active *TinyAFE* device protections. Clicking links to the *Status* 19 tab.
6	General Inputs/Outputs status of active *TinyAFE* device. Clicking links to the *Status* 19 tab.

7	Duration (intensity) of active *TinyAFE* device cell balancing.
8	Calculation of active *TinyAFE* device module voltage as the sum of voltages of all *TinyAFE* module cells connected in series. Correct module voltage calculation requires setting the correct *Number of Series Cells* 37 parameter in the *Settings* 24 tab.
9	Minimum cell voltage of active *TinyAFE* device.
10	Maximum cell voltage of active *TinyAFE* device.
11	Temperature value from the onboard *NTC* temperature sensor of the active *TinyAFE* device.
12	Temperature value from External *NTC* or *Active Multipoint* temperature sensor #1.
13	Temperature value from External *NTC* or *Active Multipoint* temperature sensor #2.
14	Setting for communication port selection. Selecting *Auto* scans all available communication ports automatically to detect connected devices.
15	Status of *TinyAFE Insider* connection to *TinyAFE* device. Clicking links to *Device Manager*.
16	Communication status of *TinyAFE Insider* on the *TX* and *RX* lines.
17	Version of *TinyAFE Insider.* Clicking links to *Maintenance* 48 tab*.*
18	Current date and time.

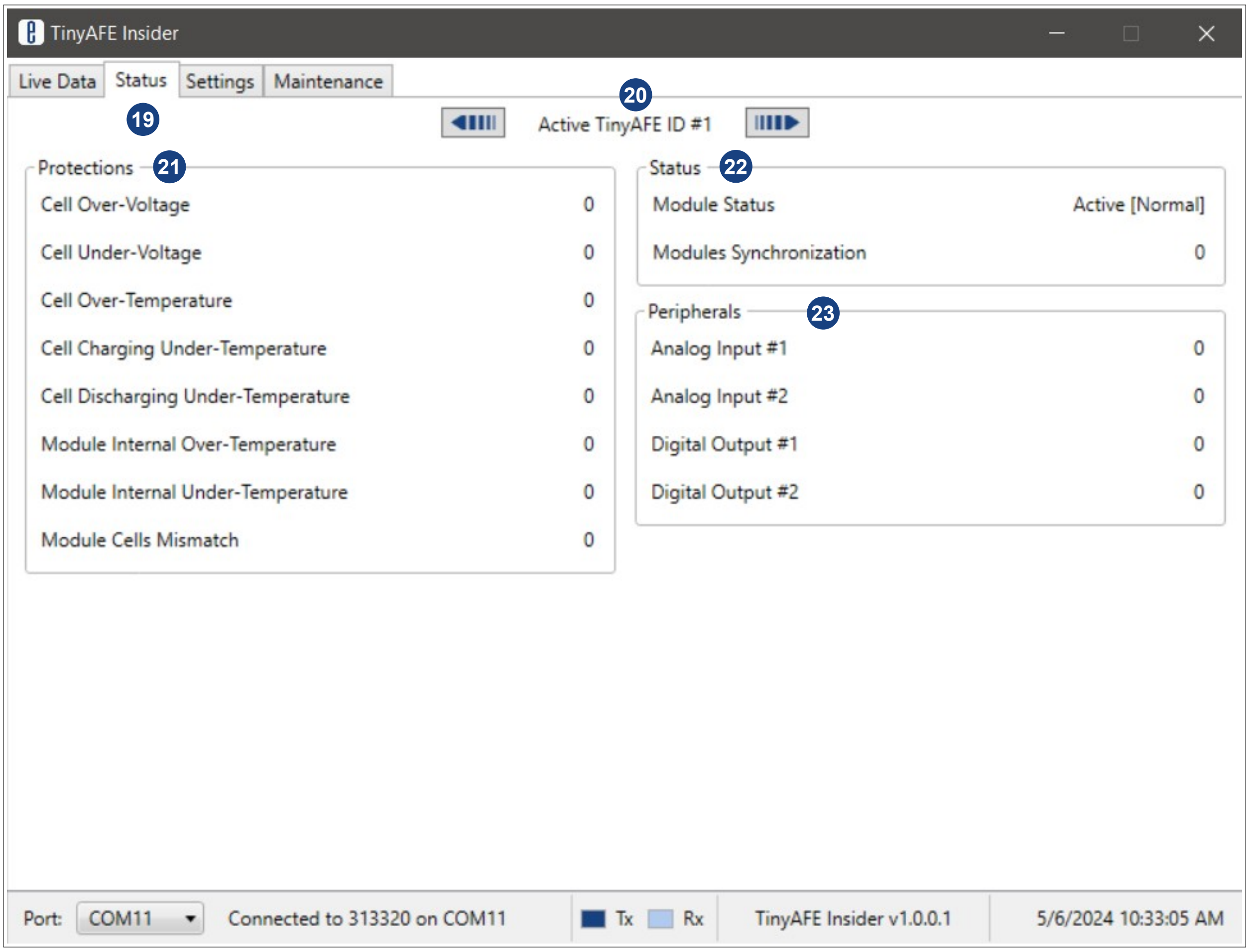


Figure 8: TinyAFE Insider device Status tab screenshot

Table 3: TinyAFE Insider Device Status tab information

19	***Status* tab**
20	Buttons for selecting active *TinyAFE* device (*ID* numbers *1-20*). Only the status of the selected device in the network will be displayed.
21	Status of active *TinyAFE* device protections (*0* – protection not triggered, *1* – protection triggered).
22	Status of active *TinyAFE* device operation. Modules synchronization status indicates whether data acquisition of all modules in the network is synchronized. Special command must be sent to synchronize modules on the network.
23	Status of analog inputs (*0* – input not detected, *1* – input detected) and digital outputs (*0* – output disabled, *1* – output enabled) of active *TinyAFE* device.

