



User Manual

Baby-LIN-MB-II V4.3

Phone: 010-2601-9622

E-Mail: info@haehong.com

Website: <https://haehongtec.com/>

1 Imprint	6
2 Glossary	7
3 Safety instructions	7
3.1 Warning signs.....	7
3.2 Safety precautions.....	8
4 Preface	9
4.1 Updates.....	9
4.1.1 Update philosophy.....	9
4.1.2 Downloads.....	9
4.1.3 Installation.....	10
4.1.4 Check version.....	10
5 Handle voucher and activation codes	11
5.1 Convert a voucher code into an activation code.....	11
5.2 Redeem an activation code.....	12
5.3 Check the installed activation codes.....	13
5.4 Support information.....	13
6 Hardware	14
6.1 Overview.....	14
6.1.1 Hardware revision C and following.....	14
6.1.2 Hardware revision B.....	15
6.1.3 Hardware revision A.....	16
6.2 MIF extension modules.....	16
6.3 Connectors.....	18
6.3.1 X3 - USB host.....	18
6.3.2 X7 - RS-232.....	18
6.3.3 X8 - Logic supply.....	20
6.3.4 X9 - LIN, CAN and IO.....	21
6.3.5 X10 - Ethernet.....	28
6.4 Power Supply.....	29
6.4.1 UPS - Uninterruptible power supply.....	29
6.5 LEDs.....	29
6.5.1 LD1, LD2 - Device state and error indicator.....	29
6.5.2 LD3-LD8 - Bus state indicator.....	30

6.6 Push buttons.....	31
6.6.1 PB1, PB2 - Push buttons.....	31
6.7 Mechanics.....	31
6.7.1 M1 - Screw thread.....	31
6.8 Hardware adaptations.....	31
6.8.1 Step by step guide: How to change the RTC battery.....	31
6.9 Installation of MIF extension modules.....	35
6.9.1 Introduction.....	35
6.9.2 Software installation.....	36
6.9.3 Hardware installation.....	36
6.9.4 Baby-LIN-MB compatibility adapter.....	41
7 Firmware	44
7.1 The connection modes of the Baby-LIN-MB-II.....	44
7.2 Firmware update.....	45
7.2.1 Introduction.....	45
7.2.2 Required software.....	45
7.2.3 Update the firmware.....	45
7.3 Import of SDF´s.....	46
7.4 Web interface.....	46
7.4.1 Overview.....	46
7.4.2 Mode switch.....	47
7.4.3 Dashboard.....	47
7.4.4 Datase files.....	48
7.4.5 System update.....	48
7.4.6 Settings.....	48
7.5 Host interface.....	51
7.5.1 Introduction.....	51
7.5.2 Serial interface.....	52
7.5.3 Network interface.....	52
7.5.4 Network configuration.....	52
7.5.5 DHCP configuration.....	52
7.5.6 Static IP adress.....	53
7.5.7 Configuration using the Baby-LIN-MB-Tool.....	53
7.5.8 Configuration using a USB mass storage device.....	53
7.6 Logging.....	54
7.6.1 Introduction.....	54
7.6.2 Configure and activate the logging.....	54

7.6.3 SimpleMenu Logging.....	55
7.6.4 Log Settings via Webinterface.....	56
7.6.5 Log data targets.....	57
7.6.6 Log data formats.....	58
7.7 ASCII host command protocol.....	59
7.7.1 Syntax.....	59
7.7.2 API modes.....	60
7.7.3 CmdDone API.....	61
7.7.4 TCP connections: Single and multi socket.....	62
7.7.5 Quick reference.....	63
7.8 Commando List.....	64
7.8.1 Command Version.....	64
7.8.2 Command SetApiMode.....	65
7.8.3 Command CMDDone.....	65
7.8.4 Command B.....	66
7.8.5 Command CurrentSdf.....	66
7.8.6 Command Start.....	67
7.8.7 Command LinStart.....	68
7.8.8 Command Stop.....	68
7.8.9 Commando LinStop.....	68
7.8.10 Commando LinSchedule.....	69
7.8.11 Commando SchedMode.....	69
7.8.12 Command RdSignal.....	71
7.8.13 Command LinRdSignal.....	72
7.8.14 Command WrSignal.....	72
7.8.15 Command LinWrSignal.....	73
7.8.16 Command VarRead.....	73
7.8.17 Command VarWrite.....	75
7.8.18 Command FormatSignals.....	77
7.8.19 Command LinMstReq.....	79
7.8.20 Command LinSivResp.....	80
7.8.21 Command LinState.....	82
7.8.22 Command RdFrameError.....	83
7.8.23 Command MacroExec.....	84
7.8.24 Command Diag22.....	85
7.8.25 Command RTCWrite.....	86
7.8.26 Command RTCRead.....	87
7.8.27 Command ReadById.....	87

7.8.28 Command ReadByldCompare.....	90
7.8.29 Command WaitSignal.....	92
7.8.30 Command SeqRun.....	93
7.8.31 Command SeqExec.....	94
7.8.32 Command Delay.....	94
7.8.33 Command DigOut.....	95
7.8.34 Command DigIn.....	96
7.8.35 Command SetConfigVar.....	96
7.8.36 Command LicencelInstall.....	97
7.9 Return codes.....	98
8 Workflow	100
8.1 Overview.....	100
8.2 Getting started.....	102
8.2.1 Introduction.....	102
8.2.2 Installation.....	102
8.3 LDF.....	102
8.3.1 LDF Example.....	103
8.3.2 LIN application frames.....	104
8.3.3 LIN Scheduling.....	105
8.3.4 LIN Diagnostic frames.....	105
8.4 Session Description File (SDF).....	107
8.4.1 How to create a LIN application.....	107
8.4.2 Introduction.....	107
8.4.3 Create a SDF.....	107
8.4.4 Common Setup.....	107
8.4.5 Example SDF.....	110
8.4.6 Start the bus communication.....	110
8.5 Stand-alone mode and autostart.....	112
8.5.1 Enable the stand-alone mode.....	112
8.5.2 Configure the autostart macro.....	112
8.5.3 Store a SDF persistently.....	112
8.5.4 Configure the device to automatically load a SDF and start a macro.....	113
9 LINWorks Software - Overview	114
10 Migration informatiion	116
10.1 Migration from Baby-LIN-MB-II to Baby-LIN-MB.....	116
10.2 Performance.....	116

10.3 LIN-Bus transceiver.....	117
10.4 Changed connectors.....	117
10.5 SDF file names.....	117
10.6 Real-time clock.....	118
10.7 Power on/off behaviour.....	118
10.8 MB-II LinSlvRsp Busy.....	118
10.9 SDF versions: SDF-V3 and SDF-V2.....	118
10.9.1 Compatibilities.....	119
10.9.2 Section.....	119
10.9.3 Target-specific options.....	120
10.9.4 Names.....	120
10.9.5 Emulation.....	120
10.9.6 Virtual signals and system variables.....	120
10.9.7 Signalfunctions.....	121
10.9.8 Macros.....	121

1 Imprint

Author	Lipowsky Industrie-Elektronik GmbH Römerstraße 57 64291 Darmstadt
Phone	+49 (0) 6151 / 93591 - 0
Fax	+49 (0) 6151 / 93591 - 28
E-Mail	info@lipowsky.de
Website	www.lipowsky.com
CEO	Andreas Lipowsky
Commercial register	Darmstadt HRB 5139
VAT-ID	DE 111647423
Quality Management	DIN EN ISO 9001:2015

Title	Baby-LIN-MB-II User Manual
Version	4.1
Date	2022-08-23
Valid for	Baby-LIN-MB-II
Copyright	© 2021, Lipowsky Industrie-Elektronik GmbH, Darmstadt

This publication is copyright protected. All rights reserved, including those to translation, performance, use of illustrations and tables, broadcasting, microfilming or reproduction by other means, or electronic storage of all material contained herein.

All other brand names and trademarks used within this manual are unlimited subject to the applicable trademark laws and the ownership rights of their registered owners.

The hardware, firmware, software and documents of the Baby-LIN-MB-II are subject to change without prior notice. Lipowsky Industrie-Elektronik GmbH thereby has no obligations.

2 Glossary

ADC	Ampere Direct Current. This is the unit of DC current values.
CAN	Controller Area Network
CAN-HS	CAN high speed. These are CAN interfaces with high data rates according to ISO-11898.
CAN-LS	CAN low speed. These are CAN interfaces with fault tolerant low data rates according to ISO-11519.
CD	Compact Disk
DBC	Database CAN: A file that contains the description of a CAN bus. It contains nearly the same information as a ARXML file.
DLL	Dynamic Link Library. It can be used to execute the DLL functions in custom applications.
ECU	Electronic control unit
EOL	End of line
ESD	Electro static discharge. The sudden flow of electricity between two electrically charged objects caused by e.g. contact.
EU	European Union. The Lipowsky Industrie-Elektronik GmbH resides inside the EU. Therefor shipping within the EU can be done without customs duties. You should definitely check out our worldwide distributors. Check chapter Distributors for more information.
LIN	Local Interconnect Network
LINWorks	Application software suite to configure the Baby-LIN devices.
PC	Personal Computer
PLC	Programmable Logic Controller
PWM	The pulse-width modulation is a modulation technique used to encode a value into a pulsing signal.
RTC	Real-time clock.
SD	Secure Digital Memory Card. This is a type of non-volatile memory cards.
SDF	Session Description File
SID	Service identifier. This number identifies a protocol service.
USB	Universal Serial Bus
VDC	Voltage Direct Current. This is the unit of DC voltage values.

3 Safety instruciotns

3.1 Warning signs

The following warning signs are used for safety precautions:



DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.



WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.



CAUTION indicates a hazardous situation which, if not avoided, could result in minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

SAFETY INSTRUCTIONS

Safety instructions signs indicate specific safety related instructions or procedures.

The following notice types are used to give you non safety precaution related information, e.g. software or configuration related problems:



Attention

This notice type signals possible problems, you should definitely pay attention to. Ignoring them probably lead to unexpected behaviour or data loss.



Version incompatibility

This notice type signals possible version incompatibilities and may lead to unexpected behaviour. These incompatibilities can be caused by old or incompatible software or firmware versions as well as missing activation codes.



Warning

This notice type signals possible problems, you should pay attention to. Ignoring them may lead to unexpected behaviour or data loss.



Attention

This notice type should inform you about useful information, that help you understand the Baby- LIN-RM-III better.






Attention

This notice type should give you tips, that help to reduce your expense and time to implement.

3.2 Safety precautions

Despite compliance with the relevant laws and regulations, residual risks can not be excluded. The following safety precautions define the hazards that can occur when operating the Baby-LIN-MB-II

 DANGER	<p>Mortal danger by automatic start of connected devices.</p> <ul style="list-style-type: none"> • Prepare for actions from connected devices. • Study safety precautions of connected devices.
 WARNING	<p>Mortal danger by electric shock.</p> <ul style="list-style-type: none"> • Operate the Baby-LIN-MB-II only within dry conditions. • Do not touch the Baby-LIN-MB-II if powered. • Do not touch the Baby-LIN-MB-II if damaged.
 CAUTION	<p>Do not touch the Baby-LIN-MB-II if wet. Injury by damaged battery.</p> <ul style="list-style-type: none"> • Operate the device only within the defined operating temperature. • Observe the correct polarity when inserting the battery. • Do not touch the battery if damaged. • Do not touch the battery if wet.
NOTICE	<p>Please recycle or dispose the battery safely and properly according to local laws and regulations.</p>

4 Preface

4.1 Updates

4.1.1 Update philosophy

The functionality and features of the Baby-LIN-MB-II are defined by the installed firmware as well as the used versions of the LINWorks and Baby-LIN-DLL.

As we are permanently working on product improvements, the software and firmware are updated periodically. These updates make new features available and solve problems, which have been discovered by our internal tests or have been reported by customers with earlier versions.

All the firmware updates are done in a way, that the updated Baby-LIN-MB-II will continue to work with an already installed, older LINWorks installation. So updating the Baby-LIN-MB-II firmware does not mean, that you necessarily have to update your LINWorks installation as well.

Therefor it is highly recommended to always update your Baby-LIN-MB-II to the latest available firmware version.

We also recommend to also update your LINWorks software and Baby-LIN-DLL, if new updates get available. Since new versions of the SessionConf may introduce new features to the SDF format, it is possible that older firmware, SimpleMenu or Baby-LIN-DLL versions are not compatible. Therefor you should also update them.

If you update your LINWorks it is highly recommended updating the firmware of your Baby-LIN-MB-II to the latest available firmware version as well as distributed the used versions of the Baby-LIN-DLL.

So the sole reason to stay with an older LINWorks version should be, that you use a Baby-LIN-MB-II with outdated firmware version, which you can't upgrade for whatever reason.

It is highly recommended updating the Baby-LIN driver to the latest version.

4.1.2 Downloads

The latest version of our software , firmware and documents can be found in the download area on our website www.lipowsky.de .



Advice

The **LINWorks** archive contains not only the **LINWorks** software but also the manuals, datasheets, application notes and examples. Only the device firmware packages are not included. The firmware is available as separate package.

Documents such as the data sheets or introductions to LIN bus communication are freely available for download. For all other documents and our LINWokrs software you have to log in. If you do not have a customer account yet you can register on our website. After your account has been activated by us you will receive an e-mail and then you have full access to our download offer.

DOWNLOADS

HERE YOU CAN DOWNLOAD DOCUMENTS FREE OF CHARGE.
FOR THE LOCKED CONTENT, PLEASE LOG IN WITH YOUR CUSTOMER ACCESS.

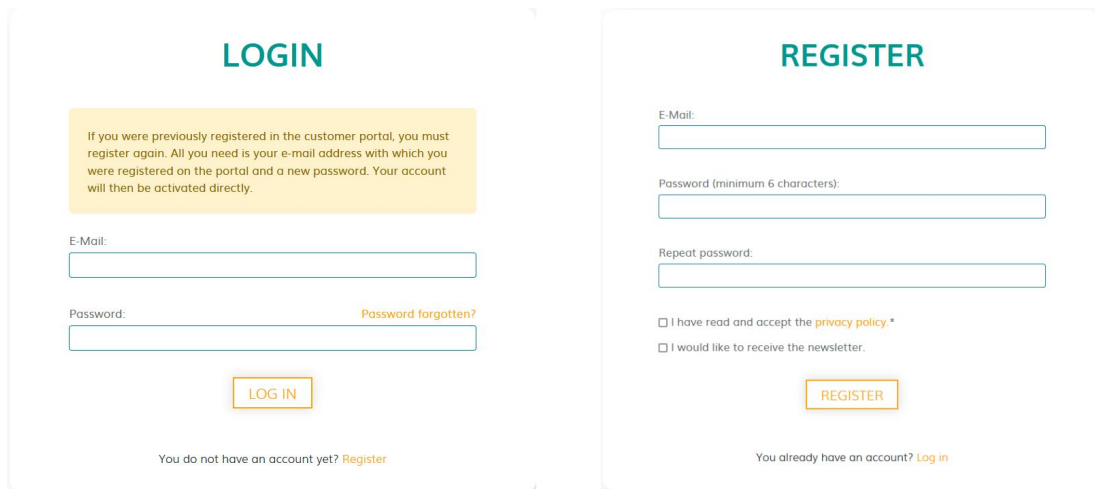
01 | Baby-LIN Software

LinWorks Software | Version 2.31.1 [More](#) 

File name: LinWorks-PCSoftware-2.X-CD.zip

Latest version of the LINWorks V2 software suite as zip archive. Contains current versions of LINWorks software, Baby-LIN DLL, associated wrappers and Baby-LIN USB drivers as well as data sheets, manuals and program examples.

