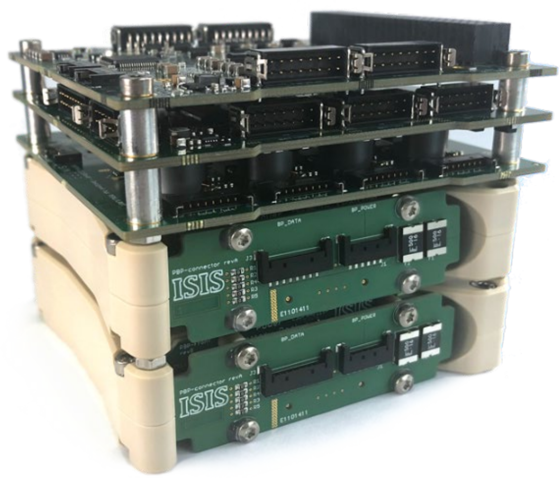
ISIS Electrical Power System 2 – Software ICD – IVID 7

ISIS.EPS2.ICD.SW.IVID.7



Issue 1.2

Release information

Prepared by:	L.B.M. Rotthier	2020-10-15
Checked by:	V. Timofeyev	2020-10-15
Approved by:	M. Kidron	2020-12-17

Distribution List:	External

Change log

Date	Issue	Modified by	Section / Pages Affected	Reason for Change
2019-01-23	0.1	L.B.M. Rotthier	All	First Draft copied from IMEPS1
2019-12-04	1.0	L.B.M. Rotthier	section 3.4	Changed explanation on safety mode thresholds, added safety mode r/w parameters.
2020-09-03	1.1	L.B.M. Rotthier	All	Updated and finetuned.
2020-10-15	1.2	L.B.M. Rotthier	sections 3.4, 3.5	IVID 7 related changes. See page 18.

Applicable Documents

#	Document ID	Title	Version
----	----	----	---

Reference Documents

#	Document ID	Title	Version
----	----	----	---

Disclaimer

The contents of this document are subject to the relevant provisions in the contract concluded between the parties. ISIS – Innovative Solutions In Space B.V. ("ISIS") shall not be liable, in full or in part, for any damage arising out from the application or use of any product or circuit described herein, in case such application or use are carried out in a manner not in line with the instructions and warranties provided in the User Manual, product information sheets or any other document provided by ISIS upon the delivery of the product ("Documents"). Further, ISIS shall not be liable for any damage caused by any use which exceeds the function(s) of the product or does not conform to such function(s) as described in the Documents. ISIS shall not be liable for any damage arising from a use which is not carried out in a manner conforming to acceptable practices in the aerospace industry.

ISIS warrants that the product is compliant with our internal quality standards and tested to be in working order, as far as these tests may indicate and predict product functionality.

List of Acronyms

BID	Board Identifier
BP	See PBP
BU	See PBU
CC	Command code
CH	Output Bus CHannel
CMD	Command
COTS	Commercial off-the-shelf
CSKB	CubeSatKit Bus
CU	See PCU
DU	See PDU
EPS	Electrical Power System
I^2C	Inter integrated circuit communication bus.
ICD	Interface control document
ICEPS	ISIS Compact EPS
IMEPS	ISIS Modular EPS
IEPS	ISIS EPS, referring to any ISIS EPS related component
PBP	Power Battery Pack
PBU	Power Battery Unit
ISIS	Innovative Solutions In Space BV.
IU	See IPIU
IVID	Interface Version Identifier
MCU	Microcontroller unit
OBC	On-board computer(s)
OBDB	On-board database
PCB	Printed circuit board
PCU	Power Conditioning Unit
PDU	Power Distribution Unit
PIU	Power Integrated Unit
RC	Response code
RSP	Response
SPI	Serial Peripheral Interface
STID	System Type Identifier
TBC	To be confirmed
TBD	To be determined
TBW	To be written
TTC	Tracking, Telemetry & Commanding

Table of contents

Table of Figures

Table of Tables

1 Introduction

The ISIS Modular Electrical Power System (IMEPS) version 2 is the second generation high power Nano- and Micro-satellite power system. The system consists of four types of elements, of which multiple can be combined in various amounts to tailor to the power and mission needs of a space platform. A schematic overview of the system is given in Figure 1-1, where the four main EPS elements are shown: Power Conditioning Unit (PCU), Power Battery Unit (PBU), Power Battery Pack (PBP) and Power Distribution Unit (PDU).

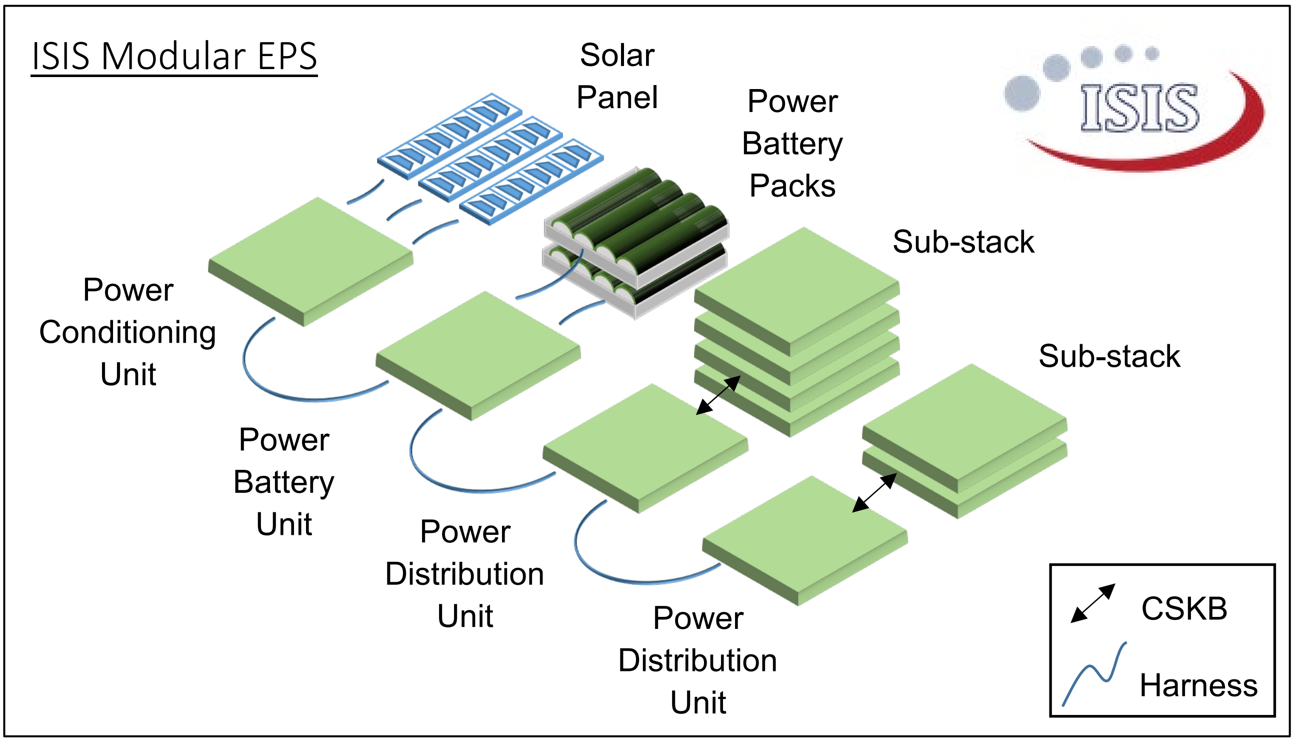


Figure 1-1: IMEPS2 high level overview

The ISIS Compact Electrical Power System (ICEPS) version 2 is the second generation small satellite power system. The system consists of two elements, the Power Integrated Unit (PIU) and the power battery pack (PBP). The PIU board provides PCU, PBU and PDU functionality in a single PC-104 board and optional accompanying daughterboard. It sacrifices some power handling capability and modularity to minimize its volume, and is intended for use in volume constrained platforms. A schematic overview is shown in figure Figure 1-2.

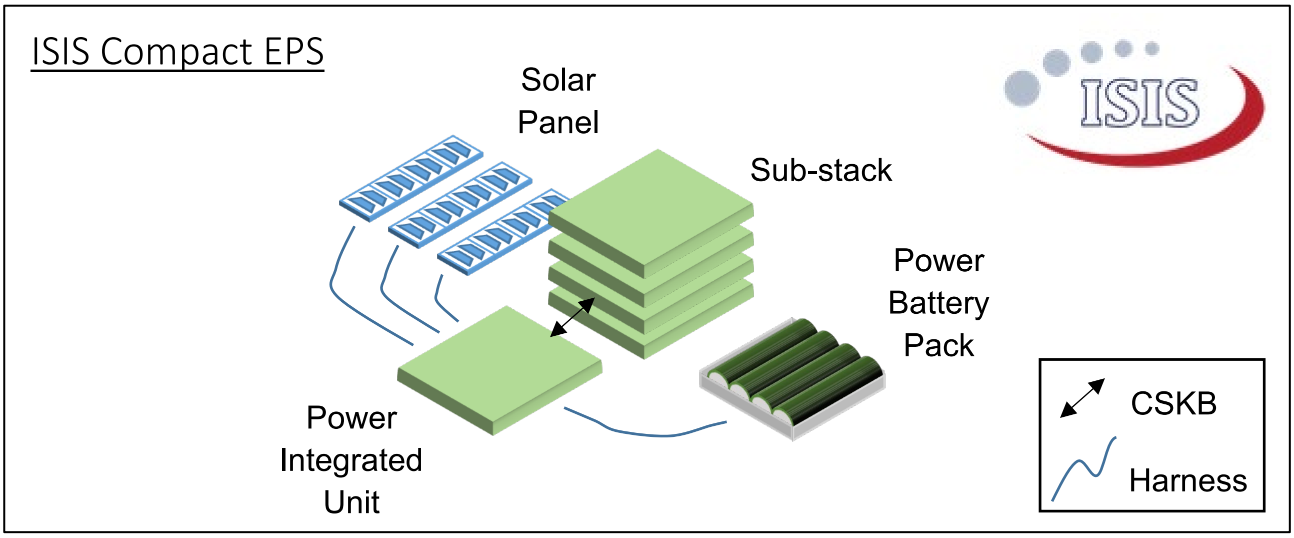


Figure 1-2: ICEPSv2 high level overview

Both the modular and compact EPS support one generic software interface. This provides a platform developer with the ability to choose the EPS that fits within the platform requirements, levering their mutual benefits, without requiring an investment in developing and maintaining two distinct software packages to interface with these systems.

1.1 Purpose and scope of document

This software ICD provides information on the generic digital interface supported by both the ISIS Modular EPS version 2 (IMEPS2) and the ISIS Compact EPS version 2 (ICEPSv2).

This document is valid until it is declared obsolete or replaced with a succeeding version. Changes with respect to the previous version will be clear from the revision. As this document may be updated without prior notice, it is advised to check the ISIS website "http://www.isispace.nl" or ask for the latest version at "support@isispace.nl" before using this document as reference.

2 Functional Division

Both the ICEPSv2 and IMEPS2 can be split out in distinct elements. In case of the modular EPS each element is hosted on a separate PCB, while for the compact EPS the elements are integrated on a single PCB.

The following elements are defined:

- Power Conditioning Unit – takes raw solar power and conditions this to a stable voltage
- Power Battery Unit – allows attaching battery packs to the eps bus
- Power Battery Pack – mechanical battery cell container
- Power Distribution Unit – distributes power to the platform

These elements all provide protection in various ways, and telemetry on their functioning is made available through data interfaces.

2.1 Application in IMEPS2

The functional elements are implemented as distinct PC-104 boards in the IMEPS2. This allows modularly building a platform EPS tailored to different power requirements. This takes some additional space when compared to the ICEPS, but also comes with greater power handling capability, ability to attach more battery packs and flexibility of the modular approach.

2.2 Application in ICEPSv2

The functional elements are integrated in a single PC-104 board for the ICEPSv2. This minimizes the required volume, at the expense of reduced power handling capability and the modularity. Overall the same components and circuits have been employed as its larger modular cousin, the IMEPS2.

The PIU can be considered a condensed IMEPS2, meaning that discussions on a PDU, PBU or PCU board in this manual also applies to the PIU.

2.3 Subsystem Level Redundancy

The IMEPS2 modular approach provides the option to implement subsystem level redundancy.

A fully redundant system is realized by implementing two distinct EPS chains with their own PCU/PBU/PBP and PDU. The PDUs can be electrically OR-ed on the CSKB output, allowing platform subsystems to be powered from both chains simultaneously. When one chain permanently fails, the other chain can still provide power to the same stack.

Other partially or multiple redundant systems can be realized by mixing different amounts of boards of each type on a platform.

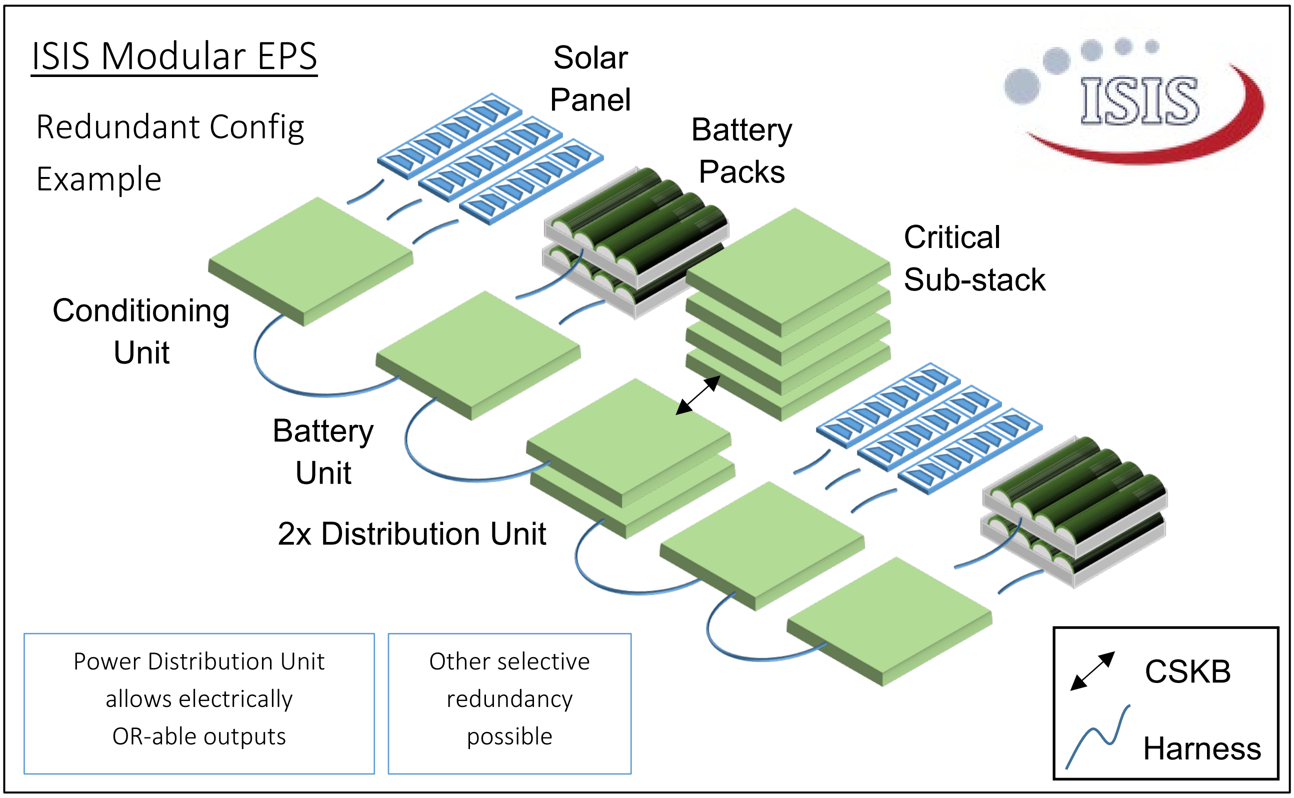


Figure 2-1: IMEPS2 redundant configuration example. Two independent chains connect to each other at the PDU CSKB (i.e. output) in an electrical-OR configuration, to provide a fully redundant EPS.

3 Functional Description

This chapter describes the functionality of the EPS software. See the relevant user manual and hardware ICD for more details on the specific functions provided by each EPS element.

3.1 Operational Modes

The system has three main operational modes described in Table 3-1.

