

Basic Small Messages Protocol (BSMP)

Version 2.00
February 12nd, 2014

Bruno Martins
bruno.martins@lnls.br
Controls Group

Revision History

Revision	Changes
2.00 02/12/2014	<ul style="list-style-type: none">• Change the name of the protocol.<ul style="list-style-type: none">◦ Was: Sirius Low Level Protocol (SLLP).
2.00-rc2 01/24/2014	<ul style="list-style-type: none">• Add the possibility of a Curve block contain less than SBLOCK bytes of data.
2.00-rc1 01/23/2014	<ul style="list-style-type: none">• First release candidate.

Index

1 Introduction.....	1
2 Transport Layer.....	2
2.1 Addressing.....	2
2.2 Multicast groups.....	2
3 Application Layer.....	3
3.1 Concepts.....	3
3.1.1 Network, Message, Command, Master and Node (or Slave).....	3
3.1.2 Protocol Type.....	3
3.1.3 Message Structure.....	3
3.2 Entities.....	4
3.2.1 Variable.....	4
3.2.2 Group of Variables.....	5
3.2.3 Curve.....	5
3.2.4 Function.....	6
3.3 Protocol Commands.....	6
3.4 (0x0_) Query commands.....	8
3.4.1 (0x00) Query Protocol Version.....	8
3.4.2 (0x01) Protocol Version.....	8
3.4.3 (0x02) Query List of Variables.....	8
3.4.4 (0x03) List of Variables.....	9
3.4.5 (0x04) Query List of Group of Variables.....	9
3.4.6 (0x05) List of Group of Variables.....	10
3.4.7 (0x06) Query Group of Variables.....	10
3.4.8 (0x07) Group of Variables.....	11
3.4.9 (0x08) Query List of Curves.....	11
3.4.10 (0x09) List of Curves.....	12
3.4.11 (0x0A) Query Curve Checksum.....	12
3.4.12 (0x0B) Curve Checksum.....	13
3.4.13 (0x0C) Query List of Functions.....	13
3.4.14 (0x0D) List of Functions.....	14
3.5 (0x1_) Reading Commands.....	15
3.5.1 (0x10) Read Variable.....	15
3.5.2 (0x11) Variable's Value.....	15
3.5.3 (0x12) Read Group of Variables.....	16
3.5.4 (0x13) Group of Variables' Values.....	16
3.6 (0x2_) Writing Commands.....	17
3.6.1 (0x20) Write Variable.....	17
3.6.2 (0x22) Write Group of Variables.....	18
3.6.3 (0x24) Binary Operation in a Variable.....	19
3.6.4 (0x26) Binary Operation in a Group.....	20
3.6.5 (0x28) Write and Read Variables.....	21
3.7 (0x30) Group of Variables' Manipulation Commands.....	22
3.7.1 (0x30) Create Group of Variables.....	22
3.7.2 (0x32) Remove all Groups of Variables.....	22
3.8 (0x4_) Curve Transfer Commands.....	23

3.8.1 (0x40) Request Curve Block.....	23
3.8.2 (0x41) Curve Block.....	24
3.8.3 (0x42) Recalculate Curve Checksum.....	25
3.9 (0x5_) Function Execution Commands.....	26
3.9.1 (0x50) Execute Function.....	26
3.9.2 (0x51) Function Return.....	27
3.9.3 (0x53) Function Error.....	27
3.10 (0xE_) Error Commands.....	28
3.10.1 (0xE0) OK.....	28
3.10.2 (0xE1) Malformed Message.....	28
3.10.3 (0xE2) Operation not supported.....	28
3.10.4 (0xE3) Invalid ID.....	28
3.10.5 (0xE4) Invalid Value.....	29
3.10.6 (0xE5) Invalid Payload Size.....	29
3.10.7 (0xE6) Read-Only.....	29
3.10.8 (0xE7) Insufficient Memory.....	29

1 Introduction

In order to standardize all communications between equipment developed for the Sirius project and connected to the Controls Network, a common communication protocol was created. The protocol was named Basic Small Messages Protocol – BSMP. This protocol describes two layers: transport and application. Those layers are independent from one another.

The Sirius' Controls Network is composed of Ethernet and RS485 networks. The devices residing in the lowest levels of the hierarchy will communicate over RS485 with Single Board Computers (SBC). The SBC's, in turn, will communicate with the computers in the higher level of the hierarchy through Ethernet, therefore having a role of a gateway between Ethernet and RS485 networks.

All RS485 devices developed for the Sirius accelerator that will connect to the Controls Network **must** use both layers described in this document. All Ethernet devices **must** use UDP/IP or TCP/IP and the application layer described in this document in order to communicate with a SBC.

2 Transport Layer

The transmission unit of the Transport Layer is the **packet**. Each packet holds a **message**. The Transport Layer **requires** all data in a packet to be in binary, which means that no byte value should have special meaning. The end of a packet is indicated with a silence on the Serial line with duration of 2 bytes. For example, in a 10 Mbps Serial network the silence after a packet should last 1.6 μ s. This layer does not impose a size limit on the message inside a packet.

Addressing	Message								Checksum
DESTINATION									CHECKSUM

Table 1- Structure of a Transport Layer packet

Packets in the Serial network have a well defined format. The first byte is used for addressing and should have a valid address of a device or group of devices. If the packet is from a node to a master, DESTINATION **must** be 0. The last byte of the packet **must** be the CHECKSUM. This structure is shown in the Table 1. The CHECKSUM is the sum of all Addressing and Message bytes, in 2's complement. Therefore, the sum of all bytes of a valid packet **must** be equal to zero (8-bit sum). Each packet carries only one message.

2.1 Addressing

The devices in a serial network **must** be given an address between 0 and 31, inclusive. This range restricts the existence of only thirty two devices in a single serial network. The device with address 0 is the **master** of the network, the other devices being **nodes** (or **slaves**). Every network must have **exactly** one master and **at least** one node.

2.2 Multicast groups

Nodes can be part of Multicast Groups. A Multicast Group can have an address between 248 and 255, which allows the existence of up to eight of such groups. The Multicast Group with address 255 is a special Group called Broadcast. All devices in a serial network **must** be in the Broadcast Group. A node can belong to more than one Multicast Group. The address ranges of serial networks are specified in the Table 2.

Address range	0	1 to 31	32 to 247	248 to 254	255
Used for	Master	Node	Reserved	<i>Multicast Group</i>	<i>Broadcast</i>

Table 2- Addresses on the Serial network

3 Application Layer

The Application Layer defines all the messages that can be exchanged between devices and the actions that must be taken in response to a message.