(376.6MiB) 21.07.30  



4.1.3 Installation

The LINWorks suite is delivered with a handy setup application. If you already have installed an older version you can simply install the newer versions. The setup application will take care of overwriting the required files. Simply follow these steps:

- Start the "Setup.exe".
- Select the components you want to install.
- Follow the instructions.



Warning

Please stop all running LINWorks applications and disconnect all Baby-LIN devices before starting the setup.



Version incompatibility

If you have used the SessionConf and SimpleMenu with version V1.x.x, the new version will be installed parallel to the old ones. Therefore you have to use the new shortcuts to start the new versions.

4.1.4 Check version

If you want to check the current version of the Baby-LIN-MB-II firmware or a LINWorks component the following table shows you how it is done:

Component	How to check the version
Baby-LIN-MB-II firmware	Start the SimpleMenu and connect to the Baby-LIN-MB-II . Now the firmware version is visible in the device list.
LINWorks: • LDFEdit • SessionConf • SimpleMenu • LogViewer • MB-Tool	Select the menu option "Help"/"About"/"Info". The info dialog will show the software version.
Baby-LIN-DLL	Call <code>BLC_getVersionString()</code> . The version is returned as string.
Baby-LIN-DLL .NET Wrapper	Call <code>GetWrapperVersion()</code> . The version is returned as string.




Advice

If you need support please always tell us the firmware and software versions you are using.

5 Handle voucher and activation codes

5.1 Convert a voucher code into an activation code

LIPOWSKY Industrie-Elektronik GmbH, Römerstr. 57, D-64291 Darmstadt



**LIPOWSKY
INDUSTRIE-ELEKTRONIK**

Embedded Solutions
CAN- and LIN-TOOLS

Record Date	23.08.2016
Record ID	62836
Contact	Frau Kerstin Lipowsky
Your Customer ID	10801
Our Supplier ID	511389
Your VAT-ID	
Your Order dated	22.08.2016
Order Reference	

Delivery Note Nr. 62836

Page 1 of 3


Pos.	Art-No.	Article Name	Amount	Batch Number
1	8000800	Option BL-HARP SDFV3-LIN Licence code for Baby-LIN-RM-II / HARP-4 to support enhanced LIN functions of LINWorks V.2.x (SDF V.3.x)	1 Stück	1 Total amount delivered batch 19722001

Convert your Voucher Code on
www.optionshop.de/lipowsky

Customs Tariffno 90309085
country of origin DE

Voucher codes: 57BC00CFB5A2

Lipowsky Industrie-Elektronik GmbH
Römerstrasse 57, 64291 Darmstadt
CEO: Dipl.-Ing. Andreas Lipowsky
Domicile: Darmstadt, Germany
Trade register: Darmstadt HRB 5139
VAT-ID: DE 111647423
WEEE-Reg-Nr. DE 43826474



Phone: +49 6151 93591-0 Fax: +49 6151 93591-28
Email: info@lipowsky.de Web: www.lipowsky.de
D-U-N-S® Nr. 341224640
Commerzbank AG BLZ 508 400 05 Account No.: 1408756
IBAN: DE19508400050140875600
SWIFT/BIC: COBADEFF508

These voucher codes have to be converted into activation codes using the target device's serial number. This can be done using the Lipowsky optionshop: www.optionshop.de/lipowsky. On this website click on "Convert voucher code".



Welcome to the Optionshop of the company Lipowsky Industrie-Elektronik GmbH!

Do you want to...

...Convert a Voucher code?

Here you can enter voucher codes and Device IDs of your designated device to generate the activation codes.

...or get activation code?

You already converted voucher codes but can not find your activation codes?

**Sprache wählen
/ Select language**

English ▾

On the next site you have to enter the voucher code and the serial number (sometime referred to as "Device ID") of the device you want the activation code for. Enter your e-mail address and click on "Get activation codes".



Convert voucher codes

Please type in your voucher codes and Device IDs for which the activation codes should be generated. Additionally, we need your email-address to which we should send the codes after generation.

Voucher code	Device ID	<input type="button" value="Remove voucher code"/>
<input type="text"/>	<input type="text"/>	
<input type="button" value="Add voucher code"/>		
E-Mail-Address	Confirm E-Mail-Address	
<input type="text"/>	<input type="text"/>	
<input type="button" value="Get activation codes"/>		

Note: Please remind that generating the codes might take up to 20 minutes, please do not send your request more than once.

A table will display all important information including the created activation code.

Device ID	Voucher code	Option	Part-no	Activation code
1382331	5333DD5EDEDE1	LIN-SDF3.X Support	8000800	EB201 HINRT BK00U OHZ1F 3DSVF IH

Additionally you will receive an e-mail with your activation codes.

Betreff: Activation code - summary / Freischaltcode - Auflistung
Von: "lipowsky@optionshop.de" <lipowsky@optionshop.de>
Datum: 14.05.2014 11:59
An: [REDACTED]

Verehrten Kunde,
 Dear customer,

wie gewünscht, senden wir Ihnen die erstellten Freischaltcodes.
 as requested, we send you the created activation codes.

Option: LIN-SDF3.X Support
 Part-no: 8000800
 Serial: 1382331
 Activation code: EB201 HINRT BK00U OHZ1F 3DSVF IH

Hinweis: Bitte beachten Sie bei der Eingabe des Freischaltcodes in Ihr Gerät den Unterschied zwischen 0 = Großbuchstabe "O" und 0 = Zahl "0"!
 Notice: When entering the activation codes into your device, please note the difference between 0 = capital letter "O" and 0 = number "0"!



Version incompatibility

The conversion of voucher codes into an activation codes may take some time. It can take up to 20 minutes until you receive your activation codes via e-mail.

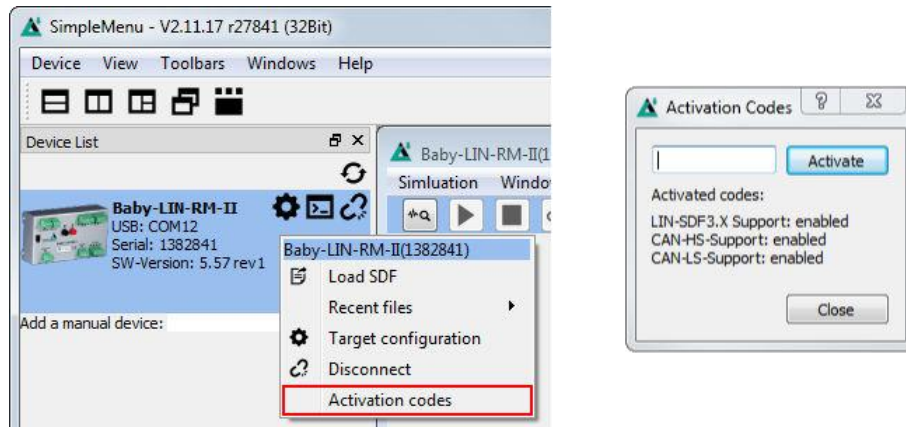


Version incompatibility

If you convert a voucher code into an activation code it will be bound to the device which serial number you entered. Once activated the voucher code can not be used for another device. There is no possibility to export an activation code from a device and regain your voucher code.

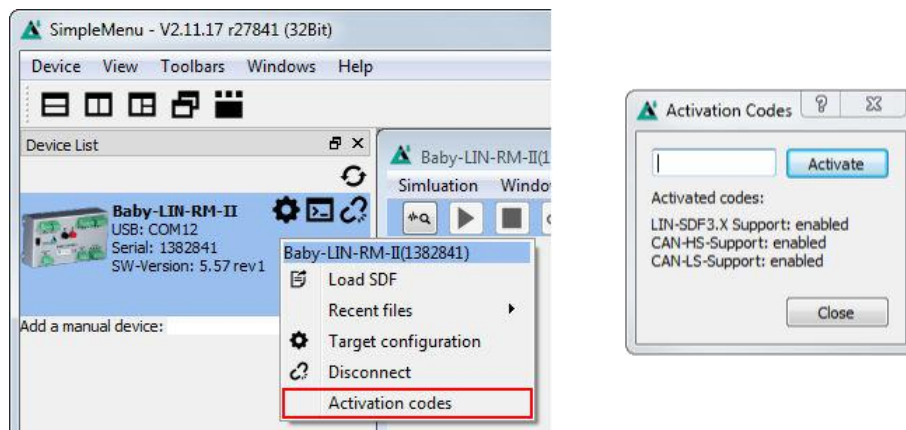
5.2 Redeem an activation code

The SimpleMenu can be used to redeem activation codes. Please connect the Baby-LIN-MB-II with a free USB port on a PC. Now start the SimpleMenu and establish a connection with your Baby-LIN-MB-II. Rightclick on the device image in the device list on the left and then choose "Activation codes". Enter your activation code in the new dialog and click on "Activate".



5.3 Check the installed activation codes

The installed activation codes can be displayed using the SimpleMenu. Please connect the Baby-LIN-MB-II with a free USB port on a PC. Now start the SimpleMenu and establish a connection with your Baby-LIN-MB-II. Right-click on the device image in the device list on the left and then choose "Activation codes". A dialog will then show you the installed activation codes.



5.4 Support information

In case of any questions you can get technical support by email or phone. We can use TeamViewer to give you direct support and help on your own PC. This way we are able to sort out problems fast and direct. We have sample code and application notes available, which will help you to make your job.

Lipowsky Industrie-Elektronik GmbH realized many successful LIN and CAN related projects and therefore we can draw upon many years of experience in these fields. We also provide turn key solutions for specific applications like EOL (End of Line) testers or programming stations.

Lipowsky Industrie-Elektronik GmbH designs, produces and applies the Baby-LIN products, so you can always expect qualified and fast support.

Contact informations	Lipowsky Industrie-Elektronik GmbH, Römerstr. 57, 64291 Darmstadt		
Website:	www.lipowsky.com	Email:	info@lipowsky.de
Telephone:	+49 (0) 6151 / 93591 - 0		

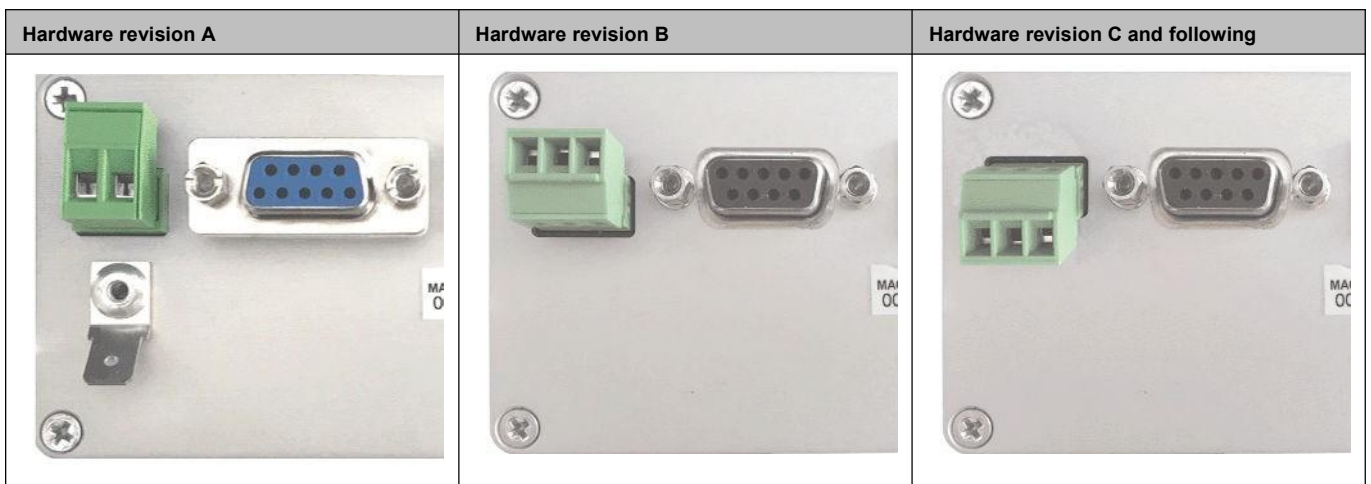
6 Hardware

6.1 Overview

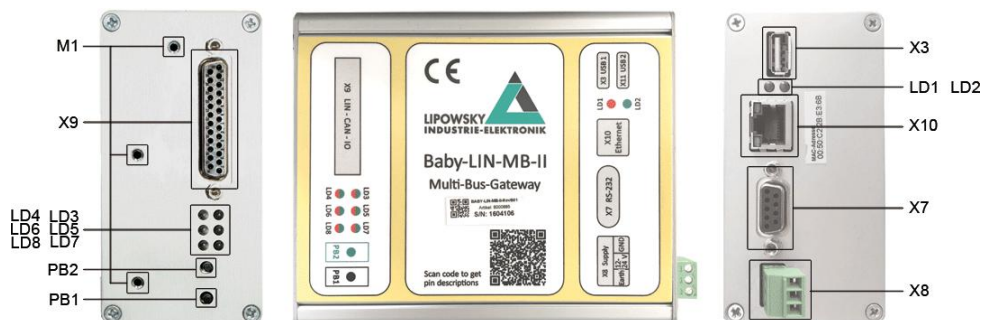
The following images show you what features the Baby-LIN-MB-II has to offer. The following features will be shown:

Abbreviation	Description
X	Connectors to access the different interfaces.
LD	LEDs that symbol certain states.
PB	Push buttons that trigger Baby-LIN-MB-II defined actions.
M	A Mechanical components that serve special purposes.

Please note that several changes to the layout of the supply and earth connectors were made. You can use the following images to find out, which hardware revision you are using.

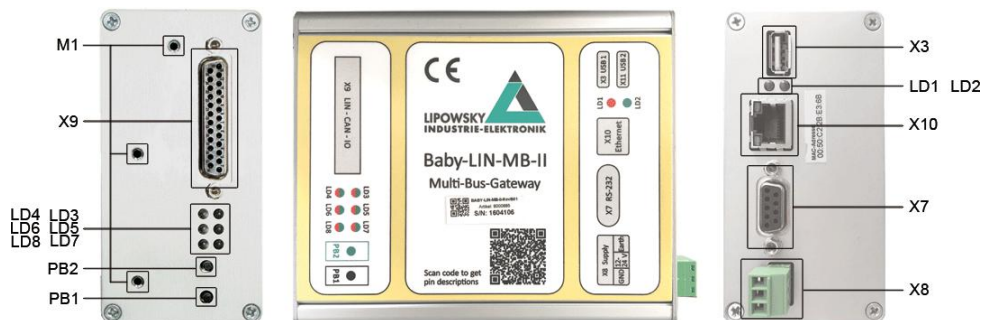


6.1.1 Hardware revision C and following



Abbreviation	Type	Description
X3	USB 2.0 type A	USB host interface connector.
X7	Sub-D-9 female	Serial RS232 connector.
X8	Socket for MC 1,5/ 3-ST-3,81 and MCVR 1,5/ 3-ST-3,81	Logic power supply with earth connection.
X9	Sub-D-25 female	Bus and I/O connector
X10	RJ-45	Ethernet port connector.
X11	-	This USB host interface is not available in the default configuration.
LD1	Red LED	Device error indicator
LD2	Green LED	Device state indicator
LD3-LD8	Red/green multi colored LED	Bus state indicator depending on installed MIF extensions.
PB1	Push button (black label)	A push button that triggers the copying of SDFs.
PB2	Push button (black label)	A push button that triggers the updating of the firmware.
M1	M3 screw thread	Is used to mount the optional Baby-LIN-MB compatibility adapter.