Table 3-1: System Operational modes

Op. Mode	Core Function	Comments
off	system is unpowered	
startup	initializes the system	system will not be responsive on external communication bus during this mode. System automatically switches modes when startup has completed.
nominal	provide full functionality	nominal mode allows commanding and full EPS functionality
safety	maintain power supply to critical systems	system automatically enters and exits this mode depending on the measured rail voltage. Only force-enable output bus channels remain powered. All output bus channel control commands are rejected in this mode.
emergency low power	protect the batteries from excessive depletion and degradation/failure by turning off all output bus channels	system automatically enters and exits this mode depending on the measured rail voltage. All output bus channels are turned off.

The mode transition diagram is given in Figure 3-1 with trigger information in Table 3-2.

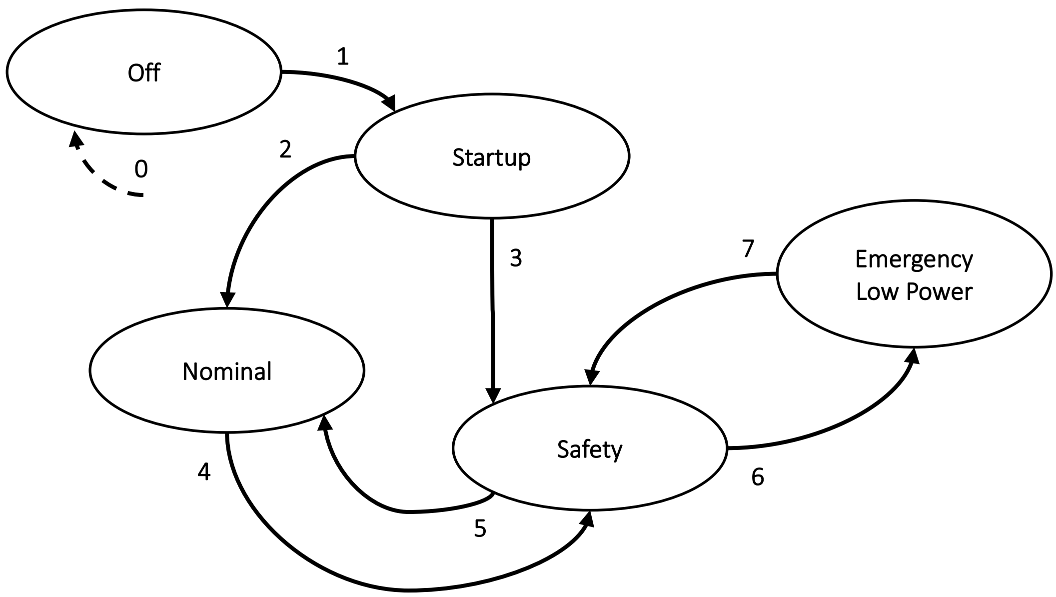


Figure 3-1: Mode Transition Diagram

Table 3-2: System Mode Transitions

Transition	From	To	Trigger
0	any	off	rail voltage falls below minimal supply voltage or upon reset of the micro-controller
1	off	startup	rail voltage exceeds minimal supply voltage
2	startup	nominal	end of startup & rail voltage above safety low threshold
3	startup	safety	end of startup & rail voltage below safety low threshold
4	nominal	safety	rail voltage falls below safety low threshold
5	safety	nominal	rail voltage rises above safety high threshold
6	safety	emlopo	rail voltage falls below emlopo low threshold
7	emlopo	safety	rail voltage rises above emlopo high threshold

3.1.1 Startup Mode

During startup mode the system will (re-)initialize all its internal components and loads the configuration parameters from non-volatile memory. In addition the reset counters are loaded/updated & saved during this mode. The system cannot be commanded during this mode. This mode lasts 500 ms and will then switch out of startup mode.

Startup mode is normally not observed by the parent system, because TT&C command handling is not performed.

3.1.2 Nominal Mode

During nominal mode all functions of the EPS are available. The parent system can command output bus channels on/off, and poll telemetry to asses platform health status.

3.1.3 Safety Mode

Safety mode is entered once the system has detected that the rail voltage has dropped below a low level. This indicates the battery cells are becoming depleted. To safeguard continued operations of critical systems attached to the force-enable output bus channels, all command enable output bus channels are disabled.

3.1.4 Emergency-Low-Power Mode

Emergency low power (emlopo) mode is entered once the system has detected that the rail voltage has dropped to a dangerously low value. This low value is an indication that the battery cells are severely depleted, and further drain could cause rapid degradation and/or permanent failure of the cells.

The system reacts by switching off all output power bus channels to minimize further power consumption. During this state of very low power consumption, the platform will still be able to harvest power from the solar panels and charge the battery. Once the rail voltage has risen sufficiently the system will exit emergency low power mode.

From platform perspective, the transition to and from emergency low power mode will resemble a power cycle. Therefore entry into emlopo and exiting again will count as a power reset, and the corresponding reset cause will indicate emlopo.

Emlopo provides a safety mechanism

3.1.5 Safety Linger & Emergency-Low-Power Period

Automatic switching into and out of safety and emlopo mode are subject to two timed safety mechanisms: *Safety linger* and the *emlopo period*.

Safety linger ensures safety mode is not automatically exited by the EPS until at least the linger time period has passed. The time starts counting when safety mode is entered. The safety linger period is a configuration parameter which is described in Section 3.5. Linger allows a small period of time for the EPS master (e.g. CDHS) to act. For example to safe state information and close down files before emergency-low-power mode is entered and all subsystems (including e.g. the CDHS) are powered down. Note that if the rail voltage rises above the safety high threshold the EPS will only switch to nominal mode after the at least the linger time period has completed (i.e. (now – safety_mode_start) > linger time).

The emlopo period defines the maximum time that the emlopo mode can be active. The time starts counting when emlopo mode is entered. The emlopo period is a configuration parameter which is described in Section 3.5. After the time period is spent, the EPS will automatically switch back into safety mode regardless of the rail voltage. Note that if the rail voltage rises above the emlopo high threshold the EPS will immediately switch to safety mode.

The combined effect of the safety linger and emlopo period will allow a platform that has entered into emlopo, to periodically (every emlopo period) power up for a limited amount of time (the safety linger time). Safety mode enables the force-enable (FE) output bus channels, and in doing so powers up the subsystems attached to them. The subsystems attached to the FE channels normally consist of a minimal set of critical subsystems, amongst which almost always the EPS master (e.g. the CDHS). When activated, the CDHS could be used to transmit satellite status and potentially perform fault detection, isolation and recovery (FDIR) steps. These steps could for example include re-configuration of EPS parameters to extend safety linger and/or change voltage thresholds for automatic mode switching. When the safety linger time is spent and the rail voltage is still below the emlopo high voltage, the EPS switches back into emlopo. Upon entering emlopo a new emlopo period is started.

3.2 TTC Watchdog

The system is equipped with watchdog functionality to safeguard the system from uncontrollability in the case of communication loss with its master. There are two mechanisms implemented, a full reset and a peripheral reset, which are described below.

> **Note:** To keep the watchdog mechanism from triggering, frequent enough communication needs to be performed with the system. Alternatively the watchdog can be disabled through configuration parameters.
>
> To maximize recovery chances without causing a watchdog trigger it is recommended that the frequency of (regular) communication is kept at least four times larger than the frequency of the watchdog timeout.
>
> It is highly recommended to keep the watchdog enabled during in-orbit operation of the mission.

3.2.1 TTC Watchdog Reset

The tracking, telemetry and commanding (TTC) watchdog is implemented as an inactivity timer that is reset when commands are received over the command interface. When the timer reaches a hard coded pre-set time threshold (i.e. a timeout occurs), the system is reset.

The timeout value is provided as a read/write parameter and can be updated or retrieved over the command interface. See Section 3.5 Digital Configuration Registers for more information.

3.2.2 TTC Peripheral Reset

An additional mechanism is in place that tries to restore communication with the master. This mechanism is triggered when the watchdog timer reaches 0.65 x watchdog timeout. When triggered only the communication peripheral internal to the system microcontroller is reset.

The peripheral reset will:

- cause unavailability of the subsystem during the peripheral reset (in the order of microseconds)
- cause loss of any ongoing transaction that started microseconds before the moment of reset
- not cause any change on the activity performed by the system

If the MCU peripheral caused the communication failure, communication will be restored by the peripheral reset. Reception of the next command will reset the watchdog timer, resulting in a more graceful recovery of the system compared to performing a full reset. To allow for graceful recovery the commanding frequency should be at least 4 times the frequency of the watchdog timeout to allow a command to fall between the peripheral reset and the system reset.

The reset timing is schematically represented in Figure 3-2, where the last transfer causes a reset of the watchdog timer, in turn circumventing the need for a system reset.

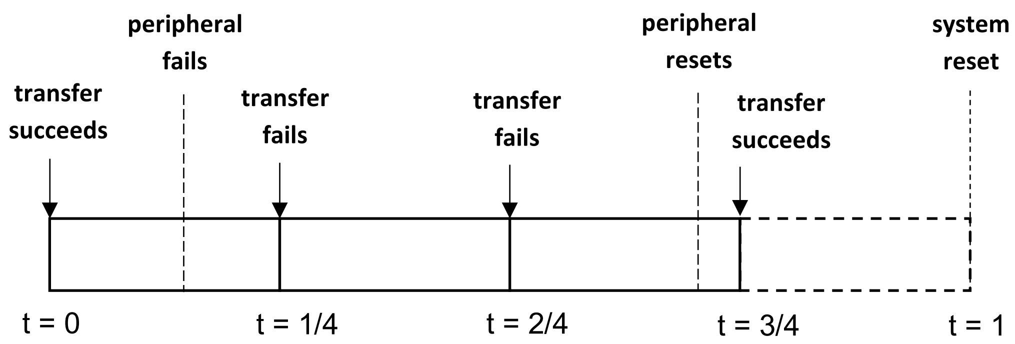


Figure 3-2: Peripheral Reset and Commanding Frequency

3.3 Digital Interface

All EPS boards (IMEPS2: PDU, PBU, PCU and ICEPSv2: PIU) provide a digital communication interface in the flavors I^{2}C and UART. In addition the IMEPS2 PDU has an additional UART/SPI interface and an independent secondary I^{2}C interface.

3.3.1 Data Transfer

There are two types of transfers available between the system and the parent system: a *command* is a directive that instructs the system to perform an action, such as switching modes or to prepare data for subsequent retrieval. A *response* contains data provided by the system to the parent system, such as status and measurement information.

The system processes any received commands only at set time intervals, independent of when the commands were provided. Only after processing has taken place will a response be made available.

The system processes commands at fixed internal time intervals of 10 milliseconds. The duration until a response is available depends on when the command was provided with respect to this processing step, and could therefore differ between 1 ms and 11 ms.

3.3.2 I^{2}C Data Transfer

The writes and reads can be in two separate I^{2}C transactions, or within a single transaction using a repeated start. All bytes within a single write transfer belong to a single command, similarly all bytes during a single read transfer belong to the same response. Hence to issue multiple commands each write transfer must be interleaved by an I^{2}C restart or an I^{2}C stop and I^{2}C start. Any additional commands provided in a single write transfer are discarded. The accepted transfer styles are illustrated in Figure 3-3, where the grey blocks represent I^{2}C protocol signals and blue blocks transferred bytes.

Figure 3-3: Issuing commands over I^{2}C. Always one command per I^{2}C start.

Note: The system will process command data at set intervals and generate a response. Reading from the system before a response is ready will generate 0xFF bytes.

Each new read transfer starts the read from the first byte of a response. A response can be read multiple times, and remains available until overwritten by a new response. Due to internal buffering reading a partially updated response (termed 'dirty read') is not possible. Each command is interpreted after the write transaction completes, and triggers the system to prepare response data. See Figure 3-4 for a symbolic overview of the behaviour.

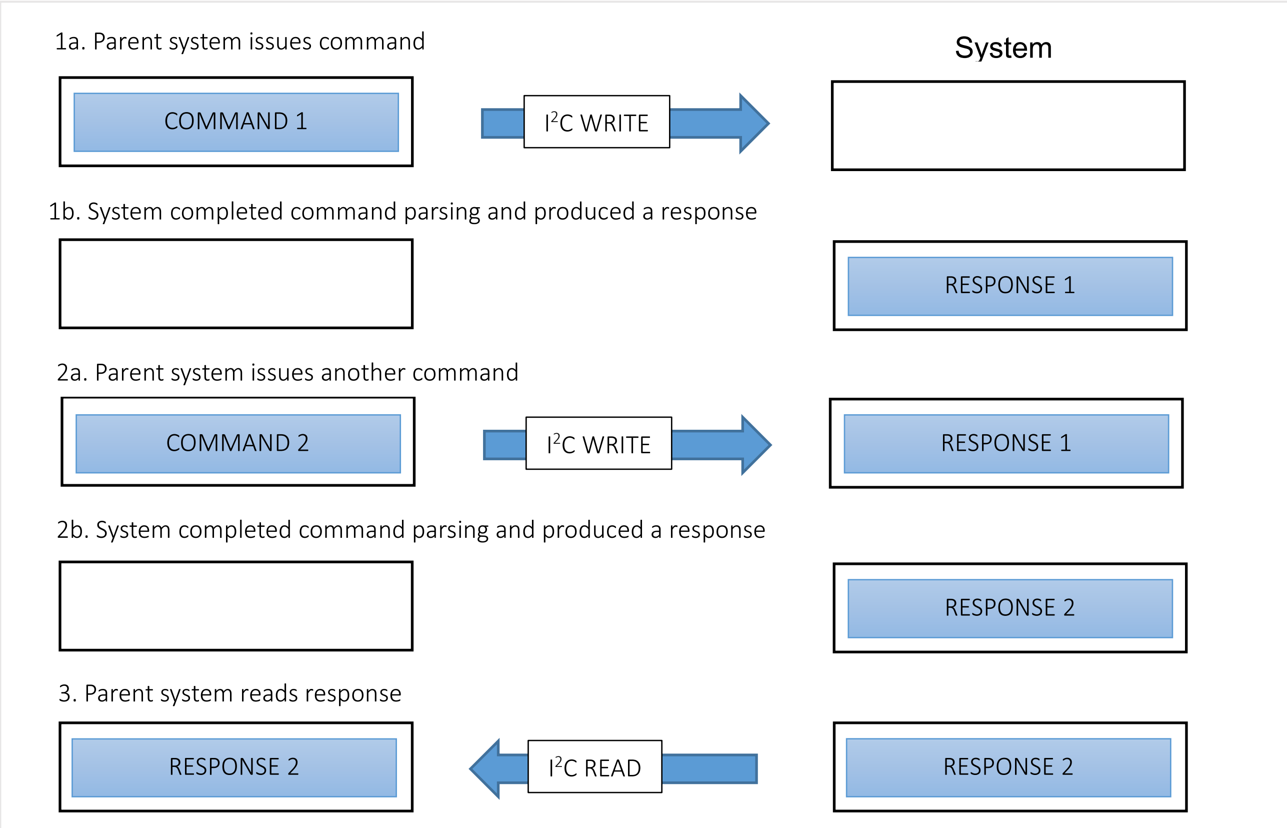


Figure 3-4: Command and response behavior.

Note: Each new command causes the previous response, which contains information on command acceptance, to be overwritten. Hence any intermediate command response is lost. It is therefore advised to perform a read transfer after each write transfer to check whether the command was accepted for processing.

During master WRITE operations the system will not perform I^{2}C NACK when the master provides more bytes than the system requires, instead any additional bytes will simply be ignored. During master READ operations that pass beyond the available response bytes a value of 0xFF (decimal: 255) will be generated. If system does not have a response available at the time of reading, the amount of available response bytes is 0, hence 0xFF bytes will be returned. Since 0xFF is not a valid CC or RC, a returned 0xFF as the first byte uniquely identifies this situation.

This mechanism therefore allows for reading a fixed amount of bytes regardless of the amount of bytes returned for a specific command. The amount of valid bytes should be inferred from the command code returned as the first byte of a read transfer.