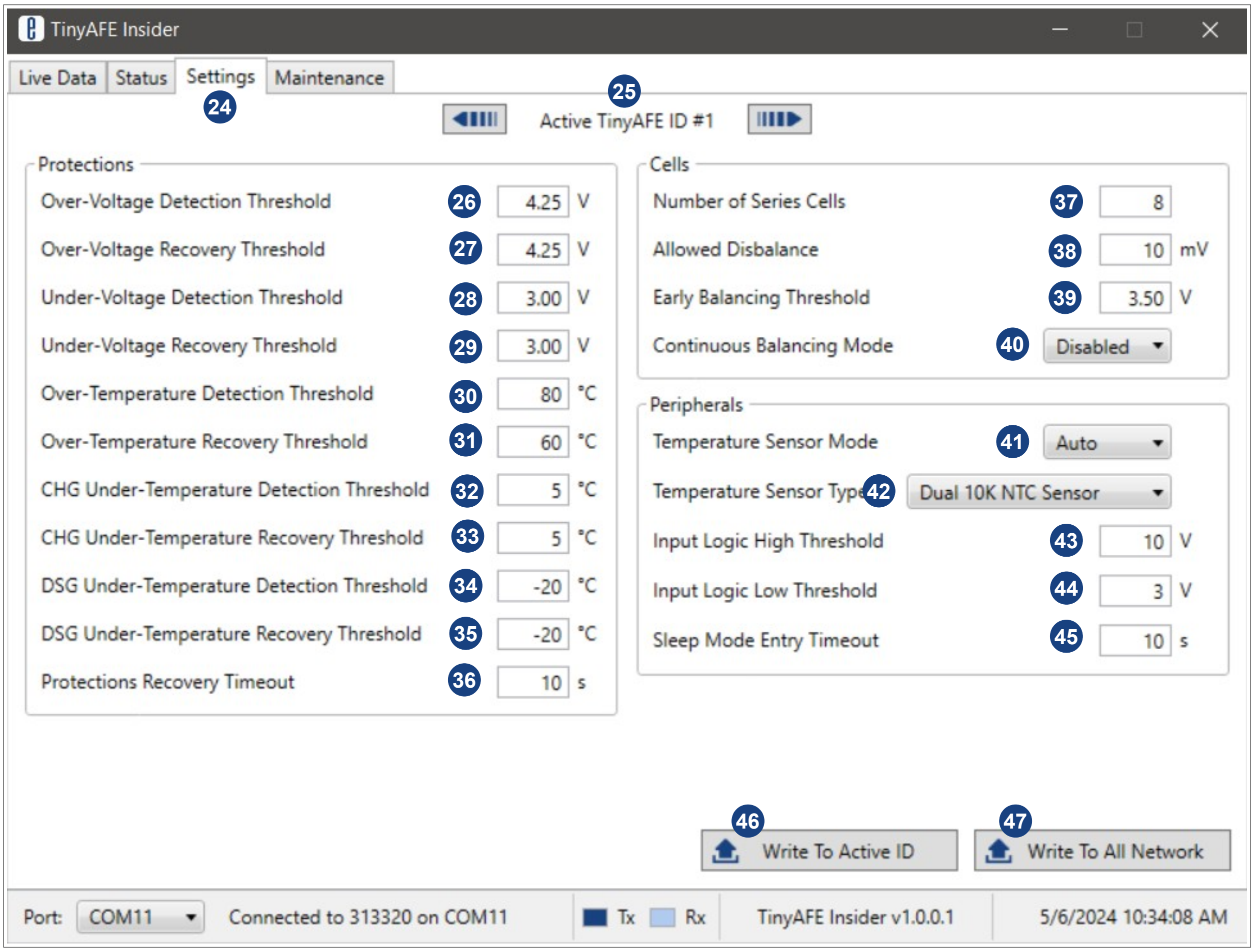


Figure 9: TinyAFE Insider device Settings tab screenshot

Table 4: TinyAFE Insider Device Settings tab information

24	***Settings* tab**
25	Buttons for selecting active *TinyAFE* device (*ID* numbers *1-20*). Only the settings of the selected device in the network will be displayed.
26	*Over-Voltage* protection detection threshold of active *TinyAFE* device: Min. value – *1.2 V*; Max. Value – *4.5 V*.
27	*Over-Voltage* protection recovery threshold of active *TinyAFE* device: Min. value – *1.2 V*; Max. Value – *4.5 V*.

28	*Under-Voltage* protection detection threshold of active *TinyAFE* device: Min. value – *0.8 V*; Max. Value – *4.0 V*.
29	*Under-Voltage* protection recovery threshold of active *TinyAFE* device: Min. value – *0.8 V*; Max. Value – *4.0 V*.
30	*Over-Temperature* protection detection threshold of active *TinyAFE* device: Min. value – *20 °C*; Max. Value – *100 °C*.
31	*Over-Temperature* protection recovery threshold of active *TinyAFE* device: Min. value – *20 °C*; Max. Value – *100 °C*.
32	*Charging Under-Temperature* protection detection threshold of active *TinyAFE* device: Min. value – *-40 °C*; Max. Value – *10 °C*.
33	*Charging Under-Temperature* protection recovery threshold of active *TinyAFE* device: Min. value – *-40 °C*; Max. Value – *10 °C*.
34	*Discharging Under-Temperature* protection detection threshold of active *TinyAFE* device: Min. value – *-40 °C*; Max. Value – *10 °C*.
35	*Discharging Under-Temperature* protection recovery threshold of active *TinyAFE* device: Min. value – *-40 °C*; Max. Value – *10 °C*.
36	Automatic recovery timeout value of active *TinyAFE* device protections: Min. value – *1 sec.* Max. Value – *3600 sec.*
37	Number of series cells of active *TinyAFE* device: Min. value – *4*; Max. Value – *16*.
38	Allowed imbalance of active *TinyAFE* device cells: Min. value – *1 mV*; Max. Value – *500 mV*.
39	Early balancing threshold (voltage threshold at which cells start balancing) of active *TinyAFE* device cells: Min. value – *1.0 V*; Max. Value – *4.0 V*.
40	Continuous balancing mode of active *TinyAFE* device cells: *Disabled* – module cells balancing process can be controlled using communication commands; *Enabled* – continuous cells balancing process (if cell voltage is higher than *Early Balancing Threshold* 39, and cell imbalance is higher than the cells *Allowed Disbalance* 38 settings).
41	Temperature sensor mode of active *TinyAFE* device: *Auto* – external temperature sensors are detected automatically (no protection detected if the sensor is disconnected). *Enabled* – external temperature sensors are always enabled (*under-temperature* protection is detected if the sensor is disconnected). *Disabled* – external temperature sensors are always disabled (not used).
42	Temperature sensor type of active *TinyAFE* device: *Dual 10K NTC Sensor* – One *NTC* temperature sensor on each device temperature channel can be connected. Supported *NTC* sensor: *10 K @ 25 °C*, Beta value *3977 K*. *Multipoint Active Sensor* – Special *ENEPAQ Multipoint Active NTC* temperature sensor for max. temperature (hot-spot) detection.
43	Logic high threshold of active *TinyAFE* device analog input: Min. value – *2 V*;

	Max. Value – *70 V*.
44	Logic low threshold of active *TinyAFE* device analog input: Min. value – *1 V*; Max. Value – *65 V*.
45	Entry timeout for sleep (ultra-low power) mode of active *TinyAFE* device: Min. value – *0 sec.* (device is always active); Max. Value – *3600 sec.* (device enters sleep mode if there is no communication within the timeout).
46	*Write To Active ID* button writes all settings only to the active *TinyAFE* device's 25 internal memory. After successful upload, the application reads back all the newest settings information, including min. and max. settings values.
47	*Write To All Network* button writes (broadcasts) all settings to all devices on the network.