3.1 Concepts

This section defines the concepts that will be used to describe the Application Layer.

3.1.1 Network, Message, Command, Master and Node (or Slave)

Devices connected with BSMP are components of a **network**. **Network** components communicate exchanging **messages**. Each **message** holds a **command**, that can be either a request or a response.

Network components can perform one of two roles: **master** or **node** (also called **slave**). There must be **exactly** one **master** per **network**. The amount of **nodes** in a **network** is not limited by the Application Layer.

3.1.2 Protocol Type

The **protocol** here described is a token protocol. Only the device with the token can initiate a transmission in the network. In this protocol the token is implicit. Therefore, all communications are initiated by the master. Once the master sends a direct message to a node, it is understood that the node has the token until it answers the master, returning the token. The protocol is stateless: each pair request/response is a whole independent transaction.

The protocol is **byte oriented**, which means that the smaller unit of the message is one byte. Multi-byte values are **big endian**.

If the Application Layer is used together with the Transport Layer, two restrictions apply:

1. Multicast packets **must not** be answered.
2. The master must establish a *timeout* to avoid the loss of the token.

3.1.3 Message Structure

A protocol message must have at least three bytes, which are part of its header: COMMAND and LENGTH (with two bytes). The COMMAND field specifies which command should be executed by the node or the response of the execution of a command. All the commands of the protocol are described in the section Protocol Commands. The field LENGTH holds the length of the Payload of the message, in **big endian**. If the message has no payload, both LENGTH bytes must be 0. The structure of the message is depicted in Table 3. The field LENGTH can assume values between 0 and 65535.

Header			Payload							
COMMAND	LENGTH	LENGTH								

Table 3- Structure of a BSMP's message

3.2 Entities

Devices communicate with each other in order to manipulate Entities of the protocol. Entities fall on one of four categories: Variable, Group of Variables, Curve and Function. The maximum amount of each entity in a node are listed in Table 4. Every Entity is identified by an ID. An ID must be unique within a category. IDs must start with 0 and be continuous within each category. For instance, the Variables of a node with 4 Variables must have the IDs 0, 1, 2 and 3. If the same node also has 8 Curves, their IDs must be 0, 1, 2, 3, 4, 5, 6 and 7.

Entity	Maximum Amount	Properties
Variable	128	ID, TYPE, SIZE
Group of Variables	8	ID, TYPE, SIZE
Curve	128	ID, TYPE, SIZE, CHECKSUM
Function	128	ID, INPUT, OUTPUT

Table 4- Amounts and properties of the protocol's Entities

3.2.1 Variable

The Variable is the central Entity of the protocol. Each node has a number of Variables. Each Variable has a value that can be read from and, for writable Variables, written to. The meaning of each Variable must be specified by the device developer. Each Variable has one value and three properties, listed in Table 5. It's important to stress that a writable Variable **can** also be read (in which case the read value would be the last written value). However, to write in a read-only Variable **must not** be allowed.

Property	Description
ID	Unique number that identifies the Variable
TYPE	The Variable may be read-only (TYPE 0) or writable (TYPE 1)
SIZE	The length of the Variable's value, between 1 and 128

Table 5- Properties of a Variable

3.2.2 Group of Variables

It is possible to create Groups of Variables so that some sets of Variables can be read from or written to with a single command. Each Group of Variables has three properties and a list of Variables in the Group, according to Table 6. A Variable can belong to more than one Group. There **must** exist, at all times, at least three Groups of Variables, presented in Table 7. Those Groups are called Standard Groups, which **must not** be deleted. A writable Group **must** contain **only** writable Variables. However a read-only Group **may** contain both writable and read-only Variables.

Property	Description
ID	Unique number that identifies the Group
TYPE	The Group may be read-only (TYPE 0) or writable (TYPE 1)
SIZE	The amount of Variables in the Group, between 1 and 128

Table 6- Properties of a Group of Variables

ID	TYPE	Group's Variables
0	0	All Variables of a node
1	0	All read-only Variables of a node
2	1	All writable Variables of a node

Table 7- Standard Groups of Variables

3.2.3 Curve

A Curve is a long sequence of bytes, which may or may not be related to each other. Values of a Curve can be transmitted in either direction (master → node or node → master). A Curve has five properties, listed in Table 8.

Property	Description
ID	Unique number that identifies the Curve
TYPE	The Curve may be read-only (TYPE 0) or writable (TYPE 1)
SBLOCK	The size of an individual block, between 1 and 65520
NBLOCKS	The amount of blocks in the Curve, between 1 and 65536
CHECKSUM	MD5 hash of all values of a Curve

Table 8- Properties of a Curve

The TYPE property indicates if the values of a Curve can (1) or cannot (0) be written to. The Curve's size is limited to 65536 (2^{16}) blocks, each block being, at most, 65520 bytes long, which gives a total of 4095 MiB per Curve. The NBLCOKS field is the number of blocks of a Curve. The SBLOCK field is the maximum size of a single block. A Curve may have a CHECKSUM associated with it, which must be calculated using the MD5 algorithm. Therefore, the length of the CHECKSUM field is 16 bytes.

3.2.4 Function

A Function is a kind of a Remote Procedure Call – RPC. A Function can receive between zero and fifteen bytes as input and return between zero and fifteen bytes as successful output or one byte as error output. A successful return is signaled by the command (0x51) Function Return; an error in the execution of the Function is indicated by the command (0x53) Function Error. The meaning of the input bytes, the output bytes and the error codes is specific to each Function and must be provided by the developer of the device. The properties of a Function are described in Table 9.

Property	Description
ID	Unique number that identifies the Function
ENTRADA	Amount of bytes taken as input (between 0 and 15)
SAIDA	Amount of bytes returned as output (between 0 and 15)

Table 9- Properties of a Function

3.3 Protocol Commands

All the codes accepted in the field COMMAND of the messages and their meaning and structure are described in this section. The commands are divided in classes, being grouped by their semantic likeness. Each command code is consisted of one byte, being the most significant nibble the indicative of the command's class. In general, the convention for the command's codes is that even codes are for commands from the master to the slave and odd commands from the slave to the master. The exceptions are the error codes (section (0xE_) Error Commands), which are always from the slave to the master, and the (0x41) Curve Block command, which can be sent both ways.