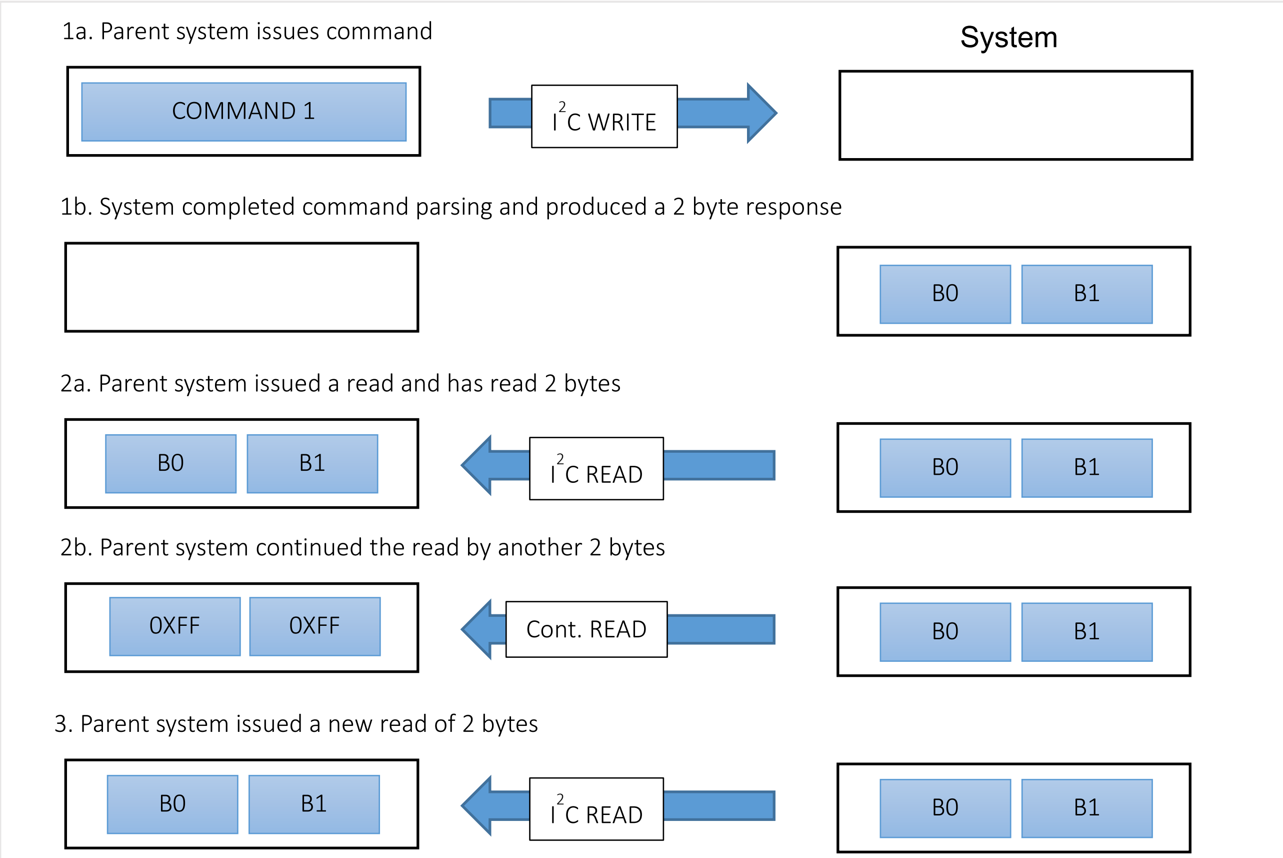


Figure 3-5: Continuous read behavior. 0xFF is returned when reading past available response bytes. A new read starts reading from the first response byte.

3.3.3 I²C Response Timing – Up to 20 milliseconds

Due to the I²C master-slave organisation the parent system will need to initiate a read of the response. Two retrieval schemes are suggested to allow for proper data exchange.

Scheme 1: fixed time wait. Read the response 20 ms after issuing the command. The response will always be available after this time period. It is advised to still check the CC/RC value for 0xFF, which indicates no response is available (yet). In this case another read could be performed after another 10 ms up to a fixed maximum amount of retries. It is up to the user application to decide what the best course of action is if no response becomes available.

Scheme 2: polling. Repeatedly read 1 byte after issuing the command, up to a predefined maximum amount. This maximum must be such that it at least allows for 30 ms to pass. If another value than 0xFF is returned, the response is available and can be read in its entirety. Preferably a pause time is added in between subsequent polls of e.g. 5 ms.

3.3.4 I²C Address

By default the system uses the 7-bit base addresses that are tied to the board ID in a system. An overview of the default addresses is given in Table 3-3. However, because the address is software-defined, the user can request other valid 7- or 10-bit address when the order is placed.

Table 3-3: Default I²C Address Allocation

System Type	Board ID	I2C Address
IPDU (0x11)	1	0x20
	2	0x21
	3	0x22
	(...)	(...)
	8	0x27
IPBU (0x12)	1	0x28
	2	0x29
	3	0x2A
	4	0x2B
IPCU (0x13)	1	0x2C
	2	0x2D
	3	0x2E
	4	0x2F
IPIU (0x1A)	1	0x20

3.3.5 UART Data Transfer

The UART interface uses marker byte sequences to mark the beginning and end of a command. Similarly the responses are packaged within byte sequences as well. An open or close byte sequence will be referred to as a *tag*. The overall structure is shown in Figure 3-6.

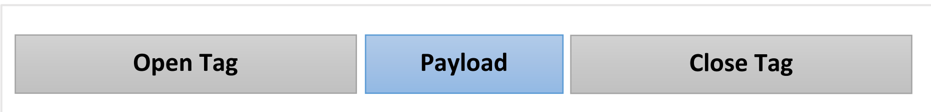


Figure 3-6: UART data framing, with open and close tags and the intermediate command/response data payload.

The payload consists of the command or response byte sequence as described in Section 3.4 on page 14. The open and close tags are modelled after XML tags and are described in Table 3-4.

Table 3-4: UART Framing Tags

Tag	Stream [hex]	Stream [ascii]
command open	3C 63 6D 64 3E	"<cmd>"
command close	3C 2F 63 6D 64 3E	"</cmd>"
response open	3C 72 73 70 3E	"<rsp>"
response close	3C 2F 72 73 70 3E 0D 0A	"</rsp>" <CR> <LF>

The command close tag can be followed by other bytes which will be ignored (e.g. a carriage return linefeed pair could be added at the end without impact to the system). The response close tag includes a carriage return and line feed pair. This is provided in support of end-of-line triggered mechanisms at the host side. Immediately after the system has processed the command, the response will be send back to the parent system.

Note: occurrence of the tag within the payload is not escaped. In very rare cases this might lead to occurrence of the tag within the data payload due to the data itself matching the tag sequence. If this is a concern, response length matching can be performed to verify complete response reception.

Example:
the NOOP command over UART looks like [hex]

3C 63 6D 64 3E <STID> <IVID> 0x02 <BID> 3C 2F 63 6D 64 3E

the system responds with

3C 72 73 70 3E <STID> <IVID> 0x03 <BID> 3C 2F 72 73 70 3E 0D 0A

where <STID>, <IVID> and <BID> are placeholders for the System Type Identifier (STID), Interface Version Identifier (IVID) and Board Identifier (BID) bytes respectively.

UART Configuration Options

The UART interface supports multiple configurations. The configuration can be set by sending a particular payload between command open and close tags. The options are shown in Table 3-5.

Table 3-5: UART Configurations

Configuration	Stream [hex]	Stream [ascii]
RAW	3C 63 66 67 3A 72 61 77 2F 3E	"<cfg:raw/>"
ASCII	3C 63 66 67 3A 61 73 63 69 69 2F 3E	"<cfg:ascii/>"

RAW is the default configuration for UART. The EPS expects raw bytes as command payload. The payload is interpreted per the command specification in this ICD.

ASCII configuration allows sending strings of ASCII text as payload that the EPS will interpret as hexadecimal numbers. The format of the ASCII strings is a space delimited set hex values, where each byte is written as two ASCII characters in the range 0 through F. Once the ASCII payload is received by the EPS, the payload is converted to a raw byte format and processed. The system responds with a payload in the ASCII format, encapsulated within response tags.

Example:
set UART into ASCII mode using

"<cmd><cfg:ascii/></cmd>"

the system responds by echoing the config string in response tags.

Send the NOOP command in ASCII to a PDU (STID=0x11) with interface version (IVID) 0x06 and board-ID (BID) 0x01

"<cmd>11 06 02 01</cmd>"

the system responds with a command accepted status and the NEW flag set (STAT = 80)

"<rsp>11 06 03 01 80</rsp>"

3.3.6 UART Response Timing – Up to 11 milliseconds

UART commands are delimited by command tags. This allows for a transaction scheme in command processing. Only when a full command close tag has been received will the command be queued for processing. Once the command is processed a response is send over the TX channel.

3.3.7 Endianness

All the parameters exchanged between the master and the system are little endian. This means each multi-byte value will start with the lowest significant byte and move up to the most significant byte. If the endianness of the parent system is big endian, the bytes of each multi-byte value should be stored in the opposite order of how they were received to have the parent system properly interpret the values.

Alternatively, bit shifting each individual byte before OR-combining them into a number allows proper value formation regardless of underlying parent system endianness. This is because the bit shift operation is independent of implementation details, such as the endianness of stored multibyte values. After formation of the number, the parent system itself will store the value according to the endianness it uses, when the number is assigned to a variable.

3.4 Digital Command Messages

A command message is formed by a sequence of command bytes send to the system. The makeup of the sequence of bytes is described for each command. The command specification does not include any bytes required by the protocol used for transferring the command bytes to the system (e.g. the I2C address in case of using the I2C protocol initiates the transfer and is therefore not part of the command byte sequence). The command specification is described as the 'Command'. The system responds by returning a sequence of bytes to the commanding system. This is described as the 'Reply'.

Both the command and the reply always start with a set of standard fields as described in Table 3-6 and Table 3-7 respectively.

Table 3-6: Standard Command Fields

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code (CC) identifying a command. Each command description states its command code at the start of its header. A command is always an even number.
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)

Table 3-7: Standard Reply Fields

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code (RC) identifying a response. A response code is a command code OR combined with 0x01. A response is always an uneven number.
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Following the standard command or response bytes a variable amount of bytes might follow depending on whether command parameters or additional reply data fields are returned for the specific command.

Note: Response bytes beyond the standard fields are only included when the STAT CMDERR indicates the command has been accepted. The system allows for an infinite amount of bytes to be read, however the data block will contain only 0xFF as reply parameters in the case of a rejected command.

All available commands and some general information on fields of these commands are listed in Table 3-8.

Apart from commands the system support a set of configuration parameters. Section 3.5 "Digital Configuration Registers" (page 83) provides information on the configuration parameters.

Table 3-8: Command Overview

Description	Page	Comments
General Information		
`System Type Identifier (STID)`	17	
`Interface Version Identifier (IVID)`	18	
`Board Identifier (BID)`	20	
`Response Status Information (STAT)`	21	
`Raw Data Conversion Information`	22	
`Complex Datatype: VIPD`	24	
`Complex Datatype: BPD`	25	for PBU/PIU
`Complex Datatype: CCD`	27	for PCU/PIU
`Complex Datatype: CCSD`	28	for PIU
Operational Commands		
`0xAA (0xAB) – System Reset`	29	
`0x02 (0x03) – No-operation`	30	
`0x04 (0x05) – Cancel Operation`	31	
`0x06 (0x07) – Watchdog`	32	
`0x10 (0x11) – Output Bus Group On`	33	for PDU/PIU
`0x12 (0x13) – Output Bus Group Off`	34	for PDU/PIU
`0x14 (0x15) – Output Bus Group State`	35	for PDU/PIU
`0x16 (0x17) – Output Bus Channel On`	36	for PDU/PIU
`0x18 (0x19) – Output Bus Channel Off`	37	for PDU/PIU
Mode Switch Commands		
`0x30 (0x31) – Switch to Nominal Mode`	38	for PDU/PIU
`0x32 (0x33) – Switch to Safety Mode`	39	for PDU/PIU
Data Request Commands		
`0x40 (0x41) – Get System Status`	40	
`0x42 (0x43) – Get PDU/PIU Overcurrent Fault State`	43	for PDU/PIU
`0x44 (0x45) – Get PBU ABF Placed State`	46	for PBU/PIU
`0x50 (0x51) – Get PDU Housekeeping Data (RAW)`	47	for PDU/PIU
`0x52 (0x53) – Get PDU Housekeeping Data (ENG)`	51	for PDU/PIU

System Type Identifier (STID)

(↑ Command Overview ↑)

The system type identifier identifies the type of a system. It differentiates this system from other system types that might be connected to a platform bus. The purpose is to ensure commands that are not fit/intended for this type of system will not be processed.

The following STID is used with the commands specified in this document:

STID = 0x11 (PDU)

STID = 0x12 (PBU)

STID = 0x13 (PCU)

STID = 0x1A (PIU)

Note: Any command with another STID (and not 0x00) will be rejected.

Note: The STID check can be bypassed by setting STID = 0x00.

The system will always provide its actual STID in its response, even when an unspecified system type was used in the command. This allows for bus scan type operations that dynamically enumerate the systems available in a given bus.

The available system types are described in Table 3-9, although this list is likely not exhaustive.

Table 3-9: System Types

System Type Identifier (STID)	Description
0x00	unspecified system type (bypass system type check)
0x1n	Electrical Power System Group
0x11	IMEPS2 Power Distribution Unit (PDU)
0x12	IMEPS2 Power Battery Unit (PBU)
0x13	IMEPS2 Power Conditioning Unit (PCU)
0x14 – 0x19	(reserved)
0x1A	ICEPSv2 Power Integrated Unit (PIU)
0x1B – 0x1F	(reserved)
0x2n	Attitude and Orbit Control System Group
0x21	ISIS Magnetorquer (IMTQv3)
0x22 – 0x2F	(reserved)
0x3n	RF Communication Group
0x31 – 0x3F	(reserved)
0x4n	Deployable/Actuator Group
0x41	ISIS Motor Drive Unit (IMDU)
0x42 – 0x4F	(reserved)
0x5n – 0xDn	(reserved)
0xEn	Non-ISIS System Group
0xE1 – 0xEF	third party system 1 through 15
0xFn	(reserved)

Interface Version Identifier (IVID)

(↑ Command Overview ↑)

The interface version identifier identifies the version of the software interface that is running on an EPS board. The IVID mentioned in this document identifies the interface that is described within this document.

The version number will be incremented when an interface change is made, resulting in a new unique IVID and new software ICD document. The changes made with each version increment are described in Table 3-10.

The following IVID is used for the interface specified in this document:

IVID = 0x07

The IVID is part of all commands and responses. As such it clearly describes what format the subsequent bytes of the command adhere to. This ensures that the EPS can properly decode the command and form an appropriate response according to the relevant interface version.

The IVID mechanism allows the EPS software to maintain backwards compatibility of interface versions. Over time the EPS software will evolve independent from software that interfaces with the EPS (e.g. CDHS, scripted clients etc). These interfacing systems would lag behind in terms of software changes, and if not for the IVID system, would need to be upgraded the moment a new EPS with a newer IVID becomes available. However, with the IVID mechanism, the EPS software will continue to properly decode commands and return appropriately formed responses to not only the newest interface it supports but also of older interface versions. The upgrade path of software interfacing with the EPS system becomes more relaxed.

It is still recommended to upgrade the master software to the latest IVID as soon as possible. This allows taking advantage of the latest improvements and limits the possibility that the IVID used by the old client code was dropped from the EPS software.

Caution: The amount of older IVIDs that are supported is not fixed. The intend is to at least support the previous IVID, but this is not guaranteed.

Caution: Even though the older IVID command and response interpretation would remain syntactically correct, the newer software version could have different EPS behaviour. It is recommended to acceptance test each new EPS that is used to ensure proper compatibility with software.

Determining IVID

The latest IVID that the EPS supports can be requested by using the IVID wildcard. When IVID = 0x00 the system will default to the newest interface it supports, and the response message will indicate the newest supported IVID of the EPS.

Note: It is recommended to use the “0x02 (0x03) – No-operation” command when using the IVID wildcard, to ensure no unexpected actions will be taken by the EPS when determining the IVID of the system. Alternatively the configuration parameter containing the IVID could be retrieved. See Section 3.5 for more information on configuration parameters.

IVID Change Log

The log in Table 3-10 describes the changes that were made when moving to a new interface version. The BWC column indicates whether an interface version is backwards compatible with the version before it.

Caution: Even though effort is made to indicate all changes in the table, it is still possible some changes were left out. It is therefore important to check the complete command set to verify the implemented software matches with the interface specification in this document.

Table 3-10: Interface Version Change Log

IVID	Description
0x01	First release of IMEPS1 interface (with IVID support)
0x02	First release of IMEPS2 interface.
0x03	Numerous changes made while maturing design.
0x04	Changed MODE field interpretation in “0x40 get system status” message.

IVID	Description
0x05	Changed BPD type and field count, impacting PBU HK TLM field sizing and offsets. Flipped CU VIP field directionality from input to output, same in CCD.
0x06	Added: "0x44 (0x45) – Get PBU ABF Placed State"
0x07	Added to "0x40 (0x41) – Get System Status": unix time and date-time fields. Added: "**Error! Reference source not found.**". Added: "0xC4 (0xC5) – Correct Time". Added: "0xC6 (0xC7) – Zero Reset Cause Counters". Added: output bus channel 16 through 31 to PDU/PIU channel control commands. Changed: Included output bus channel 16 through 31 support to PDU/PIU HK TLM. Added: Configuration parameters related to CH_EXT channels.