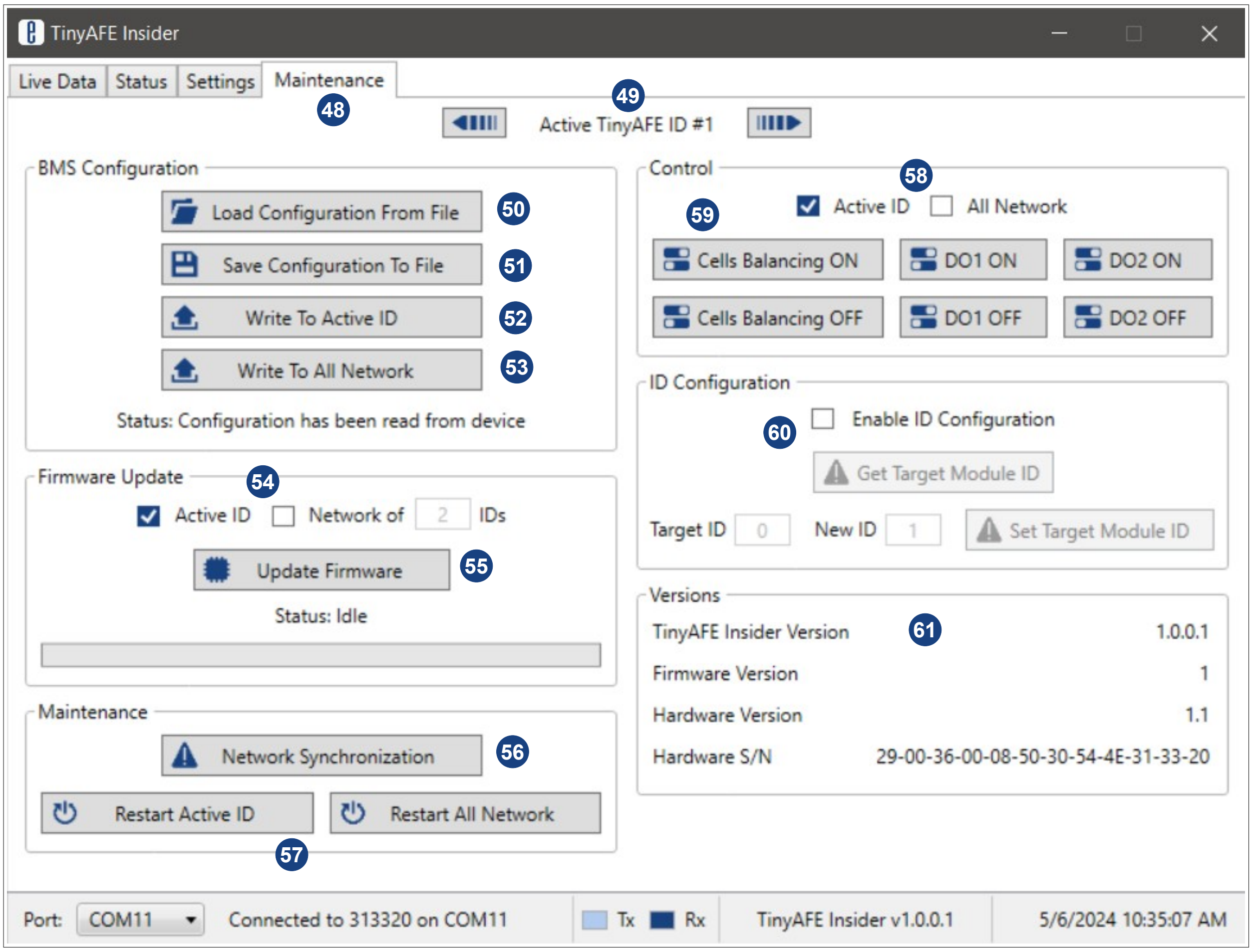


Figure 10: TinyAFE Insider Maintenance tab screenshot

Table 5: TinyAFE Insider Maintenance tab information

48	***Maintenance tab***
49	Buttons for selecting active *TinyAFE* device (*ID* numbers *1-20*).
50	Load *TinyAFE* device settings from a configuration file.
51	Save *TinyAFE* device settings to a configuration file.
52	*Write To Active ID* button writes all settings only to the active *TinyAFE* device's 49 internal memory. After successful upload, the application reads back all the newest settings information, including min. and max. settings values.

53	*Write To All Network* button writes (broadcasts) all settings to all devices on the network.
54	Firmware update mode selection: *Active ID* – update firmware only to *Active ID* 49 device on the network; *Network* – update firmware to all devices on the network (all devices on the network must be enumerated with the *ID* numbers starting from *1* to a maximum of *20 IDs*).
55	Button to start firmware update for *TinyAFE* device.
56	Network synchronization button. Synchronize data acquisition for all devices on the network. Sending a synchronization command is recommended after the device powers up, restarts, or wakes up from sleep mode.
57	Button for safe restart of *TinyAFE* device: *Restart Active ID* – restarts only active *TinyAFE* device 49; *Restart All Network* – restarts all devices on the network.
58	Selection of *TinyAFE* device control mode: *Active ID* – buttons controls only *Active ID* 49 device on the network; *All Network* – buttons controls (broadcasts) all devices on the network.
59	Buttons for controlling *TinyAFE* device cells balancing and digital outputs.
60	Mode for configuring *TinyAFE* device *ID* (a valid communication port must be selected to activate configuration mode): **Note**: In this mode, only one *TinyAFE* device must be connected to the network. Only when all devices are enumerated with different *IDs* (*1* to *20*) can they be connected to the same communication network. *Target ID* – current *ID* of *TinyAFE* device (default device *ID* from factor is *0*). *New ID* – new *ID* of *TinyAFE* device on the network (*1* to *20*).
61	Versions of *TinyAFE* device : Application version; Firmware version; Hardware version; Unique Hardware serial number.

7. Communicating with TinyAFE Devices in a High-Voltage Battery Network

Once all *TinyAFE* devices are pre-programmed with unique *ID* numbers, they can be connected to a common high-voltage battery network. The user-side master device can then communicate with all *TinyAFE* devices on the network to read data, change settings, and control peripherals. *TinyAFE* devices use a proprietary isolated communication interface that is not directly accessible to the user. Instead, the user-side master device connects to the *TinyAFE* network through a special *ENEPAQ UART-to-isolated* communication converter, allowing communication with the network via a simple *UART* interface.

The *ENEPAQ UART-to-isolated* communication converter offers two user-side interfaces: simple *UART* and *USB* (*Virtual COM Port*). However, only one interface can be used at a time. All *TinyAFE* devices on the network operate autonomously and are independent of the sequence or interval of commands sent by the user. The user can decide which data to scan, how to scan it, and at what intervals. All communication commands, as well as the internal register map of device data and settings, are provided in the *TinyAFE* communication protocol.

The first byte of each communication command indicates the *ID* number of the device on the shared network from which the data is to be retrieved. Commands for setting configuration and device control can be sent to all devices simultaneously using the *ID* value *0* (broadcast feature). Depending on the communication commands, the architecture of the user-side master device, and the interval of commands sent, an additional delay (at least *100 µs*) between separate commands may be required. Recommended communication flow charts from the user-side master device are provided.

