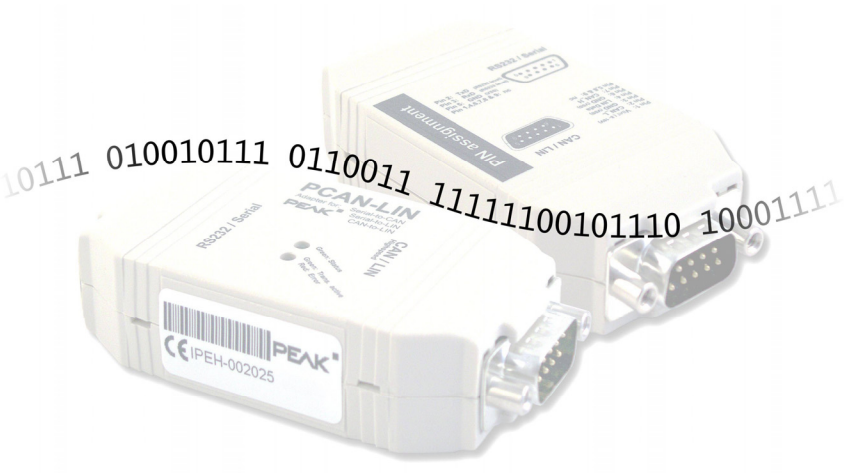
PCAN-LIN

RS-232 to LIN/CAN Interface

Protocol Definitions Documentation



Document version 1.1.0 (2016-01-14)

PEAK System

Relevant products

Product name	Firmware version
PCAN-LIN	01.02.xx

PEAK-System Technik GmbH
Otto-Roehm-Strasse 69
64293 Darmstadt
Germany

Phone: +49 (0)6151 8173-20
Fax: +49 (0)6151 8173-29

www.peak-system.com
info@peak-system.com

Document version 1.1.0 (2016-01-14)

Contents

1 Communication with a PCAN-LIN Module

The main communication with a PCAN-LIN module is done via the RS-232 interface. It is divided in structures containing information about the operation being done, the interface on the module being affected, needed parameters, data transmitted by CAN and LIN messages, or scheduler entries. Because the latter case requires more data, “Write Data” and “Scheduler Entry” commands use a more complex structure.

A total of eight different structures are used: two for the communication between host and module, two for reading and writing CAN messages, two for reading and writing LIN messages, and two for retrieving and setting entries of a LIN master scheduler.

1.1 Serial Protocol

1.1.1 Data Transmission Host – Module

Byte	Description	Bit position							
		7	6	5	4	3	2	1	0
1	STX	0	0	0	0	0	0	1	0
2	Sequence code	X	Sequence no.			Parameter byte count			
3	Control code	Target interface		Command code					
4...n	Parameters	Number of parameters depends on the parameter byte count part of the sequence code.							
n+1	Checksum	Bitwise exclusive-OR operation between all bytes of this messages except the STX byte (first byte, start byte).							

STX

This is the "Start Transmission" code for any message in a module communication. This code has a fixed value of 2 (0x2, 0b10). A message without this start byte will not be recognized as a module message.

Sequence code

The second byte in a message from host to module indicates a sequence number and the length of the data to be sent. The sequence code field length is 7 bits. The most significant bit of this field is not used and is recommended to be set to 0 in order to guaranty compatibility with future Firmware versions.

Sequence number: This number between 0 and 7 is a "User Control Number" for sending messages. Its purpose is to help with the differentiation of messages that can be sent as segmented data. A message can not have more than 15 bytes. In case that a message (command) from the host produces an answer message from the module, this response will come with the same sequence number of the message which provoked it.

Parameter byte count: These 4 bits of the sequence code indicate how many parameter/data bytes the message has. A LIN communication message cannot have more than 15 bytes (parameter byte count maximum value is 15 = 4 bits).

For example, a message with a sequence number of 3 and two parameters has the sequence code 50 (0x32, 0b0110010).

Control code

This byte indicates the command to be processed as well as the interface of the module which the command is directed to. The two most significant bits indicate the interface and the other bits the command code.

Target interface: These fields with a length of two bits indicate which of the four interfaces is associated with the command code of the "Command code" field. The codes and description of all possible commands are described in chapter 2 *Interfaces* on page 12.

Command Code: With a length of 6 bits this field can indicate a command between 0 and 63. The commands are divided in four groups. The codes and description of all possible commands are described in chapter 2 *Interfaces* on page 12.

For example, a message setting the module's bit rate for the CAN interface has the control code 95 (0x5F, 0b01011111). Code of CAN Interface = 1. Code of bit rate parameter = 31.

Parameters

Succeeding the control code, a LIN communication message will have as many parameter/data bytes as indicated in the parameter byte count field of the sequence code. For example, if the parameter byte count is 2 (0x2, 0b0010), the module will interpret the following two received bytes after the control code as parameters/data. The form of this parameters/data depends on each command.

For example, a message containing a new bit rate for the CAN interface has two parameter bytes which indicate the desired bit rate. The parameters must be arranged beginning with the least significant byte (LSB). To set a bit rate of 500 kbit/s (0x001C), a message has a value of 28 (0x1C, 0b11100) as first parameter and a value of 0 (0x0, 0b0) as second parameter. Therefore the parameter bits are 0001110000000000.

Checksum

The last byte of a LIN communication message is the checksum. This byte is the result of an Exclusive-OR operation between the bits of all components of the message with exception of the start byte (STX).

For example, a message with a sequence code of 50, a control code of 95 and two parameters, 28 and 0, has a value of 113 (0x71, 0b01110001) as checksum byte.

In summary, putting this small examples together, a LIN communication message setting the module's CAN interface to the bit rate of 500 kbit/s using a sequence number of 3 is done by sending the following data to the RS-232 port (order of appearance, from left to right):

- Hexadecimal representation: 02 – 32 – 5F – 1C – 00 – 71
- Decimal representation: 2 – 50 – 95 – 28 – 0 – 113
- Binary representation:

1	STX	0	0	0	0	0	0	**1**	**0**
2	Sequence code	**0**	0	1	1	0	0	1	0
3	Control code	0	1	0	1	1	1	1	1
4	Parameter0	0	0	0	1	1	1	0	0
5	Parameter1	0	0	0	0	0	0	0	0
6	Checksum	0	1	1	1	0	0	0	1

1.1.2 Data Transmission Module – Host

Byte	**Description**	**Bit position**							
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	STX	0	0	0	0	0	0	1	0
2	Sequence code	AR	Sequence number			Parameter byte count			
3...n	Parameters	Number of parameter depends on the parameter byte count part of the sequence code.							
n+1	Checksum	Bitwise exclusive-OR operation between all bytes of this messages except the STX byte (first byte, start byte).							

STX

This is the "Start Transmission" code for any message in a module communication. This code has a fixed value of 2 (0x2, 0b10). A message without this start byte will not be recognized as a module message.

Sequence code

The second byte in a message from the module to the host indicates a sequence number and the length of the data to be received. The most significant bit of this field is called **A**uto **R**eply bit and its use is reserved for the module. It was introduced to make a difference between forwarded messages (**AR** set to 1) and answers to commands sent by the host.

Sequence Number: This number between 0 and 7 is a "User Control Number" for messages. If the message is a response to a request made by the host, the sequence number is the same sequence number used by the request message. If the message is an Auto Reply message the sequence number is defined as follows:

Bit	Value	Description
6	0	Unused
5/4	00 = RS-232 01 = CAN 02 = LIN 03 = Module	Source

Parameter byte count: These four bits of the sequence code indicate how many bytes are interpreted as parameter/data in the incoming message after the sequence code and before the checksum byte.

For example, a response for the example in section 1.1.1 above (setting the CAN bit rate) is 49 (0x31, 0b00110001). This value corresponds in its sequence number field to 3 (the same used above) and in its parameter byte count field to 1 (one byte as answer). This data byte indicates if the operation was successful or not. For more information see chapter 3 *Error Messages* on page 103.

Parameters

An incoming LIN communication message from the module will have as many parameter/data bytes after the sequence code byte as indicated in the parameter byte count field. The way to interpret this data depends on the requested information. For more information see the possible return values for the desired command later in this documentation.

For example, if the command to set the CAN bit rate was sent and this operation was successful, the incoming message has only one parameter and this is 0 (no errors).

Checksum

The last byte of an incoming LIN communication message is the checksum. This byte is the result of an exclusive-OR operation between the bits of all components of the message with exception of the start byte (STX).

For example, a message with a sequence code of 49 and one parameter, 0, has a value of 49 (0x31, 0b00110001) as checksum byte.

In summary, putting this small examples together, an incoming LIN communication message indicating the result of setting the bit rate of the CAN interface (regarding the values used in the request messages as example in the previous section) the RS-232 port receives the following data:

- Hexadecimal representation: 02 – 31 – 00 – 31
- Decimal representation: 2 – 49 – 0 – 49
- Binary representation:

1	STX	0	0	0	0	0	0	**1**	**0**
2	Sequence code	**0**	0	1	1	0	0	0	1
3	Parameter0	0	0	0	0	0	0	0	0
4	Checksum	0	0	1	1	0	0	0	1

2 Interfaces

2.1 Notes about Interfaces and Commands

The PCAN-LIN module has four logical interfaces: RS-232, CAN, LIN and Module. A command may not be valid for all interfaces. Each interface has a code in order to direct a command to it. These codes are:

- RS-232: 0
- CAN: 1
- LIN: 2
- Module: 3

RS-232 (0)

This interface is used to configure and/or reset/erase information referring to the RS-232 interface of the module. This interface is used to make all needed configurations on the module.

CAN (1)

This interface is used to configure and/or reset/erase the settings of the CAN capabilities of the module.

LIN (2)

This interface is used to configure and/or reset/erase the settings of the LIN capabilities of the module.

Module (3)

This interface is used to read general information about the module as hardware and firmware version as well as to reset it.

The commands are arranged in four groups according to their tasks: initializing, reading, configuring, resetting/deleting. The command

"Write Data" is treated separately, because it uses more complex structures.

The following sections contain tables with the values for the sequence code and the control code of each described command message. The sequence number is 0 by default. If you use another sequence number, you have to recalculate this field.

Abbreviations

- STX: start byte. It is always 0x02.
- SC: sequence code
- CC: control code
- PX: parameters. X stands for the correlative number of the parameter (1…n).
- CHK: checksum. The checksum is calculated doing an exclusive-OR operation between the bits of all components of the message with exception of the start byte (STX).

2.2 RS-232 Interface

2.2.1 Initialization Commands

Command	on page
Initialization According to Current Parameters	13

Initialization According to Current Parameters

This command initializes the RS-232 interface with the configured parameters. In this case, the only parameter that can be configured using a configuration command is the baud rate. If a configuration command was successful, the new baud rate will not be used until this command is sent.

Sequence Code (SC)									Control Code (CC)								
X	Sequence			Param Byte Count					Interface		Command Code						
0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
0	0, 0x0			0, 0x0					0, 0x0		0, 0x0						
0, 0x0									0, 0x0								

Parameters	none
Checksum	SC(0x00) XOR CC(0x00) XOR PX(0x00) = 0x0
Message to send	STX(0x02) – SC(0x00) – CC(0x00) – CHK(0x00)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: After changing and initializing the used RS-232 baud rate, the connection with the host must be adjusted to the new value. Therefore a reconnection between host and module using the new baud rate is required because the module will not respond anymore.

2.2.2 Read Commands

Retrieve Baud Rate

This command retrieves the configured baud rate of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x9600 = P1:0x00 and P2:0x96

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	0	0	0	1	0	1	1
0	0, 0x0			0, 0x0					0, 0x0		11, 0xB					
0, 0x0									11, 0xB							

Parameters	none
Checksum	SC(0x00) XOR CC(0x0B) XOR PX(0x00) = 0x0B
Message to send	STX(0x02) – SC(0x00) – CC(0x0B) – CHK(0x0B)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x96) – CHK(0x94)

A positive response has two parameters with the configured baud rate. In this example the default baud rate is returned with P1 and P2:0x9600 = 38400 Baud

Note: The configured baud rate differs from the used baud rate. The used baud rate is the speed being used to send and receive messages from/to the module in a determined time. A configured baud rate is the result of sending a baud rate configuration message causing the change in the baud rate but not initializing it. To activate the new configured value refer to section 2.2.1 *Initialization Commands*.

Retrieve RCV Error Counter

This command retrieves the value of the receive error counter of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	0	0	1	0	1	1	0
0	0, 0x0			0, 0x0					0, 0x0		22, 0x16					
0, 0x0									22, 0x16							

Parameters	none
Checksum	SC(0x00) XOR CC(0x16) XOR PX(0x00) = 0x16
Message to send	STX(0x02) – SC(0x00) – CC(0x16) – CHK(0x16)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the RCV error counter. In this example the default value is returned with P1 and P2:0x0 = 0 errors.

Retrieve XMT Error Counter

This command retrieves the value of the transmit error counter of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	0	0	1	0	1	1	1
0	0, 0x0			0, 0x0					0, 0x0		23, 0x17					
0, 0x0									23, 0x17							

Parameters	none
Checksum	SC(0x00) XOR CC(0x17) XOR PX(0x00) = 0x17
Message to send	STX(0x02) – SC(0x00) – CC(0x17) – CHK(0x17)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the XMT error counter. In this example the default value is returned with P1 and P2:0x0 = 0 errors

Retrieve All Error Counters

This command retrieves the values of the error counters of the RS-232 interface. The returned values are (in this order) the value of the RCV error counter, the value of the XMT error counter and the value of the checksum error counter. Each value is 2 bytes long.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0, 0x0			0, 0x0				0, 0x0		24, 0x18					
0, 0x0								24, 0x18							

Parameters	none
Checksum	SC(0x00) XOR CC(0x18) XOR PX(0x00) = 0x18
Message to send	STX(0x02) – SC(0x00) – CC(0x18) – CHK(0x18)
Positive response	STX(0x02) – SC(0x06) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0x00) – CHK(0x06)

A positive response has 6 parameters with the values of the RCV, XMT and checksum error counters. In this example following is returned:

- with P1 and P2 the default value of the RCV error counter: 0x0 = 0 errors
- with P3 and P4 the default value of the XMT error counter: 0x0 = 0 errors
- with P5 and P6 the default value of the checksum error counter: 0x0 = 0 errors

2.2.3 Configuration Commands

Configure Baud Rate

The only parameter for the RS-232 interface that can be changed is the baud rate. There are a couple of specific allowed baud rates.

When you try to set a baud rate that is not in the allowed list, a negative response message is initiated by the module and the baud rate will not be changed.

Parameters: Two parameters (two bytes) are required in order to specify the desired baud rate. The following table shows the allowed baud rates and its parameter notation P1-P2.

Baud rate	Hex value	P1	P2
4,800	0x12C0	0xC0	0x12
9,600	0x2580	0x80	0x25
19,200	0x4B00	0x00	0x4B
38,400	0x9600	0x00	0x96
57,600	0xE100	0x00	0xE1

The data to be sent and received starts with the LSB. Example: 0x12C0 = P1:0xC0 and P2:0x12

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		0	0	0	1	1	1	1	1
0	0, 0x0			2, 0x2					0, 0x0		31, 0x1F					
2, 0x2									31, 0x1F							

Parameters	Baud rate, two parameters
Checksum	PX = P1(0x00) XOR P2(0x96) = 0x96 SC(0x02) XOR CC(0x1F) XOR PX(0x96) = 0x8B
Message to send	STX(0x02) – SC(0x02) – CC(0x1F) – P1(0x00) – P2(0x96) – CHK(0x0B)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: The configured baud rate differs from the used baud rate. The used baud rate is the speed being used to send and receive

messages from/to the module in a determined time. A configured baud rate is the result of sending a baud rate configuration message causing the change in the baud rate but not initializing it. To activate the new configured value refer to section 2.2.1 *Initialization Commands* on page 13.