(↑ Command Overview ↑)

Board Identifier (BID)

The board identifier is used to identify a particular board within a multi-board system, such as the IMEPS, which will likely contain multiple boards of the same system type. The board identifier is a sequentially numbered value starting from 1.

The BID allows for positive identification of the origin of the received response (measurement) data.

Commands with a BID that does not match the BID of a recipient will be rejected. The BID check can be bypassed by setting BID = 0x00, in which case the command will be accepted regardless of the recipient BID.

Example:
An IMEPS consists of 4 x PCU, 3 x PBU and 2 x PDU. The following bids are defined for the system:

- PCU BIDs: 1, 2, 3, 4
- PBU BIDs: 1, 2, 3
- PDU BIDs: 1, 2

The PCU2 responses will indicate 2 as BID, and will reject all commands with BIDs other than 2 or 0.

(↑ Command Overview ↑)

Response Status Information (STAT)

The STAT response byte reflects the response status to the preceding command. It indicates whether the response is returned for the first time (a new response) or whether it is a re-read of a previously read response. In addition it provides an error code indicating acceptance or rejection of the previous command and the grounds upon which the decision was based. The composition of the STAT bitflag field is described in Table 3-11.

Table 3-11: STAT Bitflag Field Composition

Name	Bit	Description
CMDERR	0x00	accepted
	0x01	rejected
	0x02	rejected: invalid command code
	0x03	rejected: parameter missing
	0x04	rejected: parameter invalid
	0x05	rejected: unavailable in current mode/configuration
	0x06	rejected: invalid system type, interface version or bid
	0x07	internal error occurred during processing
(reserved)	0x10	(reserved)
(reserved)	0x20	(reserved)
(reserved)	0x40	(reserved)
NEW	0x80	set when the response is read for the first time. Not set otherwise

.

(↑ Command Overview ↑)

Raw Data Conversion Information

The raw data can be converted to engineering values using a bias and a gain value.

The full computation has the following form:

$$Eng = G \times (Raw + Bias) = \frac{PreMul}{PostDiv} \times (Raw + Bias)$$

To retain precision while applying integer arithmetic, the conversion is actually implemented by upcasting to a larger variable, then applying the multiplication before the division and finally downcasting to the original variable size:

$$Eng = \frac{PreMul \times (Raw + Bias)}{PostDiv}$$

The bias, pre-multiplier and post-divider conversion parameters are given in Table 3-12.

Table 3-12: Raw to Engineering Value Conversion Formulas

Field	Bias	PreMul	PosDiv	Unit	Notes
VOLT_BRDSUP	0	1000	819	1e-3 V	
(PDU) TEMP	- 1168	220	9	1e-2 °C	PDU uses MCU factory provided calibration correction[1] which could cause slight deviaton in engineering hk data using the values provided in this document.
(PIU) TEMP	- 868	26400	819	1e-2 °C	PIU uses production calibration correction[2] on the results of this conversion. This can cause deviation of the engineering hk result wrt the result from this conversion alone.
VIP VOLT	0	125	128	1e-3 V	
VIP CURR	0	3125	10240 20480	1e-3 A	= VIP_INPUT/OUTPUT & VIP_VDx = VIP_CH
VIP POWE	0	3125	3200 6400	1e-2 W	= VIP_INPUT/OUTPUT & VIP_VDx = VIP_CH
VOLT_CELLx	- 512	3	2	1e-3 V	
BAT_TEMP1	- 1969	75	4	1e-2 °C	
BAT_TEMP2, BAT_TEMP3					Non-linear. See Figure 3-7 and/or Table 3-13.
VOLT_IN_MPPT, VOLT_OUT_MPPT	0	2625	128	1e-3 V	

[1] See Section 3.5 on page 85 for the factory calibration parameters ADC_MCU_TEMP_V25T30 and ADC_MCU_TEMP_V25T85. To convert using the calibration data use:
TEMP = 30 + 55 x (RAW - V25T30) / (V25T85 – V25T30)
This is equivalent to:
Bias = V25T30 - 30 x (V25T85 – V25T30) / 55, Premul = 55, PosDiv = V25T85 – V25T30

[2] See Section 3.5 on page 85 for the production calibration parameters MCU_TEMP_BIAS, MCU_TEMP_PREMUL and MCU_TEMP_POSDIV. The production calibration data is applied on the uncorrected engineering TEMP:
$TEMP_{ENG\_CORR}$ = MCU_TEMP_PREMUL x ($TEMP_{ENG}$ + MCU_TEMP_BIAS) / MCU_TMP_POSTDIV

Field	Bias	PreMul	PosDiv	Unit	Notes
CURR_IN_MPPT, CURR_OUT_MPPT	-41	625	192	1e-3 A	

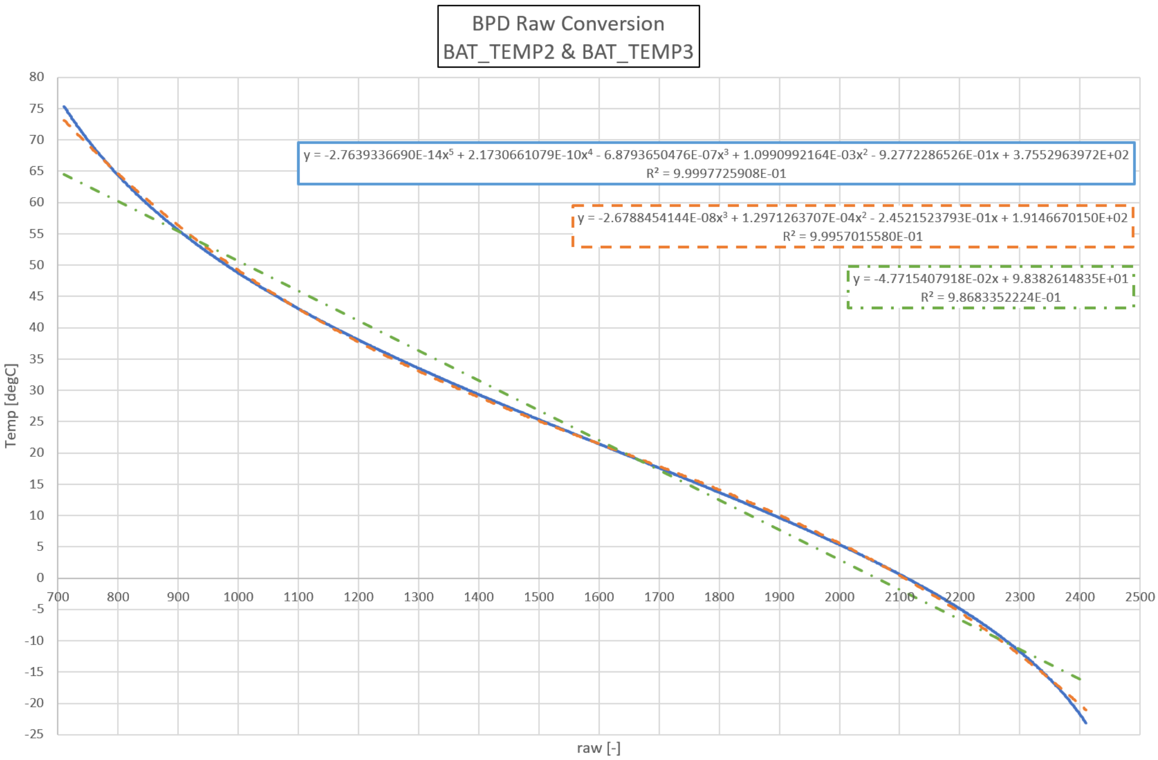


Figure 3-7: BPD raw data to temperature relation for BAT_TEMP2 and BAT_TEMP3

The non-linear relation between the raw and temperature of the NTC battery pack temperature sensors is shown in Figure 3-7 and Table 3-13. The blue solid line shows the 5th degree poly that resembles the temperature relation. Several polynomial fits are applied to this curve of which the corresponding equations and R^2 values are shown. These fits allow selection of a desired accuracy versus computational performance. Alternatively a static look-up table can be constructed using the 5th degree poly, which minimizes computational burden while maximizing accuracy, at the expense of digital storage space for the look-up table[3].

Table 3-13: Polynomial Fit to BPD: BAT_TEMP2 and BAT_TEMP3

Degree	Polynomial
Fifth	$y = -2.7639336690E-14x^5 + 2.1730661079E-10x^4 - 6.8793650476E-07x^3 + 1.0990992164E-03x^2 - 9.2772286526E-01x + 3.7552963972E+02$
Third	$y = -2.6788454144E-08x^3 + 1.2971263707E-04x^2 - 2.4521523793E-01x + 1.9146670150E+02$
Linear	$y = -4.7715407918E-02x + 9.8382614835E+01$

[3] For PIU use: $y = -3.34855677E-9x^3 + 2.214139287620E-5x^2 – 0.066728398x + 96.32440376$

Complex Datatype: VIPD

(↑ Command Overview ↑)

The voltage V - current I - power P datatype (VIPD) is a triplet of current, voltage and power values. These three elements are used in blocks across the HK telemetry.

See "Raw Data Conversion Information" on page 22 for conversion information in case manual conversion from raw to engineering values is preferred.

A subscript suffix *raw* after the name of a VIPD type data block refers to the raw values VIPD as shown in Table 3-14.

Table 3-14: VIPD Composition (raw values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
$VOLT_{raw}$	0	2	int16		Channel voltage expressed in raw form, using twos-complement signed format.
$CURR_{raw}$	2	2	int16		Channel current expressed in raw form, using twos-complement signed format.
$POWE_{raw}$	4	2	int16		Channel power expressed in raw form, using twos-complement signed format.

No subscript suffix after the name of a VIPD type data block refers to the engineering values VIPD as shown in Table 3-15.

Table 3-15: VIPD Composition (engineering values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VOLT	0	2	int16	1e-3 V	Channel voltage. Note: Negative voltage range is limited to -0.2 V due to hardware constraints.
CURR	2	2	int16	1e-3 A	Channel current
POWE	4	2	int16	1e-2 W	Channel power

All three engineering values, for both raw and engineering values, are in twos-complement format. Any multi-byte value is returned in little endian byte order. Sign allows values to indicate directionality for current and power flows, and indicate bias/offset for voltages that are slightly below zero.

Note: due to hardware constraints the negative voltage range extends only up to -0.2 V. Voltages below -0.2 V are outside the safe operating range of the EPS.

Complex Datatype: BPD

(↑ Command Overview ↑)

The battery pack datatype (BPD) contains the data available for each battery pack attached to the PBU. These are used in the PBU HK telemetry.

See "Raw Data Conversion Information" on page 22 for conversion information in case manual conversion from raw to engineering values is preferred.

A subscript suffix *raw* after the name of a BPD type data block refers to the raw values BPD as shown in Table 3-16.

Table 3-16: BPD Composition (raw values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
$\texttt{VIP\_BP\_INPUT}_{raw}$	0	6	VIPD		Input V, I and P data for the battery in raw form. Negative values indicate output flow. See "Complex Datatype: VIPD"
STAT_BP	6	2	uint16		Bitflag field indicating BP board status. See Table 3-18 for definition.
$\texttt{VOLT\_CELL1}_{raw}$	8	2	uint16		Voltage across cell 1 in raw form
$\texttt{VOLT\_CELL2}_{raw}$	10	2	uint16		Voltage across cell 2 in raw form
$\texttt{VOLT\_CELL3}_{raw}$	12	2	uint16		Voltage across cell 3 in raw form
$\texttt{VOLT\_CELL4}_{raw}$	14	2	uint16		Voltage across cell 4 in raw form
$\texttt{BAT\_TEMP1}_{raw}$	16	2	uint16		Internal temperature of battery monitor/balancer IC on the BU. NOTE: Not located on the battery pack.
$\texttt{BAT\_TEMP2}_{raw}$	18	2	uint16		Battery pack temperature in between the center battery cells.
$\texttt{BAT\_TEMP3}_{raw}$	20	2	uint16		Battery pack temperature on the front of the battery pack.

No subscript suffix after the name of a BPD type data block refers to the engineering values BPD as shown in Table 3-17.

Table 3-17: BPD Composition (engineering values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_BP_INPUT	0	6	VIPD		Input V, I and P data for the battery in raw form. Negative values indicate output flow. See "Complex Datatype: VIPD"
STAT_BP	6	2	uint16		Bitflag field indicating BP board status. See Table 3-18 for definition.
VOLT_CELL1	8	2	int16	1e-3 V	Voltage across cell 1
VOLT_CELL2	10	2	int16	1e-3 V	Voltage across cell 2
VOLT_CELL3	12	2	int16	1e-3 V	Voltage across cell 3
VOLT_CELL4	14	2	int16	1e-3 V	Voltage across cell 4

Name	Offset [byte]	Size [byte]	Type	Unit	Description
BAT_TEMP1	16	2	int16	1e-2 °C	Internal temperature of battery monitor/balancer IC on the BU. NOTE: Not located on the battery pack.
BAT_TEMP2	18	2	int16	1e-2 °C	Battery pack temperature in between the center battery cells.
BAT_TEMP3	20	2	int16	1e-2 °C	Battery pack temperature on the front of the battery pack.

All signed engineering values are in two's complement format. Any multi-byte value is returned in little endian byte order.

Table 3-18: Battery Pack Status Bitfield Definition

Bit	Hex	Description
0 (LSB)	0x0001	battery cell 1 under voltage
1	0x0002	battery cell 2 under voltage
2	0x0004	battery cell 3 under voltage
3	0x0008	battery cell 4 under voltage
4	0x0010	battery cell 1 over voltage
5	0x0020	battery cell 2 over voltage
6	0x0040	battery cell 3 over voltage
7	0x0080	battery cell 4 over voltage
8	0x0100	battery cell 1 balancing
9	0x0200	battery cell 2 balancing
10	0x0400	battery cell 3 balancing
11	0x0800	battery cell 4 balancing
12	0x1000	heaters active
13	0x2000	(reserved)
14	0x4000	(reserved)
15 (MSB)	0x8000	battery pack enabled. bitflag is set when the battery is connected to the output bus.

(↑ Command Overview ↑)

Complex Datatype: CCD

The conditioning channel datatype (CCD) contains the data available for each power conditioning chain of the PCU. These are used in the PCU HK telemetry.

See "Raw Data Conversion Information" on page 22 for conversion information in case manual conversion from raw to engineering values is preferred.

A subscript suffix *raw* after the name of a CCD type data block refers to the raw values CCD as shown in Table 3-16.

Table 3-19: CCD Composition (raw values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
$VIP\_CC\_OUTPUT_{raw}$	0	6	VIPD		Output V, I and P data for the conditioning channel in raw form. Negative values indicate input flow. See "Complex Datatype: VIPD"
$VOLT\_IN\_MPPT_{raw}$	6	2	uint16		MPPT input voltage measurement in raw form
$CURR\_IN\_MPPT_{raw}$	8	2	uint16		MPPT input current measurement in raw form
$VOLT\_OU\_MPPT_{raw}$	10	2	uint16		MPPT output voltage measurement in raw form
$CURR\_OU\_MPPT_{raw}$	12	2	uint16		MPPT output current measurement in raw form

No subscript suffix after the name of a CCD type data block refers to the engineering values CCD as shown in Table 3-17.

Table 3-20: CCD Composition (engineering values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CC_OUTPUT	0	6	VIPD		Output V, I and P data for the conditioning channel. Negative values indicate input flow. See "Complex Datatype: VIPD"
VOLT_IN_MPPT	6	2	int16	1e-3 V	MPPT input voltage measurement
CURR_IN_MPPT	8	2	int16	1e-3 A	MPPT input current measurement
VOLT_OU_MPPT	10	2	int16	1e-3 V	MPPT output voltage measurement
CURR_OU_MPPT	12	2	int16	1e-3 A	MPPT output current measurement

(↑ Command Overview ↑)

Complex Datatype: CCSD

The conditioning channel short datatype (CCSD) contains the data available for each power conditioning chain of the PIU. These are used in the PIU HK telemetry. The block is lacking the initial VIP data, but is otherwise the same as the CCD block.

See "Raw Data Conversion Information" on page 22 for conversion information in case manual conversion from raw to engineering values is preferred.

A subscript suffix *raw* after the name of a CCSD type data block refers to the raw values CCSD as shown in Table 3-16.