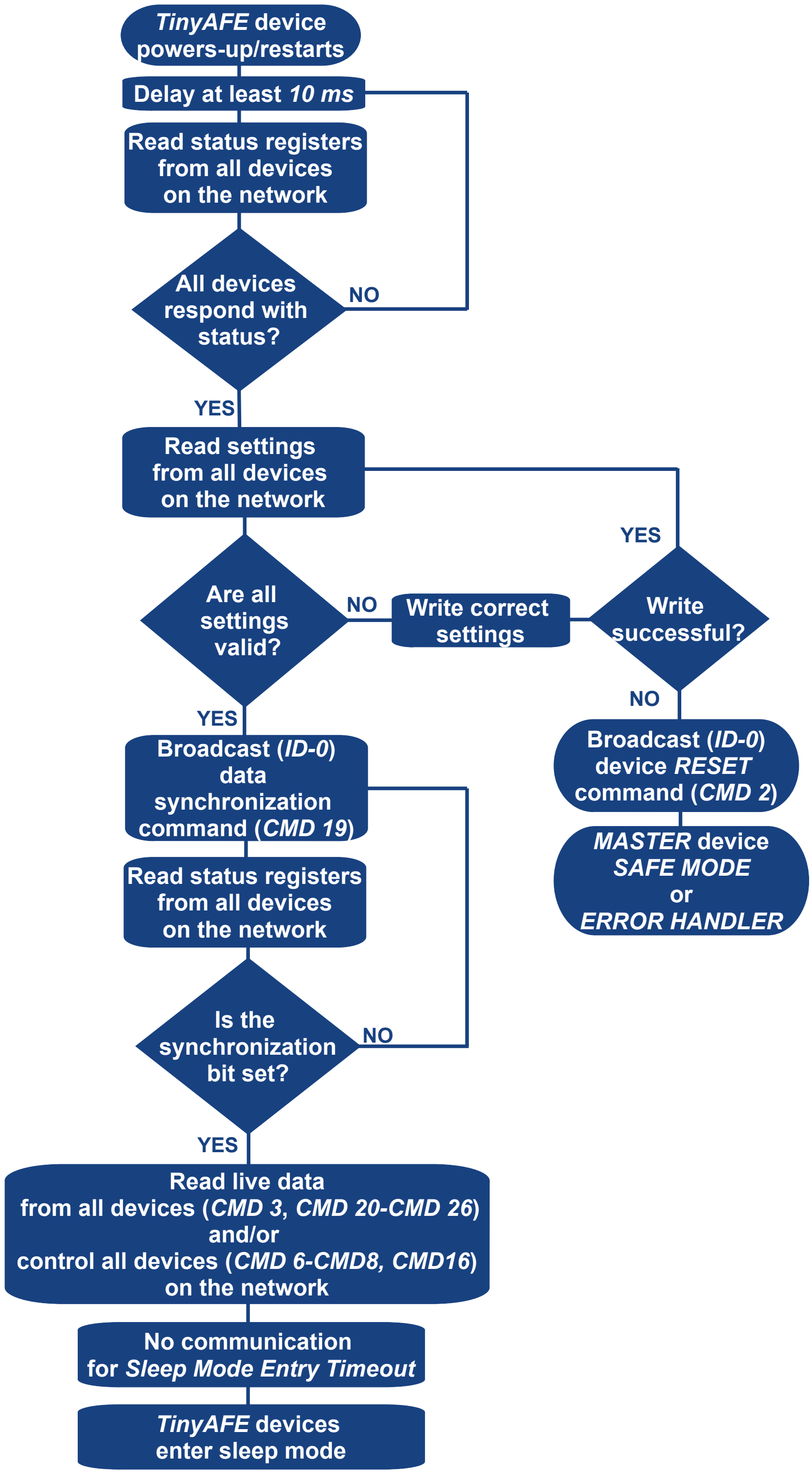


Figure 11: Recommended simplified user-side master device communication sequence with *TinyAFE* devices.

**An additional delay (at least 100 µs) between different commands may be required*

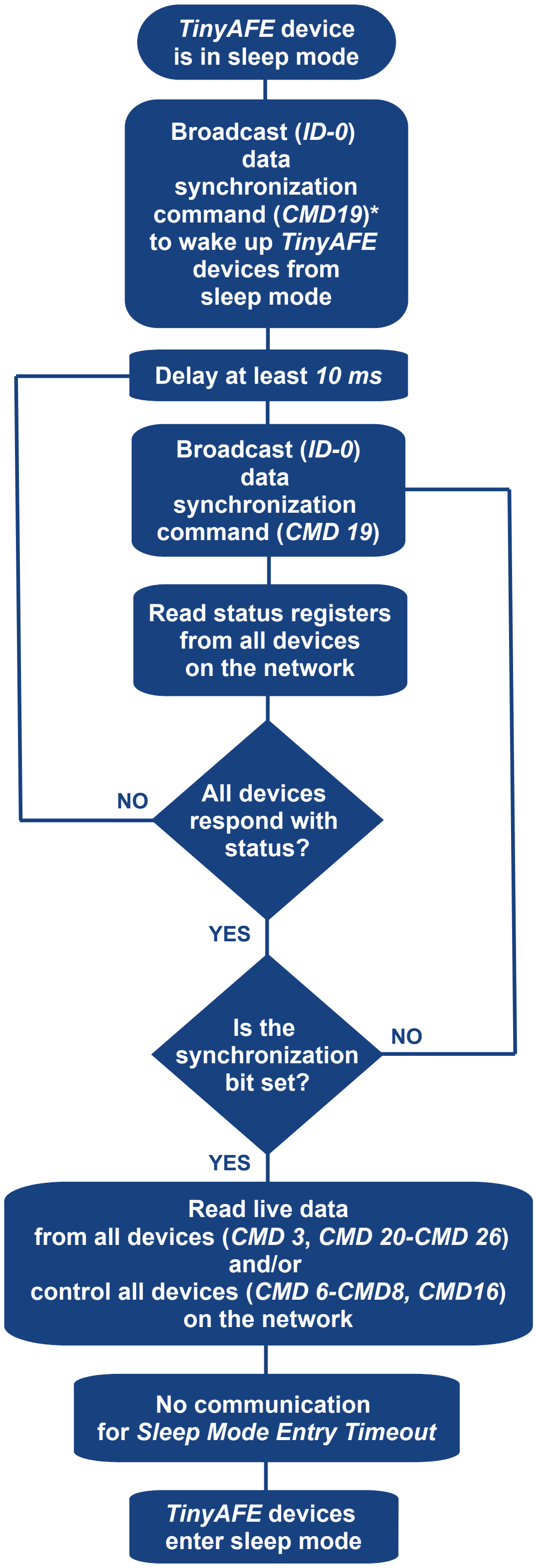


Figure 12: Recommended simplified sequence for waking up *TinyAFE* devices from sleep mode.
*Any communication command can be used to wake up the *TinyAFE* device from sleep mode, but the first command will be rejected.
***Additional delay (at least 100 µs) between different commands may be required*

8. TinyAFE Device UART Communication Protocol

The *TinyAFE* device features a single isolated communication interface, which serves to interconnect all *TinyAFE* devices within a stacked network essential for high-voltage batteries. This internal isolated communication interface is not directly accessible to the user. To facilitate user access to the data on the stacked network, a dedicated *ENEPAQ* converter is required to convert simple *UART* interface to isolated communication. Users interact with *TinyAFE* devices on the common network through this isolated converter using a simple *UART* interface with the communication commands listed below.
Each *TinyAFE* device on the network must be assigned a unique *ID* number ranging from *1* to *20*. The default *ID* number, *0*, must be changed by the user before connecting to the stacked network. The first byte of the communication commands indicates the *ID* number of the *TinyAFE* device with which communication is being conducted. A device with a modified *ID* (*1* to *20*) also accepts commands with *ID 0*. In essence, if a command is sent with *ID 0*, all devices in the network will accept that command (broadcast feature).