2.2.4 Reset/Erase Commands

Command	on page
Reset RCV Error Counter	19
Reset XMT Error Counter	20
Reset Checksum Error Counter	20

Reset RCV Error Counter

This command directs the module to reset the counter of the reception errors of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
0	0, 0x0			0, 0x0				0, 0x0		52, 0x34					
0, 0x0								52, 0x34							

Parameters	none
Checksum	SC(0x00) XOR CC(0x34) XOR PX(0x00) = 0x34
Message to send	STX(0x02) – SC(0x00) – CC(0x34) – CHK(0x34)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset XMT Error Counter

This command directs the module to reset the counter of the transmission errors of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	0	1	1	0	1	0	1
0	0, 0x0			0, 0x0					0, 0x0		53, 0x35					
0, 0x0									53, 0x35							

Parameters	none
Checksum	SC(0x00) XOR CC(0x35) XOR PX(0x00) = 0x35
Message to send	STX(0x02) – SC(0x00) – CC(0x35) – CHK(0x35)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset Checksum Error Counter

This command directs the module to reset the counter of the checksum errors of the RS-232 interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	0	1	1	0	1	1	1
0	0, 0x0			0, 0x0					0, 0x0		55, 0x37					
0, 0x0									55, 0x37							

Parameters	none
Checksum	SC(0x00) XOR CC(0x37) XOR PX(0x00) = 0x37
Message to send	STX(0x02) – SC(0x00) – CC(0x37) – CHK(0x37)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

2.3 CAN Interface

2.3.1 Initialization and Write Commands

Initialization According to Current Parameters

This command initializes the CAN interface with the configured parameters.

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	0	0	0	0	0
0	0, 0x0			0, 0x0					1, 0x1		0, 0x0					
0, 0x0									64, 0x40							

Parameters	none
Checksum	SC(0x00) XOR CC(0x40) XOR PX(0x00) = 0x40
Message to send	STX(0x02) – SC(0x00) – CC(0x40) – CHK(0x40)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Write Data on Bus

The command “Write Data” is treated separately, because it uses more complex structures. See section 2.6.1 *Message Transmission and Reception > CAN* on page 95.

2.3.2 Read Commands

Retrieve Activation Status

This command retrieves a value indicating, if the CAN features of the module are active. If not, a configuration of the interface is still possible but it does not have any effect when a CAN feature is required in order to handle an event.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0
0	0, 0x0			0, 0x0				1, 0x1		10, 0xA					
0, 0x0								74, 0x4A							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4A) XOR PX(0x00) = 0x4A
Message to send	STX(0x02) – SC(0x00) – CC(0x4A) – CHK(0x4A)
Positive response	STX(0x02) – SC(0x02) – P1(0x01) – P2(0x00) – CHK(0x03)

A positive response has two parameters. If the value formed by P1 and P2 (in this case 0x01) is greater than 0, this means “Interface is active”, otherwise “Interface is not active”.

Retrieve Bit Rate

This command retrieves the configured bit rate of the CAN interface. The bit rate is in Btr0Btr1 format.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	0	1	0	1	1
0	0, 0x0			0, 0x0					1, 0x1		11, 0xB					
0, 0x0									75, 0x4B							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4B) XOR PX(0x00) = 0x4B
Message to send	STX(0x02) – SC(0x00) – CC(0x4B) – CHK(0x4B)
Positive response	STX(0x02) – SC(0x02) – P1(0x1C) – P2(0x00) – CHK(0x1E)

A positive response has two parameters with the configured bit rate. In this example the default bit rate is returned with P1 and P2: 0x0010 = 500 kbit/s

Note: The configured bit rate differs from the used bit rate. The used bit rate is the speed being used to send and receive messages from/to the module in a determined time. A configured bit rate is the result of sending a bit rate configuration message causing the change in the bit rate but not initializing it. To acti-

vate the new configured value refer to section 2.3.1 *Initialization and Write Commands* on page 21.

Retrieve Forward Mask

This command retrieves the configured forward mask of the CAN interface in a module. This mask tells an interface to which other interface(s) the received messages shall be forwarded and if error information shall be monitored. The errors can be redirected to the RS-232 interface, too.

The forward mask value is a bit combination (bitwise OR operation) of the values in the following table:

Mask value	Meaning
0x80	Forward error information to RS-232 interface
0x40	Forward error information to CAN interface
0x04	Forward CAN messages to RS-232 interface
0x02	Forward CAN messages to LIN interface

For example, a forward mask of 0x86, means that the error information of the CAN interface are sent to the RS-232 interface and received CAN messages are forwarded to both the LIN and the RS-232 interface.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
0	0, 0x0			0, 0x0				1, 0x1		12, 0xC					
0, 0x0								76, 0x4C							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4C) XOR PX(0x00) = 0x4C
Message to send	STX(0x02) – SC(0x00) – CC(0x4C) – CHK(0x4C)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. In this example the returned value is the default value (no forwarding). If the value is different than 0, you can figure out which kind of forwarding is configured by doing a bitwise AND operation using the values from the mask table. For example, if the returned value is 0x86, the AND operation would be as follows:

- If (0x86 AND 0x80 = 0x80), then forward error information to RS-232 interface
- If (0x86 AND 0x04 = 0x04), then forward CAN messages to RS-232 interface
- If (0x86 AND 0x02 = 0x02), then forward CAN messages to LIN interface

Retrieve Filter Mask

This command retrieves the configured acceptance mask of the CAN interface in a module. The CAN acceptance mask, in conjunction with the acceptance code, is a filter used to tell the CAN interface which messages are received. This filter is referring to the ID of a CAN message.

The data to be received starts with the LSB. Example: 0x221100FF = P1:0xFF, P2:0x00, P3:0x11, and P4:0x22

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1
0	0, 0x0			0, 0x0				1, 0x1		13, 0xD					
0, 0x0								77, 0x4D							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4D) XOR PX(0x00) = 0x4D
Message to send	STX(0x02) – SC(0x00) – CC(0x4D) – CHK(0x4D)
Positive response	STX(0x02) – SC(0x04) – P1(0xFF) – P2(0xFF) – P3(0xFF) – P4(0xFF) – CHK(0x04)

A positive response has 4 parameters. In this example the returned value is the default value for a Filter Mask (0xFFFFFFFF). For more information about CAN filters, see the Application Note AN97076 for the CAN controller Philips SJA1000.

Retrieve Filter Code

This command retrieves the configured acceptance code of the CAN interface in a module. The CAN acceptance code, in conjunction with the acceptance mask, is a filter used to tell the CAN interface which messages are received. This filter is referring to the ID of the CAN messages.

The data to be received starts with the LSB. Example: 0x221100FF = P1:0xFF, P2:0x00, P3:0x11, and P4:0x22

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	0	1	1	1	0
0	0, 0x0			0, 0x0					1, 0x1		14, 0xE					
0, 0x0									78, 0x4E							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4E) XOR PX(0x00) = 0x4E
Message to send	STX(0x02) – SC(0x00) – CC(0x4E) – CHK(0x4E)
Positive response	STX(0x02) – SC(0x04) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – CHK(0x04)

A positive response has 4 parameters. In this example the returned value is the default value for a filter code (0x00000000). For more information about CAN filters, see the application note AN97076 for the CAN controller Philips SJA1000.

Retrieve CAN ID Offset for LIN Messages

This command retrieves the configured CAN ID offset for LIN messages. The length of a LIN messages ID is one byte. CAN messages need an ID offset in order to be successfully forwarded to the LIN interface.

The data to be received starts with the LSB. Example: 0x00110022 = P1:0x22, P2:0x00, P3:0x11, and P4:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	0	1	1	1	1
0	0, 0x0			0, 0x0					1, 0x1		15, 0xF					
0, 0x0									79, 0x4F							

Parameters	none
Checksum	SC(0x00) XOR CC(0x4F) XOR PX(0x00) = 0x4F
Message to send	STX(0x02) – SC(0x00) – CC(0x4F) – CHK(0x4F)
Positive response	STX(0x02) – SC(0x04) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – CHK(0x04)

Description of the positive response:

Param(s)	Description
P1	Least significant byte of the CAN ID offset
P2	2nd byte of the CAN ID offset (only 3 bits for 11-bit CAN ID)
P3	3rd byte of the CAN ID offset (only 29-bit CAN ID)
P4	Bits 0 to 4: Most significant part of the CAN ID offset (only 29-bit CAN ID) Bit 5: Not used Bit 6: Not used Bit 7: Flag for extended CAN ID (0 = 11 bits, 1 = 29 bits)

Retrieve RCV Error Counter

This command retrieves the value of the receive error counter of the CAN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	1	0	1	1	0
0	0, 0x0			0, 0x0					1, 0x1		22, 0x16					
0, 0x0									86, 0x56							

Parameters	none
Checksum	SC(0x00) XOR CC(0x56) XOR PX(0x00) = 0x56
Message to send	STX(0x02) – SC(0x00) – CC(0x56) – CHK(0x56)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the RCV error counter. In this example the default value is returned with P1 and P2: 0x0 = 0 errors

Retrieve XMT Error Counter

This command retrieves the value of the transmit error counter of the CAN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	1	0	1	1	1
0	0, 0x0			0, 0x0					1, 0x1		23, 0x17					
0, 0x0									87, 0x57							

Parameters	none
Checksum	SC(0x00) XOR CC(0x57) XOR PX(0x00) = 0x57
Message to send	STX(0x02) – SC(0x00) – CC(0x57) – CHK(0x57)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the XMT error counter. In this example the default value is returned with P1 and P2: 0x0 = 0 errors

Retrieve All Error Counters

This command retrieves the values of the errors counters of the CAN interfaces. The returned values are (in this order) the value of the RCV error counter, the value of the XMT error counter, the bus passive state error counter and the bus off state error counter. Each value is 2 bytes long.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		0	1	0	1	1	0	0	0
0	0, 0x0			0, 0x0					1, 0x1		24, 0x18					
0, 0x0									88, 0x58							

Parameters	none
Checksum	SC(0x00) XOR CC(0x58) XOR PX(0x00) = 0x58
Message to send	STX(0x02) – SC(0x00) – CC(0x58) – CHK(0x58)
Positive response	STX(0x02) – SC(0x08) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0x00) – P7(0x00) – P8(0x00) – CHK(0x08)

A positive response has 8 parameters with the value of the RCV, XMT, bus passive and bus off states error counters. In this example the following is returned:

- with P1 and P2 the default value of the RCV error counter: 0x0 = 0 errors
- with P3 and P4 the default value of the XMT error counter: 0x0 = 0 errors
- with P5 and P6 the default value of the bus passive state error counter: 0x0 = 0 errors
- with P7 and P8 the default value of the bus off state error counter: 0x0 = 0 errors

2.3.3 Configuration Commands

Set Activation Status

This command sets a value in the module which indicates, if the CAN features are activated.

The data to be sent or received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		0	1	0	1	1	1	1	0
0	0, 0x0			1, 0x1					1, 0x1		30, 0x1E					
1, 0x1									94, 0x5E							

Parameters	One parameter (one byte) is required in order to specify the activation status of the CAN interface. This parameter byte works as a boolean value. If the parameter is 0, the interface is deactivated, otherwise is set to active. (In the example the interface is activated.)
Checksum	P1 = 0x01 SC(0x01) XOR CC(0x5E) XOR PX(0x01) = 0x5E
Message to send	STX(0x02) – SC(0x01) – CC(0x5E) – P1(0x01) – CHK(0x5E)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: The activation of the interface does not require an initialization using the initialization command.

Set Bit Rate

This command directs the module to change the bit rate used by the CAN interface (Btr0Btr1 format). There are a couple of preset bit rates, but all valid CAN bit rate can be used.

Parameters: Two parameters (2 bytes) are required in order to specify the desired bit rate. The following table shows the frequently used bit rates and their parameter notation P1-P2.

Bit rate	Hex value	P1	P2
1 Mbit/s	0x0014	0x14	0x00
500 kbit/s	0x001C	0x1C	0x00
250 kbit/s	0x011C	0x1C	0x01
125 kbit/s	0x031C	0x1C	0x03
100 kbit/s	0x432F	0x2F	0x43
50 kbit/s	0x472F	0x2F	0x47
20 kbit/s	0x532F	0x2F	0x53
10 kbit/s	0x672F	0x2F	0x67
5 kbit/s	0x7F7F	0x7F	0x7F

The data to be sent and received starts with the LSB. Example: 0x12C0 = P1:0xC0 and P2:0x12

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	0	0	1	0	1	1	1	1	1
0	0, 0x0			2, 0x2				1, 0x1		31, 0x1F					
2, 0x2								95, 0x5F							

Parameters	In this example the default bit rate of the CAN interface is set: 500 kbit/s (Btr0Btr1: 0x001C). The number of needed parameters is 2.
Checksum	PX = P1(0x1C) XOR P2(0x00) = 0x1C SC(0x02) XOR CC(0x5F) XOR PX(0x1C) = 0x41
Message to send	STX(0x02) – SC(0x02) – CC(0x5F) – P1(0x1C) – P2(0x00) – CHK(0x41)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: The configured bit rate differs from the used bit rate. The used bit rate is the speed being used by the CAN interface to interact with the CAN bus in a determined time. A configured bit rate is the result of sending a bit rate configuration message causing the change in the bit rate but not initializing it. To activate the new configured value refer to section 2.3.1 *Initialization and Write Commands* on page 21.

Set Forward Mask

This command sets the forward mask of the CAN interface in a module. This mask tells an interface to which other interface(s) the received messages are forwarded and, if error information is monitored. The errors can be redirected to the RS-232 interface.

The forward mask value is a bit combination (bitwise-OR operation) of the values in the following table:

Mask value	Meaning
0x80	Forward error information to RS-232 interface
0x40	Forward error information to CAN interface
0x04	Forward CAN messages to RS-232 interface
0x02	Forward CAN messages to LIN interface

For example, a forward mask of 0x86 means that the error information of the CAN interface is sent to the RS-232 interface and received CAN messages is forwarded to both the LIN and the RS-232 interface.

Note: In order to forward error information to the RS-232 interface (mask value 0x80) the corresponding message forwarding (0x04) must be activated, too.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		0	1	1	0	0	0	0	0
0	0, 0x0			1, 0x1					1, 0x1		32, 0x20					
1, 0x1									96, 0x60							

Parameters	One parameter (one byte) is required. In this example the mask will be used to monitor errors and, if any, to forward them to the RS-232 interface. The value to send is 0x80 (forward error information to RS-232 interface).
Checksum	P1 = 0x80 SC(0x01) XOR CC(0x60) XOR P1(0x80) = 0xE1
Message to send	STX(0x02) – SC(0x01) – CC(0x60) – P1(0x80) – CHK(0xE1)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Filter Mask

This command sets an acceptance mask for the CAN interface in a module. The CAN acceptance mask, in conjunction with the acceptance code, is a filter used to tell the CAN interface which messages are received. This filter is referring to the ID of a CAN message.