Table 3-21: CCSD Composition (raw values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
$VOLT\_IN\_MPPT_{raw}$	0	2	uint16		MPPT input voltage measurement in raw form
$CURR\_IN\_MPPT_{raw}$	2	2	uint16		MPPT input current measurement in raw form
$VOLT\_OU\_MPPT_{raw}$	4	2	uint16		MPPT output voltage measurement in raw form
$CURR\_OU\_MPPT_{raw}$	6	2	uint16		MPPT output current measurement in raw form

No subscript suffix after the name of a CCSD type data block refers to the engineering values CCSD as shown in Table 3-17.

Table 3-22: CCD Composition (engineering values)

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VOLT_IN_MPPT	0	2	int16	1e-3 V	MPPT input voltage measurement
CURR_IN_MPPT	2	2	int16	1e-3 A	MPPT input current measurement
VOLT_OU_MPPT	4	2	int16	1e-3 V	MPPT output voltage measurement
CURR_OU_MPPT	6	2	int16	1e-3 A	MPPT output current measurement

0xAA (0xAB) – System Reset

(↑ Command Overview ↑)

Write length: 5 bytes.

Read length: 5 bytes.

Action: perform a software induced reset of the MCU.

Note: a reply to this command will not always be retrievable, given that the system will shut down the command interface while preparing for the hard reset. After reset the reply will not be available anymore, instead having no response available (i.e. returns 0xFF on READ)

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header (without parenthesis)
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
RST_KEY	4	1	uint8	Reset key. The value of this parameter needs to be set to 0xA6 for the command to be accepted. Any other value will cause the command to be rejected.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xAA <BID> 0xA6}

(↑ Command Overview ↑)

0x02 (0x03) – No-operation

Write length: 4 bytes.

Read length: 5 bytes.

Action: performs a no-operation. This is useful to check the availability of the system, without changing anything about the current configuration or operation.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system
{<STID> <IVID> 0x02 <BID>}

The system responds with:
{<STID> <IVID> 0x03 <BID> 0x80}

(↑ Command Overview ↑)

0x04 (0x05) – Cancel Operation

Write length: 4 bytes.

Read length: 5 bytes.

Action: Switches off any command-enable output bus channels that have been switched on after the system powered up.

Only output bus channels that can be commanded off are affected.

All force-enable channels will remain enabled.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8	See “Interface Version Identifier (IVID)” (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See “Board Identifier (BID)” (page 20)

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8	See “Interface Version Identifier (IVID)” (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See “Board Identifier (BID)” (page 20)
STAT	4	1	uint8	See “Response Status Information (STAT)” (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x04 <BID>}

The system will turn off all command-enable output bus channels. All force-enable output bus channels will remain powered.

0x06 (0x07) – Watchdog

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 5 bytes.

Action: Resets the watchdog timer keeping the system from performing a reset.

Note that any traffic with the system implicitly performs a watchdog reset, hence periodic interaction with the system through other commands removes the requirement on sending this particular command.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x06 <BID>}

(↑ Command Overview ↑)

0x10 (0x11) – Output Bus Group On

Write length: 6 or 8 bytes.

Read length: 5 bytes.

Action: Turn-on output bus channels that are marked with a 1-bit, leave bus channels that are not marked unaltered. The least-significant bit corresponds to bus channel 0 (CH0), the next bit corresponds to channel 1 (CH1), etc.

For example, providing the flag field 0b00001010 (=0x0A, decimal 10) turns on bus channels 1 and 3, while leaving the other channels unaltered.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CH_BF	4	2	uint16	Bitflag field indicating which output bus channels to turn-on. These are channels 0 through 15. A 1-bit indicates the corresponding bus channel needs to be turned-on. A 0-bit has no effect on the existing state of the corresponding bus channel. Flags corresponding to force-enable (FE) and already-on channels are ignored.
CH_EXT_BF	6	2	uint16	Optional. Bitflag field indicating which extended output bus channels to turn-on. These are channels 16 through 31. Similar to CH_BF in operation. If not provided the channels will remain unchanged.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x10 <BID> 0x03 0x05}

Output bus group mask provided is 0b1010011 (=0x0503, decimal 83). This switches output bus channels 0, 1, 4 and 6 on. All other channel states remain unaltered.

Note that force-enable (i.e. permanent, always powered, non-switchable) channels are ignored.

0x12 (0x13) – Output Bus Group Off

(↑ Command Overview ↑)

Write length: 6 or 8 bytes.

Read length: 5 bytes.

Action: Turn-off output bus channels that are marked with a 1-bit, leave bus channels that are not marked unaltered. The least-significant bit corresponds to bus channel 0 (CH0), the next bit corresponds to channel 1 (CH1), etc.

For example, providing the flag field 0b00001010 (=0x0A, decimal 10) turns off bus channels 1 and 3, while leaving the other channels unaltered.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CH_BF	4	2	uint16	Bitflag field indicating which output bus channels to turn-off. These are channels 0 through 15. A 1-bit indicates the corresponding bus channel needs to be turned-on. A 0-bit has no effect on the existing state of the corresponding bus channel. Flags corresponding to force-enable (FE) and already-off channels are ignored.
CH_EXT_BF	6	2	uint16	Optional. Bitflag field indicating which extended output bus channels to turn-off. These are channels 16 through 31. Similar to CH_BF in operation. If not provided the channels will remain unchanged.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x12 <BID> 0x03 0x05}

Output bus group mask provided is 0b1010011 (=0x0503, decimal 83). This switches output bus channels 0, 1, 4 and 6 off. All other channel states remain unaltered.

Note that force-enable (i.e. permanent, always powered, non-switchable) channels are ignored and will remain powered.

0x14 (0x15) – Output Bus Group State

(↑ Command Overview ↑)

Write length: 6 or 8 bytes.

Read length: 5 bytes.

Action: Turn-on bus channels that are marked with a 1-bit, turn-off bus channels that are not marked (i.e. 0-bit). The least-significant bit corresponds to bus channel 0 (CH00), the next bit corresponds to channel 1 (CH01), etc.

For example, providing the flag field 0b00001010 (=0x0A, decimal 10) turns on bus channels 1 and 3, while turning off all the other channels.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CH_BF	4	2	uint16	Bitflag field indicating the required output bus channel on-state. These are channels 0 through 15. A 1-bit indicates the corresponding bus channel needs to be on. A 0-bit indicates the corresponding bus channel needs to be off. Flags corresponding to force-enable (FE) channels are ignored. Bus channels already in the required state are ignored.
CH_EXT_BF	6	2	uint16	Optional. Bitflag field indicating the required extended output bus channels on-state. These are channels 16 through 31. Similar to CH_BF in operation. If not provided the channels will remain unchanged.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x14 <BID> 0x03 0x05}

Output bus group mask provided is 0b1010011 (=0x0503, decimal 83). This switches output bus channels 0, 1, 4 and 6 on. All other channels are switched off, except for the CH_EXT_BF channels, which retain their original state.

Note that force-enable (i.e. permanent, always powered, non-switchable) channels are ignored and will remain powered.

(↑ Command Overview ↑)

0x16 (0x17) – Output Bus Channel On

Write length: 5 bytes.

Read length: 5 bytes.

Action: Turn a single output bus channel on using the bus channel index. Index 0 represents channel 0 (CH0), index 1 represents channel 1 (CH1), etc.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CH_IDX	4	1	uint8	Bus channel index of the channel to turn on. An index beyond the software supported range will be rejected. A bus channel that is already on is ignored (this includes FE channels). Note: Channels may or may not be available in hardware, depending on the options chosen.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x16 <BID> 0x02}

Switches output bus channel 2 (CH2) on.

(↑ Command Overview ↑)

0x18 (0x19) – Output Bus Channel Off

Write length: 5 bytes.

Read length: 5 bytes.

Action: Turn a single output bus channel on using the bus channel index. Index 0 represents channel 0 (CH0), index 1 represents channel 1 (CH1), etc.

note: channels configured as force-enable output channels cannot be turned off and commanding this will return an error.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CH_IDX	4	1	uint8	Bus channel index of the channel to turn off. An index beyond the software supported range or corresponding to an FE channel will be rejected. A bus channel that is already in the off state is ignored. Note: Software supported channels may or may not be available in hardware, depending on the options chosen.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x18 <BID> 0x02}

Switches output bus channel 2 (CH2) off.

(↑ Command Overview ↑)

0x30 (0x31) – Switch to Nominal Mode

Write length: 4 bytes.

Read length: 5 bytes.

Action: move system to nominal mode. This provides full control of all output busses. The system automatically enters nominal mode after startup mode or when the PDU system is in safety mode or emergency low power mode and the PDU rail voltage exceeds their respective high threshold set in the configuration parameter system.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following byte to the system.
{<STID> <IVID> 0x30 <BID>}

Upon command reception the system switches to nominal mode. Auto-on channels will be enabled after any designated time periods.

0x32 (0x33) – Switch to Safety Mode

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 5 bytes.

Action: move subsystem to safety mode. This switches off all non-auto-enable output bus channels. Auto-enable lines will remain powered. Any commands to enable non-auto-enable channels will be rejected. Automatic transfer to safety mode occurs when the PDU rail voltage level falls below the threshold set in the configuration parameter system.

note: only applicable to PDU and PIU boards.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following byte to the system.
{<STID> <IVID> 0x32 <BID>}

Upon command reception the system switches to safety mode. Any non-auto-enable channels are deactivated while auto-enable channels remain active. Any commands to enable non-auto-enable channels are rejected.

(↑ Command Overview ↑)

0x40 (0x41) – Get System Status

Write length: 4 bytes.

Read length: 36 bytes.

Action: return system status information

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
MODE	5	1	uint8		Indicates the current mode of the system. The following modes are defined: 0 = startup 1 = nominal 2 = safety 3 = emergency low power
CONF	6	1	uint8		Read/write configuration parameters have been changed by the EPS master since the last parameters load/save operation. 1 = parameters have been altered since the last load/save 0 = parameters have not been altered since the last load/save Note that at system startup a configuration load operation is performed, resetting this flag to 0.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
RESET_CAUSE	7	1	uint8		Cause of last reset. The following causes are defined: 0 = power-on; system returned from an unpowered state (e.g. power cycle). 1 = watchdog; system was reset due to watchdog timeout, caused by a too-long delay between command interactions with the parent system. 2 = commanded; system was reset after having received a reset command 3 = control system reset; an upset in the EPS control system caused a reset 4 = emlopo; emergency low power mode was engaged because the input voltage dropped below the threshold.
UPTIME	8	4	uint32	1 s	Uptime since system start expressed in seconds. Will wrap around to zero on overflow.
ERROR	12	2	uint16		First internal error encountered during the system control cycle. Information purposes only.
RC_CNT_PWRON	14	2	uint16		Counter indicating amount of power-on reset occurrences since beginning-of-life. Stored in non-volatile memory.
RC_CNT_WDG	16	2	uint16		Counter indicating amount of watchdog reset occurrences since beginning-of-life. Stored in non-volatile memory.
RC_CNT_CMD	18	2	uint16		Counter indicating amount of commanded reset occurrences since beginning-of-life. Stored in non-volatile memory.
RC_CNT_MCU	20	2	uint16		Counter indicating amount of EPS controller resets that occurred since beginning of life.
RC_CNT_EMLOPO	22	2	uint16		Counter indicating amount of reset occurrences due to emergency low power mode since beginning of life.
PREVCMD_ELAPSED	24	2	uint16	1 s	Time elapsed between reception of the previous and this command.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
UNIX_TIME	26	4	uint32	1 s	UNIX time. Seconds elapsed since 1970-01-01 00:00:00 including any leap second corrections. To have unix time available, the system user needs to correct the clock to the proper time after powerup. Command "0xC4 (0xC5) – Correct Time" as described on page 81 can be used for this. Note: The clock will resume from where it stopped at the previous power down. Note2: This is an unsigned 32 bit value that will allow time keeping up to the year 2106.
UNIX_YEAR	30	1	uint8	varies	Calendar year of UNIX_TIME, without the century part. Year = 2000 + UNIX_YEAR
UNIX_MONTH	31	1	uint8	varies	Calendar month of UNIX_TIME
UNIX_DAY	32	1	uint8	86400 s	Calendar day of UNIX_TIME
UNIX_HOUR	33	1	uint8	3600 s	Calendar hour of UNIX_TIME
UNIX_MINUTE	34	1	uint8	60 s	Calendar minute of UNIX_TIME
UNIX_SECOND	35	1	uint8	1 s	Calendar second of UNIX_TIME

Note: Multi byte values are provided in little-endian ordering (i.e. least significant byte first).

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x40 <BID>}

Upon reception of the command, the system gathers system status information and places this in the read buffer. The data is returned during the next read transaction.

0x42 (0x43) – Get PDU/PIU Overcurrent Fault State

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 78 bytes.

Action: Prepare the response buffer with output bus over current events. Over current fault counters are incremented each time a bus is latched off due to an overcurrent event.

note: only available on PDU/PIU boards

note: the response can also be partially read to only include the fields that are in use.

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
STAT_CH_ON	6	2	uint16		Bitflag field indicating channel-on status for the output bus channels. When bit is 1 the corresponding output bus is enabled
STAT_CH_EXT_ON	8	2	uint16		Bitflag field indicating channel-on status for the output bus channels.
STAT_CH_OCF	10	2	uint16		Bitflag field indicating overcurrent fault status for the output bus channels. When bit is 1 the corresponding output bus is latched off due to an overcurrent event. A channel enable or disable command will clear the flag.
STAT_CH_EXT_OCF	12	2	uint16		Bitflag field indicating overcurrent fault status for the output bus channels. The output bus is latched off.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
OCF_CNT_CH00	14	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH01	16	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH02	18	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH03	20	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH04	22	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH05	24	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH06	26	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH07	28	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH08	30	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH09	32	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH10	34	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH11	36	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH12	38	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH13	40	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH14	42	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH15	44	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH16	46	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH17	48	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH18	50	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH19	52	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH20	54	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH21	56	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH22	58	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH23	60	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH24	62	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH25	64	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH26	66	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH27	68	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH28	70	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH29	72	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH30	74	2	uint16		Overcurrent fault count of output bus channel
OCF_CNT_CH31	76	2	uint16		Overcurrent fault count of output bus channel

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x42 <BID>}

Upon reception of the command, the system places the latest output bus latch-off data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

(↑ Command Overview ↑)

0x44 (0x45) – Get PBU ABF Placed State

Write length: 4 bytes.

Read length: 8 bytes.

Action: Prepare the response buffer with ABF placed state information.

note: only available on PBU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
ABF_PLACED_0	6	1	uint8		ABF placed detection circuit result. 0xAB = ABF is placed 0x00 = ABF is not placed Note: any other value than the above two is considered invalid. Upon receiving an invalid value, check ABF_PLACED_1 for a valid value and/or re-issue a new request.
ABF_PLACED_1	7	1	uint8		Same as ABF_PLACED_0, but redundant circuit on the opposite side of the ABF connector. See ABF_PLACED_0 description.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x44 <BID>}

Upon reception of the command, the system places the latest ABF state data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x50 (0x51) – Get PDU Housekeeping Data (RAW)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 258 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in raw form, as received from the hardware, unaltered by the main controller.

See section "Raw Data Conversion Information" (page 22) for information on converting to engineering values.

note: only available on PDU/PIU boards.

note: the response can also be partially read to only include the fields that are in use.