Note: *UART* configuration: baudrate *115200 bit/s*, *8* data bits, *1* stop bit, no parity, no flow control. *UART* configuration is not allowed to be changed by the user.

BMS acknowledgement

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	CMD	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*
CMD – Command code

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	CMD	ERROR	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*
ERROR – Error code

CMD 2. Reset TinyAFE Device/Settings

Request to BMS				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x00 or ID	0x02	OPTIONS	CRC:LSB	CRC:MSB
		0x00 – Restart Device		
		0x01 – Reset Default Settings		

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	0x02	CRC:LSB	CRC:MSB

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x02	ERROR	CRC:LSB	CRC:MSB

CMD 3. Read TinyAFE registers block (MODBUS compatible)

Request to BMS							
Byte1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
ID	0x03	ADDR1:MSB	ADDR1:LSB	0x00	RL	CRC:LSB	CRC:MSB
		[UINT_16]					

ID – *TinyAFE* module *ID (1-20)*
ADDR – First registers block address
RL – Registers to read. Max. *127* registers *(0x7F)*

Response from BMS [OK]									
Byte1	Byte 2	Byte 3	Byte 4	Byte 5	...	Byte n*2+2	Byte n*2+3	Byte n*2+4	Byte n*2+5
ID	0x03	PL	DATA1:MSB	DATA1:LSB	...	DATAn:MSB	DATAn:LSB	CRC:LSB	CRC:MSB
			[UINT_16]			[UINT_16]			

PL – Payload length in bytes

Response from BMS [ERROR]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x03	ERROR	CRC:LSB	CRC:MSB

CMD 4. Read TinyAFE module ID

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
0x00 or ID	0x04	CRC:LSB	CRC:MSB

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID* (*1-20*)

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Device ID	0x04	Device ID	CRC:LSB	CRC:MSB

CMD 5. Write TinyAFE module ID

Request to BMS				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x00 or ID	0x05	NEW ID	CRC:LSB	CRC:MSB

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
NEW ID	0x01	0x05	CRC:LSB	CRC:MSB

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x05	ERROR	CRC:LSB	CRC:MSB

CMD 6. Control cells balancing

Request to BMS				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x00 or ID	0x06	OPTIONS	CRC:LSB	CRC:MSB
		0x00 – Disable Balancing		
		0x01 – Enable Balancing		

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	0x06	CRC:LSB	CRC:MSB

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x06	ERROR	CRC:LSB	CRC:MSB

CMD 7. Control DO1 output

Request to BMS				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x00 or ID	0x07	OPTIONS	CRC:LSB	CRC:MSB
		0x00 – Output Disable		
		0x01 – Output Enable		

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	0x07	CRC:LSB	CRC:MSB

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x07	ERROR	CRC:LSB	CRC:MSB

CMD 8. Control DO2 output

Request to BMS				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x00 or ID	0x08	OPTIONS	CRC:LSB	CRC:MSB
		0x00 – Output Disable		
		0x01 – Output Enable		

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	0x08	CRC:LSB	CRC:MSB

Response from BMS [NACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x08	ERROR	CRC:LSB	CRC:MSB

CMD 13. Read TinyAFE settings values (min, max, default, current)

Request to *BMS*						
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
ID	0x0D	SETTINGS	0x00	RL	CRC:LSB	CRC:MSB
		0x00 – MIN				
		0x01 – MAX				
		0x02 – DEFAULT				
		0x03 – CURRENT				

ID – *TinyAFE* module *ID (1-20)*
RL – Registers to read. Max. *50* (*0x32*) registers

Response from *BMS* [OK]									
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	...	Byte n*2+2	Byte n*2+3	Byte n*2+4	Byte n*2+5
ID	0x0D	PL	DATA1:LSB	DATA1:MSB	...	DATAn:LSB	DATAn:MSB	CRC:LSB	CRC:MSB
			[UINT_16]			[UINT_16]			

PL – Payload length in bytes

Response from *BMS* [ERROR]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x0D	ERROR	CRC:LSB	CRC:MSB

CMD 16. Write TinyAFE registers block (MODBUS compatible)

Request to *BMS*									
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	...
0x00 or ID	0x10	ADDR:MSB	ADDR:LSB	0x00	RL	PL	DATA1:MSB	DATA1:LSB	...
		[UINT_16]					[UINT_16]		

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Byte n*2+6	Byte n*2+7	Byte n*2+8	Byte n*2+9
DATAn:MSB	DATAn:LSB	CRC:LSB	CRC:MSB
[UINT_16]			

Response from *BMS* [OK]							
Byte1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
ID	0x10	ADDR:MSB	ADDR:LSB	0x00	RL	CRC:LSB	CRC:MSB
		[UINT_16]					

ADDR – First registers block address
RL – Registers to write. Max. *50* registers (*0x32*)
PL – Payload length in bytes
DATA – Registers block values to write

Response from BMS [ERROR]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x00	0x10	ERROR	CRC:LSB	CRC:MSB

CMD 19. Synchronize Slave Modules Measurements

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
0x00 or ID	0x13	CRC:LSB	CRC:MSB

0x00 (Byte 1) – All devices on the network accept this command, but no response from devices is received (command broadcast)
ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
ID	0x01	0x13	CRC:LSB	CRC:MSB

CMD 20. Read Cell Voltages

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4

ID	0x14	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]									
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	...	Byte n*2+2	Byte n*2+3	Byte n*2+4	Byte n*2+5
ID	0x14	PL	DATA1:LSB	DATA1:MSB	...	DATAn:LSB	DATAn:MSB	CRC:LSB	CRC:MSB
			[UINT_16]			[UINT16]			

PL – Payload length in bytes
DATA – Cell voltage

CMD 21. Read Temperatures

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
ID	0x15	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]										
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
ID	0x15	PL	DATA1:LSB	DATA1:MSB	DATA2:LSB	DATA2:MSB	DATA3:LSB	DATA3:MSB	CRC:LSB	CRC:MSB
			[INT_16]		[INT_16]		[INT16]			