If a node happens to receive a message in which the number of bytes indicated in the LENGTH field differs from the actual length of the Payload, it will return the error (0xE1) Malformed Message. If a node receive a command that it doesn't know how to perform, it will return the error (0xE2) Operation not supported. If the number of bytes in the payload of a command differs from the number of bytes expected for that specific command, the error (0xE5) Invalid Payload Size will be returned.

A summary of all commands of the protocol and their payloads is given in Table 10. Detailed descriptions are given in the following sections.

(Code) Command	Direction	Payload
(0x00) Query Protocol Version	M → N	
(0x01) Protocol Version	M ← N	[Version, Subversion, Revision]
(0x02) Query List of Variables	M → N	
(0x03) List of Variables	M ← N	[Type Size] * (# of Vars)
(0x04) Query List of Group of Variables	M → N	
(0x05) List of Group of Variables	M ← N	[Type Size] * (# of Groups)
(0x06) Query Group of Variables	M → N	[Group ID]
(0x07) Group of Variables	M ← N	[Var ID] * (# of Vars in the Group)
(0x08) Query List of Curves	M → N	
(0x09) List of Curves	M ← N	[Type,NBlocks(2 bytes),SBlock(2 bytes)]*(# curves)
(0x0A) Query Curve Checksum	M → N	[Curve ID]
(0x0B) Curve Checksum	M ← N	16 bytes (MD5 Checksum)
(0x0C) Query List of Functions	M → N	
(0x0D) List of Functions	M ← N	[Input Output] * (# of Functions)
(0x10) Read Variable	M → N	[Var ID]
(0x11) Variable's Value	M ← N	[Value]
(0x12) Read Group of Variables	M → N	[Group ID]
(0x13) Group of Variables' Values	M ← N	[Value] * (# of Vars in the Group)
(0x20) Write Variable	M → N	[Var ID, Value]
(0x22) Write Group of Variables	M → N	[Group ID], [Value]*(# of Vars in the Group)
(0x24) Binary Operation in a Variable	M → N	[Var ID, Operation,Mask]
(0x26) Binary Operation in a Group	M → N	[Group ID, Operation],[Mask]*(# of Vars in the Group)
(0x28) Write and Read Variables	M → N	[Var ID (to be written), Var ID (to be read), Value]
(0x30) Create Group of Variables	M → N	[Var ID] * (# of desired Vars)
(0x32) Remove all Groups of Variables	M → N	
(0x40) Request Curve Block	M → N	[Curve ID, block offset (2 bytes)]
(0x41) Curve Block	M ← N	[Curve ID, block offset (2 bytes), Data (up to Sblock bytes)]*
(0x42) Recalculate Curve Checksum	M → N	[Curve ID]
(0x50) Execute Function	M → N	[Function ID, Input (between 0 and 15 bytes)]
(0x51) Function Return	M ← N	[Output (between 0 and 15 bytes)]
(0x53) Function Error	M ← N	[Error code]
Errors: (0xE0) OK, (0xE1) Malformed Message, (0xE2) Operation not supported, (0xE3) Invalid ID, (0xE4) Invalid Value, (0xE5) Invalid Payload Size, (0xE6) Read-Only, (0xE7) Insufficient Memory		

Table 10- Summary of the commands of the protocol

3.4 (0x0_) Query commands

3.4.1 (0x00) Query Protocol Version

Direction	Payload Size	Expected Answer
Master → Node	0	(0x01) Protocol Version
Description Request the Version of the protocol supported by the Node.		

3.4.2 (0x01) Protocol Version

Direction	Payload Size	Expected Answer												
Master ← Node	3	-												
Description The first byte is the Version field. The second one, the Subversion field. The third one, the Revision. A version string can be constructed with the format “Version.Subversion.Revision”. Implementations of the same Version of the Protocol must be compatible. Greater Protocol's Subversion numbers mean that new commands were added. The Revision field is used to identify different implementations.														
Structure <table><tr><th colspan="3">Payload</th></tr><tr><td>Version</td><td>Subversion</td><td>Revision</td></tr></table>			Payload			Version	Subversion	Revision						
Payload														
Version	Subversion	Revision												
Example <table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>01</td><td>00</td><td>03</td><td>02</td><td>00</td><td>00</td></tr></table> Answer with Version 2, Subversion 0 and Revision 0: “2.00.0”.			Header			Payload			01	00	03	02	00	00
Header			Payload											
01	00	03	02	00	00									

3.4.3 (0x02) Query List of Variables

Direction	Payload Size	Expected Answer
Master → Node	0	(0x03) List of Variables
Description Request the list of Variables in the node.		

3.4.4 (0x03) List of Variables

Direction	Payload Size	Expected Answer																	
Master ← Node	(number of Variables in the node)	-																	
Description																			
Contains a list with the TYPE and the SIZE of all node's Variables.																			
Structure																			
<table><tr><th colspan="3">Payload</th></tr><tr><td>First Variable</td><td></td><td>Last Variable</td></tr><tr><td>TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)</td><td>...</td><td>TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)</td></tr></table>			Payload			First Variable		Last Variable	TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)	...	TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)								
Payload																			
First Variable		Last Variable																	
TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)	...	TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)																	
For each Variable is returned one byte of information. The Variables are in ascending order of their IDs. The first Variable has the ID 0. The most significant bit of each byte indicates if the Variable is read-only (bit = 0) or writable (bit = 1). The remaining seven bits contain the SIZE of the Variable. If SIZE is 0, the Variable is 128 bytes wide.																			
Example																			
<table><tr><th colspan="3">Header</th><th colspan="5">Payload</th></tr><tr><td>03</td><td>00</td><td>06</td><td>03</td><td>03</td><td>83</td><td>83</td><td>01</td><td>81</td></tr></table>			Header			Payload					03	00	06	03	03	83	83	01	81
Header			Payload																
03	00	06	03	03	83	83	01	81											
Two read-only Variables of size 3, two writable Variables of size 3, one read-only Variable of size 1, one writable Variable of size 1.																			

3.4.5 (0x04) Query List of Group of Variables

Direction	Payload Size	Expected Answer
Master → Node	0	(0x05) List of Group of Variables
Description Request the node to return a list describing all of his Groups of Variables.		