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
$VOLT\_BRDSUP_{raw}$	6	2	uint16		Voltage level of internal board supply in raw form.
$TEMP\_MCU_{raw}$	8	2	uint16		Measured temperature provided by a sensor internal to the MCU in raw form
VIP_INPUT_{raw}	10	6	VIPD		Input V, I and P data for the unit in raw form. Negative values indicate output flow. See "Complex Datatype: VIPD"
STAT_CH_ON	16	2	uint16		Bitflag field indicating channel-on status for output bus channels 0 through 15.
STAT_CH_EXT_ON	18	2	uint16		Bitflag field indicating channel-on status for the extended output bus channels 16 through 31.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STAT_CH_OCF	20	2	uint16		Bitflag field indicating overcurrent latch-off fault status for output bus channels 0 through 15.
STAT_CH_EXT_OCF	22	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the extended output bus channels 16 through 31.
VIP_VD0_{raw}	24	6	VIPD		Output V, I and P of voltage domain 0 in raw form
VIP_VD1_{raw}	30	6	VIPD		Output V, I and P of voltage domain 1 in raw form
VIP_VD2_{raw}	36	6	VIPD		Output V, I and P of voltage domain 2 in raw form
VIP_VD3_{raw}	42	6	VIPD		Output V, I and P of (optional) voltage domain 3 in raw form
VIP_VD4_{raw}	48	6	VIPD		Output V, I and P of (optional) voltage domain 4 in raw form
VIP_VD5_{raw}	54	6	VIPD		Output V, I and P of (optional) voltage domain 5 in raw form
VIP_VD6_{raw}	60	6	VIPD		Output V, I and P of (optional) voltage domain 6 in raw form
VIP_CH00_{raw}	66	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH01_{raw}	72	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH02_{raw}	78	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH03_{raw}	84	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH04_{raw}	90	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH05_{raw}	96	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH06_{raw}	102	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH07_{raw}	108	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH08_{raw}	114	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH09_{raw}	120	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH10_{raw}	126	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH11_{raw}	132	6	VIPD		Output V, I and P of output bus channel in raw form

Name	Offset [byte]	Size [byte]	Type	Unit	Description
Sonboard (SB) items below this line. Availability depends on SB presence and type.					
VIP_CH12_{raw}	138	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH13_{raw}	144	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH14_{raw}	150	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH15_{raw}	156	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH16_{raw}	162	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH17_{raw}	168	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH18_{raw}	174	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH19_{raw}	180	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH20_{raw}	186	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH21_{raw}	192	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH22_{raw}	198	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH23_{raw}	204	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH24_{raw}	210	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH25_{raw}	216	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH26_{raw}	222	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH27_{raw}	228	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH28_{raw}	234	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH29_{raw}	240	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH30_{raw}	246	6	VIPD		Output V, I and P of (optional) output bus channel in raw form

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CH31_{raw}	252	6	VIPD		Output V, I and P of (optional) output bus channel in raw form

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x50 <BID>}

Upon reception of the command, the system places the latest raw housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x52 (0x53) – Get PDU Housekeeping Data (ENG)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 258 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in engineering form.

note: only available on PDU boards

note: the response can also be partially read to only include the fields that are in use.

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
VOLT_BRDSUP	6	2	int16	1e-3 V	Voltage of internal board supply.
TEMP_MCU	12	2	int16	1e-2 °C	Measured temperature provided by a sensor internal to the MCU
VIP_INPUT	16	6	VIPD		Input V, I and P data for the unit. Negative values indicate output flow. See "Complex Datatype: VIPD"
STAT_CH_ON	16	2	uint16		Bitflag field indicating channel-on status for output bus channels 0 through 15.
STAT_CH_EXT_ON	18	2	uint16		Bitflag field indicating channel-on status for the extended output bus channels 16 through 31.
STAT_CH_OCF	20	2	uint16		Bitflag field indicating overcurrent latch-off fault status for output bus channels 0 through 15.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STAT_CH_EXT_OCF	22	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the extended output bus channels 16 through 31.
VIP_VD0	24	6	VIPD		Output V, I and P of voltage domain 0
VIP_VD1	30	6	VIPD		Output V, I and P of voltage domain 1
VIP_VD2	36	6	VIPD		Output V, I and P of voltage domain 2
VIP_VD3	42	6	VIPD		Output V, I and P of (optional) voltage domain 3
VIP_VD4	48	6	VIPD		Output V, I and P of (optional) voltage domain 4
VIP_VD5	54	6	VIPD		Output V, I and P of (optional) voltage domain 5
VIP_VD6	60	6	VIPD		Output V, I and P of (optional) voltage domain 6
VIP_CH00	66	6	VIPD		Output V, I and P of output bus channel
VIP_CH01	72	6	VIPD		Output V, I and P of output bus channel
VIP_CH02	78	6	VIPD		Output V, I and P of output bus channel
VIP_CH03	84	6	VIPD		Output V, I and P of output bus channel
VIP_CH04	90	6	VIPD		Output V, I and P of output bus channel
VIP_CH05	96	6	VIPD		Output V, I and P of output bus channel
VIP_CH06	102	6	VIPD		Output V, I and P of output bus channel
VIP_CH07	108	6	VIPD		Output V, I and P of output bus channel
VIP_CH08	114	6	VIPD		Output V, I and P of output bus channel
VIP_CH09	120	6	VIPD		Output V, I and P of output bus channel
VIP_CH10	126	6	VIPD		Output V, I and P of output bus channel
VIP_CH11	132	6	VIPD		Output V, I and P of output bus channel
Sonboard (SB) items below this line. Availability depends on SB presence and type.					
VIP_CH12	138	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH13	144	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH14	150	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH15	156	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH16	162	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH17	168	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH18	174	6	VIPD		Output V, I and P of (optional) output bus channel

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CH19	180	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH20	186	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH21	192	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH22	198	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH23	204	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH24	210	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH25	216	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH26	222	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH27	228	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH28	234	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH29	240	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH30	246	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH31	252	6	VIPD		Output V, I and P of (optional) output bus channel

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x52 <BID>}

Upon reception of the command, the system places the latest engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x54 (0x55) – Get PDU Housekeeping Data (Running Average)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 258 bytes.

Action: Prepare the response buffer with running average housekeeping data. The housekeeping data is returned in engineering values.

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Reply is equivalent in layout to the "0x52 (0x53) – Get PDU Housekeeping Data (ENG)" (page 51) data request. The direct measurements are replaced with running averaged versions.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x54 <BID>}

Upon reception of the command, the system places the latest running average engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x60 (0x61) – Get PBU Housekeeping Data (RAW)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 84 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in raw form, as received from the hardware, unaltered by the main controller.

See section "Raw Data Conversion Information" (page 22) for information on converting to engineering values.

note: only available on PBU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
$VOLT\_BRDSUP_{raw}$	6	2	uint16		Voltage of internal board supply in raw form.
$TEMP\_MCU_{raw}$	8	2	uint16		Measured temperature provided by a sensor internal to the MCU expressed in raw form
VIP_INPUT_{raw}	10	6	VIPD		Input V, I and P data for the unit in raw form. Effectively a sum of all BPx_{raw}. Negative values indicate output flow. See "Complex Datatype: VIPD"
STAT_BU	16	2	uint16		Bitflag field showing any raised flags on any battery pack (bitwise OR of all STAT_BP fields in BPx_{raw}). See Table 3-18 for definition.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
$BP1_{raw}$	18	22	BPD		Battery pack channel information in raw form. See “Complex Datatype: BPD”
$BP2_{raw}$	40	22	BPD		Battery pack channel information in raw form.
$BP3_{raw}$	62	22	BPD		Battery pack channel information in raw form.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x60 <BID>}

Upon reception of the command, the system places the latest raw housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x62 (0x63) – Get PBU Housekeeping Data (ENG)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 84 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in engineering values.

note: only available on PBU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8		See “Interface Version Identifier (IVID)” (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See “Board Identifier (BID)” (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8		See “Interface Version Identifier (IVID)” (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See “Board Identifier (BID)” (page 20)
STAT	4	1	uint8		See “Response Status Information (STAT)” (page 21)
(reserved)	5	1	uint8		(reserved)
VOLT_BRDSUP	6	2	int16	1e-3 V	Voltage of internal board supply.
TEMP_MCU	12	2	int16	1e-2 °C	Measured temperature provided by a sensor internal to the MCU.
VIP_INPUT	10	6	VIPD		Input V, I and P data for the unit. Effectively a sum of all BPx. Negative values indicate output flow. See “Complex Datatype: VIPD”
STAT_BU	16	2	uint16		Bitflag field showing any raised flags on any battery chain (bitwise OR of all STAT_BP fields in BPx). See Table 3-18 for definition.
BP1	18	22	BPD		Battery pack channel information. See “Complex Datatype: BPD”
BP2	40	22	BPD		Battery pack channel information.
BP3	62	22	BPD		Battery pack channel information.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x62 <BID>}

Upon reception of the command, the system places the latest engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x64 (0x65) – Get PBU Housekeeping Data (Running Average)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 84 bytes.

Action: Prepare the response buffer with running average housekeeping data. The housekeeping data is returned in engineering values.

note: only available on PBU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Reply is equivalent in layout to the "0x62 (0x63) – Get PBU Housekeeping Data (ENG)" (page 57) data request. The direct measurements are replaced with running averaged versions.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x60 <BID>}

Upon reception of the command, the system places the latest running average engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x70 (0x71) – Get PCU Housekeeping Data (RAW)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 72 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in raw form, as received from the hardware, unaltered by the main controller.

See section "Raw Data Conversion Information" (page 22) for information on converting to engineering values.

note: only available on PCU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
$VOLT\_BRDSUP_{raw}$	6	2	uint16	1e-3 V	Voltage of internal board supply in raw form
$TEMP\_MCU_{raw}$	8	2	uint16	1e-2 °C	Measured temperature provided by a sensor internal to the MCU in raw form
VIP_OUTPUT_{raw}	10	6	VIPD		Output V, I and P data for the unit in raw form. Effectively a sum of all CCx_{raw}. Negative values indicate input flow. See "Complex Datatype: VIPD"
$CC1_{raw}$	16	14	CCD		Data on conditioning chain in raw form. See "Complex Datatype: CCD"
$CC2_{raw}$	30	14	CCD		Data on conditioning chain in raw form.
$CC3_{raw}$	44	14	CCD		Data on conditioning chain in raw form.
$CC4_{raw}$	58	14	CCD		Data on conditioning chain in raw form.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x70 <BID>}

Upon reception of the command, the system places the latest raw housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x72 (0x73) – Get PCU Housekeeping Data (ENG)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 72 bytes.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in engineering values.

note: only available on PCU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
VOLT_BRDSUP	6	2	int16		Voltage of internal board supply
TEMP_MCU	8	2	int16		Measured temperature provided by a sensor internal to the MCU
VIP_OUTPUT	10	6	VIPD		Output V, I and P data for the unit. Effectively a sum of all CCx. Negative values indicate input flow. See "Complex Datatype: VIPD"
CC1	16	14	CCD		Data on conditioning chain. See "Complex Datatype: CCD"
CC2	30	14	CCD		Data on conditioning chain.
CC3	44	14	CCD		Data on conditioning chain.
CC4	58	14	CCD		Data on conditioning chain.

Usage Example:

Transfer the following bytes to the system.
{<STID> <IVID> 0x72 <BID>}

Upon reception of the command, the system places the latest engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x74 (0x75) – Get PCU Housekeeping Data (Running Average)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 72 bytes.

Action: Prepare the response buffer with running average housekeeping data. The housekeeping data is returned in engineering values.

note: only available on PCU boards

Command:

Name	Offset [byte]	Size [byte]	Type		Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Reply is equivalent in layout to the "0x72 (0x73) – Get PCU Housekeeping Data (ENG)" (page 62) data request. The direct measurements are replaced with running averaged versions.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x74 <BID>}

Upon reception of the command, the system places the latest running average engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0x82 (0x83) – Get Configuration Parameter

(↑ Command Overview ↑)

Write length: 6 bytes.

Read length: 8 bytes + 1-8 parameter value bytes.

Action: get the value of a configuration parameter.

note: See Section 3.5 for more information on the parameter-id's and their values.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
PAR_ID	4	2	uint16	parameter id of the parameter to get

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8	(reserved)
PAR_ID	6	2	uint16	parameter id of the parameter to get
PAR_VAL	8	1-8	variable	parameter value encompassing n bytes. The length depends on the parameter type.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x82 <BID> 0x00 0x20}

The system places the parameter data for parameter 0x2000 (uint8) in the read buffer. The parameter data is returned during the next read transaction.

(↑ Command Overview ↑)

0x84 (0x85) – Set Configuration Parameter

Write length: 6 bytes + 1-8 parameter value bytes.

Read length: 8 bytes + 1-8 parameter value bytes.

Action: change a configuration parameter. The change will take effect immediately and any function using the parameter will use the new value.

Certain parameter are read-only and the command will return an error if the user tries to change such a parameter. Also certain parameters only allow specific settings or a reduced range. Updates outside of the allowed range will cause these values to be clamped to the minimum/maximum allowable.

note: See Section 3.5 for more information on the parameter-id's and their values.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	3	1	uint8	Command code as stated in the header
BID	2	1	uint8	See "Board Identifier (BID)" (page 20)
PAR_ID	4	2	uint16	parameter id of the parameter to get
PAR_VAL	6	1-8	variable	new parameter value encompassing n bytes. The length depends on the parameter type.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	3	1	uint8	Response code as stated in the header (inside parenthesis).
BID	2	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8	(reserved)
PAR_ID	6	2	uint16	parameter id of the parameter to get
PAR_VAL	8	1-8	variable	parameter value encompassing n bytes. The length depends on the parameter type. This is the value that exists after the set operation. This value might be different from the one provided due to restrictions on the allowable range of the parameter.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x84 <BID> 0x00 0x20 0x08}

The system updates the parameter 0x2000 (uint8) to 0x08 and verifies the parameter is in the allowed range. When the value is not within the allowed range it is altered to be within the allowed range. The resulting parameter data is returned during the next read transaction.

0x86 (0x87) – Reset Configuration Parameter

(↑ Command Overview ↑)

Write length: 6 bytes.

Read length: 8 bytes + 1-8 parameter value bytes.

Action: reset a parameter to its default hard-coded value. All parameters have this value at system power-up or after the software reset command.

note: See Section 3.5 (page 83) for more information on the parameter-id's and their values.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
PAR_ID	4	2	uint16	parameter id of the parameter to get

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8	(reserved)
PAR_ID	6	2	uint16	parameter id of the parameter to get
PAR_VAL	8	1-8	variable	parameter value encompassing n bytes. The length depends on the parameter type. This is the value that exists after the set operation. This value might be different from the one provided due to restrictions on the allowable range of the parameter.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x86 <BID> 0x00 0x20}

The system resets the parameter 0x2000 (uint8) to its hardcoded default. The resulting parameter data is returned during the next read transaction.

0x90 (0x91) – Reset Configuration

(↑ Command Overview ↑)

Write length: 5 bytes.

Read length: 5 bytes

Action: Reset all configuration parameters to hard-coded defaults, discarding any changes made, in volatile memory (only!). This is performed automatically at system startup before an attempt to load a configuration is performed. If no (valid) configuration is found that can be loaded, the system will use hard coded defaults.

The read-only parameter set contains a save-counter and the applicable checksum of the any loaded read/write parameter set which can be used to track load and save state of the configuration. Use the 0x82 (0x83) – Get Configuration Parameter command to retrieve these parameters. The save-counter and checksum will both be 0 when hard-coded defaults have been used.

Use the "0x94 (0x95) – Save Configuration" command (page 71) to store the reset configuration to non-volatile memory if desired.

Reset Configuration is performed immediately after command acceptance.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CONF_KEY	4	1	uint8	Configuration key: 0xA7. Any other value causes the reset command to be rejected with a parameter error. The reset will not be performed in that case.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x90 <BID> 0xA7}

The system resets the configuration to hard-coded defaults, discarding any volatile memory settings that might have been performed.

(↑ Command Overview ↑)

0x92 (0x93) – Load Configuration

Write length: 5 bytes.

Read length: 5 bytes

Action: Load all configuration parameters from non-volatile memory, discarding any changes made in volatile memory. This is performed automatically at system startup if a valid load configuration is encountered in non-volatile memory. If no (valid) configuration is found, the system will initialize using hard coded defaults.

The read-only parameter set contains a save-counter and the applicable checksum of the stored read/write parameter set which can be used to track load and save state. Use the "0x82 (0x83) – Get Configuration Parameter" command (page 65) to retrieve these parameters. The save-counter and checksum will both be 0 when hard-coded defaults have been used.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CONF_KEY	4	1	uint8	Configuration key: 0xA7. Any other value causes the load command to be rejected with a parameter error. The load will not be performed in that case.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x92 <BID> 0xA7}

The system (re-)loads the configuration stored in non-volatile memory, discarding any volatile memory settings that might have been performed.

0x94 (0x95) – Save Configuration

(↑ Command Overview ↑)

Write length: 7 bytes.

Read length: 5 bytes

Action: Commit all read/write configuration parameters kept in volatile memory to non-volatile memory.

A checksum taken over all read/write configuration parameters ensures no inadvertent changes have occurred between altering the volatile memory parameters and issuing the save command. Provide 0 as checksum to force-save without this protection.

The read-only parameter set contains a save-counter and the applicable checksum of the stored read/write parameter set which can be used to track load and save state. Use the "0x82 (0x83) – Get Configuration Parameter" command (page 65) to retrieve these parameters.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CONF_KEY	4	1	uint8	Configuration key: 0xA7. Any other value causes the save command to be rejected with a parameter error. The save will not be performed in that case.
CHECKSUM	5	2	uint16	Checksum of the configuration data to save. The CRC16 polynomial used is the CRC-CCITT-BR polynomial, given by f(x) = x16 + x12 + x5 + 1. The seed is 0xFFFF. The CRC is taken over all read/write parameters as defined in the configuration parameter list, from lowest ID to highest ID. To force save this value can be set to 0. The save will then proceed without performing CRC verification.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0x94 <BID> 0xA7 0xBA 0x12}

The system verifies that the CRC over the read/write parameters results in the provided checksum 0x12BA. Successful verification is followed by committing the read/write parameter set to non-volatile memory. The read-only save-counter parameter is incremented after successful save and can be retrieved using a parameter get.