PL – Payload length in bytes
DATA1 – Internal *PCB* temperature
DATA2 – External temperature sensor *#1* temperature value (value of *-32768* if not connected)
DATA3 – External temperature sensor *#2* temperature value (value of *-32768* if not connected)

CMD 22. Read Minimal Cell Voltage

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
ID	0x16	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x16	DATA:LSB	DATA:MSB	CRC:LSB	CRC:MSB
		[UINT_16]			

DATA – Minimal cell voltage

CMD 23. Read Maximal Cell Voltage

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
ID	0x17	CRC:LSB	CRC:MSB

ID – *TinyAFE module ID (1-20)*

Response from BMS [ACK]					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
ID	0x17	DATA:LSB	DATA:MSB	CRC:LSB	CRC:MSB
		[UINT_16]			

DATA – Maximal cell voltage

CMD 24. Read Module Voltage

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
ID	0x18	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
ID	0x18	DATA:LSB	DATA	DATA	DATA:MSB	CRC:LSB	CRC:MSB
		[UINT_32]					

DATA – Module Voltage

CMD 25. Read device, I/O, protection states

Request to BMS			
Byte 1	Byte 2	Byte 3	Byte 4
ID	0x19	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
ID	0x19	DATA1	DATA2	DATA3:LSB	DATA3:MSB	CRC:LSB	CRC:MSB

	[UINT_8]	[UINT_8]	[UINT_16]

DATA1 – Device state (*0-6* Bits) (*0x00-Init, 0x01-Idle, 0x02-Fault/Protection*), Synchronization status (*7 Bit*)
DATA2 – I/O status (*bit 0 – AI1, bit 1 – AI2, bit 2 – DO1, bit 3 – DO2*)
DATA3 – Protections status (*bit 0 – Cell OV, bit 1- Cell UV, bit 2 – Cell OT, bit 3 – Cell CHG UT, bit 4 – Cell DSG UT, bit 5 – PCB OT, bit 6 – PCB UT, bit 7 – Cells mismatch*)

CMD 26. Read balancing status, duration

Request to BMS

Byte 1	Byte 2	Byte 3	Byte 4
ID	0x1A	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 7	Byte 8
ID	0x1A	DATA1:LSB	DATA1:MSB	DATA2	CRC:LSB	CRC:MSB
		[UINT_16]		[UINT_8]		

DATA1 – Baancing bits
DATA2 – Balancing durations (*%*)

CMD 30. Read device version

Request to BMS

Byte 1	Byte 2	Byte 3	Byte 4
ID	0x1E	CRC:LSB	CRC:MSB

Response from BMS [ACK]

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11	Byte 12
ID	0x1E	PL	DATA1	DATA2	DATA3:LSB	DATA3:MSB	DATA4:LSB	DATA4:MSB	DATA5	CRC:LSB	CRC:MSB
			[UINT_8]	[UINT_8]	[UINT_16]		[UINT_16]		[UINT_8]		

PL – Payload length
DATA1 – Hardware version
DATA 2 – Hardware changes version
DATA 3 – Firmware public version
DATA4 – Firmware Internal version
DATA 5 – Bootloader version

CMD 31. Read device serial number

Request to BMS

Byte 1	Byte 2	Byte 3	Byte 4
ID	0x1F	CRC:LSB	CRC:MSB

ID – *TinyAFE* module *ID (1-20)*

Response from BMS [ACK]

Byte 1	Byte 2	Byte 3	Byte 4	...	Byte 15	Byte 16	Byte 17
ID	0x1F	PL	DATA1	...	DATA12	CRC:LSB	CRC:MSB
			[UINT_8]		[UINT_8]		

PL – Payload length
DATA – Device serial number (*12 bytes*)

9. TinyAFE Registers Map

9.1. Live Data

<table>
<tr><th>Reg. Nr.</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>Action</th></tr>
<tr><td>0</td><td colspan="16">Cell 1 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>1</td><td colspan="16">Cell 2 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>2</td><td colspan="16">Cell 3 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>3</td><td colspan="16">Cell 4 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>4</td><td colspan="16">Cell 5 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>5</td><td colspan="16">Cell 6 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>6</td><td colspan="16">Cell 7 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>7</td><td colspan="16">Cell 8 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>8</td><td colspan="16">Cell 9 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>9</td><td colspan="16">Cell 10 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>10</td><td colspan="16">Cell 11 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>11</td><td colspan="16">Cell 12 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>12</td><td colspan="16">Cell 13 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>13</td><td colspan="16">Cell 14 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>14</td><td colspan="16">Cell 15 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>15</td><td colspan="16">Cell 16 Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>16</td><td colspan="16">External Temp. Sensor #1 Temperature [INT16] / Resolution 0.1 °C</td><td>R</td></tr>
<tr><td>17</td><td colspan="16">External Temp. Sensor #2 Temperature [INT16] / Resolution 0.1 °C</td><td>R</td></tr>
<tr><td>18</td><td colspan="16">Internal PCB Temperature [INT_16] / Resolution 0.1 °C</td><td>R</td></tr>
<tr><td>19</td><td colspan="16">Module Minimal Cell Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>20</td><td colspan="16">Module Maximal Cell Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>21</td><td colspan="16" rowspan="2">Module Voltage [UINT_32] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>22</td><td>R</td></tr>
<tr><td rowspan="2">23</td><td colspan="6" rowspan="2">Module State [0-6 Bits] [UINT_8] /
0x00 – Initialization,
0x01 – Active [Normal],
0x02 – Active [Protection],
0x03 – Sleep</td><td colspan="2" rowspan="2">Synchronization Status [7 Bit]</td><td colspan="8">I/O Status [UINT_8]</td><td rowspan="2">R</td></tr>
<tr><td colspan="2">Bit 0 - AI1</td><td colspan="2">Bit 1 – AI2</td><td colspan="2">Bit 2 – DO1</td><td colspan="2">Bit 3 - DO2</td></tr>
<tr><td rowspan="2">24</td><td colspan="16">Protections Status [UINT_16]</td><td>R</td></tr>
<tr><td colspan="2">Bit 0 – Cell OV</td><td colspan="2">Bit 1 – Cell UV</td><td colspan="2">Bit 2 – Cell OT</td><td colspan="2">Bit 3 – Cell CHG UT</td><td colspan="2">Bit 4 – Cell DSG UT</td><td colspan="2">Bit 5 – PCB OT</td><td colspan="2">Bit 6 - PCB UT</td><td colspan="2">Bit 7 – Cells Mismatch</td><td>R</td></tr>
<tr><td>25</td><td colspan="16">Balancing Decision Bits [UINT_16] / Bit 0 – Cell 1 to Bit 15 – Cell 16</td><td>R</td></tr>
<tr><td>26</td><td colspan="16">Balancing Duration [UINT_16] / Resolution 1 %</td><td>R</td></tr>
<tr><td>27</td><td colspan="16">Battery Minimal Cell Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>28</td><td colspan="16">Battery Maximal Cell Voltage [UINT_16] / Resolution 1 mV</td><td>R</td></tr>
<tr><td>29...99</td><td colspan="16">Reserved</td><td>R</td></tr>
</table>

9.2. Settings [Non-Volatile Memory]

Note: All settings are stored in the non-volatile memory of the *TinyAFE* device. Once configured, the settings are automatically restored after powering up, restarting, or waking up from sleep mode events.