6.1.2 Hardware revision B



Abbreviation	Type	Description
X3	USB 2.0 type A	USB host interface connector.
X7	Sub-D-9 female	Serial RS232 connector.
X8	Socket for MC 1,5/ 3-ST-3,81 and MCVR 1,5/ 3-ST-3,81	Logic power supply with earth connection.
X9	Sub-D-25 female	Bus and I/O connector
X10	RJ-45	Ethernet port connector.
X11	-	This USB host interface is not available in the default configuration.
LD1	Red LED	Device error indicator
LD2	Green LED	Device state indicator
LD3-LD8	Red/green multi colored LED	Bus state indicator depending on installed MIF extensions.
PB1	Push button (black label)	A push button that triggers the copying of SDFs.
PB2	Push button (black label)	A push button that triggers the updating of the firmware.
M1	M3 screw thread	Is used to mount the optional Baby-LIN-MB compatibility adapter.

6.1.3 Hardware revision A

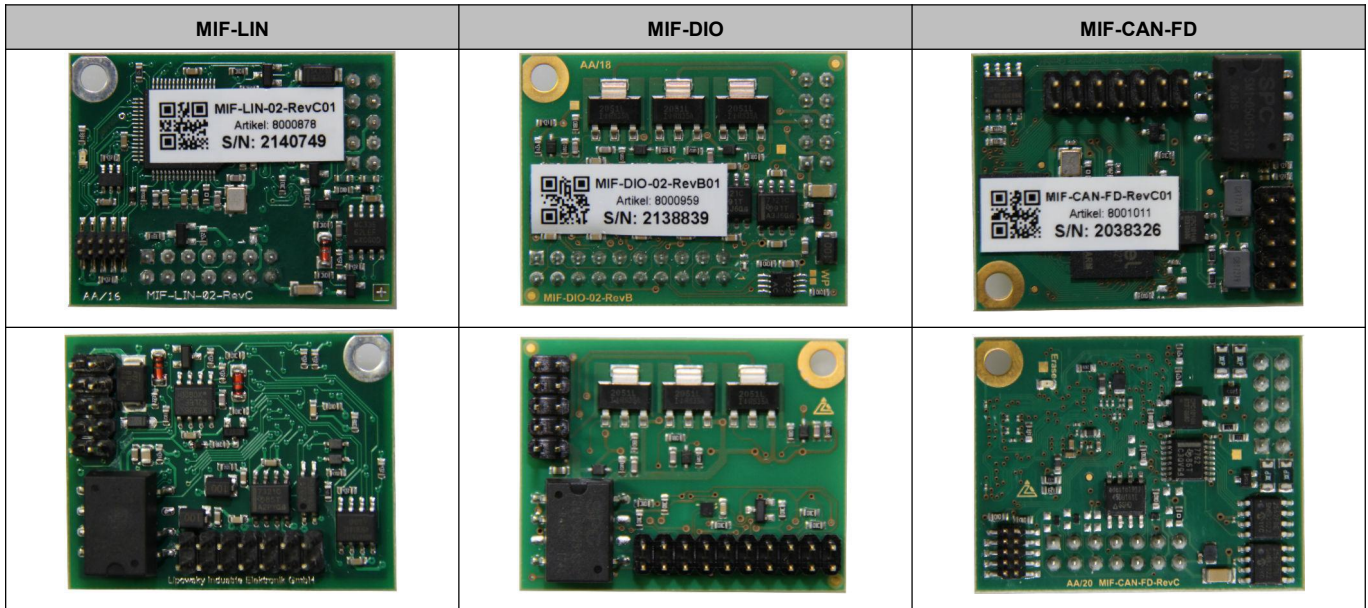


Abbreviation	Type	Description
X3	USB 2.0 type A	USB host interface connector.
X7	Sub-D-9 female	Serial RS232 connector.
X8	Socket for MSTB 2,5/ 2-ST-5,08 6,3 mm earth pin	Logic power supply with earth connection.
X9	Sub-D-25 female	Bus and I/O connector
X10	RJ-45	Ethernet port connector.
X11	-	This USB host interface is not available in the default configuration.
LD1	Red LED	Device error indicator
LD2	Green LED	Device state indicator
LD3-LD8	Red/green multi colored LED	Bus state indicator depending on installed MIF extensions.
PB1	Push button (black label)	A push button that triggers the copying of SDFs.
PB2	Push button (black label)	A push button that triggers the updating of the firmware.
M1	M3 screw thread	Is used to mount the optional Baby-LIN-MB compatibility adapter.

6.2 MIF extension modules

The Baby-LIN-MB-II features slots for up to 2 piggyback extensions. These MIF (mounted interface) extensions allow for a modularisation of the device. Therefore the device can meet strongly varying requirements (e.g. up to 6 independant LIN-Bus interfaces). The following MIF modules are available:

- MIF-LIN: adds 2 additional LIN-Bus interfaces
- MIF-DIO: adds 6 shared digital inputs/outputs
- MIF-CAN-FD: adds 2 additional CAN-Highspeed/CAN-FD Bus interfaces



Not all combinations of MIF extensions are possible. The following table shows all possible combinations:

MIF support by slot	MIF slot 1	MIF slot 2
MIF-LIN	✓	✓
MIF-DIO	✓	✗
MIF-CAN-FD	✗	✓

Version incompatibility

Please note that the second onboard LIN-Bus interface must be activated if you want to use additional LIN-Bus interfaces via MIF-LIN modules. Therefore only the following number of LINBus interfaces are possible:



Possible number of LIN-Bus interfaces	Required options
1	None
2	Option BL-MB-II LIN2
4	Option BL-MB-II LIN2, 1 x Option BL-MB-II MIF-LIN
6	Option BL-MB-II LIN2, 2 x Option BL-MB-II MIF-LIN

Version incompatibility

If the MIF-DIO extension is installed, the pins for the VLIN-Detect-1 to VLIN-Detect-4 pins are used by the MIF-DIO and therefore not usable. You will not be able to check if the LIN-Bus voltage of channel 1 or 2 is supplied.



Version incompatibility

If only one MIF-LIN is ordered, it will be installed at the first slot, therefore the LIN-Bus interfaces are available as channels 3 and 4. If one MIF-LIN and one MIF-DIO are ordered, the MIF-DIO will be installed at the first slot and the MIF-LIN will be installed at the second slot, therefore the LIN-Bus interfaces are available as channels 5 and 6.



Advice

Please note that a zero based index is used to access the channels via the host command protocol.



Channel	1	2	3	4	5	6
Index	0	1	2	3	4	5



Advice
If you have any questions or need some guidance please contact us: "Support information"

6.3 Connectors

6.3.1 X3 - USB host

This USB host interface is available via a USB 2.0 type A connector. Most USB mass storage devices can be connected and used.

This host interface is used to update the firmware, copy SDFs and as destination for log files.



Description

The connector uses the default pin assignment of a USB 2.0 type A.



Warning
The USB mass storage device has to be formatted as FAT16 or FAT32.



Warning
The maximum current output of the USB host interface is 500 mA. USB mass storage devices, that require higher currents may not work.

NOTICE

Do not remove the USB mass storage device while the Baby-LIN-MB-II is logging. Wait until the logging is finished after 5 minutes or press the black push button PB2 to abort it. Then wait until the green LED LD1 has stopped blinking.

6.3.2 X7 - RS-232

The default configuration of the Baby-LIN-MB-II is using this interface for the RS-232 MIF extension.

The RS-232 interface is available via a Sub-D-9 male connector.

This interface is used to communicate with the Baby-LIN-MB-II using the ASCII command protocol.

Hardware revision B and following:



Hardware revision A:



Pin	Signal	Description
X1	-	
2	TX	The TX signal of the RS-232 interface
3	RX	The RX signal of the RS-232 interface
4	-	
5	GND	The ground signal of the RS-232 interface
6	-	
7	-	
8	-	
9	-	



Advice

The RS-232 interface is galvanically isolated from the logic supply and the communication interfaces and the digital I/Os.

The serial port has the following properties:

- Baud rate: 9600 baud
- Configuration: 8-N-1
 - Data bits: 8
 - Parity bit: No parity bit
 - Stop bit: 1
- For the connection with a PC, a normal serial cable is used, not a null-modem cable.




Version incompatibility

The serial connection can not be used for the following features:

- Connect to the Baby-LIN-MB-II with the "SimpleMenu".
- Connect to the Baby-LIN-MB-II with the "Baby-LIN-DLL".
- Connect to the Baby-LIN-MB-II with the "Baby-LIN-MB-Tool".
- Access the "Web interface" of the Baby-LIN-MB-II.

6.3.3 X8 - Logic supply

The logic supply interface was changed at the beginning of the lifecycle of the Baby-LIN-MB-II. Please refer to chapter "Overview" to check which hardware revision you use.

Hardware revision C and following:														
Connector: MC 1,5/ 3-ST-3,81		<table border="1"> <thead> <tr> <th>Pin</th> <th>Signal</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Earth</td> <td>Earth connection (Earth switch cabinet)</td> </tr> <tr> <td>2</td> <td>24V</td> <td>Positive logic supply (8-32 VDCV)</td> </tr> <tr> <td>3</td> <td>GND</td> <td>The logic supply ground</td> </tr> </tbody> </table>	Pin	Signal	Description	1	Earth	Earth connection (Earth switch cabinet)	2	24V	Positive logic supply (8-32 VDCV)	3	GND	The logic supply ground
Pin	Signal	Description												
1	Earth	Earth connection (Earth switch cabinet)												
2	24V	Positive logic supply (8-32 VDCV)												
3	GND	The logic supply ground												
Hardware revision B:														
Connector: MC 1,5/ 3-ST-3,81		<table border="1"> <thead> <tr> <th>Pin</th> <th>Signal</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Earth</td> <td>Earth connection (Earth switch cabinet)</td> </tr> <tr> <td>2</td> <td>24V</td> <td>Positive logic supply (8-32 VDCV)</td> </tr> <tr> <td>3</td> <td>GND</td> <td>The logic supply ground</td> </tr> </tbody> </table>	Pin	Signal	Description	1	Earth	Earth connection (Earth switch cabinet)	2	24V	Positive logic supply (8-32 VDCV)	3	GND	The logic supply ground
Pin	Signal	Description												
1	Earth	Earth connection (Earth switch cabinet)												
2	24V	Positive logic supply (8-32 VDCV)												
3	GND	The logic supply ground												
Hardware revision A:														
Connector: MSTB 2,5/ 2-ST-5,08 6,3 mm earth pin		<table border="1"> <thead> <tr> <th>Pin</th> <th>Signal</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>24V</td> <td>Positive logic supply (8-32 VDCV)</td> </tr> <tr> <td>2</td> <td>GND</td> <td>The logic supply ground</td> </tr> <tr> <td>3</td> <td>Earth</td> <td>Earth connection (Earth switch cabinet)</td> </tr> </tbody> </table>	Pin	Signal	Description	1	24V	Positive logic supply (8-32 VDCV)	2	GND	The logic supply ground	3	Earth	Earth connection (Earth switch cabinet)
Pin	Signal	Description												
1	24V	Positive logic supply (8-32 VDCV)												
2	GND	The logic supply ground												
3	Earth	Earth connection (Earth switch cabinet)												



Advice

The logic supply interface is galvanically isolated from the the communication interfaces and the digital I/Os.



Warning

This earth connector should be connected with the local earth potential to guarantee a good EMI performance.

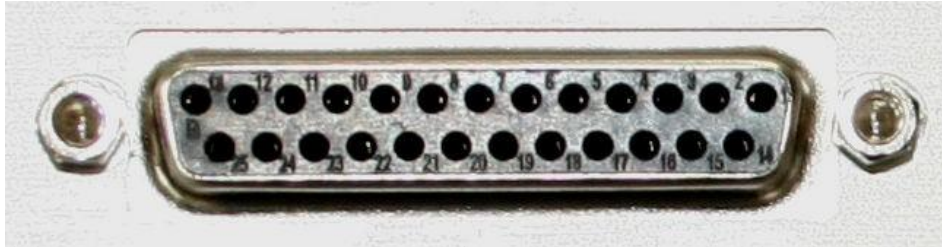


Attention

Do not operate the logic supply outside the voltage range of 8-32 VDCV.

6.3.4 X9 - LIN, CAN and IO

The LIN- and CAN-Bus interface as well as the digital I/Os are available via a Sub-D-25 connector.



Warning

Please be aware of the different indexing types. Check chapter "Channels" for more information.

Pin	Signal	Description	Required MIF module
X9-1	GND	Logic supply ground; also connected with X8-GND	
X9-2	VLIN-Detect-6	LIN supply detection for LIN-6 (typically connected to LIN-6 node supply)	MIF-LIN-2
	CAN2-H	CAN2-High signal of high speed MIF-CAN-FD interface	MIF-CAN-FD
X9-3	LIN-5	LIN-5-Bus signal	MIF-LIN-2
	CAN1-H	CAN1-High signal of high speed MIF-CAN-FD interface	MIF-CAN-FD
X9-4	MIF-DIO-IN/OUT-2	Digital input/output pin 2 of the MIF-DIO	MIF-DIO
X9-5	VLIN-Detect-4	LIN supply detection for LIN-4 (typically connected to LIN-4 node supply)	MIF-LIN-1
	MIF-DIO-IN/OUT-1	Digital input/output pin 1 of the MIF-DIO	MIF-DIO
X9-6	VLIN-Detect-2	LIN supply detection for LIN-2 (typically connected to LIN-2 X9-6 node supply)	
	MIF-DIO-IN/OUT-5	Digital input/output pin 5 of the MIF-DIO	MIF-DIO
X9-7	LIN-4	LIN-4-Bus signal	MIF-LIN-1
	MIF-DIO-IN/OUT-3	Digital input/output 3 of the MIF-DIO	MIF-DIO
X9-8	LIN-1	LIN-1-Bus signal	
X9-9	LIN-Supply	Supply connection for all LIN channels (typically connected to 8-26 VDC supply)	
X9-10	CAN-HS-L	CAN-Low signal of high speed CAN-Bus interface	
X9-11	CAN-GND	Ground for CAN-Bus interface	
X9-12	Switch-Port-2	Digital output; Normally open semiconductor relay Max: $I_{DCIAC} = 190 \text{ mA}$, $V_{DCIAC} = 30 \text{ V}$	
X9-13	DigIn	Digital input (Ground on X9-25)	
X9-14	Logic-Supply	Positive logic supply; also connected with X8-24V	
X9-15	LIN-6	LIN-6-Bus signal	MIF-LIN-2
	CAN1-L	CAN1-Low signal of high speed MIF-CAN-FD interface	MIF-CAN-FD
X9-16	VLIN-Detect-5	LIN supply detection for LIN-5 (typically connected to LIN-5 node supply)	MIF-LIN-2
	CAN2-L	CAN2-Low signal of high speed MIF-CAN-FD interface	MIF-CAN-FD
X9-17	VLIN-Detect-3	LIN supply detection for LIN-3 (typically connected to LIN-3 node supply)	MIF-LIN-1
	MIF-DIO-GND	Ground for all MIF-DIO signals	MIF-DIO
X9-18	VLIN-Detect-1	LIN supply detection for LIN-1 (typically connected to LIN-1 X9-18 node supply)	
	MIF-DIO-IN/OUT-6	Digital input/output pin 6 of the MIF-DIO	MIF-DIO
X9-19	LIN-3	LIN-3-Bus signal	MIF-LIN-1
	MIF-DIO-IN/OUT-4	Digital input/output pin 4 of the MIF-DIO	MIF-DIO

Pin	Signal	Description	Required MIF module
X9-20	LIN-GND	Ground connection for all LIN channels	
	MIF-CAN-FD-GND	Ground connection for all CAN channels	MIF-CAN-FD
X9-21	LIN-2	LIN-2-Bus signal	
X9-22	PWR-Switch	Switchable power output of LIN-Supply (X9-9).	
X9-23	CAN-HS-H	CAN-High signal of high speed CAN-Bus interface	
X9-24	Switch-Port-1	Digital output; Normally open semiconductor relay Max: $I_{DC/AC} = 190 \text{ mA}$, $V_{DC/AC} = 30 \text{ V}$	
X9-25	DigIn-GND	Ground for digital input	

CAUTION



Keep the LIN-Bus voltage within the following range: 8-26 VDC.