The data to be received starts with the LSB. Example: 0x221100FF = P1:0xFF, P2:0x00, P3:0x11, and P4:0x22

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	1	0	0		0	1	1	0	0	0	0	1
0	0, 0x0			4, 0x4					1, 0x1		33, 0x21					
4, 0x4									97, 0x61							

Parameters	Four parameters (4 bytes) are required. An acceptance mask is four bytes long because it uses the 29-bit identifier. For more information about CAN filters, see the application note AN97076 for the CAN controller Philips SJA1000. In this example a filter mask of 0x1FFFFFFF will be used.
Checksum	PX = P1(0xFF) XOR P2(0xFF) XOR P3(0xFF) XOR P4(0x1F) = 0xE0 SC(0x04) XOR CC(0x61) XOR PX(0xE0) = 0x85
Message to send	STX(0x02) – SC(0x04) – CC(0x61) – P1(0xFF) – P2(0xFF) – P3(0xFF) – P4(0x1F) – CHK(0x85)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has one parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Filter Code

This command sets the acceptance code for the CAN interface in a module. The CAN acceptance code, in conjunction with the acceptance mask, is a filter used to tell the CAN interface which messages are received. This filter is referring to the ID of the CAN messages.

The data to be received starts with the LSB. Example: 0x221100FF = P1:0xFF, P2:0x00, P3:0x11, and P4:0x22

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0
0	0, 0x0			4, 0x4				1, 0x1		34, 0x22					
4, 0x4								98, 0x62							

Parameters	Four parameters (4 bytes) are required. An acceptance code is four bytes long because it uses the 29-bit identifier. For more information about CAN filters, see the application note AN97076 for the CAN controller Philips SJA1000. In this example a filter code of 0xABBCCDEF will be used.
Checksum	PX = P1(0xEF) XOR P2(0xCD) XOR P3(0xBC) XOR P4(0xAB) = 0x35 SC(0x04) XOR CC(0x62) XOR PX(0x35) = 0x53
Message to send	STX(0x02) – SC(0x04) – CC(0x62) – P1(0xEF) – P2(0xCD) – P3(0xBC) – P4(0xAB) – CHK(0x53)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has one parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set CAN ID Offset for LIN Messages

This command sets the CAN ID offset for LIN messages in a module. The length of a LIN message's ID is one byte. CAN messages need an ID offset in order to be successfully forwarded to the LIN interface.

The data to be received starts with the LSB. Example: 0x80110022 = P1:0x22, P2:0x00, P3:0x11, and P4:0x80

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	1	0	0		0	1	1	0	0	0	1	1
0	0, 0x0			4, 0x4					1, 0x1		35, 0x23					
4, 0x4									99, 0x63							

Parameters	Four parameters (4 bytes) are required. A CAN ID offset is four bytes long (see extra table below). In this example an ID mask of 0x00110022 will be used.
Checksum	PX = P1(0x22) XOR P2(0x00) XOR P3(0x11) XOR P4(0x80) = 0xB3 SC(0x04) XOR CC(0x63) XOR PX(0xB3) = 0xD4
Message to send	STX(0x02) – SC(0x04) – CC(0x63) – P1(0x22) – P2(0x00) – P3(0x11) – P4(0x80) – CHK(0xD4)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has one parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Param(s)	Description
P1	Least significant byte of the CAN ID offset
P2	2nd byte of the CAN ID offset (only 3 bits for 11-bit CAN ID, other bits = 0)
P3	3rd byte of the CAN ID offset (only 29-bit CAN ID, else set to 0)
P4	Bits 0 to 4: Most significant part of the CAN ID offset (only 29-bit CAN ID, else set to 0) Bit 5: Not used (set to 0) Bit 6: Not used (set to 0) Bit 7: Flag for extended CAN ID (0 = 11 bits, 1 = 29 bits)

2.3.4 Reset/Erase Commands

The CAN interface does not have reset or erase commands.

2.4 LIN Interface

2.4.1 Initialization and Write Commands

Command	on page
Initialization According to Current Parameters	37
Write Data on Bus	37

Initialization According to Current Parameters

This command initializes the LIN interface with the configured parameters.

Sequence Code (SC)									Control Code (CC)								
X	Sequence			Param Byte Count					Interface		Command Code						
0	0	0	0	0	0	0	0		1	0	0	0	0	0	0	0	
0	0, 0x0			0, 0x0					2, 0x2		0, 0x0						
0, 0x0									128, 0x80								

Parameters	none
Checksum	SC(0x00) XOR CC(0x80) XOR PX(0x00) = 0x80
Message to send	STX(0x02) – SC(0x00) – CC(0x80) – CHK(0x80)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has one parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Write Data on Bus

The command “Write Data” is treated separately, because it uses more complex structures. See section 2.6.2 *Message Transmission and Reception > LIN* on page 98.

2.4.2 Read Commands

Retrieve Activation Status

This command retrieves a value indicating, if the LIN features of the module are active. If not, a configuration of the interface is still possible but does not have an effect when a LIN feature is required in order to handle an event.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	0	1	0	1	0
0	0, 0x0			0, 0x0					2, 0x2		10, 0xA					
0, 0x0									138, 0x8A							

Parameters	none
Checksum	SC(0x00) XOR CC(0x8A) XOR PX(0x00) = 0x8A
Message to send	STX(0x02) – SC(0x00) – CC(0x8A) – CHK(0x8A)
Positive response	STX(0x02) – SC(0x02) – P1(0x01) – P2(0x00) – CHK(0x03)

A positive response has two parameters. If the value formed by P1 and P2 (in this case 0x01) is greater than 0, this means “interface is active”, otherwise “interface is not active”.

Retrieve Bit Rate

This command retrieves the configured bit rate of the LIN interface.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	0	1	0	1	1
0	0, 0x0			0, 0x0					2, 0x2		11, 0xB					
0, 0x0									139, 0x8B							

Parameters	none
Checksum	SC(0x00) XOR CC(0x8B) XOR PX(0x00) = 0x8B
Message to send	STX(0x02) – SC(0x00) – CC(0x8B) – CHK(0x8B)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x4B) – CHK(0x49)

A positive response has two parameters with the configured bit rate. In this example the default bit rate is returned with P1 and P2: 0x4B00 = 19200 bit/s

Note: The configured bit rate differs from the used bit rate. The used bit rate is the speed being used by the LIN interface to interact with the LIN bus in a determined time. A configured bit rate is the result of sending a bit rate configuration message causing the change in the bit rate but not initializing it. To activate the new configured value refer to section 2.4.1 *Initialization and Write Commands*.

Retrieve Forward Mask

This command sets the forward mask of the LIN interface in a module. This mask tells an interface to which other interface(s) the received messages are forwarded and, if error information is monitored. The errors can be redirected to the RS-232 and the CAN interface.

The forward mask value is a bit combination (bitwise-OR operation) of the values in the following table:

Mask value	Meaning
0x80	Forward error information to RS-232 interface
0x40	Forward error information to CAN interface
0x04	Forward LIN messages to RS-232 interface
0x01	Forward LIN messages to CAN interface

For example, a forward mask of 0xC5 means that the error information as well as the messages of the LIN Interface will be forwarded to both the CAN and the RS-232 interface.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	0	1	1	0	0
0	0, 0x0			0, 0x0					2, 0x2		12, 0xC					
0, 0x0									140, 0x8C							

Parameters	none
Checksum	SC(0x00) XOR CC(0x8C) XOR PX(0x00) = 0x8C
Message to send	STX(0x02) – SC(0x00) – CC(0x8C) – CHK(0x8C)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. In this example the returned value is the default value (no forwarding). If the value is different than 0, you can figure out which kind of forwarding is

configured by doing a bitwise AND operation using the values from the mask table. For example, if the returned value is 0xC5, the AND operation would be as follows:

- If (0xC5 AND 0x80 = 0x80), then forward error information to RS-232 interface
- If (0xC5 AND 0x40 = 0x40), then forward error information to CAN interface
- If (0xC5 AND 0x04 = 0x04), then forward LIN messages to RS-232 interface
- If (0xC5 AND 0x02 = 0x02), then forward LIN messages to CAN interface

Retrieve Filter Mask

This command retrieves the configured acceptance mask of the LIN interface in a module. The LIN acceptance mask, in conjunction with the acceptance code, is a filter used to tell the LIN interface which messages are received. This filter is referring to the ID of a LIN message.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	0	1	1	0	1
0	0, 0x0			0, 0x0					2, 0x2		13, 0xD					
0, 0x0									141, 0x8D							

Parameters	none
Checksum	SC(0x00) XOR CC(0x8D) XOR PX(0x00) = 0x8D
Message to send	STX(0x02) – SC(0x00) – CC(0x8D) – CHK(0x8D)
Positive response	STX(0x02) – SC(0x02) – P1(0xFF) – P2(0x00) – CHK(0xFD)

A positive response has two parameters. In this example the returned value is the default value for a filter mask (0xFF).

Retrieve Filter Code

This command retrieves the configured acceptance code of the LIN interface in a module. The LIN acceptance code, in conjunction with the acceptance mask, is a filter used to tell the LIN interface which messages are received. This filter is referring to the ID of a LIN message.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)										Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code							
0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0		
0	0, 0x0			0, 0x0				2, 0x2		14, 0xE							
0, 0x0								142, 0x8E									

Parameters	none
Checksum	SC(0x00) XOR CC(0x8E) XOR PX(0x00) = 0x8E
Message to send	STX(0x02) – SC(0x00) – CC(0x8E) – CHK(0x8E)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. In this example the returned value is the default value for a filter code (0x00).

Retrieve Slave Mask Status

This command retrieves the current slave mask status (Publisher or Subscriber) for one or all LIN message IDs. This command needs two parameters (two bytes). The first parameter (P1) has two possible values, 0x00 or 0x01. Using 0 means that the slave mask status of all IDs is returned. In this case the second parameter does not matter, but still must be sent. Specifying 1 means that the second parameter (P2) indicates a valid LIN ID. For this LIN ID the slave mask status is returned.

P1	P2	Description
0	Ignored	Retrieves the slave mask status of all 64 IDs (0x00 to 0x3F)
1	LIN ID (0x00…0x3F)	Retrieves the slave mask status of the LIN ID passed with parameter P2

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Case A: request of slave mask status for all LIN IDs (P1 = 0)

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		1	0	0	0	1	1	1	1
0	0, 0x0			2, 0x2					2, 0x2		15, 0xF					
2, 0x2									143, 0x8F							

Parameters	Two parameters (two bytes) are required. The first parameter indicates that the operation should be done for all IDs (0x00), the second parameter is not used but must be sent, too.
Checksum	PX = P1(0x00) XOR P2(0x00) = 0x00 SC(0x02) XOR CC(0x8F) XOR PX(0x00) = 0x8D
Message to send	STX(0x02) – SC(0x02) – CC(0x8F) – P1(0x00) – P2(0x00) – CHK(0x8D)
Positive response	STX(0x02) – SC(0x08) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5 (0x00) – P6(0x00) – P7(0x00) – P8(0x00) – CHK(0x02)

A positive response has 8 parameters. In this example the returned value is the default value for a slave mask where all slave mask IDs are Subscribers.

Case B: request of slave mask status for a single ID (P1 = 1)

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		1	0	0	0	1	1	1	1
0	0, 0x0			2, 0x2					2, 0x2		15, 0xF					
2, 0x2									143, 0x8F							

Parameters	Two parameters (two bytes) are required. The first parameter indicates that the operation should be done for a single ID (0x01), the second parameter indicates the corresponding ID. In this example this is ID 57 (0x39).
Checksum	PX = P1(0x01) XOR P2(0x39) = 0x38 SC(0x02) XOR CC(0x8F) XOR PX(0x38) = 0xB5
Message to send	STX(0x02) – SC(0x02) – CC(0x8F) – P1(0x01) – P2(0x39) – CHK(0xB5)
Positive response	STX(0x02) – SC(0x02) – P1(0x39) – P2(0x00) – CHK(0x3B)

A positive response has two parameters. In this example the returned value is the default value for a slave mask of ID 57 (0x39) which is a Subscriber. P1 contains the requested ID and P2 the associated value.

The mask status is treated as boolean value. If the returned value is 1, the ID is a Publisher, if it is 0, then it is a Subscriber. The slave mask status for all IDs is returned in 8 data bytes, each status is delivered through a single bit.

Data Bytes	**ID/Bit Assignment**							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P1 (IDs 0…7)	0x07	0x06	0x05	0x04	0x03	0x02	0x01	0x00
P2 (IDs 8…15)	0x0F	0x0E	0x0D	0x0C	0x0B	0x0A	0x09	0x08
P3 (IDs 16…23)	0x17	0x16	0x15	0x14	0x13	0x12	0x11	0x10
P4 (IDs 24…31)	0x1F	0x1E	0x1D	0x1C	0x1B	0x1A	0x19	0x18
P5 (IDs 32…39)	0x27	0x26	0x25	0x24	0x23	0x22	0x21	0x20
P6 (IDs 40…47)	0x2F	0x2E	0x2D	0x2C	0x2B	0x2A	0x29	0x28
P7 (IDs 48…55)	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30
P8 (IDs 56…63)	0x3F	0x3E	0x3D	0x3C	0x3B	0x3A	0x39	0x38

For example, if you receive P3 with a value of 0x82 (0b10000010), this means that the IDs 16 (0x10), 18 (0x12), 19 (0x13), 20 (0x14), 21 (0x15), and 22 (0x16) have the slave mask status Subscriber and the IDs 17 (0x11) and 23 (0x17) have the slave mask status Publisher.

Retrieve Master Status

This command retrieves a value indicating the master status of the LIN interface of a module (active/inactive). The status “active” means that the master scheduler has entries and they are transmitted.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	0	0	0	0
0	0, 0x0			0, 0x0					2, 0x2		16, 0x10					
0, 0x0									144, 0x90							

Parameters	none
Checksum	SC(0x00) XOR CC(0x90) XOR PX(0x00) = 0x90
Message to send	STX(0x02) – SC(0x00) – CC(0x90) – CHK(0x90)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. If the value formed by P1 and P2 (in this case 0x00) is equals 0, the LIN master is inactive, otherwise the LIN master is active.

Retrieve LIN Bus Termination (Master Termination)

This command retrieves a value which indicates, if the module is configured as master or slave (the so called termination). The default termination for a module is “Slave”.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	0	0	0	1
0	0, 0x0			0, 0x0					2, 0x2		17, 0x11					
0, 0x0									145, 0x91							

Parameters	none
Checksum	SC(0x00) XOR CC(0x91) XOR PX(0x00) = 0x91
Message to send	STX(0x02) – SC(0x00) – CC(0x91) – CHK(0x91)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. If the value formed by P1 and P2 (in this case 0x00) equals 0, this means “slave termination”, otherwise “master termination”.