<table>
<tr><th>Reg. Nr.</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>Action</th></tr>
<tr><td>100</td><td colspan="16">Cells In Series [UINT_16] [4-16]</td><td>R/W</td></tr>
<tr><td>101</td><td colspan="16">Cell Over-Voltage Detection Threshold [UINT_16] [120-450] / Resolution 10 mV</td><td>R/W</td></tr>
<tr><td>102</td><td colspan="16">Cell Over-Voltage Recovery Threshold [UINT_16] [120-450] / Resolution 10 mV</td><td>R/W</td></tr>
<tr><td>103</td><td colspan="16">Cell Under-Voltage Detection Threshold [UINT_16] [80-400] / Resolution 10 mV</td><td>R/W</td></tr>
<tr><td>104</td><td colspan="16">Cell Under-Voltage Recovery Threshold [UINT_16] [80-400] / Resolution 10 mV</td><td>R/W</td></tr>
<tr><td>105</td><td colspan="16">Cell Over-Temperature Detection Threshold [INT_16] [20-100] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>106</td><td colspan="16">Cell Over-Temperature Recovery Threshold [INT_16] [20-100] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>107</td><td colspan="16">Cell CHG Under-Temperature Detection Threshold [INT_16] [-40-10] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>108</td><td colspan="16">Cell CHG Under-Temperature Recovery Threshold [INT_16] [-40-10] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>109</td><td colspan="16">Cell DSG Under-Temperature Detection Threshold [INT_16] [-40-10] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>110</td><td colspan="16">Cell DSG Under-Temperature Recovery Threshold [INT_16] [-40-10] / Resolution 1 °C</td><td>R/W</td></tr>
<tr><td>111</td><td colspan="16">Protections Recovery Timer Value [UINT_16] [1-3600] / Resolution 1 sec.</td><td>R/W</td></tr>
<tr><td>112</td><td colspan="16">Temperature Sensor Type [UINT_16] / 0x00 - Dual 10K NTC, 0x01 - Multipoint Active Sensor</td><td>R/W</td></tr>
<tr><td>113</td><td colspan="16">Temperature Sensors Detection Mode [UINT_16] / 0x00 – Auto, 0x01 – Enabled, 0x02 - Disabled</td><td>R/W</td></tr>
<tr><td>114</td><td colspan="16">Allowed Cells Disbalance [UINT_16] [1-500] / Resolution 1 mV</td><td>R/W/</td></tr>
</table>

115	Early Balancing Threshold [UINT_16] [100-400] / Resolution 1 mV	R/W
116	Continuous Balancing Mode [UINT_16] / 0x00 – Disabled, 0x01 - Enabled	R/W
117	Analog Input Logic High Threshold [UINT_16] [2-70] / Resolution 1V	R/W
118	Analog Input Logic Low Threshold [UINT_16] [1-65] / Resolution 1V	R/W
119	Sleep Activation Timer Value [UINT_16] [0-3600] / Resolution 1 sec.	R/W
120...149	Reserved	R/W

9.3. Information Data

Reg. Nr.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Action
150	Hardware Version [8 bits LSB]								Hardware Changes Version [8 bits MSB]								R
151	Public Release Firmware Version [UINT_16]																R
152	Internal Firmware Version [UINT_16]																R
153	Bootloader Version [8 bits LSB]								Reserved								R
154	Device Serial Number [96 bits]																R
155																	
156																	
157																	
158																	
159																	

9.4. CRC checksum calculation

CRC stands for *Cyclic Redundancy Check*. It is two bytes added to the end of every command message for error detection. Every byte in the message is used to calculate the *CRC* value. The receiving device also must calculate the *CRC* and compare it to the *CRC* from sending device. If even one bit in the message is received incorrectly, the *CRC* values will be different and will result in an error. In the *TinyAFE UART* communication protocol the *CRC* checksum is *16* bit value, calculated based on standard *MODBUS CRC* polynomial $x^{16}+x^{15}+x^{2}+1$ (*0x8005* in *HEX* format). Below is the function example in *C* programming language that can be used as a reference to calculate the *16* bit *CRC* value:

```
const static uint16_t crcTable[256]={
        0x0000, 0xC0C1, 0xC181, 0x0140, 0xC301, 0x03C0, 0x0280, 0xC241,
        0xC601, 0x06C0, 0x0780, 0xC741, 0x0500, 0xC5C1, 0xC481, 0x0440,
        0xCC01, 0x0CC0, 0x0D80, 0xCD41, 0x0F00, 0xCFC1, 0xCE81, 0x0E40,
        0x0A00, 0xCAC1, 0xCB81, 0x0B40, 0xC901, 0x09C0, 0x0880, 0xC841,
        0xD801, 0x18C0, 0x1980, 0xD941, 0x1B00, 0xDBC1, 0xDA81, 0x1A40,
        0x1E00, 0xDEC1, 0xDF81, 0x1F40, 0xDD01, 0x1DC0, 0x1C80, 0xDC41,
        0x1400, 0xD4C1, 0xD581, 0x1540, 0xD701, 0x17C0, 0x1680, 0xD641,
        0xD201, 0x12C0, 0x1380, 0xD341, 0x1100, 0xD1C1, 0xD081, 0x1040,
        0xF001, 0x30C0, 0x3180, 0xF141, 0x3300, 0xF3C1, 0xF281, 0x3240,
        0x3600, 0xF6C1, 0xF781, 0x3740, 0xF501, 0x35C0, 0x3480, 0xF441,
        0x3C00, 0xFCC1, 0xFD81, 0x3D40, 0xFF01, 0x3FC0, 0x3E80, 0xFE41,
        0xFA01, 0x3AC0, 0x3B80, 0xFB41, 0x3900, 0xF9C1, 0xF881, 0x3840,
        0x2800, 0xE8C1, 0xE981, 0x2940, 0xEB01, 0x2BC0, 0x2A80, 0xEA41,
        0xEE01, 0x2EC0, 0x2F80, 0xEF41, 0x2D00, 0xEDC1, 0xEC81, 0x2C40,
        0xE401, 0x24C0, 0x2580, 0xE541, 0x2700, 0xE7C1, 0xE681, 0x2640,
        0x2200, 0xE2C1, 0xE381, 0x2340, 0xE101, 0x21C0, 0x2080, 0xE041,
        0xA001, 0x60C0, 0x6180, 0xA141, 0x6300, 0xA3C1, 0xA281, 0x6240,
        0x6600, 0xA6C1, 0xA781, 0x6740, 0xA501, 0x65C0, 0x6480, 0xA441,
        0x6C00, 0xACC1, 0xAD81, 0x6D40, 0xAF01, 0x6FC0, 0x6E80, 0xAE41,
        0xAA01, 0x6AC0, 0x6B80, 0xAB41, 0x6900, 0xA9C1, 0xA881, 0x6840,
        0x7800, 0xB8C1, 0xB981, 0x7940, 0xBB01, 0x7BC0, 0x7A80, 0xBA41,
        0xBE01, 0x7EC0, 0x7F80, 0xBF41, 0x7D00, 0xBDC1, 0xBC81, 0x7C40,
        0xB401, 0x74C0, 0x7580, 0xB541, 0x7700, 0xB7C1, 0xB681, 0x7640,
        0x7200, 0xB2C1, 0xB381, 0x7340, 0xB101, 0x71C0, 0x7080, 0xB041,
        0x5000, 0x90C1, 0x9181, 0x5140, 0x9301, 0x53C0, 0x5280, 0x9241,
        0x9601, 0x56C0, 0x5780, 0x9741, 0x5500, 0x95C1, 0x9481, 0x5440,
        0x9C01, 0x5CC0, 0x5D80, 0x9D41, 0x5F00, 0x9FC1, 0x9E81, 0x5E40,
        0x5A00, 0x9AC1, 0x9B81, 0x5B40, 0x9901, 0x59C0, 0x5880, 0x9841,
        0x8801, 0x48C0, 0x4980, 0x8941, 0x4B00, 0x8BC1, 0x8A81, 0x4A40,
```

```
        0x4E00, 0x8EC1, 0x8F81, 0x4F40, 0x8D01, 0x4DC0, 0x4C80, 0x8C41,
        0x4400, 0x84C1, 0x8581, 0x4540, 0x8701, 0x47C0, 0x4680, 0x8641,
        0x8201, 0x42C0, 0x4380, 0x8341, 0x4100, 0x81C1, 0x8081, 0x4040
};

uint16_t CRC16 (const uint8_t* data, uint16_t length)
{
        uint8_t tmp;
        uint16_t crcWord = 0xFFFF;

        while (length--)
        {
                tmp = *data++ ^ crcWord;
                crcWord >>= 8;
                crcWord ^= crcTable[tmp];
        }
        return crcWord;
}
```