- Injury by damaged Baby-LIN-RM-III.
- The Baby-LIN-RM-III may get damaged.

CAUTION



Check LIN-Bus node specifications before using voltages above 18 VDC. If voltages in excess of 18 VDC are used as LIN-Bus supply voltage, it must be ensured that all connected nodes can cope with this voltage level. It is possible, that some nodes will function incorrectly in voltages exceeding 18 VDC, since the LIN specification states a maximum voltage of 18 VDC.

- Injury by damaged LIN-Bus nodes.
- LIN-Bus nodes may get damaged.



Advice

The following interfaces are galvanically isolated from each other:

- The LIN-Bus interfaces
- The CAN-Bus interface
- The digital I/Os of the MIF-DIO
- The RS-232 interface
- The Ethernet interface



Version incompatibility

All LIN-Bus interfaces, including the optional LIN-3-6 share the same supply ground and therefore have the same potential (LIN-GND).



Warning




The LIN-Bus supply must be provided by an external power supply and must not be interrupted during the LIN communication.



Advice

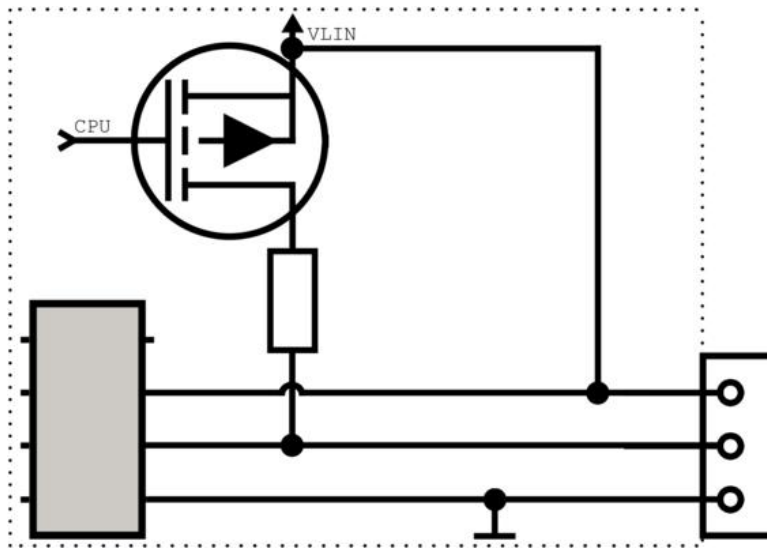
A terminating resistor can be switched active for the CAN interface and must not be connected externally. The size of the terminating resistor is 120 Ohm.

The following table gives you more details about the functional units and the pins they use:

Function	Pins	Signal	Description
Logic supply	X9-1 X9-14	GND Logic-Supply	This pins are used for the power supply of the logic. The logic supply voltage should be in the range of 8-32 VDCV. The ground and the positive supply voltage are connected with the ground and positive supply voltage of the X8 connector.
LIN-BUS 1	X9-20 X9-9 X9-8 X9-18	LIN-GND LIN-Supply LIN-1 VLIN-Detect-1	<p>These are the LIN-Bus interfaces. The LIN-Bus supply voltage has to be in the range of 8-26 VDC. The LIN-Bus interfaces support baudrates up to 115200 Baud. This means they also support protocols outside of the LIN specification. Supported LIN-versions are V1.2, V1.3, ... V2.2</p> <div style="background-color: #0070C0; color: white; padding: 5px; border-radius: 5px;">  Advice All LIN-Bus interfaces use the same ground and supply lines. </div> <p>The detect pins are used to overwatch the LIN-Bus supply voltage. The maximum signal cable length for the LIN-Bus interface is 30 m.</p> <div style="background-color: #FF8C00; color: white; padding: 5px; border-radius: 5px;">  Version incompatibility Not all LIN-Bus interfaces are available in the base version of the device. </div>
LIN-BUS 2	X9-20 X9-9 X9-21 X9-6	LIN-GND LIN-Supply LIN-2 VLIN-Detect-2	
LIN-BUS 3	X9-20 X9-9 X9-19 X9-17	LIN-GND LIN-Supply LIN-3 VLIN-Detect-3	
LIN-BUS 4	X9-20 X9-9 X9-7 X9-5	LIN-GND LIN-Supply LIN-4 VLIN-Detect-4	
LIN-BUS 5	X9-20 X9-9 X9-3 X9-16	LIN-GND LIN-Supply LIN-5 VLIN-Detect-5	
LIN-BUS 6	X9-20 X9-9 X9-15 X9-2	LIN-GND LIN-Supply LIN-6 VLIN-Detect-6	
CAN-HS	X9-23 X9-10 X9-11	CAN-HS-H CAN-HS-L CAN-GND	This is the CAN-Bus high speed interface according to ISO-11898. The maximum signal cable length for the CAN-Bus interface is 30 m.
MIF-CAN-FD	X9-2 X9-3 X9-15 X9-16 X9-20	CAN2-H CAN1-H CAN1-L CAN2-L MIF-CAN-FD-GND	These are two CAN-Bus high speed interface according to ISO-11898-1:2015, capable of a flexible data-rate up to 8Mbit/s. The maximum signal cable length for the CAN-Bus interface is 30 m.
Digital input	X9-13 X9-25	DigIn DigIn-GND	This is a digital input.
Switch port	X9-24 X9-12	Switch-Port-1 Switch-Port-2	This is a digital output. When activated the pins are switched conducting.
Power switch	X9-22	PWR-Switch	<p>This pin can be used to loop the LIN-Bus supply voltage via the Baby-LIN-MB-II through the LIN-Bus nodes. It can be deactivated and therefor switch off the LIN-Bus supply for the LIN nodes.</p> <div style="background-color: #0070C0; color: white; padding: 5px; border-radius: 5px;">  Advice The internal switch requires a minimum voltage of at least 12 V. </div>

Function	Pins	Signal	Description
MIF-DIO	X9-17	MIF-DIO-GND	These are digital I/Os. They share the same ground.
	X9-5	MIF-DIO-IN/OUT-1	
	X9-4	MIF-DIO-IN/OUT-2	
	X9-7	MIF-DIO-IN/OUT-3	
	X9-19	MIF-DIO-IN/OUT-4	
	X9-6	MIF-DIO-IN/OUT-5	
	X9-18	MIF-DIO-IN/OUT-6	

6.3.4.1 Equivalent circuit of the LIN-Bus interface:



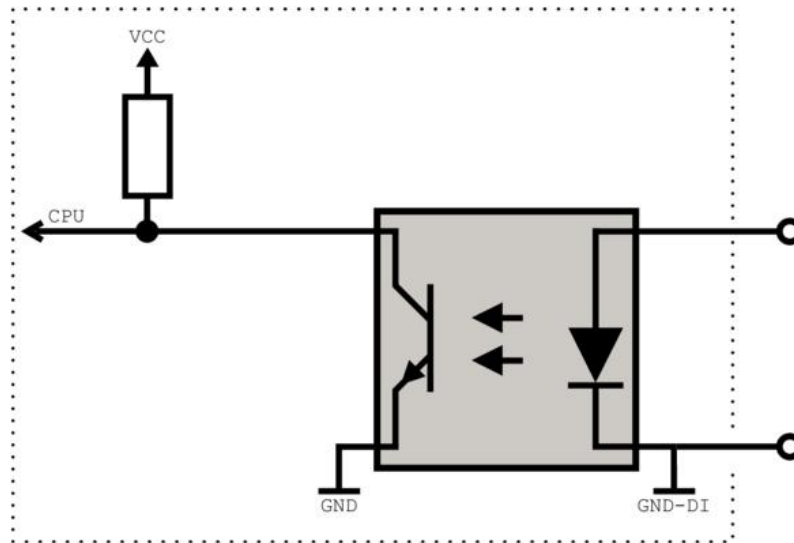
Electrical characteristics	Value	Unit
Maximum operational voltage	26	V
Maximum LIN voltage	32	V
Maximum LIN current	700	mA
Maximum LIN supply voltage	51	V
Maximum LIN supply current	700	mA

Electrical characteristics	Voltage threshold for LIN detection			Voltage threshold for LIN transceiver
	LIN interface	off	on	
Minimum	1	5.4 V	6.8 V	6.8 V
	2	3.1 V	4.8 V	
	3	5.0 V	6.8 V	
	4	2.8 V	4.8 V	
	5	5.0 V	6.8 V	
	6	2.8 V	4.8 V	

The pull-up resistor of the LIN-Bus driver is switched to 1 kOhm, if the master node is emulated and to 30 kOhm, if only slave nodes are emulated.

6.3.4.2 Equivalent circuit of the digital input of the Baby-LIN-MB-II:

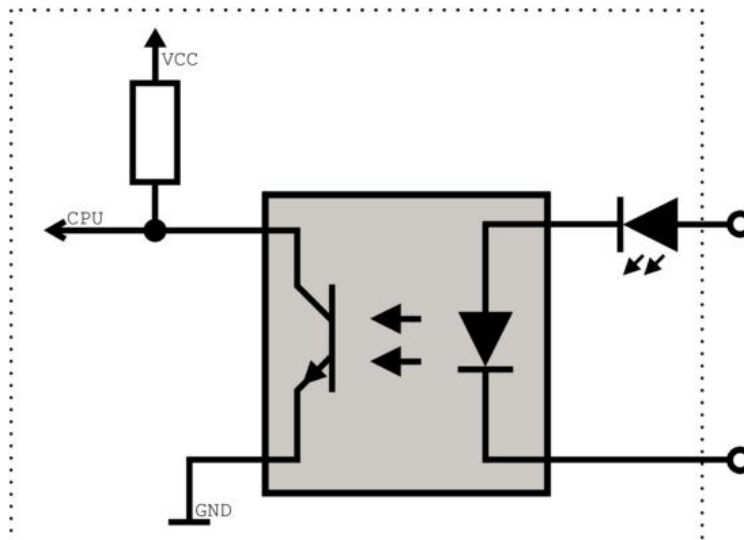
Pins: X9-13, X9-25



Electrical characteristics	Value	Unit
Maximum voltage for low level	1	V
Minimum voltage for high level	4	V
Typical current at 24V	8.5	mA
Maximum current	18	mA
Maximum voltage	44	V

6.3.4.3 Equivalent circuit of the digital inputs of the MIF-DIO:

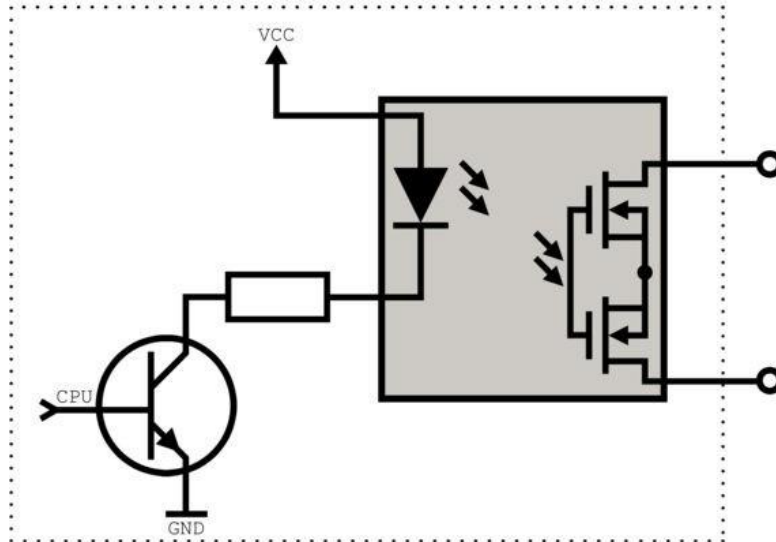
Pins: X9-17, X9-5, X9-4, X9-7, X9-19, X9-6, X9-18



Electrical characteristics	Value	Unit
Maximum voltage for low level	0.8	V
Minimum voltage for high level	7.2	V
Typical current at 24V	150	μ A
Maximum current	700	μ A
Maximum voltage	33	V

6.3.4.4 Equivalent circuit of the switch port of the Baby-LIN-MB-II:

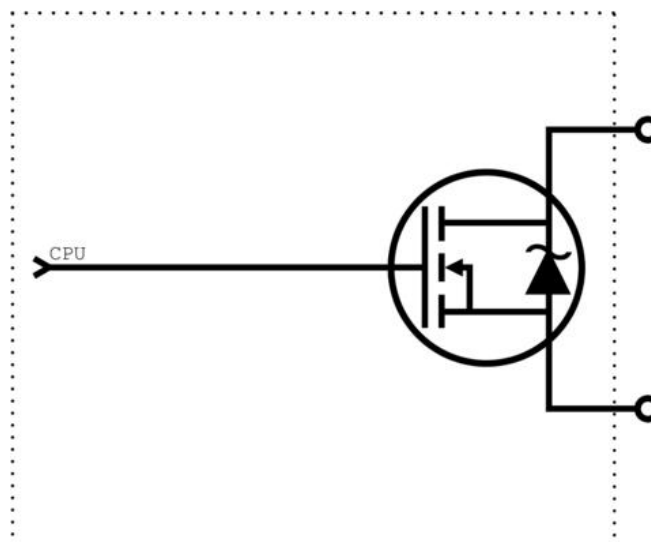
Pins: X9-24, X9-12



Electrical characteristics	Value	Unit
Minimum current for high level	190	mA
Maximum current	380	mA
Maximum voltage	24	V

6.3.4.5 Equivalent circuit of the power switch of the Baby-LIN-MB-II:

Pins: X9-22, X9-20



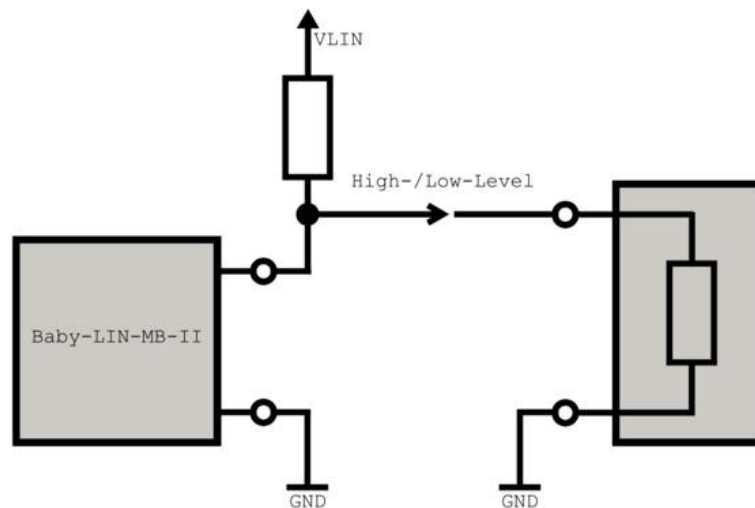
Electrical characteristics	Value	Unit
Minimum current for high level	500	mA
Maximum current	2.4	A
Maximum voltage	32	V

6.3.4.6 Equivalent circuit of the digital outputs of the Baby-LIN-MB-II:

Pins: X9-17, X9-5, X9-4, X9-7, X9-19, X9-6, X9-18

Electrical characteristics	Value	Unit
Maximum current	3.2	A
Maximum voltage	33	V

6.3.4.7 Connecting an input (e.g. a PLC input) to the digital output of the MIF-DIO:



The calculation of the pull-up resistor value must meet two criteria:

- The voltage threshold for the digital input of your device must be exceeded.

$$V_{Threshold} < V_{LIN} * (R_{IN} / (R_{in} + R_{PullUp}))$$

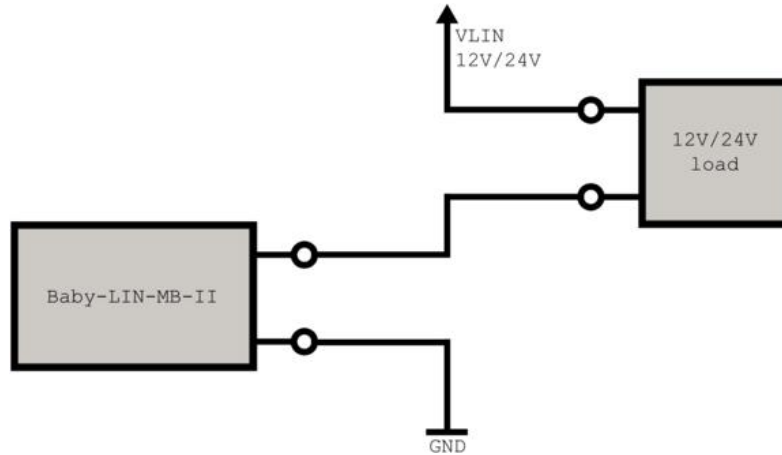
- The current through the digital output must be limited.

$$R_{PullUp} > V_{LIN} / 0.5A$$

Hence the pull-up resistor must meet the following inequation:

$$V_{LIN} / 0.5A < R_{PullUp} < R_{IN} * (V_{LIN} / V_{Threshold} - 1)$$

6.3.4.8 Connecting a load to a digital output of the MIF-DIO:



If a load is connected, please make sure the maximum current through the output of the Baby-LIN-MB-II is lower than 900 mA.