0xA0 (0xA1) – Get PIU Housekeeping Data (RAW)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 116 bytes w/o daughterboard or 274 bytes with daughterboard.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in raw form, as received from the hardware, unaltered by the main controller.

See section "Raw Data Conversion Information" (page 22) for information on converting to engineering values.

note: only applicable to PIU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8		See "Response Status Information (STAT)" (page 21)
(reserved)	5	1	uint8		(reserved)
$VOLT\_BRDSUP_{raw}$	6	2	uint16		Voltage of internal board supply in raw form
$TEMP_{raw}$	8	2	uint16		Measured temperature provided by a sensor internal to the MCU in raw form
$VIP\_DIST\_INPUT_{raw}$	10	6	VIPD		Input V, I and P data taken at the input of the distribution part of the unit in raw form. Negative values indicate output flow. See "Complex Datatype: VIPD"
$VIP\_BATT\_INPUT_{raw}$	16	6	VIPD		Input V, I and P data taken at the input of the battery part of the unit in raw form. Negative values indicate output flow.
STAT_CH_ON	22	2	uint16		Bitflag field indicating channel-on status for the output bus channels.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STAT_CH_OCF	24	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the output bus channels.
BAT_STAT	26	2	uint16		Bitflag field indicating BP board status. See Table 3-18 for definition.
BAT_TEMP2$_{raw}$	28	2	uint16		2 and 4 cell battery pack: Battery pack temperature in between the center battery cells.
BAT_TEMP3$_{raw}$	30	2	uint16		2 cell battery pack: not used 4 cell battery pack: Battery pack temperature on the front of the battery pack.
VOLT_VD0$_{raw}$	32	2	uint16		Voltage of voltage domain 0 in raw form
VOLT_VD1$_{raw}$	34	2	uint16		Voltage of voltage domain 1 in raw form
VOLT_VD2$_{raw}$	36	2	uint16		Voltage of voltage domain 2 in raw form
VIP_CH00$_{raw}$	38	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH01$_{raw}$	44	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH02$_{raw}$	50	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH03$_{raw}$	56	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH04$_{raw}$	62	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH05$_{raw}$	68	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH06$_{raw}$	74	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH07$_{raw}$	80	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH08$_{raw}$	86	6	VIPD		Output V, I and P of output bus channel in raw form
CC1$_{raw}$	92	8	CCSD		Data on conditioning chain in raw form. See “Complex Datatype: CCSD”
CC2$_{raw}$	100	8	CCSD		Data on conditioning chain in raw form.
CC3$_{raw}$	108	8	CCSD		Data on conditioning chain in raw form.
Daughterboard (DB) items below this line. Availability depends on DB presence and type.					

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CH09_{raw}	116	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH10_{raw}	122	6	VIPD		Output V, I and P of output bus channel in raw form
VIP_CH11_{raw}	128	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH12_{raw}	134	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH13_{raw}	140	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH14_{raw}	146	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH15_{raw}	152	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
$CC4_{raw}$	158	8	CCSD		Data on conditioning chain in raw form.
$CC5_{raw}$	166	8	CCSD		Data on conditioning chain in raw form.
STAT_CH_EXT_ON	174	2	uint16		Bitflag field indicating channel-on status for the extended output bus channels 16 through 31.
STAT_CH_EXT_OCF	176	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the extended output bus channels 16 through 31.
VIP_CH16_{raw}	178	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH17_{raw}	184	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH18_{raw}	190	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH19_{raw}	196	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH20_{raw}	202	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH21_{raw}	208	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH22_{raw}	214	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH23_{raw}	220	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH24_{raw}	226	6	VIPD		Output V, I and P of (optional) output bus channel in raw form

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CH25_{raw}	232	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH26_{raw}	238	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH27_{raw}	244	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH28_{raw}	250	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH29_{raw}	256	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH30_{raw}	262	6	VIPD		Output V, I and P of (optional) output bus channel in raw form
VIP_CH31_{raw}	268	6	VIPD		Output V, I and P of (optional) output bus channel in raw form

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xA0 <BID>}

Upon reception of the command, the system places the latest raw housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0xA2 (0xA3) – Get PIU Housekeeping Data (ENG)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 116 bytes w/o daughterboard or 274 bytes with daughterboard.

Action: Prepare the response buffer with housekeeping data. The housekeeping data is returned in engineering values.

note: only applicable to PIU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8		See “Interface Version Identifier (IVID)” (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See “Board Identifier (BID)” (page 20)

Reply:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See “System Type Identifier (STID)” (page 17)
IVID	1	1	uint8		See “Interface Version Identifier (IVID)” (page 18)
RC	2	1	uint8		Response code as stated in the header (inside parenthesis).
BID	3	1	uint8		See “Board Identifier (BID)” (page 20)
STAT	4	1	uint8		See “Response Status Information (STAT)” (page 21)
(reserved)	5	1	uint8		(reserved)
VOLT_BRDSUP	6	2	int16	1e-3 V	Voltage of internal board supply
TEMP	8	2	int16	1e-2 °C	Measured temperature provided by a sensor internal to the MCU
VIP_DIST_INPUT	10	6	VIPD		Input V, I and P data taken at the input of the distribution part of the unit in raw form. Negative values indicate output flow. See “Complex Datatype: VIPD”
VIP_BATT_INPUT	16	6	VIPD		Input V, I and P data taken at the input of the battery part of the unit in raw form. Negative values indicate output flow.
STAT_CH_ON	22	2	uint16		Bitflag field indicating channel-on status for the output bus channels.
STAT_CH_OCF	24	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the output bus channels.

Name	Offset [byte]	Size [byte]	Type	Unit	Description
BAT_STAT	26	2	uint16		Bitflag field indicating BP board status. See Table 3-18 for definition.
BAT_TEMP2	28	2	int16		2 and 4 cell battery pack: Battery pack temperature in between the center battery cells.
BAT_TEMP3	30	2	int16		2 cell battery pack: not used 4 cell battery pack: Battery pack temperature on the front of the battery pack.
VOLT_VD0	32	2	int16		Voltage of voltage domain 0
VOLT_VD1	34	2	int16		Voltage of voltage domain 1
VOLT_VD2	36	2	int16		Voltage of voltage domain 2
VIP_CH00	38	6	VIPD		Output V, I and P of output bus channel
VIP_CH01	44	6	VIPD		Output V, I and P of output bus channel
VIP_CH02	50	6	VIPD		Output V, I and P of output bus channel
VIP_CH03	56	6	VIPD		Output V, I and P of output bus channel
VIP_CH04	62	6	VIPD		Output V, I and P of output bus channel
VIP_CH05	68	6	VIPD		Output V, I and P of output bus channel
VIP_CH06	74	6	VIPD		Output V, I and P of output bus channel
VIP_CH07	80	6	VIPD		Output V, I and P of output bus channel
VIP_CH08	86	6	VIPD		Output V, I and P of output bus channel
CC1	92	8	CCSD		Data on conditioning chain. See “Complex Datatype: CCSD”
CC2	100	8	CCSD		Data on conditioning chain
CC3	108	8	CCSD		Data on conditioning chain
Daughterboard (DB) items below this line. Availability depends on DB presence and type.					
VIP_CH09	116	6	VIPD		Output V, I and P of output bus channel
VIP_CH10	122	6	VIPD		Output V, I and P of output bus channel
VIP_CH11	128	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH12	134	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH13	140	6	VIPD		Output V, I and P of (optional) output bus channel

Name	Offset [byte]	Size [byte]	Type	Unit	Description
VIP_CH14	146	6	VIPD		Output V, I and P of (optional) output bus channel
VIP_CH15	152	6	VIPD		Output V, I and P of (optional) output bus channel
CC4	158	8	CCSD		Data on conditioning chain
CC5	166	8	CCSD		Data on conditioning chain
STAT_CH_EXT_ON	174	2	uint16		Bitflag field indicating channel-on status for the extended output bus channels 16 through 31.
STAT_CH_EXT_OCF	176	2	uint16		Bitflag field indicating overcurrent latch-off fault status for the extended output bus channels 16 through 31.
VIP_CH16	178	6	VIPD		Output V, I and P of output bus channel
VIP_CH17	184	6	VIPD		Output V, I and P of output bus channel
VIP_CH18	190	6	VIPD		Output V, I and P of output bus channel
VIP_CH19	196	6	VIPD		Output V, I and P of output bus channel
VIP_CH20	202	6	VIPD		Output V, I and P of output bus channel
VIP_CH21	208	6	VIPD		Output V, I and P of output bus channel
VIP_CH22	214	6	VIPD		Output V, I and P of output bus channel
VIP_CH23	220	6	VIPD		Output V, I and P of output bus channel
VIP_CH24	226	6	VIPD		Output V, I and P of output bus channel
VIP_CH25	232	6	VIPD		Output V, I and P of output bus channel
VIP_CH26	238	6	VIPD		Output V, I and P of output bus channel
VIP_CH27	244	6	VIPD		Output V, I and P of output bus channel
VIP_CH28	250	6	VIPD		Output V, I and P of output bus channel
VIP_CH29	256	6	VIPD		Output V, I and P of output bus channel
VIP_CH30	262	6	VIPD		Output V, I and P of output bus channel
VIP_CH31	268	6	VIPD		Output V, I and P of output bus channel

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xA2 <BID>}

Upon reception of the command, the system places the latest engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0xA4 (0xA5) – Get PIU Housekeeping Data (Running Average)

(↑ Command Overview ↑)

Write length: 4 bytes.

Read length: 116 bytes w/o daughterboard or 274 bytes with daughterboard.

Action: Prepare the response buffer with running average housekeeping data. The housekeeping data is returned in engineering values.

note: only applicable to PIU boards

Command:

Name	Offset [byte]	Size [byte]	Type	Unit	Description
STID	0	1	uint8		See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8		See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8		Command code as stated in the header
BID	3	1	uint8		See "Board Identifier (BID)" (page 20)

Reply:

Reply is equivalent in layout to the "0xA2 (0xA3) – Get PIU Housekeeping Data (ENG)" (page 77) data request. The direct measurements are replaced with running averaged versions.

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xA4 <BID>}

Upon reception of the command, the system places the latest running average engineering housekeeping data in the read buffer. The data is returned during the next read transaction. The STAT NEW bit is set when the accompanying data represents new measurement information that is retrieved for the first time.

0xC4 (0xC5) – Correct Time

(↑ Command Overview ↑)

Write length: 8 bytes.

Read length: 5 bytes.

Action: Correct the unit's unix time with the specified amount of seconds.

The unix time value is returned as part of the "0x40 (0x41) – Get System Status" response, described on page 40.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
CORRECTION	4	4	int32	Amount of seconds to correct the unix time with. Positive values are added and negative values are subtracted from the unix time. Reminder: provide multibyte signed values in little endian order using two's complement encoding.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xC4 <BID> 0xF0 0xF1 0xFF 0xFF}

Upon reception of the command, the system corrects the unix time by 0xFF FF F1 F0 (= -3600 in decimal) seconds, shifting its time value back by one hour.

0xC6 (0xC7) – Zero Reset Cause Counters

(↑ Command Overview ↑)

Write length: 5 bytes.

Read length: 5 bytes.

Action: Write all reset cause counters to zero in persistent memory.

The reset cause counters are returned as part of the "0x40 (0x41) – Get System Status" response, described on page 40.

Command:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
CC	2	1	uint8	Command code as stated in the header
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
ZERO_KEY	4	1	int32	Zero key: 0xA7. Any other value causes this command to be rejected with a parameter error. The write to zero is not performed in that case.

Reply:

Name	Offset [Bytes]	Size [Bytes]	Type	Description
STID	0	1	uint8	See "System Type Identifier (STID)" (page 17)
IVID	1	1	uint8	See "Interface Version Identifier (IVID)" (page 18)
RC	2	1	uint8	Response code as stated in the header (inside parenthesis).
BID	3	1	uint8	See "Board Identifier (BID)" (page 20)
STAT	4	1	uint8	See "Response Status Information (STAT)" (page 21)

Usage Example:
Transfer the following bytes to the system.
{<STID> <IVID> 0xC6 <BID> 0xA7}

Upon reception of the command, the system writes all reset cause counters to 0.

3.5 Digital Configuration Registers

This section describes the available configuration parameters for the system. They allow influencing system behaviour and provide measurement conversion values. The configuration parameters are altered through command messages as described in Section 3.4 "Digital Command Messages". Parameters are first grouped according to whether they are read/write or read only, then subsequently grouped according to their datatype.

Table 3-23: Group Overview

Group	Comment
CP – Sequential Overview Read/Write	Read/write parameters
CP – Sequential Overview Read-Only	Read only parameters

The remainder of this section will first discuss topics that are important for understanding the configuration parameter system and then present the available parameters themselves.

3.5.1 Param-ID

Each parameter has been attributed a 2-byte number that uniquely identifies that parameter. This identifier is referred to as the *configuration parameter identifier* (or *param-id*). Each parameter also has an accompanying data type. The data type determines how many bytes need to be supplied as the parameter-value. The possible types are shown in Table 3-24.

Table 3-24: Possible Parameter Data Types

Type	Sign	Bytes	Code	Comments
int8	Signed	1	0x1000	
uint8	Unsigned	1	0x2000	
int16	Signed	2	0x3000	
uint16	Unsigned	2	0x4000	
int32	Signed	4	0x5000	
uint32	Unsigned	4	0x6000	
float	Signed	4	0x7000	IEEE754 Single Prec. Floating Point
int64	Signed	8	0x8000	
uint64	Unsigned	8	0x9000	
double	Signed	8	0xA000	IEEE754 Double Prec. Floating Point

3.5.2 Endianess

Endianess of all multi-byte data, including float/double, is little-endian (i.e. least significant byte to most significant byte). E.g. a float value 1.0 is represented by the number 0x3FF00000. This is transferred as 0x00 0x00 0xF0 0x3F in little-endian format.

Message Example

The config set command-code is 0x84. The parameter-id for a parameter is 0x5002, with int32 as type. The value therefore consists of 4 bytes. To update this parameter to the value 0x11332255 (decimal: 288563797), the bytes to send and its response are:

Send: STID IVID 0x84 BID | 0x02 0x50 | 0x55 0x22 0x33 0x11

Response: STID IVID 0x84 BID 0x80 | 0x02 0x50 | 0x55 0x22 0x33 0x11

Note that the STID, IVID and BID are placeholders for their respective byte values. The STAT value of 0x80 in the response indicates the command was accepted and the response is NEW.

CP – Sequential Overview Read/Write (↑ Group Overview ↑)

A sequential overview of the available read/write parameters are given in Table 3-25.

Table 3-25: Read/Write Configuration Parameter List

ID	Name	Unit	Type	Default	Description
0x6xxx – uint32 (read/write)					
0x6000 – 0x6001	(internal)	-	uint32	n/a	used for internal purposes
0x6002	CH_STARTUP_ENA_BF	-	uint32	PDU 0x0000 0111 PIU 0x0000 0023	Bitflag field indicating which output bus channels need to be automatically enabled upon entering safety or nominal mode. A delay duration can be configured with the OBUS_STARTUP_DELAY parameter. Note: Force enable channels implicitly have startup enable set. Setting or clearing these channels in this field has no effect. Note2: Changing the default requires setting the appropriate key. An incorrect key causes the default value to be used.
0x6003	CH_STARTUP_KEY	-	uint32	PDU 0x1111 1222 PIU 0x1111 1134	The startup enable key is the startup enable bitflag plus 0x1111 1111. Example: startup_enable_bf = 0x0111 1111 key = 0x0111 + 0x1111 1111 = 0x1111 1222
0x6004	CH_LATCHOFF_ENA_BF	-	uint32	PDU 0x0000 0111 PIU 0x0000 0023	Bitflag field indicating which output bus channels need to be automatically re-enabled upon overcurrent latch off. The off duration can be configured with the OBUS_LATCHOFF_DELAY parameter. Note: Force enable channels implicitly have latchoff enable set. Setting or clearing these channels in this field has no effect. Note2: Changing the default requires setting the appropriate key. An incorrect key causes the default value to be used.