3.4.6 (0x05) List of Group of Variables

Direction	Payload Size	Expected Answer												
Master ← Node	(number of Groups in the node)	-												
Description														
Contains a list with the TYPE and the SIZE of all node's Groups.														
Structure														
<table><tr><th colspan="3">Payload</th></tr><tr><td>First Group</td><td rowspan="2">...</td><td>Last Group</td></tr><tr><td>TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)</td><td>TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)</td></tr></table>			Payload			First Group	...	Last Group	TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)	TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)				
Payload														
First Group	...	Last Group												
TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)		TYPE (1 <i>bit</i>) SIZE (7 <i>bits</i>)												
One byte of information is returned for each Group in the node. The Groups are in their ascending ID order. The first Group has the ID 0. The most significant bit of each byte indicates the TYPE of the Group (0 for read-only, 1 for writable). The seven remaining bits contain the SIZE of the Group. If SIZE is 0, the Group has 128 Variables.														
Example														
<table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>05</td><td>00</td><td>03</td><td>0A</td><td>05</td><td>85</td></tr></table> <div>Three Groups. The first one is read-only and has 10 Variables. The second one is also read-only and has 5 Variables. The last one is writable and has 5 Variables.</div>			Header			Payload			05	00	03	0A	05	85
Header			Payload											
05	00	03	0A	05	85									

3.4.7 (0x06) Query Group of Variables

Direction	Payload Size	Expected Answer					
Master → Node	1	(0x07) Group of Variables					
Description							
Request the node to return the list of Variables of the specified Group.							
Structure							
<table><tr><th>Payload</th><td rowspan="2">The Payload must contain the ID of the Group to be queried.</td></tr><tr><td>Group ID</td></tr></table>			Payload	The Payload must contain the ID of the Group to be queried.	Group ID		
Payload	The Payload must contain the ID of the Group to be queried.						
Group ID							
Example							
<table><tr><th>Header</th><th>Payload</th><td rowspan="2">Query the Group with ID 2.</td></tr><tr><td>06</td><td>00 01</td></tr></table>			Header	Payload	Query the Group with ID 2.	06	00 01
Header	Payload	Query the Group with ID 2.					
06	00 01						
Possible Errors							
(0xE3) Invalid ID: There is no Group with the specified ID.							

3.4.8 (0x07) Group of Variables

Direction	Payload Size	Expected Answer															
Master ← Node	(number of Variables in the Group)	-															
Description																	
Contains the list of all Variables in the Group.																	
Structure																	
<table><tr><th colspan="3">Payload</th></tr><tr><td>First Variable</td><td>...</td><td>Last Variable</td></tr><tr><td>ID</td><td></td><td>ID</td></tr></table>		Payload			First Variable	...	Last Variable	ID		ID	The Variable's IDs are listed in their ascending order.						
Payload																	
First Variable	...	Last Variable															
ID		ID															
Example																	
<table><tr><th colspan="3">Header</th><th colspan="4">Payload</th></tr><tr><td>07</td><td>00</td><td>05</td><td>04</td><td>05</td><td>06</td><td>07</td><td>09</td></tr></table>		Header			Payload				07	00	05	04	05	06	07	09	Group of 5 Variables with IDs 4, 5, 6, 7 e 9.
Header			Payload														
07	00	05	04	05	06	07	09										

3.4.9 (0x08) Query List of Curves

Direction	Payload Size	Expected Answer
Master → Node	0	(0x09) List of Curves
Description Request the list of Curves in the node.		

3.4.10 (0x09) List of Curves

Direction	Payload Size	Expected Answer																								
Master ← Node	5*(number of Curves in the node)	-																								
Description																										
Contains a list with TYPE, SBLOCK and NBLOCKS fields of all node's Curves.																										
Structure																										
Payload																										
First Curve					...	Last Curve																				
TYPE	SBLOCK		NBLOCKS			TYPE	SBLOCK		NBLOCKS																	
	most sig.	least sig.	most sig.	least sig.			most sig.	least sig.	most sig.	least sig.																
<p>The Curves are listed in the ascending order of their IDs. The first Curve has the ID 0. There are five bytes for each Curve. The first byte is the TYPE of the Curve (0 for read-only, 1 for writable). Second and third bytes contain the size of a block (SBLOCK). Fourth and fifth bytes contain the number of blocks (NBLOCKS). If NBLOCKS is 0, the Curve has 65536 blocks.</p>																										
Example																										
<table><tr><th colspan="3">Header</th><th colspan="5">Payload</th></tr><tr><td>09</td><td>00</td><td>03</td><td>00</td><td>40</td><td>00</td><td>02</td><td>00</td></tr></table>								Header			Payload					09	00	03	00	40	00	02	00	<p>List with only one Curve. It is read-only (00) and has 512 (200h) 16384-byte (4000h) blocks.</p>		
Header			Payload																							
09	00	03	00	40	00	02	00																			

3.4.11 (0x0A) Query Curve Checksum

Direction	Payload Size	Expected Answer									
Master → Node	1	(0x0B) Curve Checksum									
Description Request the CHECKSUM of a Curve in the node.											
Structure <table><tr><th>Payload</th><td rowspan="2">The ID of the Curve to be queried is specified.</td></tr><tr><td>Curve ID</td></tr></table>			Payload	The ID of the Curve to be queried is specified.	Curve ID						
Payload	The ID of the Curve to be queried is specified.										
Curve ID											
Example <table><tr><th colspan="3">Header</th><th>Payload</th><td rowspan="2">Query the CHECKSUM of the Curve with ID 2.</td></tr><tr><td>0A</td><td>00</td><td>01</td><td>02</td></tr></table>			Header			Payload	Query the CHECKSUM of the Curve with ID 2.	0A	00	01	02
Header			Payload	Query the CHECKSUM of the Curve with ID 2.							
0A	00	01	02								
Possible Errors (0xE3) Invalid ID: There's no Curve with the ID specified.											

3.4.12 (0x0B) Curve Checksum

Direction	Payload Size	Expected Answer																																																	
Master ← Node	16	-																																																	
Description																																																			
Contains the CHECKSUM of a Curve.																																																			
Structure																																																			
<table><tr><td colspan="16">Payload</td></tr><tr><td colspan="16">MD5 Checksum</td></tr><tr><td>most significant</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>least significant</td></tr></table>			Payload																MD5 Checksum																most significant																least significant
Payload																																																			
MD5 Checksum																																																			
most significant																least significant																																			
The 16 bytes of the MD5 CHECKSUM are returned from the most to the least significant byte.																																																			
Example																																																			
<table><tr><td colspan="3">Header</td><td colspan="17">Payload</td></tr><tr><td>0B</td><td>00</td><td>10</td><td>01</td><td>23</td><td>45</td><td>67</td><td>89</td><td>AB</td><td>CD</td><td>EF</td><td>FE</td><td>DC</td><td>BA</td><td>98</td><td>76</td><td>54</td><td>32</td><td>10</td></tr></table>			Header			Payload																	0B	00	10	01	23	45	67	89	AB	CD	EF	FE	DC	BA	98	76	54	32	10										
Header			Payload																																																
0B	00	10	01	23	45	67	89	AB	CD	EF	FE	DC	BA	98	76	54	32	10																																	
Curve with CHECKSUM 0123456789abcdeffedcba9876543210.																																																			

3.4.13 (0x0C) Query List of Functions

Direction	Payload Size	Expected Answer
Master ← Node	0	(0x0D) List of Functions
Description Request the list of Functions in the node.		