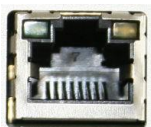
6.3.5 X10 - Ethernet

This Ethernet interface is available via a RJ-45 connector and supports transfer rates of 10/100 MBit. Additionally, it has an auto MDI-X feature.

Both dynamic and static IP addresses can be configured. Check chapter "Search and configure devices" for more information.

This interface is used to communicate with the Baby-LIN-MB-II using the "ASCII host command protocol".

Additionally, the web interface is accessible, which features system information as well as an easy SDF upload. Check chapter "Web interface" for more information.



Description

The connector uses the default pin assignment of a typical Ethernet RJ-45 connector.

The connector features a green and a yellow LED that signal different states:

LED	State	Description
Yellow LED	Off	10 MBit connection
	On	100 MBit connection
Green LED	Blinking	data are sent
	On	Connection exist



Advice

The Ethernet interface is galvanically isolated from the logic supply, the communication interfaces and the digital I/Os. The shield is connected with the case and the ground potential of the Baby-LIN-MB-II.

6.4 Power Supply

The Baby-LIN-MB-II can be powered by the following source:

- Power supply over connector: "X8 - Logic supply" Supported power supply voltage: 8-32 VDCV

The Baby-LIN-MB-II has a typical power consumption of 420 mA @ 24 VDCmA.

6.4.1 UPS - Uninterruptible power supply

The Baby-LIN-MB-II has an integrated UPS (uninterruptible power supply) that allows the safe shut down of the system during power fail events or keeps the system running on short power drops.

The power supply of the Baby-LIN-MB-II requires a voltage of 8-32 VDCV on connector "X8 - Logic supply" Below that minimum voltage a threshold exist (just below 7 VDC), under which the device will be powered for a short time by the UPS. This time will be used to signal a power failure, bridge a short power outage and guarantee a safe shut down.

During this time the error LED LD2 is blinking. The blinking will accelerate. Check chapter "LD1, LD2 - Device state and error indicator". for more information.

If the power supply is restored within about 2 s, the Baby-LIN-MB-II will continue to operate normally. If the power supply is not restored, the UPS will trigger a restart of the Baby-LIN-MB-II.

If the restart is triggered the device will check the power supply. If the power supply is still low, the device might not start or take a longer time. If the voltage is high enough but still below the threshold, it will signal the low power by a blinking of the error LED LD2 Check chapter "LD1, LD2 - Device state and error indicator".

The drawback of this feature is the missing possibility to reset the system by cutting the supply voltage for a short moment, which worked on the Baby-LIN-MB.

As a replacement for this method, the system offers a reboot feature, which can be triggered by pressing both buttons PB1 and PB2 together for at least 5 seconds. Check chapter "PB1, PB2 - Push buttons" for more information.

6.5 LEDs

6.5.1 LD1, LD2 - Device state and error indicator

LD2 is a green LED and indicates the state of the device. LD1 is a red LED and indicates errors.

Depending on the type of operation the LEDs show different patterns. Some of these patterns progresses through several stages.

Operation	Stage	Approximate Duration	LD2 Green LED	LD1 Red LED
While booting	1	System start up	On	On
	2	while Linux booting	Off	On
While booting and the power supply voltage is too low.		Until the power supply voltage is high enough	Off	Blinking
Normal operation			On	Off
System error		While the error lasts	Off	Blinking
System error: power failure		9 s	Off	Blinking with increasing rate
Temporary operations <ul style="list-style-type: none"> • Import SDFs • Firmware update • Log file export • Import of a network configuration file 	1	During update	Blinking (100 ms on, 100 ms off)	Off
	2	3s	On if operation succeeded	On if operation failed
Identification using the Baby-LIN-MB-Tool		Until the user stops it	Blinking alternately	

6.5.2 LD3-LD8 - Bus state indicator

The six LEDs LD3 to LD8 are multicolored LEDs. The following bus interfaces are assigned to the six LEDs:

LED	Bus interfaces	Requirements
LD3	LIN-1	None
LD4	LIN-2	"Option BL-MB-II LIN2"
LD5	LIN-3	First "Option BL-MB-II MIF-LIN"
	CAN-1	First "Option BL-MB-II MIF-CAN-FD"
LD6	LIN-4	First "Option BL-MB-II MIF-LIN"
	CAN-2	First "Option BL-MB-II MIF-CAN-FD"
LD7	CAN-HS	First "Option BL-MB-II CAN-HS"
	LIN-5	Second "Option BL-MB-II MIF-LIN"
LD8	LIN-6	

The following LED states are used to signal bus states and errors:

Bus interface	Green	Red	Information
LIN	Off	Off	No SDF ist loaded
	Blinking (500 ms on, 500 ms off)	Off	SDF is loaded but noch LIN-Bus voltage was detected
	Permanent	Off	SDF ist loaded and LIN-Bus voltage was detected
	Off	Blinking	An error has occurred. The error can have e.g. one of the following reasons: <ul style="list-style-type: none"> • Checksum error • Missing response
CAN-HS	Off	Blinking	An error has occurred on the CAN-Bus

6.6 Push buttons

6.6.1 PB1, PB2 - Push buttons

The push buttons are used for different actions depending on the time the button was pressed:

Push button	Pressing duration	Triggered action	Chapter with more details
PB1 (black)	300 ms < 1 s	Start logging to USB mass storage device.	"Logging"
PB1 (black)	>=5 s	Firmware update from USB mass storage device	"Firmware update"
PB2 (green)	300 ms < 1 s	Copy SDFs from USB mass storage device.	"Import of SDFs"
PB1 (black) and PB2 (green)	>= 5 s	Restart device	

6.7 Mechanics

6.7.1 M1 - Screw thread

The three M3 screw threads are used to mount the optional Baby-LIN-MB compatibility adapter.



Check chapter "Baby-LIN-MB compatibility adapter" for more information.

6.8 Hardware adaptations

6.8.1 Step by step guide: How to change the RTC battery

6.8.1.1 Introduction

The battery, that powers the RTC of the Baby-LIN-MB-II will usually last over 7Years years. Once its power is used up, the Baby-LIN-MB-II will loose the date and time. Then it is time to replace the RTC battery.

This step by step guide will show you how to change the battery of the real-time clock.

6.8.1.2 Required tools and materials

Please make sure you have the following tools and materials available before starting:

- A 5mm hex socket bit or nut spinner
- A PH1 screw driver
- A fresh 3V CR2430 button cell

6.8.1.3 Circuit board unboxing

Unscrew the 2 screws of the Sub-D-25 female connector using the 5 mm hex socket bit or nut spinner:



Turn around the device and unscrew the 4 case screws on the opposite site using the PH1 screw driver.



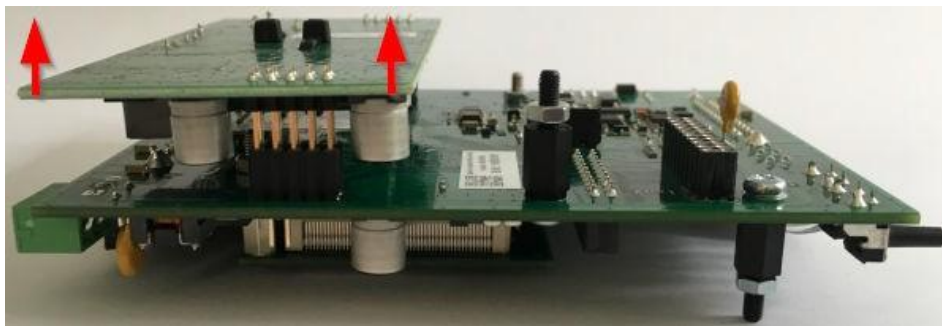
Now you can grasp the outer edges of the unscrewed panel and start to pull out the inner circuit boards.



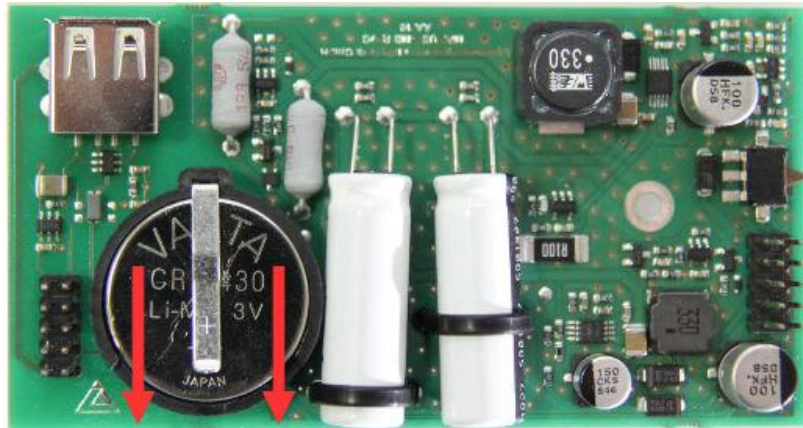
6.8.1.4 Disassembly of the UPS board

Hardware Revision A-C

Turn the device upside down so you can see the UPS board. Grab it on the outer edges and slowly pull it straight up.

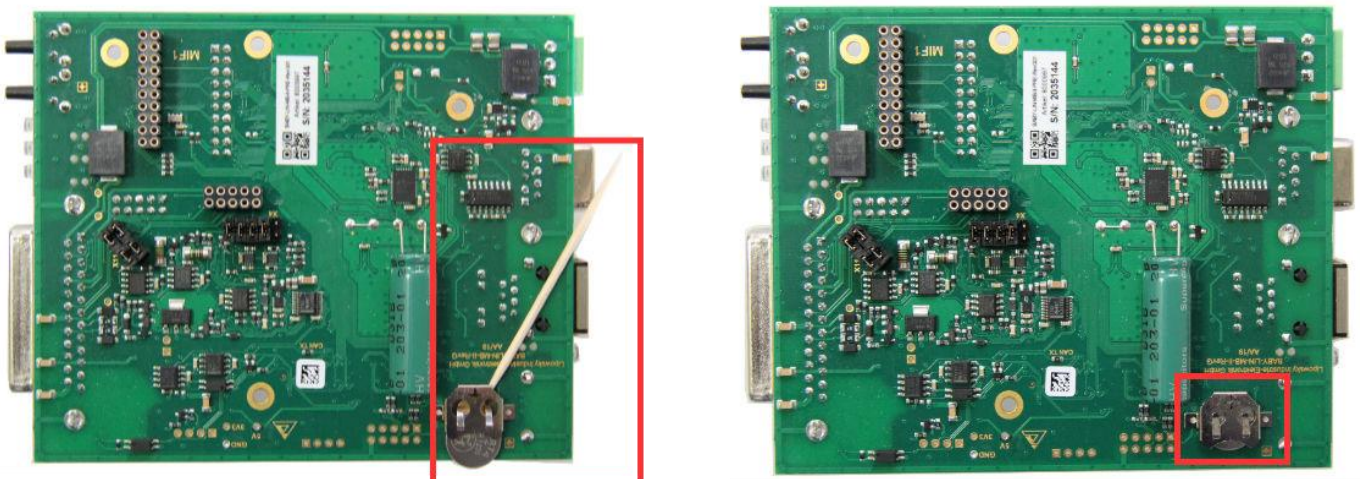


Turn around the UPS board. Now you should be able to see the button cell within the battery holder.



Hardware Revision D and following

The battery cells are located on the underside of the device. The battery can be easily pushed out of the holder with a narrow, non-conductive object.



6.8.1.5 The battery change

Now you can remove the old button cell and add a fresh 3V CR2430 button cell.

The spring clip needs to contact the + side of the battery.



Version incompatibility

The date and time of the RTC is stored within the processor of the SBC board. It is supplied by the battery of the UPS board. If this supply is interrupted, the RTC is reset. An interruption may have one of the following reasons:

- The SBC board is removed. • The RTC battery is removed from the UPS board.
- The UPS board is removed. • The RTC battery is empty.

If one of these events occur, you have to set the RTC date and time again.

6.8.1.6 Reassembly of the UPS board

Plug the UPS board back on the bottom side of the circuit board.

6.8.1.7 Circuit board boxing

Now you can reassemble the Baby-LIN-MB-II by reversing the steps:

- Push the circuit boards back into the case. The big base circuit board needs to be placed in the fifth notch from the top. The bottom UPC board needs to be placed in the second notch from the bottom.

NOTICE

Please make sure you push the circuit boards on the correct rails back in. Please make sure the LEDs and all connectors fit into the corresponding openings.



- Screw back the 4 case screws (Torque: 0.75 Nm).
- Screw back the 2 screws of the Sub-D-25 female connector (Torque: 0.5 Nm).

6.9 Installation of MIF extension modules

6.9.1 Introduction

This guide will show you how to install MIF extension modules.

Not all MIF extension modules can be placed on any MIF slot on the Baby-LIN-MB-II. Do not start until you know exactly, which MIF slot should be used. Please check chapter "MIF extension modules" for more informations.

NOTICE

MIF-DIOs can not always be installed by customers. This depends on the revision of the Baby-LIN-MB-II:

Baby-LIN-MB-II Revision	Installation	Instructions
Rev. A, Rev. B	Lipowsky Industrie-Elektronik GmbH	Please contact us to prepare the mailing of your Baby-LIN-MB-II. Check chapter "Support information" for more information.
Rev. C and following	Customer	Follow this installation guide

SAFETY INSTRUCTIONS

Please read this guide completely before you start the installation. Make sure you understood everything and have all the tools and materials required available.

NOTICE

The Baby-LIN-MB-II could be damaged. - Please observe ESD measures before modifying the Baby-LIN-MB-II, opening the case and touching the circuit boards! Ideally you would use an ESD-pad and ESD-wristband.




Advice

If you have any questions or need some guidance please contact us: "Support information"

6.9.2 Software installation

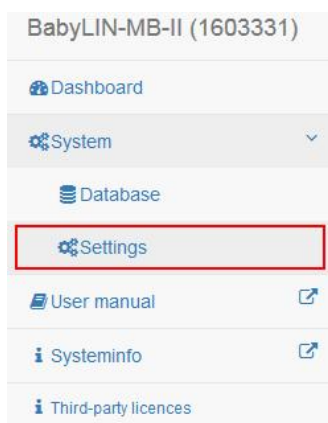
NOTICE

Please update the firmware of the Baby-LIN-MB-II before you install any MIF extension module. The recognition of the installed MIF extension modules may be influenced.