ID	Name	Unit	Type	Default	Description
0x6005	CH_LATCHOFF_KEY	-	uint32	PDU 0x1111 1222 PIU 0x1111 1134	The latchoff enable key is the latchoff enable bitflag plus 0x1111 1111. Example: latchoff_enable_bf = 0x1111 1111 key = 0x0111 + 0x1111 1111 = 0x1111 1222
0x4xxx – uint16 (read/write)					
0x4000	TTC_WDG_TIMEOUT	s	uint16	300	system resets when no data interaction occurred with the parent system within this timeout. After 0.65 x timeout the system's command interface peripheral is reset (default: 195 [s]). Set to 0 to disable. Note: to set a value below 30 [s] the correct wdg timeout key must be provided. If the incorrect key is set with a value below 30 [s] (including 0), a timeout of 30 [s] is used.
0x4001	TTC_WDG_TIMEOUT_KEY	-	uint16	0	The timeout key needs to be set to a predefined value for timeout values below 30 [s]. For 30 > timeout > 0 [s]: Key = 0xDB30 For timeout = 0 (disable): Key = 0xD0FF
0x4002 – 0x4021	CH_STARTUP_DELAY	ms	uint16	0 (each channel)	delay between completing startup mode and enabling of a startup enable channel. A delay can be set per channel individually. Note: max = 20000 ms = 20 s. Attempts to set a lower value than min will result in the minimum value being set. Note2: accuracy is limited to ~10 ms increments.
0x4022 – 0x4041	CH_LATCHOFF_DELAY	ms	uint16	5000 (each channel)	delay between detection of overcurrent latchoff and re-enable of a latchoff enable channel. A delay can be set per channel individually. Note: min = 50 ms. max = 20000 ms. Attempts to set a value outside the indicated range will clamp the value to the closest value within the range. Note2: accuracy is limited to ~10 ms increments.

ID	Name	Unit	Type	Default	Description
0x4042	SAFETY_VOLT_LOTHR	mV	uint16	12400	Sets the requested low voltage threshold for safety mode. This value is used as input for a range and threshold check comparing low and high values of safe and emlopo modes. The resulting threshold value actually used by the system is SAFETY_VOLT_LOTHR_USED as described in the read-only section.
0x4043	SAFETY_VOLT_HITHR	mV	uint16	14000	Sets the requested high voltage threshold for safety mode. This value is used as input for a range and threshold check comparing low and high values of safe and emlopo modes. The resulting threshold value actually used by the system is SAFETY_VOLT_HITHR_USED as described in the read-only section.
0x3xxx – int16 (read/write)					
0x3000 – 0x3002	LOTHR_BP1_HEATER – LOTHR_BP3_HEATER	0.01 ˚C	int16	200 (each battery pack)	Battery heater low temperature threshold. Once the battery temperature falls below this threshold the battery heater is turned on (if enabled). A threshold can be set per battery channel individually. Caution: It is recommended to always keep the low threshold well above 0 degrees to keep excessive battery degradation in sub-zero temperatures from occurring. Note: min = -2000. max = 3000. Attempts to set a value outside the indicated range will clamp the value to the closest value within the range.
0x3003 – 0x3005	HITHR_BP1_HEATER – HITHR_BP3_HEATER	0.01 ˚C	int16	600 (each battery pack)	Battery heater high temperature threshold. Once the battery temperature rises above this threshold the battery heater is turned off (if enabled). A threshold can be set per battery channel individually. Note: min = -2000. max = 3400. Attempts to set a value outside the indicated range will clamp the value to the closest value within the range.

ID	Name	Unit	Type	Default	Description
0x3006 – 0x3008	LOTHR_BP1_UNBAL – LOTHR_BP3_UNBAL	mV	int16	30 (each battery pack)	Battery balancer low threshold. When the voltage difference between the lowest and highest cell voltage in a battery pack is below this value, cell balancing is stopped. A threshold can be set per battery channel individually. Note: min = 30 mV. Attempts to set a lower value than min will result in the minimum value being set.
0x3009 – 0x300B	HITHR_BP1_UNBAL – HITHR_BP3_UNBAL	mV	int16	50 (each battery pack)	Battery balancer high threshold. When the voltage difference between the lowest and highest cell voltage in a battery pack is above this value, cell balancing is started. Cell balancing drains the highest voltage cell in a pack. A threshold can be set per battery channel individually. Note: min = 30 mV. Attempts to set a lower value than min will result in the minimum value being set.
0x300C	MCU_TEMP_BIAS	0.01 ˚C	int16	0	MCU temperature measurement bias. This bias is added to the measurement value, after factory calibration corrections using TLV.
0x300D	MCU_TEMP_PREMUL	-	int16	1	MCU temperature measurement pre-multiplier. Multiplied after the bias has been added.
0x300E	MCU_TEMP_POSDIV	-	int16	1	MCU temperature measurement post-divider. Divided after the bias is added and pre-multiplier multiplied.
0x300F – 0x3011	BP1_TEMP1_BIAS – BP1_TEMP3_BIAS	0.01 ˚C	int16	0 (each temp)	Battery temperature measurement bias for each of the temperature sensors. This bias is added to the measurement value.
0x3012 – 0x3014	BP2_TEMP1_BIAS – BP2_TEMP3_BIAS	0.01 ˚C	int16	0 (each temp)	Battery temperature measurement bias for each of the temperature sensors. This bias is added to the measurement value.
0x3015 – 0x3017	BP3_TEMP1_BIAS – BP3_TEMP3_BIAS	0.01 ˚C	int16	0 (each temp)	Battery temperature measurement bias for each of the temperature sensors. This bias is added to the measurement value.
0x3018 – 0x301A	BP1_TEMP1_PREMUL – BP1_TEMP3_PREMUL	-	int16	1 (each temp)	Battery temperature measurement pre-multiplier for each of the temperature sensors. This value is multiplied after the bias has been added.

ID	Name	Unit	Type	Default	Description
0x301B – 0x301D	BP2_TEMP1_PREMUL – BP2_TEMP3_PREMUL	-	int16	1 (each temp)	Battery temperature measurement pre-multiplier for each of the temperature sensors. This value is multiplied after the bias has been added.
0x301E – 0x3020	BP3_TEMP1_PREMUL – BP3_TEMP3_PREMUL	-	int16	1 (each temp)	Battery temperature measurement pre-multiplier for each of the temperature sensors. This value is multiplied after the bias has been added.
0x3021 – 0x3023	BP1_TEMP1_POSDIV – BP1_TEMP3_POSDIV	-	int16	1 (each item)	Battery temperature measurement post-divider for each of the temperature sensors. This value is used for dividing after the bias is added and pre-multiplier multiplied.
0x3024 – 0x3026	BP2_TEMP1_POSDIV – BP2_TEMP3_POSDIV	-	int16	1 (each item)	Battery temperature measurement post-divider for each of the temperature sensors. This value is used for dividing after the bias is added and pre-multiplier multiplied.
0x3027 – 0x3029	BP3_TEMP1_POSDIV – BP3_TEMP3_POSDIV	-	int16	1 (each item)	Battery temperature measurement post-divider for each of the temperature sensors. This value is used for dividing after the bias is added and pre-multiplier multiplied.
0x2xxx – uint8 (read/write)					
0x2000	BOARD_IDENTIFIER	-	uint8	1	Board identifier. See “Board Identifier (BID)” for more information. Note: Changing the default requires setting the appropriate key. An incorrect key will cause the unit to use BID = 0.
0x2001	BOARD_IDENTIFIER_KEY	-	uint8	0x12	The board identifier key is the board identifier plus 0x11. Example: bid = 0x01 key = 0x01 + 0x11 = 0x12
0x2002	RAVG_STRENGTH_P2	-	uint8	3	running average filtering strength expressed as a 2-power. e.g. setting 3 gives a strength of 2**3 = 8
0x1xxx – int8 (read/write)					
0x1000	(internal)	-	int8	n/a	used for internal purposes

ID	Name	Unit	Type	Default	Description
0x1001 – 0x1003	AUTO_HEAT_ENA_BP1 – AUTO_HEAT_ENA_BP3	-	int8	0 (each battery pack)	non-zero: automatic battery pack heater control according to the high and low temperature thresholds is performed. zero: heaters are disabled.
0x1004 – 0x1006	AUTO_BAL_ENA_BP1 – AUTO_BAL_ENA_BP3	-	int8	0 (each battery pack)	non-zero: automatic cell balancing according to the high and low balancing thresholds is performed. zero: balancing is disabled.
0x1007 – 0x100D	VD1_ALWAYS_ENA – VD6_ALWAYS_ENA	-	int8	0 (each VD)	When non-zero forces the VD regulator to be always enabled. When zero the VD regulator only starts up when one of the VD output channels is enabled. Note: Not all voltage domains support enable/disable functionality.
0x100E – 0x1014	VD1_ALWAYS_DISA – VD6_ALWAYS_DISA	-	int8	0 (each VD)	When non-zero forces the VD regulator to be always disabled. When zero the VD regulator only starts up when one of the VD output channels is enabled. Note: Any force-enable bus channels on the VD will still cause the VD to go into regulation, regardless of this setting. Note2: Not all voltage domains support enable/disable functionality.

CP – Sequential Overview Read-Only

(↑ Group Overview ↑)

A sequential overview of the available read-only parameters are given in Table 3-26.

Table 3-26: Read-Only Configuration Parameter List

ID	Name	Unit	Type	Default	Description
0x68xx – uint32 (read-only)					
0x6800 – 0x6808	(internal)	-	uint32	n/a	used for internal purposes
0x6809	CH_FORCE_ENA_USE_BF	-	uint32	PDU 0x0000 0111 PIU 0x0000 0023	bitflag field indicating which output bus channels belong to the force-enable channel group. Default shown for PDU. Note: This can be different depending on customer request upon order time.
0x680A	CH_STARTUP_ENA_USE_BF	-	uint32	PDU 0x0000 0111 PIU 0x0000 0023	bitflag field indicating which output bus channels belong to the startup enable channel group. Default shown for PDU.
0x680B	CH_LATCHOFF_ENA_USE_BF	-	uint32	PDU 0x0000 0111 PIU 0x0000 0023	bitflag field indicating which output bus channels belong to the latchoff enable channel group. Default shown for PDU.
0x680C – 0x6812	VD1_ALLOC_CH_BF – VD6_ALLOC_CH_BF	-	uint32	PDU 0x0000 000F 0x0000 00F0 0x0000 0F00 0x0000 1000 0x0000 2000 0x0000 4000 0x0000 8000 PIU 0x0000 0E01 0x0000 001E 0x0000 01E0 0x0000 3000 0x0000 C000 0x0001 0000 0x0000 0000	Output bus channel mapping to voltage domains. Each voltage domain parameter is a bitflag field indicating the channels that belong to it. Eg: Param of PDU VD2 = 0x0000 0F00. Hence channels 8, 9, 10 and 11 belong to VD2.

ID	Name	Unit	Type	Default	Description
0x6813	SWCI_CH_CMD_ENA_BF	-	uint32	n/a	bitflag field showing which channels have received an enable command through the single wire command interface (SWCI). Note: Subsequent disable commands over the interface will reset the corresponding enable flag again. **Caution:** this flag field does **not** show the channel state currently in effect. It only gives information on correctly received switch commands on the SWCI.
0x6814	SWCI_CH_CMD_DISA_BF	-	uint32	n/a	bitflag field showing which channels have received a disable command through the single wire command interface (SWCI). Note: Subsequent disable commands over the interface will reset the corresponding enable flag again. **Caution:** this flag field does **not** show the channel state currently in effect. It only gives information on correctly received switch commands on the SWCI.
0x48xx – uint16 (read-only)					
0x4800	TTC_I2C_SLAVE_ADDR	-	uint16	variable	right aligned 7-bit I2C slave address of the system. Note: by default standard addresses are provided to each of the boards. A custom address can be requested upon order placement. Note2: All I^2C addresses on an I^2C bus must be unique. Hence when ordering multiple subsystems with custom addresses make sure each board has an unique address.
0x4801	CONF_NVM_SAVE_CNTR	-	uint16	n/a	configuration parameter non-volatile memory save counter. Increments with each successful nvm save.

ID	Name	Unit	Type	Default	Description
0x4802	CONF_NVM_SAVE_CHKS	-	uint16	n/a	CRC16 checksum[4] for the read/write configuration parameters.
0x4803	RST_CAUSE	-	uint16	n/a	cause of last reset 0 = power-on 1 = watchdog 2 = commanded 3 = mcu 4 = emlopo
0x4804	RST_CNTR_PWRON	-	uint16	n/a	amount of power-on resets of the system since its firmware was flashed
0x4805	RST_CNTR_WDG	-	uint16	n/a	amount of watchdog resets of the system since its firmware was flashed
0x4806	RST_CNTR_CMD	-	uint16	n/a	amount of commanded resets of the system since its firmware was flashed
0x4807	RST_CNTR_MCU	-	uint16	n/a	amount of mcu resets of the system since its firmware was flashed
0x4808	RST_CNTR_EMLOPO	-	uint16	n/a	amount of emergency low power mode resets of the system since its firmware was flashed.
0x4809	RST_CODE_MCU_RAW	-	uint16	n/a	cause of last reset code read from MCU hardware
0x480A	EMLOPO_VOLT_LOTHR	mV	uint16	12000	when the rail voltage falls below this value the system will enter emergency-low-power mode. This shuts off all output busses.
0x480B	EMLOPO_VOLT_HITHR	mV	uint16	12400	when the rail voltage climbs above this value the system will switch to safety mode.
0x480C	EMLOPO_PERIOD	s	uint16	600	maximum time period emlopo mode is maintained. After this time period elapses the system will move to safety mode.

4 The CRC16 polynomial used is the CRC-CCITT-BR polynomial, given by f(x) = x16 + x12 + x5 + 1. The seed is 0xFFFF. The CRC is taken over all read/write parameters described in Table 3-25 (page 57), in the same order as they are presented and taking into account the little-endian byte ordering for multi-byte values.

ID	Name	Unit	Type	Default	Description
0x480D	SAFETY_VOLT_LOTHR_USED	mV	uint16	n/a	when the rail voltage falls below this value the system will enter safety mode. This shuts off all non cmd-enable output busses. This is the value used by the system after the read/write values are checked for range and relative threshold value. Safety low threshold can never be lower than emlopo low threshold and will be corrected accordingly.
0x480E	SAFETY_VOLT_HITHR_USED	mV	uint16	n/a	when the rail voltage climbs above this value the system will switch to nominal mode. This is the value used by the system after the read/write values are checked for range and relative threshold value. Safety high threshold can never be lower than safety low threshold or emlopo low threshold and will be corrected accordingly.
0x480F	SAFETY_LINGER	s	uint16	60	minimum time period safety mode is maintained. The system will only automatically switch out of safety mode after this period has elapsed. Note: The mode can be changed using a TT&C command at any time.
0x4810	TTC_WDG_TIMOUT_USED	s	uint16	n/a	TT&C watchdog timeout value used in the system. This can differ from the read/write timeout setting in case invalid keys are provided. Note: the default value used is the default of the read/write parameter (assuming the default key is not altered).

ID	Name	Unit	Type	Default	Description
0x4811	TTC_PREVCMD_ELAPSED	s	uint16	n/a	elapsed time since previous received command. once this value exceeds the watchdog timeout used value the watchdog will trigger. Note: requesting this parameter is considered a valid command, and will therefore reset the watchdog.
0x4812 – 0x4813	(internal)	-	uint16	n/a	used for internal purposes
0x38xx – int16 (read-only)					
0x3800	ADC_MCU_TEMP_V25T30	-	int16	factory (eg. 1300)	burned-in factory defined calibrated values for the MCU on this system
0x3801	ADC_MCU_TEMP_V25T85	-	int16	factory (eg. 1500)	burned-in factory defined calibrated values for the MCU on this system
0x28xx – uint8 (read-only)					
0x2800	STID	-	uint8	see STID	subsystem type identifier. see "System Type Identifier (STID)" for more information.
0x2801	IVID	-	uint8	see IVID	interface version identifier. see "Interface Version Identifier (IVID)" for more information.
0x2802	BID_USED	-	uint8	variable	BID used by the board. This value depends on the read/write BOARD_IDENTIFIER and BOARD_IDENTIFIER_KEY values.
0x2803	BOOT_RESUME_SHORT	-	uint8	0	indicates the amount of times a reset occurred shortly after a boot resume was performed. When this value exceeds a fixed threshold a full power cycle is performed instead.
0x18xx – int8 (read-only)					
0x1800	CONF_PARAM_CHANGED	-	int8	0	becomes nonzero when a parameter change was performed that has not (yet) been saved in non-volatile memory.