3.4.14 (0x0D) List of Functions

Direction	Payload Size	Expected answer												
Master ← Node	3*(number of Functions in the node)	-												
Description														
Contains a list with the INPUT and the OUTPUT of all node's Functions.														
Structure														
<table><tr><th colspan="3">Payload</th></tr><tr><td>First Function</td><td rowspan="2">...</td><td>Last Function</td></tr><tr><td>INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)</td><td>INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)</td></tr></table>			Payload			First Function	...	Last Function	INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)	INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)				
Payload														
First Function	...	Last Function												
INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)		INPUT (4 <i>bits</i>) OUTPUT (4 <i>bits</i>)												
The list of Functions are listed in the ascending order of their IDs. The first Function has the ID 0. There is one byte for each Function. The most significant nibble of the byte contains the INPUT length of the Function (between 0 and 15). Likewise, the least significant nibble contains the OUTPUT length of the Function (also between 0 and 15).														
Example														
<table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>0D</td><td>00</td><td>03</td><td>F0</td><td>0F</td><td>22</td></tr></table>			Header			Payload			0D	00	03	F0	0F	22
Header			Payload											
0D	00	03	F0	0F	22									
Three Functions. The Function with ID 0 takes 15 bytes as input and returns 0 bytes as output. The Function with ID 1 returns 15 bytes as output and takes no input. The Function with ID 2 takes 2 bytes as input and returns 2 bytes as output.														

3.5 (0x1_) Reading Commands

3.5.1 (0x10) Read Variable

Direction	Payload Size	Expected answer								
Master → Node	1	(0x11) Variable's Value								
Description										
Request the VALUE of a Variable.										
Structure										
<table><tr><th>Payload</th></tr><tr><td>Variable ID</td></tr></table>		Payload	Variable ID	The payload contains the ID of the Variable to be read.						
Payload										
Variable ID										
Example										
<table><tr><th colspan="3">Header</th><th>Payload</th></tr><tr><td>10</td><td>00</td><td>01</td><td>03</td></tr></table>		Header			Payload	10	00	01	03	Request to read the Variable with ID 3.
Header			Payload							
10	00	01	03							
Possible Errors										
(0xE3) Invalid ID: There's no Variable with the ID specified.										

3.5.2 (0x11) Variable's Value

Direction	Payload Size	Expected answer												
Master ← Node	(SIZE of the Variable)	-												
Description														
Contains the VALUE of the Variable. The meaning of the VALUE of a Variable must be specified by the developer of the device.														
Structure														
<table><tr><th colspan="3">Payload</th></tr><tr><td colspan="3">Variable's VALUE</td></tr><tr><td>First byte</td><td>...</td><td>Last byte</td></tr></table>			Payload			Variable's VALUE			First byte	...	Last byte			
Payload														
Variable's VALUE														
First byte	...	Last byte												
The VALUE of the Variable is given byte by byte.														
Example														
<table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>11</td><td>00</td><td>03</td><td>03</td><td>FF</td><td>FF</td></tr></table> Variable with the VALUE 03h FFh FFh.			Header			Payload			11	00	03	03	FF	FF
Header			Payload											
11	00	03	03	FF	FF									

3.5.3 (0x12) Read Group of Variables

Direction	Payload Size	Expected answer				
Master → Node	1	(0x13) Group of Variables' Values				
Description						
Request the VALUE of all Variables in a Group.						
Structure						
<table><tr><th>Payload</th></tr><tr><td>Group ID</td></tr></table>		Payload	Group ID	The payload contains the ID of the Group to be read.		
Payload						
Group ID						
Example						
<table><tr><th>Header</th><th>Payload</th></tr><tr><td>120001</td><td>01</td></tr></table>		Header	Payload	120001	01	Request the VALUEs of the Variables in the Group with ID 1.
Header	Payload					
120001	01					
Possible Errors						
(0xE3) Invalid ID: There's no Group with the specified ID.						

3.5.4 (0x13) Group of Variables' Values

Direction	Payload Size	Expected answer																																	
Master ← Node	(sum of SIZES of the Variables of a Group)	-																																	
Description																																			
Contains the VALUEs of all the Variables in a Group.																																			
Structure																																			
<table><tr><th colspan="9">Payload</th></tr><tr><td colspan="6">VALUE of the First Variable</td><td></td><td colspan="2">VALUE of the Last Variable</td></tr><tr><td>First byte</td><td>...</td><td colspan="3">Last byte</td><td></td><td>First <i>byte</i></td><td>...</td><td>Last byte</td></tr></table>			Payload									VALUE of the First Variable							VALUE of the Last Variable		First byte	...	Last byte				First <i>byte</i>	...	Last byte						
Payload																																			
VALUE of the First Variable							VALUE of the Last Variable																												
First byte	...	Last byte				First <i>byte</i>	...	Last byte																											
The VALUEs of the Variables of the Group are listed in the ascending order of the Variables' IDs.																																			
Example																																			
<table><tr><th colspan="3">Header</th><th colspan="14">Payload</th></tr><tr><td>13</td><td>00</td><td>0C</td><td>03</td><td>FF</td><td>FF</td><td>03</td><td>FF</td><td>FF</td><td>03</td><td>FF</td><td>FF</td><td>03</td><td>FF</td><td>FF</td><td>AA</td></tr></table>			Header			Payload														13	00	0C	03	FF	FF	03	FF	FF	03	FF	FF	03	FF	FF	AA
Header			Payload																																
13	00	0C	03	FF	FF	03	FF	FF	03	FF	FF	03	FF	FF	AA																				
Sequence of the VALUEs of the Variables of a Group. It's possible to interpret those VALUEs once it's known which Variables are in the Group (with the command (0x06) Query Group of Variables).																																			