9.5. *UART* communication examples

Note: When the *TinyAFE* device is in sleep mode, the first command must be sent twice. Upon receiving the first command, *TinyAFE* wakes up from sleep mode, but it will respond to the command only upon receiving it for the second time. Subsequently, *TinyAFE* does not re-enter sleep mode while communication is ongoing.

9.5.1. *MODBUS* write registers example

Below is an example, of how to configure cell *Over-Voltage Detection Threshold* to *4.2 V* value and *Over-Voltage Recovery Threshold* to 4*.15V* value only to *TinyAFE* device *ID 1* using *MODBUS* write command:

Over-Voltage Detection Threshold register address is *101* (*0x0065*)
Over-Voltage Recovery Threshold register address is *102* (*0x0066*)
According to *CMD 16 Write TinyAFE registers block (MODBUS compatible)*:
ADDR – *0x0065* (according to *TinyAFE* registers map);
RL – *0x02* (write two registers);
PL – *0x04* (all *TinyAFE* registers contains two bytes);
DATA – *4.2 V=4200 mV / 10 mV* (*0x01A4*), 4*.15 V*=4150 *mV / 10 mV* (*0x019F*);
CRC – *0x5F34*.
Command request bytes sequence to send:
0x01 0x10 0x00 0x65 0x00 0x02 0x04 0x01 0xA4 0x01 0x9F 0x34 0x5F.

If the command was sent successfully, *TinyAFE device ID 1* responds with data:
0x01 0x10 0x00 0x65 0x00 0x02 0x51 0xD7
Configured registers block start address – *0x0065*;
Configurred two registers – *0x0002*;
CRC – *0xD751*.

The second example demonstrates how to configure the module *Cells In Series* parameter to *8* cells for all the *TinyAFE* devices connected in a stacked network (broadcast settings):

Cells In Series register address is *100* (*0x0064*)
According to *CMD 16 Write TinyAFE registers block (MODBUS compatible)*:
ADDR – *0x0064* (according to *TinyAFE* registers map);
RL – *0x01* (write two registers);
PL – *0x02* (all *TinyAFE* registers contains two bytes);

DATA – *8 cells* (*0x0008*);
CRC – *0x22A2*.
Command request bytes sequence to send:
0x00 0x10 0x00 0x64 0x00 0x01 0x02 0x00 0x08 0xA2 0x22.

No response is received when the broadcast *ID* is *0x00*. Users can read individual device settings to confirm successful writing.

9.5.2. MODBUS read registers example

Below is an example, of how to read five cells voltage (cell *12* to cell *16*) from *TinyAFE* device with *ID 1* using *MODBUS* read command:
According to *CMD 3 Read BMS registers block (MODBUS compatible)*:
ADDR – *0x000B* (cell *12* address according to *TinyAFE* registers map);
RL – *0x05* (read five registers);
CRC – *0x0BF4*.
Command request bytes sequence to send:
0x01 0x03 0x00 0x0B 0x00 0x05 0xF4 0x0B.

If command was sent successfully, *TinyAFE device with ID 1* responds with data:
0x01 0x03 0x0A 0x0E 0x19 0x0D 0xEE 0x0E 0x1D 0x0E 0x33 0x0E 0x07 0x77 0x2D
Payload length – *0x0A* (*10* bytes);
Cell *1* voltage – *0x0E19* (*3609* decimal or *3.609 V* according to *TinyAFE* registers map)
Cell *2* voltage – *0x0DEE* (*3566* decimal or *3.566 V* according to *TinyAFE* registers map)
Cell *3* voltage – *0x0E1D*(*3613* decimal or *3.613 V* according to *TinyAFE* registers map)
Cell *4* voltage – *0x0E33*(*3635* decimal or *3.635 V* according to *TinyAFE* registers map)
Cell *5* voltage – *0x0E07* (*3591* decimal or *3.591 V* according to *TinyAFE* registers map)
CRC – *0x2D77*.

9.5.3. TinyAFE read temperatures example

Below is an example, of how to read temperature values (*ID 1*) using the *read TinyAFE* device temperatures command:

Command request bytes sequence to send:
0x01 0x15 0xC1 0xEF
CRC – *0xEFC1*.

If command was sent successfully, *BMS* responds with data:
0x01 0x15 0x06 0xEF 0x00 0xEE 0x00 0xEF 0x00 0xCF 0xD4
Payload length – *0x06* (*6* bytes);
TinyAFE internal temperature – *0x00EF* (*239* decimal or *23.9 °C* according to *TinyAFE* registers map)
TinyAFE external #*1* temperature – *0x00EE* (*276* decimal or *23.8 °C* according to *TinyAFE* registers map)
TinyAFE external #*2* temperature – *0x00EF* (*278* decimal or *23.9 °C* according to *TinyAFE* registers map)
CRC – *0xD4CF*.

Document revision history

Revision	Date	Description
A	2024-05-30	Initial release.