Retrieve Message Transmission Timeouts

This command retrieves a couple of values which indicate the time-outs (in milliseconds) for the transmission of messages. The time-outs are for bus idle messages, messages with the frame length 8, frame length 4, and frame length 2. The response to this command expresses the timeouts in that order. The length of each timeout is 2 bytes.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0
0	0, 0x0			0, 0x0				2, 0x2		18, 0x12					
0, 0x0								146, 0x92							

Parameters	none
Checksum	SC(0x00) XOR CC(0x92) XOR PX(0x00) = 0x92
Message to send	STX(0x02) – SC(0x00) – CC(0x92) – CHK(0x92)
Positive response	STX(0x02) – SC(0x08) – P1(0x16) – P2(0x05) – P3(0x0A) – P4(0x00) – P5(0x07) – P6(0x00) – P7(0x05) – P8(0x00) – CHK(0x13)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1, P2	Bus idle	0x0516 (default)	1302 ms
P3, P4	Frame length 8	0x000A (default)	10 ms
P5, P6	Frame length 4	0x0007 (default)	7 ms
P7, P8	Frame length 2	0x0005 (default)	5 ms

Retrieve Message Retries

This command retrieves a value which indicates the number of retries for a LIN response message causing a “slave not responding” timeout (only when the master status is inactive).

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	1
0	0, 0x0			0, 0x0				2, 0x2		19, 0x13					
0, 0x0								147, 0x93							

Parameters	none
Checksum	SC(0x00) XOR CC(0x93) XOR PX(0x00) = 0x93
Message to send	STX(0x02) – SC(0x00) – CC(0x93) – CHK(0x93)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. In this example the returned value is the default value 0 for message retries.

Retrieve Scheduler Entries Count

This command retrieves a value which indicates the count of configured and valid entries of the master scheduler in a module. The maximum number of scheduler entries is 50.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	0	1	0	0
0	0, 0x0			0, 0x0					2, 0x2		20, 0x14					
0, 0x0									148, 0x94							

Parameters	none
Checksum	SC(0x00) XOR CC(0x94) XOR PX(0x00) = 0x94
Message to send	STX(0x02) – SC(0x00) – CC(0x94) – CHK(0x94)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters. In this example the returned “count of entries” is the default value 0, since as default the scheduler is empty.

Retrieve Scheduler Entry

This command retrieves a couple of values which indicate a configuration of a valid entry from the master scheduler in a module. A scheduler entry is a conjunction of a LIN message ID and its corresponding timestamp. A timestamp is the time difference between two frames measured from its initiation (trailing edge of the signal “sync break” from each frame). The entry to be returned from the scheduler list is given as parameter for this command. The parameter corresponds with the index of the entry in the scheduler list. The first possible scheduler entry index is 1 and the last is 50.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	0	1	0	1	0	1
0	0, 0x0			1, 0x1					2, 0x2		21, 0x15					
1, 0x1									149, 0x95							

Parameters	One parameter (one byte) is required to indicate the index of the scheduler entry to be returned. In this example it is assumed that the scheduler has an entry with the following configuration: ID: 0x10, timestamp: 50 ms
Checksum	PX = P1(0x01) = 0x01 SC(0x01) XOR CC(0x95) XOR PX(0x01) = 0x95
Message to send	STX(0x02) – SC(0x01) – CC(0x95) – P1(0x01) – CHK(0x95)
Positive response	STX(0x02) – SC(0x03) – P1(0x32) – P2(0x00) – P3(0x10) – CHK(0x21)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1, P2	Timestamp	0x0032	50 ms
P3	LIN ID	0x10	LIN ID 0x10

Retrieve RCV Error Counter

This command retrieves the value of the receive error counter of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	0	1	1	0
0	0, 0x0			0, 0x0					2, 0x2		22, 0x16					
0, 0x0									150, 0x96							

Parameters	none
Checksum	SC(0x00) XOR CC(0x96) XOR PX(0x00) = 0x96
Message to send	STX(0x02) – SC(0x00) – CC(0x96) – CHK(0x96)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the RCV error counter. In this example the default value is returned with P1 and P2: 0x0 = 0 errors

Retrieve XMT Error Counter

This command retrieves the value of the transmit error counter of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	0	1	1	1
0	0, 0x0			0, 0x0					2, 0x2		23, 0x17					
0, 0x0									151, 0x97							

Parameters	none
Checksum	SC(0x00) XOR CC(0x97) XOR PX(0x00) = 0x97
Message to send	STX(0x02) – SC(0x00) – CC(0x97) – CHK(0x97)
Positive response	STX(0x02) – SC(0x02) – P1(0x00) – P2(0x00) – CHK(0x02)

A positive response has two parameters with the value of the XMT error counter. In this example the default value is returned with P1 and P2: 0x0 = 0 errors

Retrieve All Error Counters

This command retrieves the values of the errors counters of the LIN interface. The returned values are (in this order) from the RCV error counter, XMT error counter, bit error counter, checksum error counter, ID checksum error counter, “Slave not responding” error counter, and “Inconsistent synchronization bytes” error counter. Each value is 2 bytes long.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	1	0	0	0
0	0, 0x0			0, 0x0					2, 0x2		24, 0x18					
0, 0x0									152, 0x98							

Parameters	none
Checksum	SC(0x00) XOR CC(0x98) XOR PX(0x00) = 0x98
Message to send	STX(0x02) – SC(0x00) – CC(0x98) – CHK(0x98)
Positive response	STX(0x02) – SC(0x0E) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0x00) – P7(0x00) – P8(0x00) – P9(0x00) – P10(0x00) – P11(0x00) – P12(0x00) – P13(0x00) – P14(0x00) – CHK(0x0E)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1, P2	RCV error counter	0x0 (default)	0 errors
P3, P4	XMT error counter	0x0 (default)	0 errors
P5, P6	Bit error counter	0x0 (default)	0 errors
P7, P8	Checksum error counter	0x0 (default)	0 errors
P9, P10	ID checksum error counter	0x0 (default)	0 errors
P11, P12	“Slave not responding” error counter	0x0 (default)	0 errors
P13, P14	“Inconsistent synchronization bytes” error counter	0x0 (default)	0 errors

Retrieve Slave ID + Data Configuration Information

This command retrieves information about the slave configuration of a module. As a response to this command the following information is received: the status of the “Slave ID + Data Configuration” field (enabled or disabled) and the ID identifying the slave and the configured data associated with that slave ID. The length of the data depends on the slave ID. A LIN ID has a length of 6 bits (0x00…0x3F) and a data length as follows:

ID range	Data length
0x00 to 0x1F	2 bytes
0x20 to 0x2F	4 bytes
0x30 to 0x3F	8 bytes

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	1	0	0	1
0	0, 0x0			0, 0x0					2, 0x2		25, 0x19					
0, 0x0									153, 0x99							

Parameters	none
Checksum	SC(0x00) XOR CC(0x99) XOR PX(0x00) = 0x99
Message to send	STX(0x02) – SC(0x00) – CC(0x99) – CHK(0x99)
Positive response	STX(0x02) – SC(0x0A) – P1(0x00) – P2(0x3F) – P3(0xFF) – P4(0xFF) – P5(0xFF) – P6(0xFF) – P7(0xFF) – P8(0xFF) – P9(0xFF) – P10(0xFF) – CHK(0x35)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1	Field status (0 = disabled, 1 = enabled)	0x00 (default)	disabled
P2	Slave ID	0x3F	Results in data length of 8 bytes
P3...P10	Configuration data (bytes 1 to 8)	each 0xFF (default)	

In this example it is assumed that the lengths of the IDs are correlated to the default lengths described in the table “ID range / Data length” above. A new feature of the module allows you to set any length for any ID. This results in a varying length of a positive response without following the rules in the mentioned table.

Retrieve Bit Rate Recognition for Synchronization Field Status

This command retrieves a value which indicates, if the bit rate recognition is active or not. The automatic bit rate recognition only works for received frames and when a synchronization break was correctly recognized.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	1	0	1	0
0	0, 0x0			0, 0x0					2, 0x2		26, 0x1A					
0, 0x0									154, 0x9A							

Parameters	none
Checksum	SC(0x00) XOR CC(0x9A) XOR PX(0x00) = 0x9A
Message to send	STX(0x02) – SC(0x00) – CC(0x9A) – CHK(0x9A)
Positive response	STX(0x02) – SC(0x02) – P1(0x01) – P2(0x00) – CHK(0x03)

A positive response has two parameters. In this example the default value for the bit rate recognition field 0x01 (active) is returned.

Retrieve Frame Length

This command retrieves the assigned frame length for one or all 64 LIN message IDs. As shown for the command *Retrieve Slave ID + Data Configuration Information* on page 51 the frame length of a LIN message depends on the ID. An extension on the capabilities of the module lets you set the frame length of any ID from 0 to 8 bytes, independent of the ID (the default value for the frame lengths follows the rule of the ID as mentioned above). For this reason the frame lengths of all IDs or the frame length of a single ID can be retrieved.

This command has two parameters. Depending on the needed data, the parameters are as follows:

P1	P2	Description
0	Not used, ignored	Retrieves the frame lengths of IDs 0x00 to 0x0F
1	Not used, ignored	Retrieves the frame lengths of IDs 0x10 to 0x1F
2	Not used, ignored	Retrieves the frame lengths of IDs 0x20 to 0x2F
3	Not used, ignored	Retrieves the frame lengths of IDs 0x30 to 0x3F
4	LIN ID (0x00…0x3F)	Retrieves the frame length of the ID passed by parameter P2

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Case A: Retrieving a set of frame lengths (P1 within 0 to 3)

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	0	1	0	0	1	1	0	1	1
0	0, 0x0			2, 0x2				2, 0x2		27, 0x1B					
0, 0x0								155, 0x9B							

Parameters	This command needs two parameters (two bytes). The first parameter indicates the range of IDs. The second parameter is not used, but must be sent. In this example the frame length of the last 16 IDs (0x30 to 0x3F) is retrieved.
Checksum	PX = P1(0x03) XOR P2(0x00) = 0x03 SC(0x02) XOR CC(0x9B) XOR PX(0x03) = 0x9A
Message to send	STX(0x02) – SC(0x02) – CC(0x9B) – P1(0x03) – P2(0x00) – CHK(0x9A)
Positive response	STX(0x02) – SC(0x08) – P1(0x88) – P2(0x88) – P3(0x88) – P4(0x88) – P5(0x88) – P6(0x88) – P7(0x88) – P8(0x88) – CHK(0x08)

A positive response has 8 parameters. In this example the default value for the frame lengths of the IDs 0x30 to 0x3F is returned. Each data byte contains the lengths for two IDs (order: four least significant bits, 4 most significant bits). For example, 0x84 means that one ID (0x16) is 4 bytes and the next one (0x17) is 8 bytes long.

Param(s)	Description	Example value	Meaning
P1…P8	Length of frames for given IDs	0x88	8 bytes for each frame

Case B: Retrieving the frame length of a single ID (P1 = 4)

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	0	1	0	0	1	1	0	1	1
0	0, 0x0			2, 0x2				2, 0x2		27, 0x1B					
0, 0x0								155, 0x9B							

Parameters	This command needs two parameters (two bytes). The first parameter indicates that the second parameter contains an ID. The frame length for this ID shall be returned. In this example the frame length of ID 0x07 is retrieved.
Checksum	PX = P1(0x04) XOR P2(0x07) = 0x03 SC(0x02) XOR CC(0x9B) XOR PX(0x03) = 0x9A
Message to send	STX(0x02) – SC(0x02) – CC(0x9B) – P1(0x04) – P2(0x07) – CHK(0x9A)
Positive response	STX(0x02) – SC(0x02) – P1(0x07) – P2(0x02) – CHK(0x07)

A positive response has two parameters. In this example the default value for the frame length of the ID 0x07 is returned. The first data byte contains the ID and the second data byte the frame length:

Param(s)	Description	Example value	Meaning
P1	LIN ID	0x07	ID 0x07
P2	Frame length	0x02 (default)	2 bytes

Retrieve Checksum Type

This command retrieves the current checksum type for a single or all 64 LIN message IDs. This can either be the classic type (LIN 1.x) or the enhanced type (LIN 2.0). This command has two parameters. Depending on the needed data, the parameters can be as follows:

P1	P2	Description
0	Not used, ignored	Retrieves the checksum type of all 64 LIN IDs (0x00 to 0x3F)
1	LIN ID (0x0…0x3F)	Retrieves the checksum type of the LIN ID passed as parameter P2

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Case A: Retrieving all checksum types (P1 = 0)

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		1	0	0	1	1	1	0	0
0	0, 0x0			2, 0x2					2, 0x2		28, 0x1C					
2, 0x2									156, 0x9C							

Parameters	This command needs two parameters (two bytes). The first parameter indicates that in this case the checksum types of all IDs are returned. The second parameter is not used, but must be sent.
Checksum	PX = P1(0x00) XOR P2(0x00) = 0x00 SC(0x02) XOR CC(0x9C) XOR PX(0x00) = 0x9E
Message to send	STX(0x02) – SC(0x02) – CC(0x9C) – P1(0x00) – P2(0x00) – CHK(0x9E)
Positive response	STX(0x02) – SC(0x08) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0x00) – P7(0x00) – P8(0x00) – CHK(0x08)

A positive response has 8 parameters. In this example the default value for the checksum type of all IDs (classic) is returned. Each data byte contains the format of 8 IDs, beginning with the IDs 0x00 to 0x07 in data byte 1 to IDs 0x38 to 0x3F in data byte 8. IDs are ordered incrementing from the least significant bit. A 0 represents the classic checksum type; a 1 represents the enhanced checksum type.

Param(s)	Description	Example value	Meaning
P1…P8	Format of checksum for given IDs (each bit)	0b00000000 (default)	Classic checksum type for all IDs

Case B: Retrieving the checksum type of a single ID (P1 = 1)

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		1	0	0	1	1	1	0	0
0	0, 0x0			2, 0x2					2, 0x2		28, 0x1C					
2, 0x2									156, 0x9C							

Parameters	This command needs two parameters (two bytes). The first parameter indicates that in this case the checksum type of a specific ID is returned. The second parameter contains the LIN ID. In this example the checksum type for ID 0x07 is retrieved.
Checksum	PX = P1(0x01) XOR P2(0x07) = 0x06 SC(0x02) XOR CC(0x9C) XOR PX(0x06) = 0x98
Message to send	STX(0x02) – SC(0x02) – CC(0x9C) – P1(0x01) – P2(0x07) – CHK(0x98)
Positive response	STX(0x02) – SC(0x02) – P1(0x07) – P2(0x00) – CHK(0x05)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1	LIN ID	0x07	LIN ID 0x07
P2	Checksum type	0x00 (default)	Classic checksum type

Retrieve CAN ID for Slave Activation

This command retrieves the configured CAN ID used to activate a module as LIN slave through CAN messaging as well as an indication, if the CAN ID is 11 or 29 bits long and if the status of this field is active.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	0	1	1	1	0	1
0	0, 0x0			0, 0x0					2, 0x2		29, 0x1D					
0, 0x0									157, 0x9D							

Parameters	none
Checksum	SC(0x02) XOR CC(0x9D) XOR PX(0x00) = 0x9D
Message to send	STX(0x02) – SC(0x00) – CC(0x9D) – CHK(0x9D)
Positive response	STX(0x02) – SC(0x04) – P1(0xF0) – P2(0x07) – P3(0x00) – P4(0x00) – CHK(0xF3)

Description of the positive response:

Param(s)	Description	Example value	Meaning
P1	Least significant byte of the CAN ID	0xF0	CAN ID 0x07F0
P2	2nd byte of the CAN ID (only 3 bits for 11-bit CAN ID)	0x07	
P3	3rd byte of the CAN ID (only 29-bit CAN ID)	0x00	
P4	Bits 0 to 4: Most significant part of 29-bit CAN ID Bit 5: Not used Bit 6: Slave active Bit 7: Extended CAN ID	0x00	Slave not active Standard CAN ID (11 bits)

2.4.3 Configuration Commands

Set Activation Status

This command sets a value in the module which indicates, if the CAN features are active.