Attention

The older MIF-DIO-01 generation of MIF-DIOs require the following configuration step when installing or removing a MIF-DIO. MIF-DIO-02 are recognized automatically by the Baby-LIN-MB-II and do not require this configuration step.

Before you are installing or removing MIF expansions you have to check the registry variable "mifdio". Open the web interface of the Baby-LIN-MB-II and enter the "Settings" page. If a MIF-DIO-01 will be present after your installation/switch/removal of MIF expansions the variable "mifdio" has to be set to "mifdio-01". If no MIFDIO will be present the variable "mifdio" has to be set to "none".



Registry variables		
Show	10	entries
		Search: <input type="text"/>
Name	Type	Value
mifdio	Choice	mifdio-01

Please keep the following order when changing a variable:

1. Start the Baby-LIN-MB-II before changing anything.
2. Change the registry variable "mifdio". The change will not come into effect.
3. Shut down the Baby-LIN-MB-II.
4. Install/switch/remove the MIF expansion hardware.
5. Start the Baby-LIN-MB-II. The change will now come into effect.

NOTICE


Advice

The change of the registry variable will not come into effect, before the Baby-LIN-MB-II is restarted.

6.9.3 Hardware installation

6.9.3.1 Required tools and materials

Please make sure you have the following tools and materials available before starting:

- A 5 mm hex socket bit or nut spinner
- A 5.5 mm hex socket bit or nut spinner
- A PH1 screw driver

paragraphCircuit board unboxing

Unscrew the 2 screws of the Sub-D-25 female connector using the 5 mm hex socket bit or nut spinner:



Turn around the device and unscrew the 4 case screws on the opposite site using the PH1 screw driver.

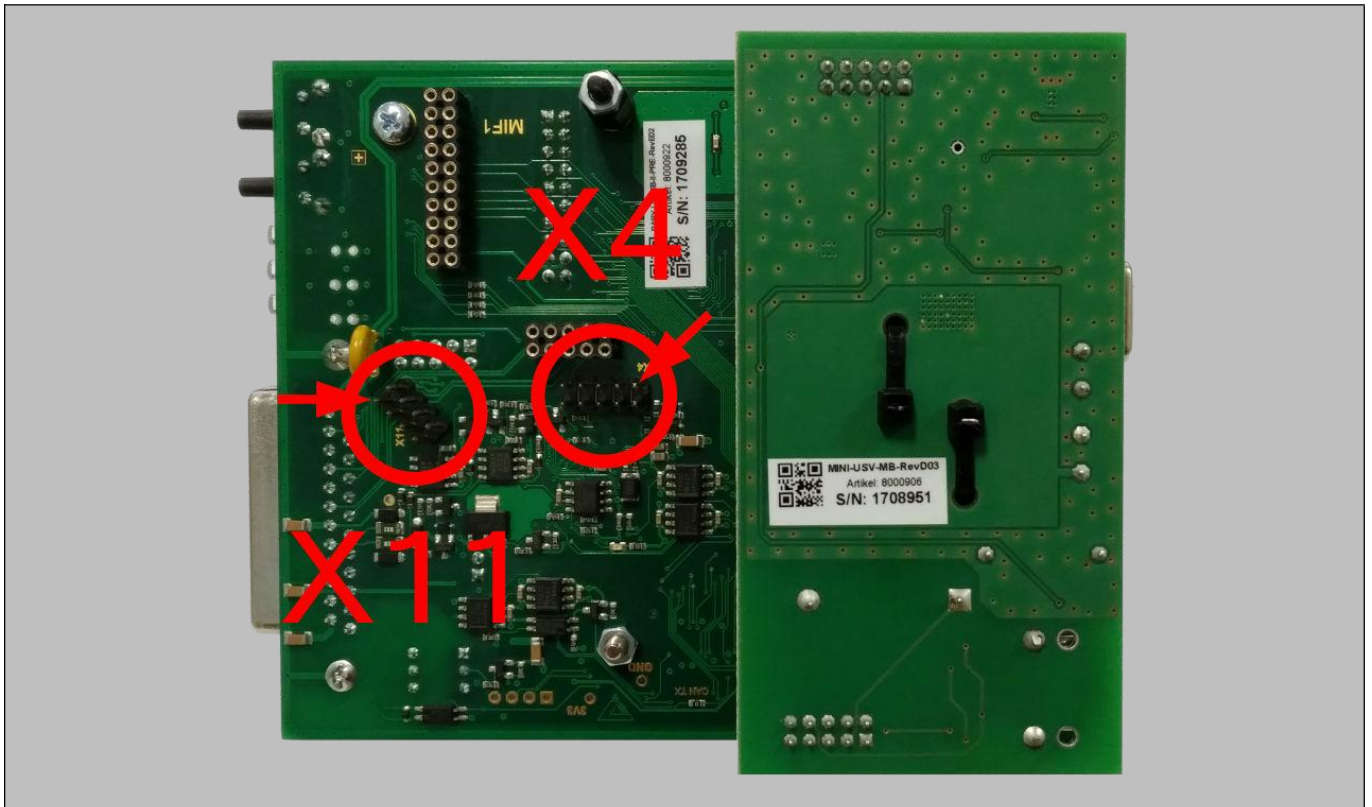


Now you can grasp the outer edges of the unscrewed panel and start to pull out the inner circuit boards.



6.9.3.2 Configure jumper

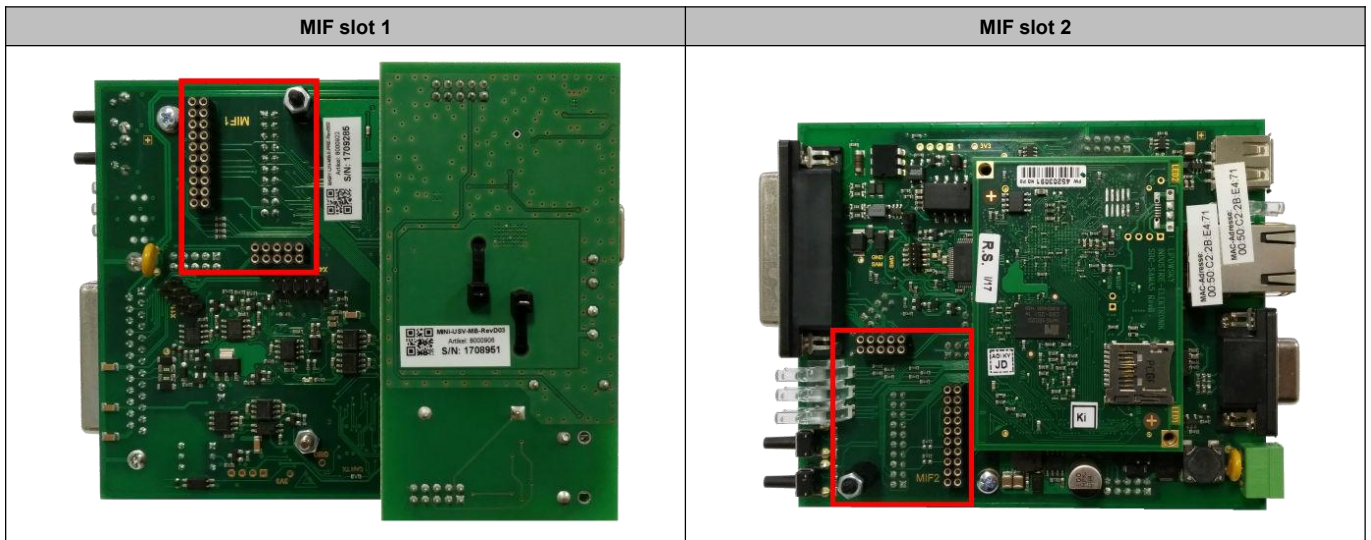
Each MIF slot on the circuit board has a dedicated set of jumpers. Depending on the MIF type that is installed, different combinations are required:



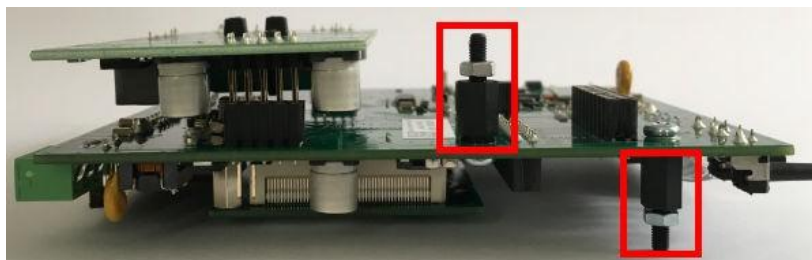
Jumper configuration [1]	MIF slot 1 X4	MIF Slot 2 X11
Not used [2]		
MIF-LIN [2]		
MIF-DIO [2]		Not supported
MIF-CAN-FD [2]	Not supported	

6.9.3.3 Install the MIF extension modules

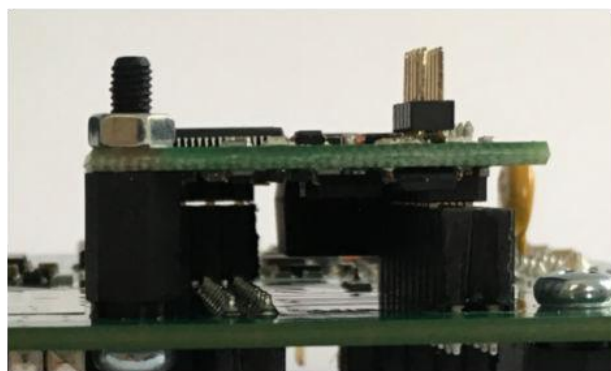
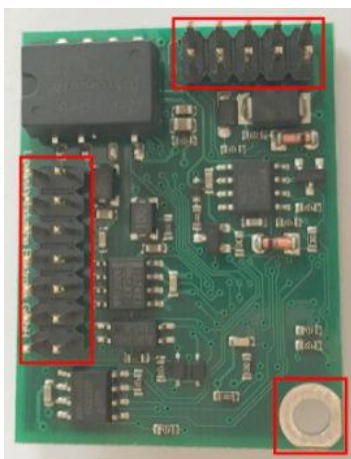
First you have to find the MIF slot you want to plug in the MIF extension. MIF slot 1 is on the bottom side while MIF slot 2 is on the top side.



Unscrew the nut off the spacer of the MIF slot, you want to use, using the 5.5 mm hex socket bit or nut spinner.



Now align the multi pin connectors of your MIF expansion with the slots on your Baby-LIN-MB-II. The spacer then will fit through the mounting hole of the MIF circuit board. Now screw back the nut to the spacer (Torque: 0.1 Nm).



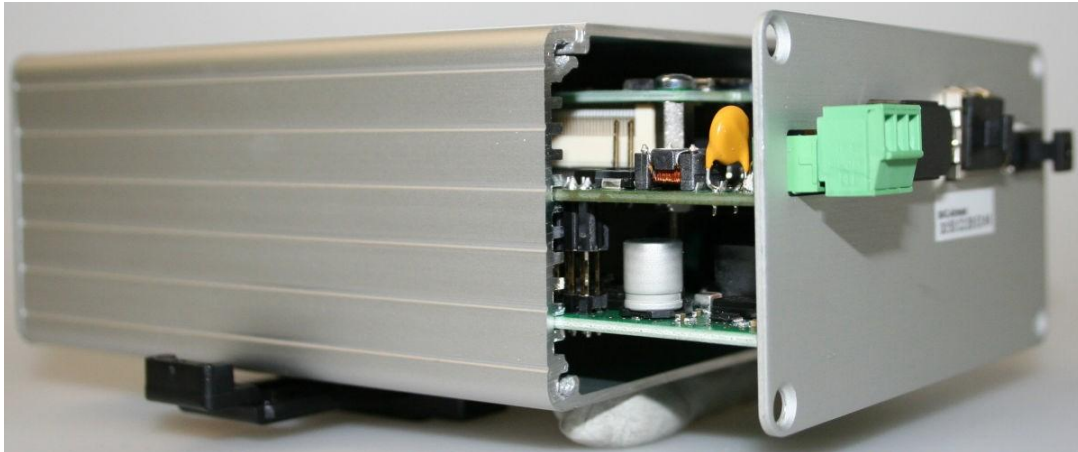
6.9.3.4 Circuit board boxing

Now you can reassemble the Baby-LIN-MB-II by reversing the steps:

- Push the circuit boards back into the case. The big base circuit board needs to be placed in the fifth notch from the top. The bottom UPC board needs to be placed in the second notch from the bottom.

NOTICE

Please make sure you push the circuit boards on the correct rails back in. Please make sure the LEDs and all connectors fit into the corresponding openings.



- Screw back the 4 case screws (Torque: 0.75 Nm).
- Screw back the 2 screws of the Sub-D-25 female connector (Torque: 0.5 Nm).

6.9.4 Baby-LIN-MB compatibility adapter

6.9.4.1 Introduction

The Baby-LIN-MB compatibility adapter is an optional hardware item that converts the Sub-D-25 female connector of the Baby-LIN-MB-II to two Sub-D-9 male connectors, that are pin compatible with the connectors of the Baby-LIN-MB. Therefore you can use the Baby-LIN-MB-II instead of the Baby-LIN-MB without the need to prepare a compatibility cable yourself.



The ordering information for the Baby-LIN-MB compatibility adapter are:

Optional hardware components		
Item number	Item	Description
8000916	BLMB-II-Dual-SUB-D9	Compatibility adapter to offer 2 Sub-D-9 connectors on the Baby-LIN-MB-II to recover the original Baby-LIN-MB pinning. Includes mounting materials.

Scope of delivery:

- Baby-LIN-MB compatibility adapter
- 3 x 11mm distance bolts
- 3 x M3x6mm screws
- 3 x M3 toothed lock washer

**SAFETY
INSTRUCTIONS**



Please read this guide completely before you start the installation. Make sure you understood everything and have all the tools and materials required available.

NOTICE



The Baby-LIN-MB-II could be damaged. - Please observe ESD measures before modifying the Baby- LIN-MB-II, opening the case and touching the circuit boards! Ideally you would use an ESD-pad and ESD-wristband.

6.9.4.2 Connectors

Front	Back
	
Compatible to the Baby-LIN-MB.	Connected to the Baby-LIN-MB-II.

This is the pin assignment for X5 - LIN:

Pin	Signal	Description
1	-	not connected
2	-	not connected
3	LIN-GND	Ground connection for the LIN-Bus
4	-	not connected
5	-	not connected
6	-	not connected
7	-	not connected
8	LIN	LIN-Bus signal
9	LIN-Supply	Supply connection for the LIN-Bus

This is the pin assignment for X6 - DIG-IO:

Pin	Signal	Description
1	-	IO-GND
2	-	not connected
3	IN	Ground connection for the LIN-Bus
4	-	not connected
5	-	not connected
6	-	not connected
7	-	OUT
8	-	not connected
9	-	not connected


Advice

The Baby-LIN-MB-II does not require a MIF-DIO to support the input and output of the Baby-LINMB's MIF-DIO. The digital input (X9-13, X9-25) and the switch port (X9-24, X9-12) of the basic Baby-LIN-MB-II will be used.


Advice

The SDFile does not require a change, if a Baby-LIN-MB is replaced with a Baby-LIN-MB-II with Baby-LIN-MB compatibility adapter. The mapping of the I/O system variables (SYSDIGIN and SYSDIGOUT) is compatible.


Warning

Since the digital output logic of the MIF-DIO of the Baby-LIN-MB was inverted to the logic of the Baby-LIN-MB-II the following configuration variable of the Baby-LIN-MB-II must be set to 1:

```
firmware.ios.digout0_invert
```

6.9.4.3 MB Adapter installation

The Baby-LIN-MB compatibility adapter can be easily installed. Simply follow these steps:


Advice

All distance bolts and screws have to be tighten with a torque of 0.75Nm.