3.6 (0x2_) Writing Commands

3.6.1 (0x20) Write Variable

Direction	Payload Size	Expected answer															
Master → Node	1 + (SIZE of the Variable)	(0xE0) OK															
Description																	
Writes in the VALUE of a Variable. The Variable must be writable.																	
Structure																	
<table><tr><th colspan="4">Payload</th></tr><tr><td rowspan="2">Variable ID</td><td colspan="3">Variable's VALUE</td></tr><tr><td>First byte</td><td>...</td><td>Last byte</td></tr></table>			Payload				Variable ID	Variable's VALUE			First byte	...	Last byte				
Payload																	
Variable ID	Variable's VALUE																
	First byte	...	Last byte														
The payload contains the ID of the Variable followed by the sequence of bytes to be written in the Variable's VALUE.																	
Example																	
<table><tr><th colspan="3">Header</th><th colspan="4">Payload</th><td rowspan="2">Request to write 01h BBh BBh in the VALUE of the Variable with ID 4.</td></tr><tr><td>20</td><td>00</td><td>04</td><td>04</td><td>01</td><td>BB</td><td>BB</td></tr></table>			Header			Payload				Request to write 01h BBh BBh in the VALUE of the Variable with ID 4.	20	00	04	04	01	BB	BB
Header			Payload				Request to write 01h BBh BBh in the VALUE of the Variable with ID 4.										
20	00	04	04	01	BB	BB											
Possible Errors																	
(0xE3) Invalid ID: There's no Variable with the specified ID.																	
(0xE6) Read-Only: Variable can't be written (its TYPE is read-only).																	

3.6.2 (0x22) Write Group of Variables

Direction	Payload Size	Expected answer														
Master → Node	1+(sum of the SIZES of the Variable's of a Group)	(0xE0) OK														
Description																
Contains values to be written to Variables in a specific Group. The Group must be writable.																
Structure																
Payload																
Group ID	VALUE of the first Variable				VALUE of the last Variable											
	First byte	...	Last byte		...	First byte	...	Last byte								
The payload contains the ID of the Group followed by the bytes to be written in the VALUE field of all Group's Variables.																
Example																
Header		Payload														
22	00	0E	02	01	BB	BB	01	BB	BB	01	BB	BB	01	BB	BB	CC
Sequence of VALUES to be written in the Variables of the Group with ID 2. It's possible to interpret this particular sequence of values knowing the Variables of the Group (with the command (0x06) Query Group of Variables).																
Possible Errors																
(0xE3) Invalid ID:		There's no Group with the specified ID.														
(0xE6) Read-Only:		Group couldn't be written (it's TYPE is read-only).														

3.6.3 (0x24) Binary Operation in a Variable

Direction	Payload Size	Expected answer																					
Master → Node	2+(SIZE of the Variable)	(0xE0) OK																					
<div>Description</div> <p>Perform a binary operation in the VALUE of a Variable with a specified mask. The available operations are listed in Table 11. The Variable must be writable.</p> <table><tr><th>Code</th><th>Operation</th><th>Description</th></tr><tr><td>0x53 ('S')</td><td>SET</td><td>'Turn on' (make 1) the bits specified in the mask.</td></tr><tr><td>0x43 ('C')</td><td>CLEAR</td><td>'Turn off' (make 0) the bits specified in the mask.</td></tr><tr><td>0x54 ('T')</td><td>TOGGLE</td><td>Invert the bits specified in the mask.</td></tr><tr><td>0x41 ('A')</td><td>AND</td><td>Perform a logical AND between the Variable's VALUE and the mask.</td></tr><tr><td>0x4F ('O')</td><td>OR</td><td>Perform a logical OR between the Variable's VALUE and the mask.</td></tr><tr><td>0x58 ('X')</td><td>XOR</td><td>Perform a logical XOR between the Variable's VALUE and the mask.</td></tr></table> <div>Table 11- Binary operations</div>			Code	Operation	Description	0x53 ('S')	SET	'Turn on' (make 1) the bits specified in the mask.	0x43 ('C')	CLEAR	'Turn off' (make 0) the bits specified in the mask.	0x54 ('T')	TOGGLE	Invert the bits specified in the mask.	0x41 ('A')	AND	Perform a logical AND between the Variable's VALUE and the mask.	0x4F ('O')	OR	Perform a logical OR between the Variable's VALUE and the mask.	0x58 ('X')	XOR	Perform a logical XOR between the Variable's VALUE and the mask.
Code	Operation	Description																					
0x53 ('S')	SET	'Turn on' (make 1) the bits specified in the mask.																					
0x43 ('C')	CLEAR	'Turn off' (make 0) the bits specified in the mask.																					
0x54 ('T')	TOGGLE	Invert the bits specified in the mask.																					
0x41 ('A')	AND	Perform a logical AND between the Variable's VALUE and the mask.																					
0x4F ('O')	OR	Perform a logical OR between the Variable's VALUE and the mask.																					
0x58 ('X')	XOR	Perform a logical XOR between the Variable's VALUE and the mask.																					
<div>Structure</div> <table><tr><th colspan="5">Payload</th></tr><tr><th rowspan="2">Variable ID</th><th rowspan="2">Operation Code</th><th colspan="3">Mask</th></tr><tr><th>First byte</th><th>...</th><th>Last byte</th></tr></table> <p>The payload contains the ID of the Variable and the code of the operation to be performed, followed by the bytes of the mask.</p>			Payload					Variable ID	Operation Code	Mask			First byte	...	Last byte								
Payload																							
Variable ID	Operation Code	Mask																					
		First byte	...	Last byte																			
<div>Example</div> <table><tr><th>Header</th><th>Payload</th><td rowspan="2">Perform a SET operation (53h) in the Variable with ID 09h with the mask F0h, which will cause the most significant nibble of the Variable's VALUE to have all 1's.</td></tr><tr><td>24 00 03</td><td>09 53 F0</td></tr></table>			Header	Payload	Perform a SET operation (53h) in the Variable with ID 09h with the mask F0h, which will cause the most significant nibble of the Variable's VALUE to have all 1's.	24 00 03	09 53 F0																
Header	Payload	Perform a SET operation (53h) in the Variable with ID 09h with the mask F0h, which will cause the most significant nibble of the Variable's VALUE to have all 1's.																					
24 00 03	09 53 F0																						
<div>Possible Errors</div> <div>(0xE2) Operation not supported:</div> <div>(0xE3) Invalid ID:</div> <div>(0xE6) Read-Only:</div> <div>The binary operation requested isn't valid.</div> <div>There's no Variable with the specified ID.</div> <div>Variable couldn't be written (its TYPE is read-only).</div>																							