The data to be sent or received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	0	1	1	1	1	0
0	0, 0x0			1, 0x1					2, 0x2		30, 0x1E					
1, 0x1									158, 0x9E							

Parameters	One parameter (one byte) is required in order to specify the activation status of the LIN interface. This parameter byte works as a boolean value. If the parameter is 0, the interface will be deactivated, otherwise it will be activated. In the example the interface will be activated.
Checksum	P1 = 0x01 SC(0x01) XOR CC(0x9E) XOR PX(0x01) = 0x9E
Message to send	STX(0x02) – SC(0x01) – CC(0x9E) – P1(0x01) – CHK(0x9E)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: The activation of the LIN interface does not require an initialization using the initialization command.

Set Bit Rate

This command directs the module to change the bit rate used by the LIN interface. The possible values for a LIN bit rate are in the range of 1000 to 20000 bit/s.

The data to be sent and received starts with the LSB. Example: 0x12C0 = P1:0xC0 and P2:0x12

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	1	0		1	0	0	1	1	1	1	1
0	0, 0x0			2, 0x2					2, 0x2		31, 0x1F					
2, 0x2									159, 0x9F							

Parameters	Two parameters (two bytes) are required in order to specify the desired bit rate. In this example the default bit rate of the LIN interface is used: 19200 bit/s (0x4B00).
Checksum	PX = P1(0x00) XOR P2(0x4B) = 0x4B SC(0x02) XOR CC(0x9F) XOR PX(0x4B) = 0xD6
Message to send	STX(0x02) – SC(0x02) – CC(0x9F) – P1(0x00) – P2(0x4B) – CHK(0xD6)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: The configured bit rate differs from the used bit rate. The used bit rate is the speed being used to send and receive messages from/to the module in a determined time. A configured bit rate is the result of sending a bit rate configuration message causing the change in the bit rate but not initializing it. To activate the new configured value refer to section 2.4.1 *Initialization and Write Commands* on page 37.

Set Forward Mask

This command sets the forward mask of the LIN interface in a module. This mask tells an interface to which other interface(s) the received messages are forwarded, and if error information should be monitored. The errors can be redirected to the RS-232 and/or CAN interface, too.

The forward mask value is a bit combination (bitwise-OR operation) of the values in the following table:

Mask value	Meaning
0x80	Forward error information to RS-232 interface
0x40	Forward error information to CAN interface
0x04	Forward LIN messages to RS-232 interface
0x01	Forward LIN messages to CAN interface

For example, a forward mask of 0xC5 means that the error information as well as the messages of the LIN interface will be forwarded to both the CAN and the RS-232 interfaces.

Note: In order to forward error information to the RS-232 interface (mask value 0x80) the corresponding message forwarding (0x04) must be activated, too.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	1	0	0	0	0	0
0	0, 0x0			1, 0x1					2, 0x2		32, 0x20					
1, 0x1									160, 0xA0							

Parameters	One parameter (one byte) is required. In this example the mask to monitor errors and, if any, to forward it to the CAN interface is used. The value to send is 0x40 (forward error information to CAN interface).
Checksum	P1 = 0x40 SC(0x01) XOR CC(0xA0) XOR PX(0x40) = 0xE1
Message to send	STX(0x02) – SC(0x01) – CC(0xA0) – P1(0x40) – CHK(0xE1)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Filter Mask

This command sets an acceptance mask for the LIN interface in a module. The LIN acceptance mask, in conjunction with the acceptance code, is a filter used to tell the LIN interface which messages are received. This filter is referring to the ID of a LIN message.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	1	0	0	0	0	1
0	0, 0x0			$1_{(1)}$, $0x1_{16)}$					2, 0x2		33, $0x21_{(6)}$					
1, 0x1									161, 0xA1							

Parameters	One parameter (one byte) is required. An acceptance mask is 1 byte long (a LIN message ID is one byte long). In this example the default value for the LIN filter mask (0xFF) is used.
Checksum	PX = P1(0xFF) = 0xFF SC(0x01) XOR CC(0xA1) XOR PX(0xFF) = 0x5F
Message to send	STX(0x02) – SC(0x01) – CC(0xA1) – P1(0xFF) – CHK(0x5F)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Filter Code

This command sets the acceptance code of the LIN interface in a module. The LIN acceptance code, in conjunction with the acceptance mask, is a filter used to tell the LIN interface which messages are received. This filter is referring to the ID of a LIN message.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	1	0	0	0	1	0
0	0, 0x0			1, 0x1					1, 0x1		34, 0x22					
1, 0x1									162, 0xA2							

Parameters	One parameter (one byte) is required. An acceptance code is 1 byte long (a LIN message ID is one byte long). In this example the default value for the LIN filter code (0x00) is used.
Checksum	PX = P1(0x00) = 0x00 SC(0x01) XOR CC(0xA2) XOR PX(0x00) = 0xA3
Message to send	STX(0x02) – SC(0x01) – CC(0xA2) – P1(0x00) – CHK(0xA3)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Master Status

This command sets the status of the LIN master in a module. The activation of this feature means that the master scheduler has entries and they are transmitted. If the scheduler is empty, trying to activate the master will return an error.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0
0	0, 0x0			1, 0x1				2, 0x2		36, 0x24					
1, 0x1								164, 0xA4							

Parameters	One parameter (one byte) is required in order to specify the activation status of the LIN master. This parameter byte works as a boolean value. If the parameter is 0, the master will be inactive, otherwise it will be active. In the example the master is active and it is assumed that the master scheduler has at least one valid entry. For more information about inserting a new scheduler entry into the master scheduler see command *Insert Scheduler Entry* on page 68.
Checksum	PX = P1(0x01) = 0x01 SC(0x01) XOR CC(0xA4) XOR PX(0x01) = 0xA4
Message to send	STX(0x02) – SC(0x01) – CC(0xA4) – P1(0x01) – CHK(0xA4)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set LIN Bus Termination (Master Termination)

This command sets the module termination to Master or Slave. The default termination for a module is Slave.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)								
X	Sequence			Param Byte Count					Interface		Command Code						
0	0	0	0	0	0	0	1		1	0	1	0	0	1	0	1	
0	0, 0x0			1, 0x1					2, 0x2		37, 0x25						
1, 0x1									165, 0xA5								

Parameters	One parameter (one byte) is required in order to specify the termination of the module. This parameter byte works as a boolean value. If the parameter is 0, the termination is set to Slave (default), otherwise it is set to Master. In the example, the termination is set to Master using the value 0x01 as parameter.
Checksum	PX = P1(0x01) = 0x01 SC(0x01) XOR CC(0xA5) XOR PX(0x01) = 0xA5
Message to send	STX(0x02) – SC(0x01) – CC(0xA5) – PX(0x01) – CHK(0xA5)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Message Transmission Timeouts

This command sets the timeouts (in milliseconds) for the transmission of the different kind of messages. The timeouts to set are for bus idle messages as well as messages with frame lengths 8, 4, 2 bytes. Every timeout has a length of 2 bytes resulting in 8 needed parameters. The order for the parameters is:

- P1 – P2: bus idle
- P3 – P4: frame length 8
- P5 – P6: frame length 4
- P7 – P8: frame length 2

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	1	0	0	0		1	0	1	0	0	1	1	0
0	0, 0x0			8, 0x8					2, 0x2		38, 0x26					
8, 0x8									166, 0xA6							

<table>
<tr><td>Parameters</td><td>Eight parameters (8 bytes) are required in order to specify all 4 timeouts (two bytes per timeout starting with the LSB). In this example the default timeouts are set.
<table>
<tr><th>Timeout for</th><th>Period</th><th>Parameter Value</th></tr>
<tr><td>Bus idle</td><td>1302 ms</td><td>0x0516</td></tr>
<tr><td>Frame length 8</td><td>10 ms</td><td>0x000A</td></tr>
<tr><td>Frame length 4</td><td>7 ms</td><td>0x0007</td></tr>
<tr><td>Frame length 2</td><td>5 ms</td><td>0x0005</td></tr>
</table></td></tr>
<tr><td>Checksum</td><td>PX = P1(0x16) XOR P2(0x05) XOR P3(0x0A) XOR P4(0x00) XOR P5 (0x07) XOR P6(0x00) XOR P7(0x05) XOR P8(0x00) = 0x1B
SC(0x08) XOR CC(0xA6) XOR PX(0x1B) = 0xB5</td></tr>
<tr><td>Message to send</td><td>STX(0x02) – SC(0x08) – CC(0xA6) – P1(0x16) – P2(0x05) – P3(0x0A) – P4(0x00) – P5(0x07) – P6(0x00) – P7(0x05) – P8(0x00) – CHK(0x5B)</td></tr>
<tr><td>Positive response</td><td>STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)</td></tr>
</table>

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Message Retries

This command sets the number of retries for a LIN response message which causes a “Slave not responding” timeout (only when the master status is inactive). The default value is 0 (no repetitions). The maximum value is 255 retries.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

<table>
<tr><th colspan="5">Sequence Code (SC)</th><th colspan="8">Control Code (CC)</th></tr>
<tr><th>X</th><th colspan="3">Sequence</th><th colspan="4">Param Byte Count</th><th colspan="2">Interface</th><th colspan="6">Command Code</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td colspan="3">0, 0x0</td><td colspan="4">1, 0x1</td><td colspan="2">2, 0x2</td><td colspan="6">39, 0x27</td></tr>
<tr><td colspan="5">1, 0x1</td><td colspan="8">167, 0xA7</td></tr>
</table>

Parameters	One parameter (one byte) is required in order to specify the number of retries for LIN response messages. In this example this field will be set to its maximum value which is 255 retries (0xFF).
Checksum	PX = P1(0xFF) = 0xFF SC(0x01) XOR CC(0xA7) XOR PX(0xFF) = 0x59
Message to send	STX(0x02) – SC(0x01) – CC(0xA7) – P1(0xFF) – CHK(0x59)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Insert Scheduler Entry

This command inserts a new scheduler entry into the master scheduler in a module. A scheduler entry is a combination of an LIN message ID and its corresponding timestamp. A timestamp is the time difference between two frames measured from their initiation (falling edge of the signal “Sync. break” of a frame), in other words, the frequency in which a frame is transmitted. By using this command new entries can be included in the scheduler. Up to 50 entries are permitted. An existing entry can not be modified. In order to make changes, the whole list must be erased and filled again with the modified values. The length of the LIN message ID is 1 byte and of the timestamp 2 bytes.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	1	1	0	1	0	1	0	0	0
0	0, 0x0			3, 0x3				2, 0x2		40, 0x28					
3, 0x3								168, 0xA8							

Parameters	Three parameters (three bytes) are required to indicate a valid LIN scheduler entry, one for the LIN message ID and two for its timestamp. In this example an entry is inserted in the scheduler conformed by an ID 0x05 and a timestamp of 100 ms (0x64). The order of the parameters is as follows: Timestamp: 100 ms (0x0064) LIN ID: 0x05
Checksum	PX = P1(0x64) XOR P2(0x00) XOR P3(0x05) = 0x01 SC(0x03) XOR CC(0xA8) XOR PX(0x61) = 0xCA
Message to send	STX(0x02) – SC(0x03) – CC(0xA8) – P1(0x64) – P2(0x00) – P3(0x05) – CHK(0xCA)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Slave ID + Data Configuration

This command sets the slave configuration of a module. This configuration is formed by the status of the "Slave ID + Data Configuration" field (enabled or disabled). This ID identifies the slave and the configured data associated with the slave ID. The length of the data depends on the slave ID. If the used frame length is the default, the length to be used can be found using the following table:

ID Range	Data Length
0x00 to 0x1F	2 Bytes
0x20 to 0x2F	4 Bytes
0x30 to 0x3F	8 Bytes

If the frame length of the used LIN ID was configured with another length, than that length must be used; otherwise this command will fail. For more information about reading the frame length see command *Retrieve Frame Length* on page 53. For frame length configuration see command *Set Frame Length* on page 71.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	1	0	0		1	0	1	0	1	0	0	1
0	0, 0x0			4, 0x4					2, 0x2		41, 0x29					
4, 0x4									169, 0xA9							

Parameters	In this example the ID 0x010 is set as slave ID. The use of the default length is assumed. The field “Slave ID + Data Configuration” is set active, data[0] to 0x1A, and data[1] to 0xBB (in this case 2 data bytes, see the table above). Therefore four parameters (four bytes) are required to indicate the activation status of this field (one byte), the ID to be set (one byte), and the data bytes (2 bytes). The order of the parameters is as follows: Field status: active (1) LIN ID: 0x10 Data byte 0: 0x1A Data byte 1: 0xBB
Checksum	PX = P1(0x01) XOR P2(0x10) XOR P3(0x1A) XOR P4(0xBB) = 0xB0 SC(0x04) XOR CC(0xA9) XOR PX(0xB0) = 0x1D
Message to send	STX(0x02) – SC(0x04) – CC(0xA9) – P1(0x01) – P2(0x10) – P3(0x1A) – P4(0xBB) – CHK(0x1D)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Configure Bit Rate Recognition for Synchronization Field

This command sets the bit rate recognition status. The automatic bit rate recognition only works for frames received by the module and when a synchronization break is correctly recognized.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	1		1	0	1	0	1	0	1	0
0	0, 0x0			1, 0x1					2, 0x2		42, 0x2A					
1, 0x1									170, 0xAA							

Parameters	One parameter (one byte) is required in order to specify the status of the automatic bit rate recognition feature of the module. If the parameter is 1, this feature is activated (default), else it is deactivated. In this example the bit rate recognition is activated.
Checksum	PX = P1(0x01) = 0x01 SC(0x01) XOR CC(0xAA) XOR PX(0x01) = 0xAA
Message to send	STX(0x02) – SC(0x01) – CC(0xAA) – P1(0x01) – CHK(0xAA)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Frame Length

This command sets the frame length of one or all 64 LIN message IDs. As shown for the command *Set Slave ID + Data Configuration* on page 69, the frame length of a LIN message depends on the ID. An extension on the capabilities of the module makes it possible to set the frame length of any ID from 0 to 8 bytes (the default value for the frame lengths follows the rule for the ID as mentioned above).

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

This command can be implemented in two ways. The first is setting the frame length for ranges of IDs in only one operation and the second is setting the frame length for a single ID. Depending on the kind of operation, the parameter count is either 3 bytes (for single ID) or 9 bytes (for range of IDs).