NOTICE

The device could be damaged.

- Make sure the distance bolts and screws are thightened with the correct torque.
- Tighten the distance bolts at the screw threads M1. Check chapter "Overview" for more information.
- Plug the Baby-LIN-MB compatibility adapter onto the Baby-LIN-MB-II.
- Put the toothed lock washer on the screws and tighten the screws.

7 Firmware

7.1 The connection modes of the Baby-LIN-MB-II

The Baby-LIN-MB-II can be operated in two different modes. Certain operations are only available in one of the modes. The following table gives you an overview over the features available in each mode:

Feature	Description	Conenction mode	
		Stand-alone mode	SimpleMenu mode
"ASCII host command protocol"	The "ASCII host command protocol" can be used to control the Baby-LIN-MB-II via Ethernet or RS-232.	✓	X
"Stand-alone mode and autostart"	The autostart plugin loads a SDFile and starts the autostart macro after power on. In this stand-alone mode no PC or PLC is required.	✓	X
"UDP push logger connection"	This logging plugin pushes the bus frames using UDP messages to a server.	✓	X
Customer specific plugins	The Baby-LIN-MB-II supports customer specific plugins, that e.g. realize special ASCII commands.	✓	X
"Web interface"	The web interface shows detailed information about the Baby-LIN-MB-II. It also allows to switch the connection mode.	✓	(✓)
"SimpleMenu"	The "SimpleMenu" can be used to control all features of the Baby-LIN-MB-II with or without loading SDFiles.	X	✓
"Baby-LIN-DLL"	The "Baby-LIN-DLL" can be used by custom applications to control the Baby-LIN-MB-II.	X	✓
"Baby-LIN-MB-Tool"	The BM-Tool can be used to connect to the Baby-LIN-MB via Ethernet interface and execute text-based scripts with host commands.	✓	X

The following methods to change the connection mode are available:

Method/tool to change the mode	Steps to change the mode
Web interface	<ol style="list-style-type: none"> 1 Enter the IP address of the Baby-LIN-MB-II in a web browser. 2 Select <code>System</code> and then <code>Mode switch</code> in the menu on the left. 3 Select the mode 4 Click on <code>Set mode</code>
Baby-LIN-MB-Tool	<ol style="list-style-type: none"> 1 Start the "Baby-LIN-MB-Tool". 2 Select the Baby-LIN-MB-II from the <code>Devices in network</code> section. 3 Click on <code>Switch connection mode on the right</code>. 4 Select the mode. 5 Click on <code>Switch connection mode</code>.

Beside the connection mode you can also choose the duration:

Duration	Description
Static	Mode will be kept even after reboot.
Temporary	Mode will be reset after reboot.


Warning

After a firmware update the connection mode is reset to the Stand-alone mode. If you have operated the Baby-LIN-MB-II in the SimpleMenu mode before the firmware update, you have to switch it afterwards.

7.2 Firmware update

7.2.1 Introduction

The firmware is stored in the flash memory of the Baby-LIN-MB-II and can easily be programmed from a PC. The update process uses an ISP (In-system programming) operation and can be executed in the field.

As we are permanently working on product improvements, the firmware is updated periodically. These updates make new features available and solve problems, which have been discovered by our internal tests or have been reported by customers with earlier versions. All the firmware updates are done in a way, that the updated Baby-LIN-MB-II will continue to interwork with an already installed, older LINWorks installation. So updating the Baby-LIN firmware does not mean, that you necessarily have to update your LINWorks installation as well. Therefore it is highly recommended to always update your Baby-LIN-MB-II to the latest available firmware version.

7.2.2 Required software

The following downloads are required to update the firmware of the Baby-LIN-MB-II:

Download archive	Description
BabyLin-MB-II-SystemUpdate.zip	This package contains the firmware for the Baby-LIN-MB-II.

7.2.3 Update the firmware

To update the firmware of the Baby-LIN-MB-II you have to follow these steps:

- Unpack the firmware archive.
- Format a USB mass storage device with a FAT-16/32 file system.


Version incompatibility

Only FAT-16 or FAT-32 file systems are supported. The Baby-LIN-MB-II can not read from NTFS or any other file system.

- Create a folder named `BL-MB-01` within the root directory.
- Create a folder named `update` within the `BL-MB-01` directory.
- Copy the firmware file `BLMBUPD.tar` into the `update` directory.


Advice

The full path is: `USB: \BL-MB-01\update\BLMBUPD.tar`

- Unplug the USB mass storage device from the PC and plug it into the USB host interface of the Baby-LIN-MB-II.
- Press the black push button for at least 4 seconds.
- Wait until the flashing finished. During the flashing the green and red LED will flash alternatively.

- After the flashing the LED show, if the firmware update was successful:

LED	Result
Green LED on for 3 seconds	The firmware update was successful.
Red LED on for 3 seconds	The firmware update was not successful

See chapter "PB1, PB2 - Push buttons" for more details about the push buttons.

See chapter "LD1, LD2 - Device state and error indicator" for more details about the LEDs.

7.3 Import of SDF's

The operation of the Baby-LIN-MB-II system is basically defined by the SDFs available on the internal memory. These SDF files are typically created and prepared on a PC using the SessionConf.

To transfer SDFs to the Baby-LIN-MB-II you have to follow these steps:

- Format an USB mass storage device with a FAT-16/32 file system.



Version incompatibility

Only FAT-16 or FAT-32 file systems are supported. The Baby-LIN-MB-II can not read from NTFS or any other file system.

- Create a folder named BL-MB-01 within the root directory.
- Create a folder named database within the BL-MB-01 directory.
- Copy your SDFs into the database directory.



Advice

The full path is: USB: \BL-MB-01\database\filename.sdf

- Unplug the USB mass storage device from the PC and plug it into the USB host interface of the Baby-LINMB- II.
- Press the green push button for at least 300 ms but no longer than 1 second.
- Wait until the import finished. During the import the green LED will flash.
- After the flashing the LEDs show, if the SDF import was successful:

LED	Result
Green LED on for 3 seconds	The SDF import was successful.
Red LED on for 3 seconds	The SDF import was not successful

See chapter "PB1, PB2 - Push buttons" for more details about the push buttons.

See chapter "LD1, LD2 - Device state and error indicator" for more details about the LEDs.

7.4 Web interface

7.4.1 Overview

The Baby-LIN-MB-II offers a web interface. This web interface offers the following features:

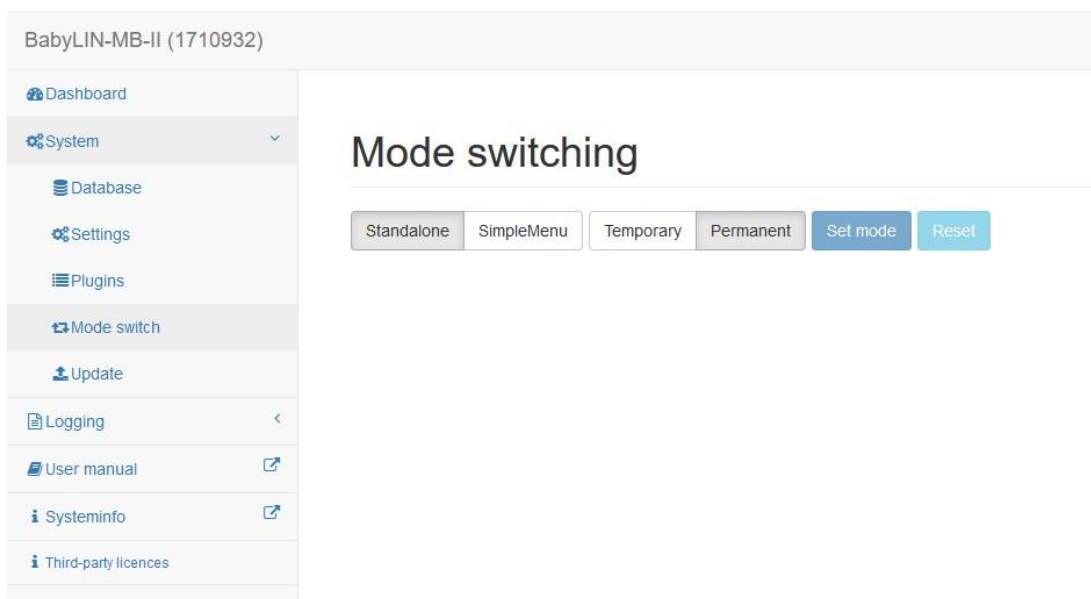
- The states of each bus interface.
- Information about the firmware and plugin versions.
- Show, upload and delete the SDFfiles on the Baby-LIN-MB-II.

- Set the date and time of the real-time clock.
- Install activation codes.
- Configure the Baby-LIN-MB-II.
- Install and remove plugins.
- Switch between the SimpleMenu and Stand-alone mode.
- View the user manual.
- Create system information and send them for debugging purposes via email.
- Check the third-party licences.

You can access the web interface by typing the IP address of the Baby-LIN-MB-II in a browser. The Baby-LINMB- Tool again is a comfortable way to get all IP addresses and you are also able to open the web interface of any listed Baby-LIN-MB-II by pushing a button.

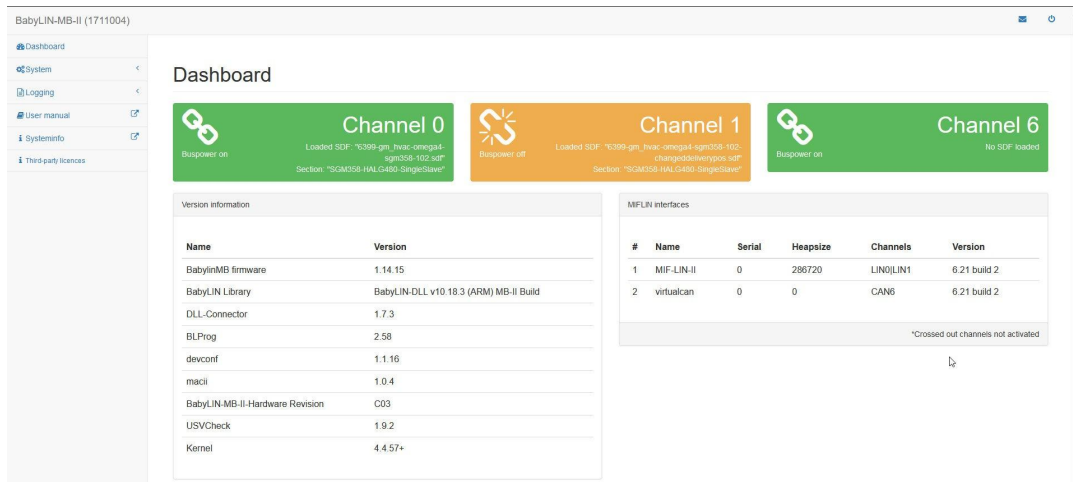
7.4.2 Mode switch

The Baby-LIN-MB-II has 2 operating modes that can be changed via the web interface. For full access to the functions of the web interface, the Baby-LIN-MB-II must be in stand-alone mode. The different modes can be set temporarily or permanently. It should be noted that the Baby-LIN-MB-II cannot be connected via the SimpleMenu if the unit is in stand-alone mode.



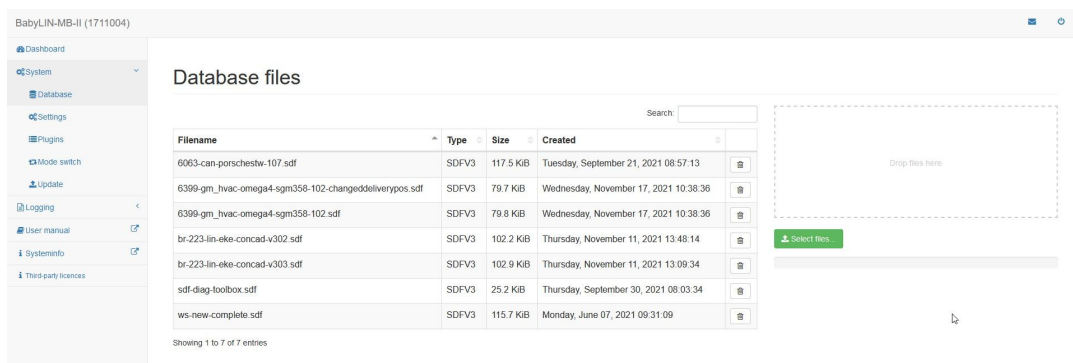
7.4.3 Dashboard

If you open the web interface of the Baby-LIN-MB-II, you will see the Dashboard page. Here you can find general information about the device. Furthermore, the active channels of the Baby-LIN-MB-II are listed with the status of the bus power and whether an SDF is loaded. The overview can thus be used to check the correct wiring.



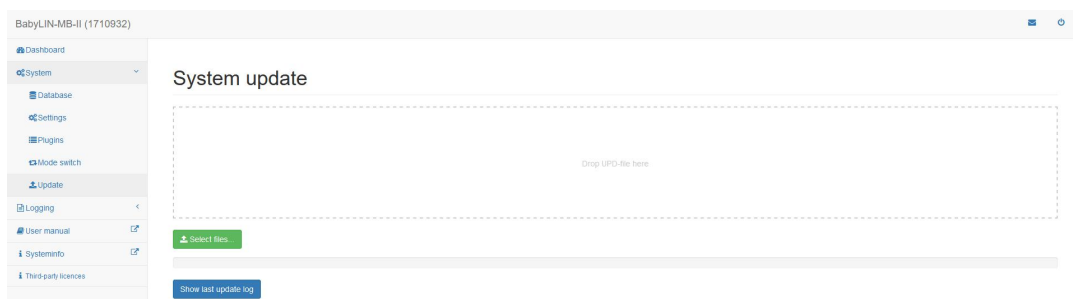
7.4.4 Datas files

The Database files page shows the stored SDFs that can be executed on the Baby-LIN-MB-II. Additional files can be added via drag and drop.



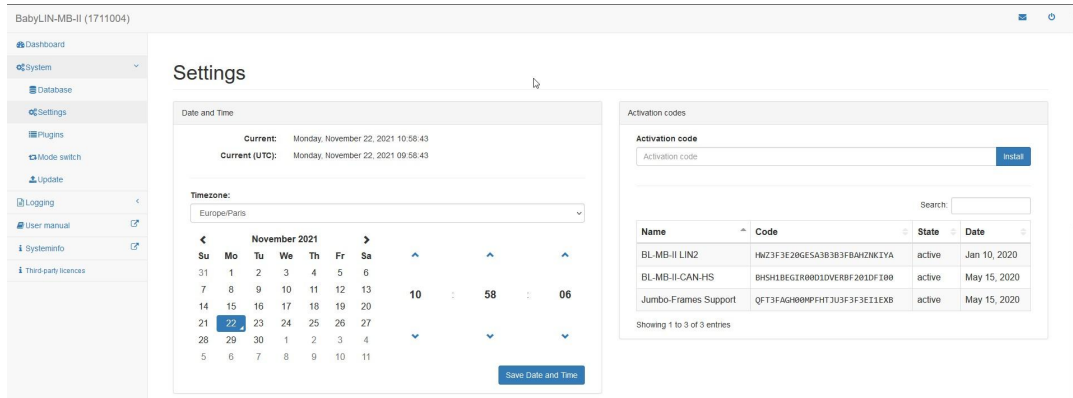
7.4.5 System update

The firmware of the Baby-LIN-MB-II is updated via the web interface. Under the System Update tab, the update file can be uploaded to the unit by drag and drop. The individual UDP file or the complete zip file can be used for the update.