3.6.4 (0x26) Binary Operation in a Group

Direction	Payload Size	Expected answer																																
Master → Node	1+(sum of the SIZES of the Variable's of a Group)	(0xE0) OK																																
Description																																		
Perform a binary operation in the VALUES of the Variables of a Group with a specified mask. The available operations are listed in Table 11. The Group must be writable.																																		
Structure																																		
<table><tr><th colspan="10">Payload</th></tr><tr><th rowspan="2">Group ID</th><th rowspan="2">Operation Code</th><th colspan="3">First Mask</th><th rowspan="2">...</th><th colspan="3">Last Mask</th></tr><tr><th>First byte</th><th>...</th><th>Last byte</th><th>First byte</th><th>...</th><th>Last byte</th></tr></table>										Payload										Group ID	Operation Code	First Mask			...	Last Mask			First byte	...	Last byte	First byte	...	Last byte
Payload																																		
Group ID	Operation Code	First Mask			...	Last Mask																												
		First byte	...	Last byte		First byte	...	Last byte																										
The payload contains the ID of the Group and the code of the operation to be performed, followed by the bytes of the masks.																																		
Example																																		
<table><tr><th colspan="3">Header</th><th colspan="4">Payload</th></tr><tr><td>26</td><td>00</td><td>05</td><td>02</td><td>4F</td><td>55</td><td>55</td><td>55</td></tr></table> <div>Perform an OR operation (4Fh) with the mask 55h in all bytes of all Variables' VALUES in the Group with ID 02h.</div>										Header			Payload				26	00	05	02	4F	55	55	55										
Header			Payload																															
26	00	05	02	4F	55	55	55																											
Possible Errors																																		
<div>(0xE2) Operation not supported: The requested binary operation is invalid.</div> <div>(0xE3) Invalid ID: There's no Group with the specified ID.</div> <div>(0xE6) Read-Only: Group couldn't be written (its TYPE is read-only).</div>																																		

3.6.5 (0x28) Write and Read Variables

Direction	Payload Size	Expected answer																
Master → Node	2 + (SIZE of the Variable to be written)	(0xE0) OK																
Description																		
Writes in the VALUE of a Variable. The Variable must be writable. Returns the VALUE of a second Variable, to be read.																		
Structure																		
<table><tr><th colspan="5">Payload</th></tr><tr><td rowspan="2">ID of the Variable to be written</td><td rowspan="2">ID of the Variable to be read</td><td colspan="3">VALUE of the Variable to be written</td></tr><tr><td>First byte</td><td>...</td><td>Last byte</td></tr></table>			Payload					ID of the Variable to be written	ID of the Variable to be read	VALUE of the Variable to be written			First byte	...	Last byte			
Payload																		
ID of the Variable to be written	ID of the Variable to be read	VALUE of the Variable to be written																
		First byte	...	Last byte														
The payload contains the ID of the Variable to be written, followed by the ID of the Variable to be read and by sequence of bytes to be written.																		
Example																		
<table><tr><th colspan="3">Header</th><th colspan="5">Payload</th></tr><tr><td>20</td><td>00</td><td>05</td><td>04</td><td>05</td><td>01</td><td>BB</td><td>BB</td></tr></table> <div>Request to write 01h BBh BBh to the VALUE of the Variable with ID 4. Request the VALUE of the Variable with ID 5.</div>			Header			Payload					20	00	05	04	05	01	BB	BB
Header			Payload															
20	00	05	04	05	01	BB	BB											
Possible Errors																		
<div>(0xE3) Invalid ID: There's no Variable with the specified ID.</div> <div>(0xE6) Read-Only: Variable can't be written (its TYPE is read-only).</div>																		

3.7 (0x30) Group of Variables' Manipulation Commands

3.7.1 (0x30) Create Group of Variables

Direction	Payload Size	Expected answer														
Master → Node	(number of Variables in the Group)	(0xE0) OK														
Description																
Create a new Group of Variables with the Variables specified in the payload. The ID of the newly created Group is equal to the ID of the last Group in the node, plus 1.																
Structure																
<table><tr><th colspan="3">Payload</th></tr><tr><td>First Variable</td><td rowspan="2">...</td><td>Last Variable</td></tr><tr><td>ID</td><td>ID</td></tr></table>		Payload			First Variable	...	Last Variable	ID	ID	The IDs of the Variables to be added to the new Group.						
Payload																
First Variable	...	Last Variable														
ID		ID														
Example																
<table><tr><th colspan="3">Header</th><th colspan="4">Payload</th></tr><tr><td>30</td><td>00</td><td>04</td><td>04</td><td>05</td><td>06</td><td>07</td></tr></table>		Header			Payload				30	00	04	04	05	06	07	Create a Group with the Variables with IDs 4, 5, 6 and 7.
Header			Payload													
30	00	04	04	05	06	07										
Possible Errors																
(0xE3) Invalid ID:	At least one of the specified ID's doesn't exist.															
(0xE5) Invalid Payload Size:	Number of Variables is zero or greater than the number of Variables in the node.															
(0xE7) Insufficient Memory:	There's no memory available to create the Group.															

3.7.2 (0x32) Remove all Groups of Variables

Direction	Payload Size	Expected answer
Master → Node	0	(0xE0) OK
Description Request for the node to remove all his Groups, except for the Standard Groups.		

3.8 (0x4_) Curve Transfer Commands

3.8.1 (0x40) Request Curve Block

Direction	Payload Size	Expected answer												
Master → Node	3	(0x41) Curve Block												
Description														
Request for the node to send a specific block of the specified Curve.														
Structure														
<table><tr><th colspan="3">Payload</th></tr><tr><td rowspan="2">Curve ID</td><td colspan="2">Block offset</td></tr><tr><td>Most significant byte</td><td>Least significant byte</td></tr></table>			Payload			Curve ID	Block offset		Most significant byte	Least significant byte				
Payload														
Curve ID	Block offset													
	Most significant byte	Least significant byte												
The payload contains the ID of the Curve and two bytes for the block offset (in Big Endian). The first block has the offset zero (0000h).														
Example														
<table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>40</td><td>00</td><td>03</td><td>03</td><td>00</td><td>04</td></tr></table> Request the fifth block (0004h) of the Curve with ID 03h.			Header			Payload			40	00	03	03	00	04
Header			Payload											
40	00	03	03	00	04									
Possible Errors														
(0xE3) Invalid ID: There's no Curve with the specified ID.														
(0xE4) Invalid Value: The block offset specified is invalid.														