Assigning a frame length to a range of IDs

There are 4 ranges of IDs (from 0 to 3). Each range of IDs configures the frame length for 16 IDs of the 64 existing in a module. The first parameter (P1) of a frame length assignment indicates the range of IDs being configured. The frame length of those 16 IDs is assigned by using a parameter (one byte) for a pair of IDs (8 bytes, P2 to P9). The following tables show the ID ranges:

Range code (P1)	ID range	
	from	to
0	ID 00 (0x00)	ID 15 (0x0F)
1	ID 16 (0x10)	ID 31 (0x1F)
2	ID 32 (0x20)	ID 47 (0x2F)
3	ID 48 (0x30)	ID 63 (0x3F)

According to the selected range, the next 8 parameters (8 bytes) contain the frame length of the associated 16 IDs of that range. 4 bits of each parameter express the frame length of one ID. The following table shows which bits represent the frame length of an ID:

P1	Bits of P2…P9	P2	P3	P4	P5	P6	P7	P8	P9
0	0…3	ID 00	ID 02	ID 04	ID 06	ID 08	ID 10	ID 12	ID 14
	4…7	ID 01	ID 03	ID 05	ID 07	ID 09	ID 11	ID 13	ID 15
1	0…3	ID 16	ID 18	ID 20	ID 22	ID 24	ID 26	ID 28	ID 30
	4…7	ID 17	ID 19	ID 21	ID 23	ID 25	ID 27	ID 29	ID 31
2	0…3	ID 32	ID 34	ID 36	ID 38	ID 40	ID 42	ID 44	ID 46
	4…7	ID 33	ID 35	ID 37	ID 39	ID 41	ID 43	ID 45	ID 47
3	0…3	ID 48	ID 50	ID 52	ID 54	ID 56	ID 58	ID 60	ID 62
	4…7	ID 49	ID 51	ID 53	ID 55	ID 57	ID 59	ID 61	ID 63

For example, the selected range is 0 and the desired frame length for each ID should be 8, except the IDs 10 (4 bytes) and ID 11 (6 bytes). The parameters then are:

P2	P3	P4	P5	P6	P7	P8	P9
0x88	0x88	0x88	0x88	0x88	0x**64**	0x88	0x88

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	1	0	0	1		1	0	1	0	1	0	1	1
0	0, 0x0			9, 0x9					2, 0x2		43, 0x2B					
9, 0x9									171, 0xAB							

Parameters	This command needs 9 parameters (9 bytes). The first parameter indicates the range of IDs used for frame length setting. The remaining 8 parameters P2 to P9 indicate the frame length for the 16 IDs associated with the selected range. In this example the frame lengths for the last 16 IDs (range number 3) are set to their default value (8 bytes).
Checksum	PX = P1(0x03) XOR P2(0x88) XOR P3(0x88) XOR P4(0x88) XOR P5(0x88) XOR P6(0x88) XOR P7(0x88) XOR P8(0x88) XOR P9(0x88) = 0x03 SC(0x09) XOR CC(0xAB) XOR PX(0x03) = 0xA1
Message to send	STX(0x02) – SC(0x09) – CC(0xAB) – P1(0x03) – P2(0x88) – P3(0x88) – P4(0x88) – P5(0x88) – P6(0x88) – P7(0x88) – P8(0x88) – P9(0x88) – CHK(0xA1)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Assigning a frame length to a single ID

To set the frame length for a single ID the parameter 1 (P1) must be set to 4. In this case the command accepts only two further parameters (P2 and P3). P2 is used to identify the ID, P3 is used to specify the corresponding frame length. The value of P3 must be in the range of 1 to 8, otherwise this command will return an error. If you want to set IDs to a frame length of 0, you must use the assignment by range (see above).

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	1	1	0	1	0	1	0	1	1
0	0, 0x0			3, 0x3				2, 0x2		43, 0x2B					
3, 0x3								171, 0xAB							

Parameters	This command needs 3 parameters (3 bytes). The first parameter has a value of 4 (0x04) indicating that a single ID is configured. The second parameter indicates the desired ID (0x00 to 0x3F). The third parameter indicates the length of the frame (0x01 to 0x08). In this example the ID 64 (0x3F) is set to a frame length of 5.
Checksum	PX = P1(0x04) XOR P2(0x3F) XOR P3(0x05) = 0x3E SC(0x03) XOR CC(0xAB) XOR PX(0x3E) = 0x96
Message to send	STX(0x02) – SC(0x03) – CC(0xAB) – P1(0x04) – P2(0x3F) – P3(0x05) – CHK(0x96)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Note: If an ID also used by the command "Slave ID + Data Configuration" is modified, that setting is changed, too. If the frame length is set to a number smaller than that used by “Slave ID + Data Configuration”, its data will be truncated. If the frame length is set to a larger number, the data will be padded with the value 0xFF.

Set Checksum Type

This command sets the checksum type for a single or all 64 LIN message IDs. This can be either the classic format (LIN 1.x) or the enhanced format (LIN 2.0).

This command can be implemented in two different ways. The first is setting the checksum type for all IDs in a single operation. The second is setting the checksum type for a single ID. Depending on

the kind of operation, the parameter count is either 3 bytes (for single ID) or 9 bytes (for all IDs).

Assigning a checksum type to all IDs

If the first parameter (P1) is 0, the checksum type is set for all IDs. The next 8 parameters (8 bytes) contain the checksum type for the IDs, using each bit of each data byte as boolean variable (0 means classic format and 1 means enhanced format). The following table shows the place of each ID in the eight data bytes:

Param	Bit assignment for LIN IDs							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P2	0x07	0x06	0x05	0x04	0x03	0x02	0x01	0x00
P3	0x0F	0x0E	0x0D	0x0C	0x0B	0x0A	0x09	0x08
P4	0x17	0x16	0x15	0x14	0x13	0x12	0x11	0x10
P5	0x1F	0x1E	0x1D	0x1C	0x1B	0x1A	0x19	0x18
P6	0x27	0x26	0x25	0x24	0x23	0x22	0x21	0x20
P7	0x2F	0x2E	0x2D	0x2C	0x2B	0x2A	0x29	0x28
P8	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30
P9	0x3F	0x3E	0x3D	0x3C	0x3B	0x3A	0x39	0x38

For example, setting the format for the first 8 IDs (using P2) to enhanced except for ID 3 the parameter P2 must be 0xF7 (0b11110111). The bit set to 0 corresponds to ID 3 (0x03) which will use the classic checksum type.

Sequence Code (SC)									Control Code (CC)							
X	**Sequence**			**Param Byte Count**					**Interface**		**Command Code**					
0	0	0	0	1	0	0	1		1	0	1	0	1	1	0	0
0	0, 0x0			9, 0x9					2, 0x2		44, 0x2C					
9, 0x9									172, 0xAC							

Parameters	This command needs 9 parameters (9 bytes). The first parameter indicates that the checksum type for all IDs is set. The remaining 8 parameters P2 to P9 indicate the checksum type for the IDs. In this example the checksum type for the IDs 0 to 31 is set to their default value (classic), for the IDs 32 to 64 to enhanced.
Checksum	PX = P1(0x00) XOR P2(0x00) XOR P3(0x00) XOR P4(0x00) XOR P5(0x00) XOR P6(0xFF) XOR P7(0xFF) XOR P8(0xFF) XOR P9(0xFF) = 0x00 SC(0x09) XOR CC(0xAC) XOR PX(0x00) = 0xA5
Message to send	STX(0x02) – SC(0x09) – CC(0xAC) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0xFF) – P7(0xFF) – P8(0xFF) – P9(0xFF) – CHK(0xA5)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Assigning a checksum type to a single ID

To set a frame length for a single ID the parameter P1 must be set to 1. In this case the command needs two additional parameters (P2 and P3). P2 is used to identify the ID. P3 is used to specify the checksum type. The value of this parameter must be either 0 for classic or 1 for enhanced, otherwise this command will return an error.

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	1	1	0	1	0	1	1	0	0
0	0, 0x0			3, 0x3				2, 0x2		44, 0x2C					
3, 0x3								172, 0xAC							

Parameters	This command needs 3 parameters (3 bytes). The first parameter has a value of 1 (0x01) which indicates that a single ID is configured. The second parameter has the desired ID (0x00 to 0x3F). The third parameter has the format of the frame (0x00 for classic or 0x01 for enhanced). In this example the ID 64 (0x3F) is set to the enhanced checksum type.
Checksum	PX = P1(0x01) XOR P2(0x3F) XOR P3(0x01) = 0x3F SC(0x03) XOR CC(0xAC) XOR PX(0x3F) = 0x90
Message to send	STX(0x02) – SC(0x03) – CC(0xAC) – P1(0x01) – P2(0x3F) – P3(0x01) – CHK(0x90)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set CAN ID for Slave Activation

This command sets the CAN ID used to activate a module as LIN slave through CAN messaging and its status (active or inactive).

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	1	0	0	1	0	1	0	1	1	0	1
0	0, 0x0			4, 0x4				2, 0x2		45, 0x2D					
4, 0x4								173, 0xAD							

<table>
<tr><td>Parameters</td><td>Four parameters (4 bytes) are required. The 4 bytes form a 32 bit value. Bits 5 to 7 of P4 are each used as boolean value to specify the following configuration:
<table>
<tr><th>Bit</th><th>Set to 1</th><th>Set to 0</th></tr>
<tr><td>7</td><td>Extended CAN ID (29 bits)</td><td>Standard CAN ID (11 bits)</td></tr>
<tr><td>6</td><td>Feature is enabled</td><td>Feature is disabled</td></tr>
<tr><td>5</td><td colspan="2">Reserved</td></tr>
</table>
The remaining 29 bits of this value identify the CAN ID.
For example, to represent the CAN ID 0x1AABBCCD (29 bits) and to activate this feature, the parameters must be: P1(0xCD), P2(0xBC), P3(0xAB), P4(0xDA). In this example the CAN ID is set to its default value (ID: 0x7F0, length: 11 bits, status: disabled).</td></tr>
<tr><td>Checksum</td><td>PX = P1(0xF0) XOR P2(0x07) XOR P3(0x00) XOR P4(0x00) = 0xF7
SC(0x04) XOR CC(0xAD) XOR PX(0xF7) = 0x5E</td></tr>
<tr><td>Message to send</td><td>STX(0x02) – SC(0x04) – CC(0xAD) – P1(0xF0) – P2(0x07) – P3(0x00) – P4(0x00) – CHK(0x5E)</td></tr>
<tr><td>Positive response</td><td>STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)</td></tr>
</table>

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Set Slave Mask Status

This command sets the status (Publisher or Subscriber) of the slave mask of one or all LIN message IDs. The command can be implemented in two different ways. The first is setting the slave mask status for all IDs. The second is setting the slave mask status for a single ID. Depending on the kind of operation, the parameter count is either 3 bytes (for a single ID) or 9 bytes (for all IDs).

Assigning the Slave Mask Status to all IDs

If the first parameter (P1) is 0, the slave mask status is set for all IDs. The next 8 parameters (8 bytes) contain the slave mask status for the IDs, using each bit of each data byte as boolean variable (0

means Subscriber and 1 means Publisher). The following table shows the place of each ID in the eight data bytes:

Param	Bit assignment for LIN IDs							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P2	0x07	0x06	0x05	0x04	0x03	0x02	0x01	0x00
P3	0x0F	0x0E	0x0D	0x0C	0x0B	0x0A	0x09	0x08
P4	0x17	0x16	0x15	0x14	0x13	0x12	0x11	0x10
P5	0x1F	0x1E	0x1D	0x1C	0x1B	0x1A	0x19	0x18
P6	0x27	0x26	0x25	0x24	0x23	0x22	0x21	0x20
P7	0x2F	0x2E	0x2D	0x2C	0x2B	0x2A	0x29	0x28
P8	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30
P9	0x3F	0x3E	0x3D	0x3C	0x3B	0x3A	0x39	0x38

For example, if you send P4 with a value of 0x82 (0b10000010), this means that the slave mask of the IDs 16 (0x10), 18 (0x12), 19 (0x13), 20 (0x14), 21 (0x15), and 22 (0x16) are set as Subscriber and the IDs 17 (0x11) and 23 (0x17) are set as Publisher.

The data to be received starts with the LSB. Example: 0x011C = P1:0x1C and P2:0x01

Sequence Code (SC)								Control Code (CC)							
X	**Sequence**			**Param Byte Count**				**Interface**		**Command Code**					
0	0	0	0	1	0	0	1	1	0	1	0	1	1	1	0
0	0, 0x0			9, 0x9				2, 0x2		46, 0x2E					
9, 0x9								174, 0xAE							

Parameters	This command needs 9 parameters (9 bytes). The first parameter indicates that the slave mask status for all IDs is set. The remaining 8 parameters P2 to P9 indicate the slave mask status for the IDs. In this example the slave mask status for the IDs 0 to 31 is set to their default value (Subscriber).
Checksum	PX = P1(0x00) XOR P2(0x00) XOR P3(0x00) XOR P4(0x00) XOR P5(0x00) XOR P6(0x00) XOR P7(0x00) XOR P8(0x00) XOR P9(0x00) = 0x00 SC(0x09) XOR CC(0xAE) XOR PX(0x00) = 0xA7
Message to send	STX(0x02) – SC(0x09) – CC(0xAE) – P1(0x00) – P2(0x00) – P3(0x00) – P4(0x00) – P5(0x00) – P6(0x00) – P7(0x00) – P8(0x00) – P9(0x00) – CHK(0xA7)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Assigning the Slave Mask Status to a Single ID

To set a slave mask status for a single ID the parameter P1 must be set to 1. In this case the command needs two additional parameters (P2 and P3). P2 is used to identify the ID. P3 is used to specify the status. The value of this parameter must be either 0 for Subscriber or 1 for Publisher, otherwise this command will return an error.

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	1	1	1	0	1	0	1	1	1	0
0	0, 0x0			3, 0x3				2, 0x2		46, 0x2E					
3, 0x3								174, 0xAE							

Parameters	This command needs 3 parameters (3 bytes). The first parameter has a value of 1 (0x01) which indicates that a single ID is configured. The second parameter has the desired ID (0x00 to 0x3F). The third parameter has the status of slave mask (0x00 for Subscriber or 0x01 for Publisher). In this example the slave mask of ID 64 (0x3F) is set to Publisher.
Checksum	PX = P1(0x01) XOR P2(0x3F) XOR P3(0x01) = 0x3F SC(0x03) XOR CC(0xAE) XOR PX(0x3F) = 0x92
Message to send	STX(0x02) – SC(0x03) – CC(0xAE) – P1(0x01) – P2(0x3F) – P3(0x01) – CHK(0x92)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

2.4.4 Reset/Erase Commands

Reset Message Transmission Timeouts

This command directs the module to reset the timeouts for the different LIN transmission messages. There are timeouts for bus idle messages as well as messages with frame length 8, 4, and 2 bytes. After reset, these timouts will contain the default values:

- Bus idle: 0x516 = 1302 ms
- Frame length 8: 0x0A = 10 ms
- Frame length 4: 0x07 = 7 ms
- Frame length 2: 0x05 = 5 ms

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0
0	0, 0x0			0, 0x0				2, 0x2		50, 0x32					
0, 0x0								178, 0xB2							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB2) XOR PX(0x00) = 0xB2
Message to send	STX(0x02) – SC(0x00) – CC(0xB2) – CHK(0xB2)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Erase Master Scheduler List

This command directs the module to remove all entries from the master scheduler in a module. If the master is active, it is set to inactive. The scheduler entries cannot be recovered afterwards.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	0	0	1	1
0	0, 0x0			0, 0x0					2, 0x2		51, 0x33					
0, 0x0									179, 0xB3							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB3) XOR PX(0x00) = 0xB3
Message to send	STX(0x02) – SC(0x00) – CC(0xB3) – CHK(0xB3)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset RCV Error Counter

This command directs the module to reset the counter of the reception errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	0	1	0	0
0	0, 0x0			0, 0x0					2, 0x2		52, 0x34					
0, 0x0									180, 0xB4							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB4) XOR PX(0x00) = 0xB4
Message to send	STX(0x02) – SC(0x00) – CC(0xB4) – CHK(0xB4)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset XMT Error Counter

This command directs the module to reset the counter for transmission errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	0	1	0	1
0	0, 0x0			0, 0x0					2, 0x2		53, 0x35					
0, 0x0									181, 0xB5							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB5) XOR PX(0x00) = 0xB5
Message to send	STX(0x02) – SC(0x00) – CC(0xB5) – CHK(0xB5)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset Bit Error Counter

This command directs the module to reset the counter for bit errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	0	1	1	0
0	0, 0x0			0, 0x0					2, 0x2		54, 0x36					
0, 0x0									182, 0xB6							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB6) XOR PX(0x00) = 0xB6
Message to send	STX(0x02) – SC(0x00) – CC(0xB6) – CHK(0xB6)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset Checksum Error Counter

This command directs the module to reset the counter for checksum errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	0	1	1	1
0	0, 0x0			0, 0x0					2, 0x2		55, 0x37					
0, 0x0									183, 0xB7							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB7) XOR PX(0x00) = 0xB7
Message to send	STX(0x02) – SC(0x00) – CC(0xB7) – CHK(0xB7)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset ID Checksum Error Counter

This command directs the module to reset the counter for ID checksum errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	1	0	0	0
0	0, 0x0			0, 0x0					2, 0x2		56, 0x38					
0, 0x0									184, 0xB8							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB8) XOR PX(0x00) = 0xB8
Message to send	STX(0x02) – SC(0x00) – CC(0xB8) – CHK(0xB8)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset "Slave not responding" Error Counter

This command directs the module to reset the counter for "Slave not responding" errors of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	1	0	0	1
0	0, 0x0			0, 0x0					2, 0x2		57, 0x39					
0, 0x0									185, 0xB9							

Parameters	None
Checksum	SC(0x00) XOR CC(0xB9) XOR PX(0x00) = 0xB9
Message to send	STX(0x02) – SC(0x00) – CC(0xB9) – CHK(0xB9)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset Inconsistent Synchronization Bytes Counter

This command directs the module to reset the counter for inconsistent synchronization bytes of the LIN interface.

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	1	0	1	0
0	0, 0x0			0, 0x0					2, 0x2		58, 0x3A					
0, 0x0									186, 0xBA							

Parameters	None
Checksum	SC(0x00) XOR CC(0xBA) XOR PX(0x00) = 0xBA
Message to send	STX(0x02) – SC(0x00) – CC(0xBA) – CHK(0xBA)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

Reset All Error Counters

This command directs the module to reset all error counters of the LIN interface. Those counters are:

- RCV errors
- XMT errors
- Bit errors
- Checksum errors
- ID checksum errors
- “Slave not responding” errors
- Inconsistent synchronization bytes

The data to be received starts with the LSB. Example: 0x1 = P1:0x01 and P2:0x00

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	0	1	1	1	0	1	1
0	0, 0x0			0, 0x0					2, 0x2		59, 0x3B					
0, 0x0									187, 0xBB							

Parameters	None
Checksum	SC(0x00) XOR CC(0xBB) XOR PX(0x00) = 0xBB
Message to send	STX(0x02) – SC(0x00) – CC(0xBB) – CHK(0xBB)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

2.5 Module Interface

2.5.1 Initialization Commands

Command	on page
Module Default Values	88

Module Default Values

This command directs the module to set its current configuration to the default values stored in the memory and to initialize all.

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	1	0	0	0	0	1	0
0	0, 0x0			0, 0x0					3, 0x3		2, 0x2					
0, 0x0									194, 0xC2							

Parameters	None
Checksum	SC(0x00) XOR CC(0xC2) XOR PX(0x00) = 0xC2
Message to send	STX(0x02) – SC(0x00) – CC(0xC2) – CHK(0xC2)
Positive response	On successful execution the RS-232 is initialized, too, and the connection is interrupted. Therefore no response is sent. This is not an error.

2.5.2 Read Commands

Read Hardware Version

This command retrieves the hardware version of the module.

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	1	0	0	0	0	0	0
0	0, 0x0			0, 0x0					3, 0x3		0, 0x0					
0, 0x0									192, 0xC0							

Parameters	None
Checksum	SC(0x00) XOR CC(0xC0) XOR PX(0x00) = 0xC0
Message to send	STX(0x02) – SC(0x00) – CC(0xC0) – CHK(0xC0)
Positive response	STX(0x02) – SC(0x0F) – P1(0x30) – P2(0x32) – P3(0x2E) – P4(0x30) – P5(0x32) – P6(0x2E) – P7(0x30) – P8(0x31) – P9(0x20) – P10(0x48) – P11(0x53) – P12(0x2D) – P13(0x43) – P14(0x41) – P15(0x4E) – CHK(0x37)

A positive response has 15 parameters. The returned data bytes are ASCII characters. The interpretation is:

Information	Parameters
Hardware – major version	P1, P2
Hardware – minor version	P4, P5
Hardware – release version	P7, P8
CAN transceiver	P10…P15
Data separators	P3, P6, P9

Hardware – major version

The major version is received as ASCII value, normally numeric characters. For example, character "0" (P1= 0x30) and character "1" (P2= 0x31) indicate the major version 01. Special kinds of hardware versions can have a non-numeric character to identify a special property. For example, S1 could stand for a special major version 01.

Hardware – minor version

The minor version is received as ASCII value (numeric characters). For example, character "0" (P4= 0x30) and character "2" (P5= 0x32) indicate the minor Version 02.

Hardware – release version

The release version is received as ASCII value (numeric characters). For example, character "5" (P7= 0x35) and character "1" (P8= 0x31) indicate the release version 51.

CAN transceiver

The CAN transceiver is returned as text (ASCII characters). This text is composed by two upper cased characters which indicate the kind of transceiver:

Parameters	Characters	CAN transceiver
P10 = 0x48, P11 = 0x53	HS	High-speed according to ISO 11898-2
P10 = 0x4C, P11 = 0x53	LS	Low-speed according to ISO 11898-3

The string "-CAN" (P12 = 0x2D, P13 = 0x43, P14 = 0x41, P15 = 0x4E) is used as suffix. For example, if a High-speed CAN transceiver is implemented in the module, the returned text is "HS-CAN".

Data separators

Three separators are used in a response to this command. A period character (P3, P6 = 0x2E = ".") is used two times to separate the major, the minor, and the release version of the hardware. A blank character (P9 = 0x20 = " ") is used to separate the hardware version from the type of CAN transceiver.

The interpretation of the positive response example above results in the following string concatenation:

- Hardware version = "02.02.01"
  P1(0x30) P2(0x32) P3(0x2E) P4(0x30) P5(0x32) P6(0x2E) P7(0x30) P8(0x31)
- Blank separator = " "
  P9(0x20)
- CAN transceiver = "HS-CAN"
  P10(0x48) P11(0x53) P12(0x2D) P13(0x43) P14(0x41) P15(0x4E)

Read Firmware Version

This command retrieves the firmware version of the module.

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1
0	0, 0x0			0, 0x0				3, 0x3		1, 0x1					
0, 0x0								193, 0xC1							

Parameters	None
Checksum	SC(0x00) XOR CC(0xC1) XOR PX(0x00) = 0xC1
Message to send	STX(0x02) – SC(0x00) – CC(0xC1) – CHK(0xC1)
Positive response	STX(0x02) – SC(0x08) – P1(0x30) – P2(0x31) – P3(0x2E) – P4(0x30) – P5(0x31) – P6(0x2E) – P7(0x30) – P8(0x31) – CHK(0x0D)

A positive response has 8 parameters. The returned data bytes are ASCII characters. The interpretation is:

Information	Parameters
Firmware – major version	P1, P2
Firmware – minor version	P4, P5
Firmware – release version	P7, P8
Data separators	P3, P6

Firmware – major version

The major version is received as ASCII value (numeric characters). For example, character "0" (P1= 0x30) and character "1" (P2= 0x31) indicate the major version 01.

Firmware – minor version

The minor version is received as ASCII value (numeric characters). For example, character "0" (P4= 0x30) and character "2" (P5= 0x32) indicate the minor Version 02.

Firmware – release version

The release version is received as ASCII value (numeric characters). For example, character "5" (P7= 0x35) and character "1" (P8= 0x31) indicate the release version 51.

Data separators

Two separators are used in a response to this command. A period character (P3, P6 = 0x2E = ".") is used two times to separate the major, the minor, and the release version of the firmware.

The interpretation of the positive response example above results in the following string concatenation:

- Firmware version = "01.01.01"
  P1(0x30) P2(0x31) P3(0x2E) P4(0x30) P5(0x31) P6(0x2E) P7(0x30) P8(0x31)

2.5.3 Configuration Commands

Flash Current Configuration

This command directs the module to take the current configuration on the module and save it into the module's flash memory. Afterwards, each time the module is reset or restarted the configuration is reloaded from the flash memory.

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	1	0	0	0	0	1	1
0	0, 0x0			0, 0x0					3, 0x3		3, 0x3					
0, 0x0									195, 0xC3							

Parameters	None
Checksum	SC(0x00) XOR CC(0xC3) XOR PX(0x00) = 0xC3
Message to send	STX(0x02) – SC(0x00) – CC(0xC3) – CHK(0xC3)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

2.5.4 Reset/Erase Commands

Reset Module

This command directs the module to initialize with values loaded from the flash memory.

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	0	0	0		1	1	0	0	0	1	0	0
0	0, 0x0			0, 0x0					3, 0x3		4, 0x4					
0, 0x0									196, 0xC4							

Parameters	None
Checksum	SC(0x00) XOR CC(0xC4) XOR PX(0x00) = 0xC4
Message to send	STX(0x02) – SC(0x00) – CC(0xC4) – CHK(0xC4)
Positive response	Information text in ASCII format

A positive response for this command is a descriptive text. This response does not follow the same rules as the responses of other commands, i.e. it does not use the STX, SC or CC bytes. The returned text is a conjunction of the name of the module, hardware and firmware versions, and the manufacturer. An example of this text:

```
PCAN-LIN Ver. 02.02.01 HS-CAN/01.01.01 (c) PEAK-System Technik
  GmbH
```

The end of the response is marked with a carriage return + line feed. The information and length of this response can vary.

A reset of the module causes the same LED test procedure as after powering on the module (see user manual). If the used baud rate at the moment of the reset is different as the baud rate saved in the flash memory, the RS-232 connection with the module will be lost.

2.6 Message Transmission and Reception

The module is able to transmit and receive CAN as well as LIN messages via the RS-232 interface.

2.6.1 CAN

Write Data

This command uses a message structure like other commands mentioned earlier in this documentation. The module forwards the embedded message to the CAN interface. The data part of the message structure can vary according to the configuration of the CAN message to be transmitted.

Description		Byte count	Bit content							
			7	6	5	4	3	2	1	0
STX		1	0	0	0	0	0	0	1	0
Sequence code		1	0	Sequence number: 0b000…0b111 (0…7)			Parameter byte count: 1 + ID length (2, 4) + data length n (0...8)			
Control code		1	Target interface: 0b01 (1 = CAN)		Command code: 0b000001 (1 = Write data on bus)					
Parameters	CAN message info	1	0: STD 1: EXT	0: Data 1: RTR	0	0	CAN data length n			
	CAN ID STD/EXT	STD: 2 EXT: 4	CAN ID STD: 0...0x07FF, 11 bits CAN ID EXT: 0…0x1FFFFFFF, 29 bits from LSB to MSB							
	CAN data bytes	n (0…8)	Data bytes							
Checksum		1	Bitwise XOR operation between all bytes of this messages except the STX byte							

- If a CAN message to be transmitted is a remote request frame (RTR), the CAN data length in the CAN message info must be 0. Otherwise the command returns an error and the message is not forwarded to the CAN interface.
- A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

For example, to send a message with standard ID (2 bytes) and 3 data bytes the sequence code must be 0x06:

Sequence Code (SC)									Control Code (CC)							
X	Sequence			Param Byte Count					Interface		Command Code					
0	0	0	0	0	1	1	0		0	1	0	0	0	0	0	1
0	0, 0x0			6, 0x6					1, 0x1		1, 0x1					
6, 0x6									65, 0x41							

Parameters	In this example a standard CAN message (ID: 0x123) with 3 data bytes (0x0A, 0x0B, 0x0C) is transmitted.
Checksum	PX = P1(0x03) XOR P2(0x23) XOR P3(0x01) XOR P4(0x0A) XOR P5(0x0B) XOR P6(0x0C) = 0x2C SC(0x06) XOR CC(0x41) XOR PX(0x2C) = 0x6B
Message to send	STX(0x02) – SC(0x06) – CC(0x41) – P1(0x03) – P2(0x23) – P3(0x01) – P4(0x0A) – P5(0x0B) – P6(0x0C) – CHK(0x6B)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

Receive Data

When a CAN message is forwarded to the RS-232 interface (see command *Retrieve Forward Mask* on page 24), it is embedded into the command structure as data part.

Description		Byte count	Bit content							
			7	6	5	4	3	2	1	0
STX		1	0	0	0	0	0	0	1	0
Sequence code		1	1 (AR)	Sequence number: 0b001 (1)			Parameter byte count (1 + ID length + data length n)			
Parameters	CAN message info	1	0: STD 1: EXT	0: Data 1: RTR	ERR	0	CAN data length n			
	CAN ID STD/EXT	STD: 2 EXT: 4	CAN ID STD: 0...0x7FF, 11 bits CAN ID EXT: 0...0x1FFFFFFF, 29 bits from LSB to MSB							
	CAN data bytes	n (0...8)	Data bytes							
Checksum		1	Bitwise XOR operation between all bytes of this messages except the STX byte							

- The auto reply bit (AR) is set, because a CAN message is forwarded.
- The sequence number is 1, meaning that the following data is a forwarded message from the CAN interface.
- If a received CAN message is a remote request frame (RTR), the CAN data length is 0.
- If the error bit (ERR) is 1, a CAN error frame is received. Error frames are only received if the forward mask of the CAN interface is set to monitor and forward CAN errors (for more information see command *Set Forward Mask* on page 32). For more information about error codes see chapter 3 *Error Messages* on page 103.

For example, receiving a forwarded message with a standard ID 0x456 (2 bytes) and 3 data bytes (0x0D, 0x0E, 0x0F) has a form as follows:

Message received: STX(0x02) – SC(0x96) – P1(0x03) – P2(0x56) – P3(0x04) – P4(0x0D) – P5(0x0E) – P6(0x0F) – CHK(0xCB)

Sequence code (SC = 0x96)								
Bit	7	6	5	4	3	2	1	0
Description	Auto reply	Sequence number			Parameter byte count			
Contents	1	0	0	1	0	1	1	0

CAN message info (P1 = 0x03)								
Bit	7	6	5	4	3	2	1	0
Description	Frame format	RTR	Error	Fixed	CAN data length			
Contents	0 (STD)	0	0 (None)	0	0	0	1	1

Frame format is standard: CAN ID is two bytes long (P2 and P3):

- CAN ID = 0x56 0x04

CAN data length is 3:

- P4: CAN data [0] = 0x0D
- P5: CAN data [1] = 0x0E
- P6: CAN data [2] = 0x0F

2.6.2 LIN

Write Data

This command uses a message structure like in other commands mentioned earlier in this documentation. The module forwards the embedded message to the LIN interface. The data part of the message structure can vary according to the configuration of the LIN message to be transmitted.