3.8.2 (0x41) Curve Block

Direction	Payload Size	Expected answer																				
Master ↔ Node	3 + (from 0 to SBLOCK)	(0xE0) OK																				
Description																						
Transmission of a Curve block sent either byte the node or by the master. If the block is sent by the master, it means that it is a request to write in the values of the specified block; the CHECKSUM of the Curve must be zeroed if the write operation is successful. When the master is done writing blocks to a Curve of the node, it should then send the (0x42) Recalculate Curve Checksum command. The block data can have less than SBLOCK bytes.																						
Structure																						
Payload																						
Curve ID	Block Offset		Block Data																			
	Most significant byte	Least significant byte	First byte	...	Last byte																	
The payload contains the ID of a Curve and two bytes for the offset of the Curve's block, followed by 16384 bytes containing the data of the specified block.																						
Example																						
<table><tr><th colspan="3">Header</th><th colspan="5">Payload</th></tr><tr><td>41</td><td>40</td><td>03</td><td>07</td><td>04</td><td>00</td><td>DD</td><td>...</td><td>DD</td></tr></table>			Header			Payload					41	40	03	07	04	00	DD	...	DD	Block offset 1024 (0400h) of the Curve with ID 07h that contains 16384 bytes DDh.		
Header			Payload																			
41	40	03	07	04	00	DD	...	DD														
Possible Errors																						
(0xE3) Invalid ID:		There's no Curve with the specified ID.																				
(0xE4) Invalid Value:		The block offset specified is invalid.																				
(0xE6) Read-Only:		Curve couldn't be written (its TYPE is read-only).																				

3.8.3 (0x42) Recalculate Curve Checksum

Direction	Payload Size	Expected answer				
Master → Node	1	(0x0B) Curve Checksum				
Description						
Request that the CHECKSUM of a Curve be recalculated.						
Structure						
<table><tr><th>Payload</th></tr><tr><td>Curve ID</td></tr></table>		Payload	Curve ID	The payload contains the ID of the Curve to have its CHECKSUM recalculated.		
Payload						
Curve ID						
Example						
<table><tr><th>Header</th><th>Payload</th></tr><tr><td>420001</td><td>00</td></tr></table>		Header	Payload	420001	00	Request to recalculate the CHECKSUM of the Curve with ID 0.
Header	Payload					
420001	00					
Possible Errors						
(0xE3) Invalid ID:		There's no Curve with the specified ID.				

3.9 (0x5_) Function Execution Commands

3.9.1 (0x50) Execute Function

Direction	Payload Size	Expected answers												
Master → Node	1+(Function INPUT)	(0x51) Function Return or (0x53) Function Error												
Description														
Request a specific Function to be executed with the given parameters.														
Structure														
<table><tr><th colspan="4">Payload</th></tr><tr><td rowspan="2">Function ID</td><td colspan="3">Input parameters</td></tr><tr><td>First byte</td><td>...</td><td>Last Byte</td></tr></table>			Payload				Function ID	Input parameters			First byte	...	Last Byte	
Payload														
Function ID	Input parameters													
	First byte	...	Last Byte											
The payload contains the ID of the Function to be executed followed by a list of bytes to be passed as input parameters. The amount of bytes for the input parameters must be exactly INPUT bytes.														
Example														
<table><tr><th colspan="3">Header</th><th colspan="3">Payload</th></tr><tr><td>50</td><td>00</td><td>03</td><td>01</td><td>BE</td><td>57</td></tr></table>		Header			Payload			50	00	03	01	BE	57	Execute the Function with ID 01h passing BEh 57h as input parameters.
Header			Payload											
50	00	03	01	BE	57									
Possible Errors														
(0xE3) Invalid ID:		There's no Function with the specified ID.												
(0xE5) Invalid Payload Size:		The number of bytes passed as input differs from the expected.												

3.9.2 (0x51) Function Return

Direction	Payload Size	Expected answer									
Master ← Node	(Function OUTPUT)	-									
Description											
Contains the result of the execution of a Function.											
Structure											
<table><tr><td colspan="3">Payload</td></tr><tr><td colspan="3">Function output</td></tr><tr><td>First byte</td><td>...</td><td>Last byte</td></tr></table>		Payload			Function output			First byte	...	Last byte	The payload contains all bytes returned by the Function as output.
Payload											
Function output											
First byte	...	Last byte									
Example											
<table><tr><td colspan="3">Header</td><td>Payload</td></tr><tr><td>51</td><td>00</td><td>01</td><td>00</td></tr></table>		Header			Payload	51	00	01	00	Response for the execution of a Function that returned just 1 byte, 00h.	
Header			Payload								
51	00	01	00								

3.9.3 (0x53) Function Error

Direction	Payload Size	Expected answer								
Master ← Node	1	-								
Description										
Indicates that there was an error returned by the execution of a Function.										
Structure										
<table><tr><th>Payload</th></tr><tr><td>Error Code</td></tr></table>		Payload	Error Code	The payload contains the error code returned by the Function, which is specific to the Function and must have its meaning described by the developer of the device.						
Payload										
Error Code										
Example										
<table><tr><th colspan="3">Header</th><th>Payload</th></tr><tr><td>53</td><td>00</td><td>01</td><td>BB</td></tr></table>		Header			Payload	53	00	01	BB	An error return code of BBh.
Header			Payload							
53	00	01	BB							

3.10(0xE_) Error Commands

All Error Commands are in the direction Master ← Node and don't have payload.

3.10.1 (0xE0) OK

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description The last command was successfully executed.		

3.10.2 (0xE1) Malformed Message

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description The number of bytes received in the payload differs from what was specified in the message's SIZE field.		

3.10.3 (0xE2) Operation not supported

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description The requested command is not supported.		

3.10.4 (0xE3) Invalid ID

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description One of the IDs specified was invalid.		

3.10.5 (0xE4) Invalid Value

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description		
A value passed is out of the acceptable range.		

3.10.6 (0xE5) Invalid Payload Size

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description		
The payload size is different than the size expected by the command.		

3.10.7 (0xE6) Read-Only

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description		
Tried to write on a read-only Entity.		

3.10.8 (0xE7) Insufficient Memory

Direction	Payload Size	Expected answer
Master ← Node	0	-
Description		
There wasn't enough memory to complete the request.		