Description		Byte count	Bit content							
			7	6	5	4	3	2	1	0
STX		1	0	0	0	0	0	0	1	0
Sequence code		1	0	Sequence number: 0b000…0b111 (0…7)			Parameter byte count: 1 + data length n			
Control code		1	Target interface: 0b10 (2 = LIN)		Command code: 0b000001 (1 = Write data on bus)					
Parameters	LIN message info	1	0: REQ 1: RESP	0	LIN ID: 0…0x3F					
	LIN data bytes	n (1…8)	Data bytes							
Checksum		1	Bitwise XOR operation between all bytes of this messages except the STX byte							

- Request frame (REQ): Message is used to do a data requisition.
- Response frame (RESP): Message comes from a LIN slave.
- If the LIN ID uses the default length (LIN 1.x specification), the data length depends on the given LIN ID (see command *Retrieve Frame Length* on page 53). Otherwise the number of data bytes as configured for the LIN ID must be included.
- If the number of transmitted data bytes does not align with the configured length for the used LIN ID, an error is returned and the message is not forwarded to the LIN interface.
- A positive response has 1 parameter with the value 0 meaning no errors. For more information about error codes see chapter 3 *Error Messages* on page 103.

For example, to transmit a message with a standard ID (LIN 1.x) and 4 data bytes (an ID in the range of 0x20 to 0x2F), the sequence code must be 0x05:

Sequence Code (SC)								Control Code (CC)							
X	Sequence			Param Byte Count				Interface		Command Code					
0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	1
0	0, 0x0			5, 0x5				2, 0x2		1, 0x1					
5, 0x5								129, 0x81							

Parameters	Needed parameters are 1 byte for the LIN message info and the needed bytes for the LIN message itself. In this example a standard LIN request with 4 bytes (ID: 0x20; data bytes: 0x0A, 0x0B, 0x0C, 0x0D) is transmitted.
Checksum	PX = P1(0x20) XOR P2(0x0A) XOR P3(0x0B) XOR P4(0x0C) XOR P5(0x0D) = 0x20 SC(0x05) XOR CC(0x81) XOR PX(0x20) = 0xA4
Message to send	STX(0x02) – SC(0x05) – CC(0x81) – P1(0x20) – P2(0x0A) – P3(0x0B) – P4(0x0C) – P5(0x0D) – CHK(0xA4)
Positive response	STX(0x02) – SC(0x01) – P1(0x00) – CHK(0x01)

Receive Data

When a LIN message is forwarded to the RS-232 interface (see command *Retrieve Forward Mask* on page 40), it is embedded into the command structure as data part.

Description		Byte count	Bit content							
			7	6	5	4	3	2	1	0
STX		1	0	0	0	0	0	0	1	0
Sequence code		1	1 (AR)	Sequence no.: 0b010 (2)			Parameter byte count (1 + data length n)			
Para-meters	LIN message info	1	0: REQ 1: RESP	ERR	LIN ID: 0…0x3F or error code					
	LIN data bytes	n (0…8)	Data bytes (0 bytes in case of LIN error)							
Checksum		1	Bitwise XOR operation between all bytes of this messages except the STX byte							

- The auto reply bit (AR) is set, because a LIN message is forwarded.
- The sequence number is 2, meaning that the following data is a forwarded message from the LIN interface.
- REQ indicates a request for data, and RESP that the message comes from a slave.
- If the error bit (ERR) is 1, a LIN error is received. Error frames are only received if the forward mask of the LIN interface is set to monitor and forward LIN errors (see command *Set Forward Mask* on page 61 for more information). In this case the LIN message info contains the error code. No data is transmitted if an error occurs.
- The data length to be read depends on the parameter byte count of the LIN sequence code. Because the LIN message info already contains the LIN ID, the rest of the parameters are data bytes. In this case the number of data bytes is the total of parameters minus one byte.

For example, receiving a forwarded message with a standard ID 0x21 (format LIN 1.x, see *Retrieve Frame Length* on page 53 for more information) and the data bytes (0x11, 0x22, 0x33, 0x44) has the following form:

Message received: STX(0x02) – SC(0xA5) – P1(0xA1) – P2(0x11) – P3(0x22) – P4(0x33) – P5(0x44) – CHK(0x40)

Sequence code (SC = 0xA4)								
Bit	7	6	5	4	3	2	1	0
Description	Auto reply	Sequence number			Parameter byte count			
Contents	1	0	1	0	0	1	0	1

LIN message info (P1 = 0xA0)								
Bit	7	6	5	4	3	2	1	0
Description	Frame type	Error	LIN ID					
Contents	1 (RESP)	0	1	0	0	0	0	1

- LIN data length = parameter byte count - 1 byte = 0b0101 - 1 = 4
- LIN ID = 0b100001 = 0x21
- P2: LIN data [0] = 0x11
- P3: LIN data [1] = 0x22
- P4: LIN data [2] = 0x33
- P5: LIN data [3] = 0x44

3 Error Messages

3.1 RS-232 Return Codes

As response to each request message a code is returned, i.e. a valid RS-232/LIN message with one data byte. The structure of a response has a length of 4 bytes and is defined as follows:

Byte	Description
1	STX (0x02)
2	Sequence code (0x01)
3	Data (error code)
4	Checksum (0x01 XOR error code)

The possible codes are listed in the following table:

Error code	Label	Description
0x00	OK	No errors
0x01	INV_CHECKSUM	Invalid checksum
0x02	SER_BUFF_OVERRUN	Buffer overrun at the RS-232 communication
0x03	INV_MASK	Invalid forward mask was passed
0x04	NOT_POSSIBLE	Command is not allowed for the given interface
0x05	OUT_OF_RANGE	Parameter out of range of valid values
0x06	NO_SCHEDULE_ENTRIES	LIN scheduler has no entries
0x07	INV_ARGUMENT	Invalid argument was passed
0x08	INV_GROUP	Invalid interface was passed
0x09	NOT_SUPPORTED	Unsupported command
0x0A	FAILED	The execution of a valid command produced an error condition
0x0B	ACTIVE_MASTER	Cyclic processing of LIN frames (LIN master operation) is currently active. At this time it is not allowed to transmit single frames.
0x0C	INV_TIMESTAMP	Invalid time stamp was passed

3.2 CAN Error Codes

If an error occurs within the CAN interface and the CAN forward mask is configured to send CAN errors to the RS-232 interface (see command *Retrieve Forward Mask* on page 24 for more information), the error is returned as a RS-232/CAN message structure with a data length of 4 bytes and a total of 8 bytes:

Byte	Description
1	STX (0x02)
2	Sequence code (0x94)
3	CAN message info (0x24)
4	Bit mask with general error information
5	Error capture code from the CAN controller (SJA1000)
6	RX error counter of the CAN controller (SJA1000)
7	TX error counter of the CAN controller (SJA1000)
8	Checksum (0x01 XOR [Bytes 2 - 7])

Byte 2: Sequence code

Bit	7	6	5	4	3	2	1	0
Description	AR		CAN		Param byte count			
Contents	1	0	0	1	0	1	0	0

Byte 3: CAN message info

Bit	7	6	5	4	3	2	1	0
Description	Frame type	Error	CAN Error		Length			
Contents	0	1	1	0	0	1	0	0

The bit mask with general information about the CAN error can be one of the following values:

Error code	Value	Description
0x01	CAN_ERR_BUS_OFF	The CAN controller is in bus-off state
0x02	CAN_ERR_BUS_PASSIVE	The CAN controller is in bus-passive state

Error code	Value	Description
0x04	CAN_ERR_BUS_ACTIVE	The CAN controller is back to active state (the state before was either bus-off or bus-passive)
0x08	CAN_ERR_DATA_OVERRUN	Receive queue was read too late (losing messages)
0x10	CAN_ERR_WARNLIMIT	One of the internal error counters of the CAN controller has reached a predefined limit
0x20	CAN_ERR_EWL_EXCEEDED	The predefined error warning limit has been exceeded

3.3 LIN Error Codes

If an error occurs within the LIN interface and the LIN forward mask is configured to send LIN errors to the RS-232 interface (see command *Retrieve Forward Mask* on page 40 for more information), the error is returned as a RS-232/LIN message structure with a data length of 1 byte and a total of 4 bytes:

Byte	Description
1	STX (0x02)
2	Sequence code (0xA1)
3	Data (error code)
4	Checksum (0xA1 XOR error code)

Byte 2: Sequence code								
Bit	7	6	5	4	3	2	1	0
Description	AR		LIN		Param byte count			
Contents	1	0	1	0	0	0	0	1

Byte 3: Data (error code)								
Bit	7	6	5	4	3	2	1	0
Description	Frame type	Error	Error code					
Contents	0	1	X	X	X	X	X	X

Bit 6 in Byte 3 is on (1) indicating that the place in the byte usually used for the LIN ID is now occupied by the LIN error code.

Error code	Description
0x01	Buffer overflow (lost information during message transmission)
0x02	Invalid stop bit found (possibly due to bit rate deviation)
0x03	Invalid break found in the protocol stack
0x04	Invalid synchronization byte found (possibly due to bit rate deviation)
0x05	Error in LIN ID parity bit 0
0x06	Error in LIN ID parity bit 1
0x07	Invalid data received (bit error)
0x08	Invalid frame checksum received
0x09	Frame timeout (slave not responding or interframe gaps are too long)
0x0A	Slave not responding
0x0B	LIN header is complete (break, sync, ID)
0x0C	LIN wake-up detected
0x0D	No receive IRQ, but several XMT IRQs: LIN line possibly shorted to power
0x0E	Scheduler delay is too short
0x0F	Not available
0x12	LIN line possibly shorted to GND

4 Serial Trace Analysis

If you have a trace of incoming messages from the RS-232 interface, you can analyze it with help of the following tables.

1st byte: STX

Bin	Hex	Description
0000 0010	02	Mandatory “Start Transmission” code for any message in a module communication

2nd byte: Sequence Code

Bin	Hex	Description
0 *NNN PPPP*	*NP*	Command auto response
1 000 *PPPP*	8*P*	Forwarded message from **RS-232** interface
1 001 *PPPP*	9*P*	Forwarded message from **CAN** interface
1 010 *PPPP*	A*P*	Forwarded message from **LIN** interface
1 011 *PPPP*	B*P*	Forwarded message from **Module** interface

NNN, *N* = Arbitrary Sequence Number (0 - 7, 0x0 - 0x7)
PPPP, *P* = Parameter byte count (1 - 15, 0x1 - 0xF)

3rd byte: Control Code

Bin	Hex	Interface	Command name
00 00 0000	00	RS-232	Initialization According to Current Parameters
00 00 1011	0B	RS-232	Retrieve Baud Rate
00 01 0101	16	RS-232	Retrieve RCV Error Counter
00 01 0111	17	RS-232	Retrieve XMT Error Counter
00 01 1000	18	RS-232	Retrieve All Error Counters
00 01 1111	1F	RS-232	Configure Baud Rate
00 11 0100	34	RS-232	Reset RCV Error Counter
00 11 0101	35	RS-232	Reset XMT Error Counter
00 11 0111	37	RS-232	Reset Checksum Error Counter
01 00 0000	40	CAN	Initialization According to Current Parameters
01 00 0001	41	CAN	Write Data
01 00 1010	4A	CAN	Retrieve Activation Status

Bin	Hex	Interface	Command name
01 00 1011	4B	CAN	Retrieve Bit Rate
01 00 1100	4C	CAN	Retrieve Forward Mask
01 00 1101	4D	CAN	Retrieve Filter Mask
01 00 1110	4E	CAN	Retrieve Filter Code
01 00 1111	4F	CAN	Retrieve CAN ID Offset for LIN Messages
01 01 0110	56	CAN	Retrieve RCV Error Counter
01 01 0111	57	CAN	Retrieve XMT Error Counter
01 01 1000	58	CAN	Retrieve All Error Counters
01 01 1110	5E	CAN	Set Activation Status
01 01 1111	5F	CAN	Set Bit Rate
01 10 0000	60	CAN	Set Forward Mask
01 10 0001	61	CAN	Set Filter Mask
01 10 0010	62	CAN	Set Filter Code
01 10 0011	63	CAN	Set CAN ID Offset for LIN Messages
10 00 0000	80	LIN	Initialization According to Current Parameters
10 00 0001	81	LIN	Write Data
10 00 1010	8A	LIN	Retrieve Activation Status
10 00 1011	8B	LIN	Retrieve Bit Rate
10 00 1100	8C	LIN	Retrieve Forward Mask
10 00 1101	8D	LIN	Retrieve Filter Mask
10 00 1110	8E	LIN	Retrieve Filter Code
10 00 1111	8F	LIN	Retrieve Slave Mask Status
10 01 0000	90	LIN	Retrieve Master Status
10 01 0001	91	LIN	Retrieve LIN Bus Termination (Master Termination)
10 01 0010	92	LIN	Retrieve Message Transmission Timeouts
10 01 0011	93	LIN	Retrieve Message Retries
10 01 0100	94	LIN	Retrieve Scheduler Entries Count
10 01 0101	95	LIN	Retrieve Scheduler Entry
10 01 0110	96	LIN	Retrieve RCV Error Counter
10 01 0111	97	LIN	Retrieve XMT Error Counter
10 01 1000	98	LIN	Retrieve All Error Counters
10 01 1001	99	LIN	Retrieve Slave ID + Data Configuration Information

Bin	Hex	Interface	Command name
10 01 1010	9A	LIN	Retrieve Bit Rate Recognition for Synchronization Field Status
10 01 1011	9B	LIN	Retrieve Frame Length
10 01 1100	9C	LIN	Retrieve Checksum Type
10 01 1101	9D	LIN	Retrieve CAN ID for Slave Activation
10 01 1110	9E	LIN	Set Activation Status
10 01 1111	9F	LIN	Set Bit Rate
10 10 0000	A0	LIN	Set Forward Mask
10 10 0001	A1	LIN	Set Filter Mask
10 10 0010	A2	LIN	Set Filter Code
10 10 0100	A4	LIN	Set Master Status
10 10 0101	A5	LIN	Set LIN Bus Termination (Master Termination)
10 10 0110	A6	LIN	Set Message Transmission Timeouts
10 10 0111	A7	LIN	Set Message Retries
10 10 1000	A8	LIN	Insert Scheduler Entry
10 10 1001	A9	LIN	Set Slave ID + Data Configuration
10 10 1010	AA	LIN	Configure Bit Rate Recognition for Synchronization Field
10 10 1011	AB	LIN	Set Frame Length
10 10 1100	AC	LIN	Set Checksum Type
10 10 1101	AD	LIN	Set CAN ID for Slave Activation
10 10 1110	AE	LIN	Set Slave Mask Status
10 11 0010	B2	LIN	Reset Message Transmission Timeouts
10 11 0011	B3	LIN	Erase Master Scheduler List
10 11 0100	B4	LIN	Reset RCV Error Counter
10 11 0101	B5	LIN	Reset XMT Error Counter
10 11 0110	B6	LIN	Reset Bit Error Counter
10 11 0111	B7	LIN	Reset Checksum Error Counter
10 11 1000	B8	LIN	Reset ID Checksum Error Counter
10 11 1001	B9	LIN	Reset “Slave not responding” Error Counter
10 11 1010	BA	LIN	Reset Inconsistent Synchronization Bytes Counter
10 11 1011	BB	LIN	Reset all Error Counters
11 00 0000	C0	Module	Read Hardware Version
11 00 0001	C1	Module	Read Firmware Version

Bin	Hex	Interface	Command name
11 00 0010	C2	Module	Module Default Values
11 00 0011	C3	Module	Flash Current Configuration
11 00 0100	C4	Module	Reset Module