7.4.6 Settings

In the "Settings" area, the current activation codes can be found and the date and time for the device can be set. By setting the configuration variables, the autostart mode can be defined as well as the IP address or the UDP port.



The following table shows all configuration variables.

7.4.6.1 Configuration variables


The Baby-LIN-MB-II can be configured using a set of configuration variables. To edit them select **System** and **Settings**.

Name	Description
<code>firmware.ios.digout0_invert</code>	If set the logic of the switch port output is inverted. This is only necessary for backward compatibility issues.
<code>web.support.mail.subject</code>	The default mail subject to use for a support email.
<code>web.support.mail.recipient</code>	The mail address to use as the support email recipient.
<code>web.support.mail.body</code>	The default mail content to use for a support email.
<code>autostart.mode</code>	Autostart-Modus: 0 = No SDFFile will be started automatically. 1 = A single pre-defined SDFFile will be started automatically. 2 = One of multiple SDFFile will be loaded depending on the state of the digital inputs.
<code>autostart.single.sdf</code>	This variable is only evaluated, if <code>autostart.mode == 1</code> . The name of the single SDFFile that is loaded if <code>autostart.mode</code> is 1.
<code>autostart.inputs.debounceTime</code>	This variable is only evaluated, if <code>autostart.mode == 2</code> . After a change of the relevant digital inputs this configurable debounce time will be waited until the new SDF is loaded. The value range lies between 10 ms and 1.000 ms.
<code>autostart.inputs.startindex</code>	This variable is only evaluated, if <code>autostart.mode == 2</code> . The starting input index from which the selection number is created.
<code>autostart.inputs.vectorsize</code>	This variable is only evaluated, if <code>autostart.mode == 2</code> . The number of inputs from which the selection number is created.
<code>autostart.inputs.<X>.sdf</code>	This variable is only evaluated, if 2. The name of the SDF that will be loaded at the start, if the selection number created from the inputs represents the value of X.
<code>udppl.channel.<X>.ip</code>	This is the IP address of a server that accepts frame log data as UDP packet. X is the index of a channel. Only LIN channels are supported.
<code>udppl.channel.<X>.port</code>	This is the UDP port of a server that accepts frame log data as UDP packet. X is the index of a channel. Only LIN channels are supported.

Name	Description
responseformat.hex.digitcount	This configuration variable is only evaluated by the VarRead command in mode 1. This configuration variable defines the number of digits for one hex value. The default is 0 (=No leading digit). Leading spaces will be filled with 0. If responseformat.hex.digitCount is 4 and the hex value is 1 the resulting output will be "0001".
responseformat.hex.case	This configuration variable is only evaluated by the VarRead command in mode 1. This configuration variable decides, if the hex values will be written using upper or lower case characters. The default value is Lower. Lower = "02 fa e2"; Upper = "02 FA E2".
responseformat.dec.digitcount	This configuration variable is only evaluated by the VarRead command in mode 0. This configuration variable defines the number of digits for one decimal value. The default value is 0 (=No leading digit). Leading spaces will be filled with 0. If responseformat.dec.digitcount is 4 and the decimal value is 7 the resulting output will be "0007".
firmware.logging.archiveperiod	Interval log files will be archived (every hour, 12am-6am-12pm-6pm, 12am-12pm, every day 12am, every monday 12am)
firmware.logging.background.disabled	Disable background logging (if configuration file is present) 0=active, 1=disabled
firmware.logging.sdf.disabled	Disable sdf logging ([LOG] syntax from SDF) 0=active, 1=disabled
firmware.logging.usblog.disabled	Disable background logging to USB flash drive 0=active, 1=disabled
device.info.location	The location of the device
device.info.name	The name of the device

7.4.6.2 Registry variables

The Baby-LIN-MB-II can be configured using a set of registry variables. To edit them select **System** and **Settings**.

Name	Description
rs232-baudrate	This variable is the baud rate of the RS-232 interface.
rs232-parity	This is the configuration of the parity bit of the RS-232 interface. Possible values: <ul style="list-style-type: none"> • None • Odd • Even
rs232-stopbit	This variable is the configuration of the stop bit of the RS-232 interface. Possible values: <ul style="list-style-type: none"> • One • Two
mifdio	If a MIF-DIO-01 is installed, this variable has to be set manually. Possible values are: <ul style="list-style-type: none"> • "none": No MIF-DIO-01 is installed. • "mifdio-01": A MIF-DIO-01 is installed. • Another string (will be set automatically): A MIF-DIO-02 ist recognized.
ntp_server	This variable is the address of a NTP server. The Baby-LIN-MB-II will connect to this server to update its time.
can_terminator	This variable can activate the on-board CAN terminator of 120 Ohm. <div style="display: flex; align-items: center;">  <div style="background-color: #ff9900; padding: 5px;"> <p>Version incompatibility</p> <p>This variable is not available on hardware revisions A and B. It was introduced with hardware revision C. Check chapter "Overview" for more information.</p> </div> </div>
simplemenu-tcpcom-port	This variable sets the TCP port, the SimpleMenu and Baby-LIN-DLL are using to connect to the Baby-LIN-MB-II. The default value is 2048.
tcp-com-port	This variable sets the TCP port, that is used by the Baby-LIN-MB-II to receive commands of the "ASCII host command protocol". The default value is 10002.
tcp-dbg-port	This variable sets the TCP port, the Baby-LIN-MB-II is using to send debug informations. The default value is 2709.
tcp-low-latency	This variable enables the low latency mode for the TCP communication. This mode increases the communication speed, but results in a higher CPU load on the Baby-LIN-MB-II. The higher CPU load could lead to problems if applications with a lot of communication are using a lot of channels (> 4).

7.5 Host interface

7.5.1 Introduction

The Baby-LIN-MB-II features an autonomous "Stand-alone mode and autostart" and it can also be controlled by custom PC applications using the "Baby-LIN-DLL". But the specialty is the host interface.

The Baby-LIN-MB-II is usually used in environments, in which it is controlled by a PC or a PLC. Therefore it features a network interface and a serial interface. Each of these interfaces allow controlling the Baby-LIN-MB-II using the "ASCII host command protocol".

To test any of the two host interfaces or to develop applications using the "ASCII host command protocol", the "Baby-LIN-MB-Tool" will be a great help.

A detailed description and instructions for the use of the MB-Tool can be found in the ApplicationNote-MB-Tool.pdf. It also contains a list of all ASCII Host Commands. You can download the document under the following link: www.lipowsky.de/downloads

7.5.2 Serial interface

The serial interface uses a Sub-D-9 male RS-232 connector. Check chapter "X7 - RS-232" for more information.

To communicate with the Baby-LIN-MB-II using this interface, the following settings are required:

Setting	Value
Baud rate	9600
Data bits	8
Parity	No Parity
Stop bit	One

7.5.3 Network interface

The serial interface uses a RJ-45 ethernet connector. Check chapter "X10 - Ethernet" for more information.

To communicate with the Baby-LIN-MB-II using this interface, the IP settings must match. You can find and configure any Baby-LIN-MB-II using the "Baby-LIN-MB-Tool".

The host interface is listening on TCP port 10002.

7.5.4 Network configuration

The IP configuration of the network interface supports DHCP server and static IP addresses.

Type	Description	Chapter
DHCP	The IP address is obtained by DHCP server. This DHCP server will prevent address conflicts.	"DHCP configuration"
Static IP address	The IP address will always be fixed and never change. Therefore address conflicts of IP addresses are possible.	"Static IP address"

You can configure the network configuration of the Baby-LIN-MB-II using the following methods:

Method	Description	Chapter
Baby-LIN-MB-Tool	This is the most easiest way to configure your Baby-LIN-MB-II.	"Configuration using the Baby-LIN-MB-Tool"
USB mass storage device	You should use this method, if the Baby-LIN-MB-Tool was not able to find your Baby-LIN-MB-II.	"Configuration using a USB mass storage device"

7.5.5 DHCP configuration

Per default the Baby-LIN-MB-II is configured to obtain an IP address from a DHCP server. If you connect the Baby-LIN-MB-II to a network with a DHCP server available and your PC is connected to the same network, you should be able to establish a connection.

The Baby-LIN-MB-II will try to obtain an IP address from the DHCP server in the following situations:

- The Baby-LIN-MB-II restarts.
- The network cable is plugged in, after it was disconnected for a certain time.

Before you can establish a connection to the Baby-LIN-MB-II you have to find out the obtained IP address. The easiest way to do this is by using the "Baby-LIN-MB-Tool".



Advice

This is the default setting of the Baby-LIN-MB-II.

If no IP address can be obtained from a DHCP server, the Baby-LIN-MB-II will use the following static IP address:

IP address	169.254.0.9
Subnet mask	255.255.255.0

If your PC is in the same network you can establish a connection, if you set the IP address of your PC to e.g. 169.254.0.10 and the subnet mask to 255.255.255.0.

7.5.6 Static IP address

If your network does not have a DHCP server or you rely on static IP addresses, you can set up the Baby-LINMB- II to use a static IP address.

The advantage is, that you can rely on the fact, that the IP address will never change. The disadvantage is that you have to manage all IP addresses by hand and you have to prevent IP conflicts.

7.5.7 Configuration using the Baby-LIN-MB-Tool

The Baby-LIN-MB-Tool can be used, among other things, to search Baby-LIN-MB-II in your local network and to set the network configuration. It allows you to enable the DHCP mode or set a static IP address. It is the easiest and most comfortable way to configure one or multiple Baby-LIN-MB-II.

Check chapter "Baby-LIN-MB-Tool" for more information.

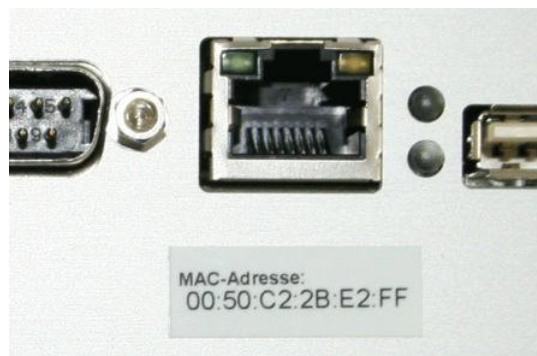
The Baby-LIN-MB-Tool is part of the LINWorks that is available for free for all Baby-LIN customers.

7.5.8 Configuration using a USB mass storage device

The network configuration of a Baby-LIN-MB-II can also be set by creating a text file on a USB mass storage device.

The name of the text file is derived from the MAC address of the network interface. Therefore it is possible, to create multiple configuration files on one USB mass storage device and then update multiple Baby-LIN-MBII with it.

You can find the MAC address on a label below the RJ-45 ethernet connector.



File name	The file name is derived from the MAC address. if the MAC address is: 00:50:C2:2B:E2:FF The configuration file name then has to be: 00-50-C2-2B-E2-FF.txt
Folder name	The folder, where the configuration files needs to be stored, is: \\BL-MB-01\\ipconfig\\
File content for a DHCP configuration	DHCP
File content for a static IP address	IP = 192.168.129.253 NETMASK = 255.255.255.0 GATEWAY = 192.168.129.1 DNS = 192.168.129.3

You can create as many configuration files on the USB mass storage device as you want. The Baby-LIN-MBII will search the matching configuration file depending on the MAC address.

If a USB mass storage device is inserted into the USB host interface of a Baby-LIN-MB-II, the device will check automatically, if a configuration file with matching MAC address exists. If such a file is found, the current and destined configuration are compared. If they are the same, nothing will happen. If a new configuration is found, it will be used to set up the new network configuration.

During this update process the network interface will shut down, re-configure itself and the start up again. All this will be signaled by the flashing red LED LD2. If the update process was successful, the green LED LD1 will be on for 3 seconds. If the update process was not successful, the red LED LD2 will be on for 3 seconds. Check chapter "LD1, LD2 - Device state and error indicator" for more information.

7.6 Logging

7.6.1 Introduction

The Baby-LIN-MB-II supports the logging of the frames on the LIN- and CAN-Bus as well as debug information. The following possibilities to log the bus data are available:

- The bus data can be logged using the SimpleMenu if the Baby-LIN-MB-II is connected to a PC.
- The log data can be written to a USB mass storage device and different network protocols without a PC.

7.6.2 Configure and activate the logging

The Baby-LIN-MB-II supports different types of logging:

Logger	Interface	Description
Local logging	"X3-USB host"	<p>The Baby-LIN-MB-II supports the logging of frames to a USB mass storage device. When the logging is triggered, the Baby-LIN-MB-II will log all frames for a duration of 5 minutes. Additionally all received host commands will be logged.</p> <p>To trigger the creation of the log file press the black push button PB1 between 300 ms and 1 s. To stop the logging, while it is active, press the black push button PB1 for less than 1 s. Check chapter "PB1, PB2 - Push buttons" for more information.</p> <p>The format of the logging file written to the USB mass storage device is the "Binary format". A conversion to the "ASCII format" is automatically executed.</p> <p>This logging mainly is used to validate the accuracy of an implemented application or debugging purposes.</p>
UDP push logger plugin	"X10 - Ethernet"	<p>The UDP push logger is a plugin for the Baby-LIN-MB-II, that sends log data to a UDP server. Each UDP packet represent a single log line.</p> <p>The UDP push logger starts sending log data after the target IP address and UDP port are set using "Configuration variables".</p> <p>The format of the logging data sent is always "ASCII format".</p>
Debug log	"X10 - Ethernet"	<p>The Baby-LIN-MB-II hosts a TCP server that distributes debug log data. The server waits for clients that connect to the TCP port 2709 and starts sending the debug log data as soon as a client is connected.</p> <p>The debug log data are sent in an ASCII format and do not contain any bus information. This debug log should only be evaluated, if you are directly asked to by the Lipowsky Industrie-Elektronik GmbH.</p>

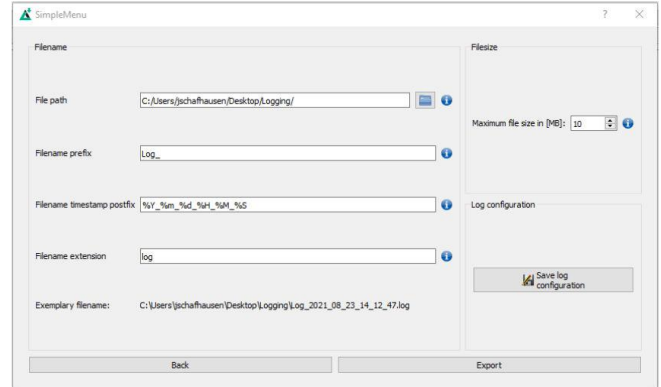
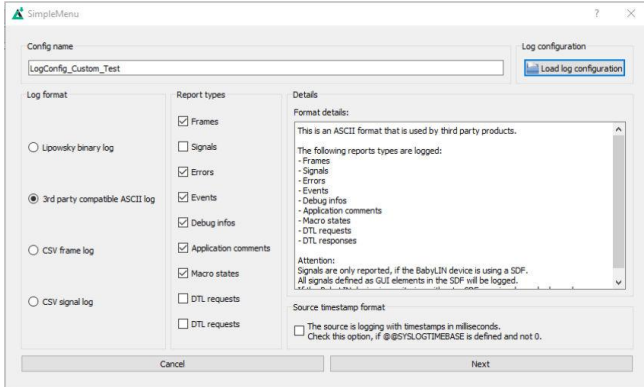
The "Baby-LIN-MB-Tool" offers different logging possibilities that allow you to receive and write the log data from the Baby-LIN-MB-II. Therefore you only have to implement your own logging component in very few cases.

7.6.3 SimpleMenu Logging

The SimpleMenu offers the possibility to record the complete bus communication of the Baby-LIN-Device and to save it locally in a file on the computer. To do this, open the integrated log viewer in the SimpleMenu.



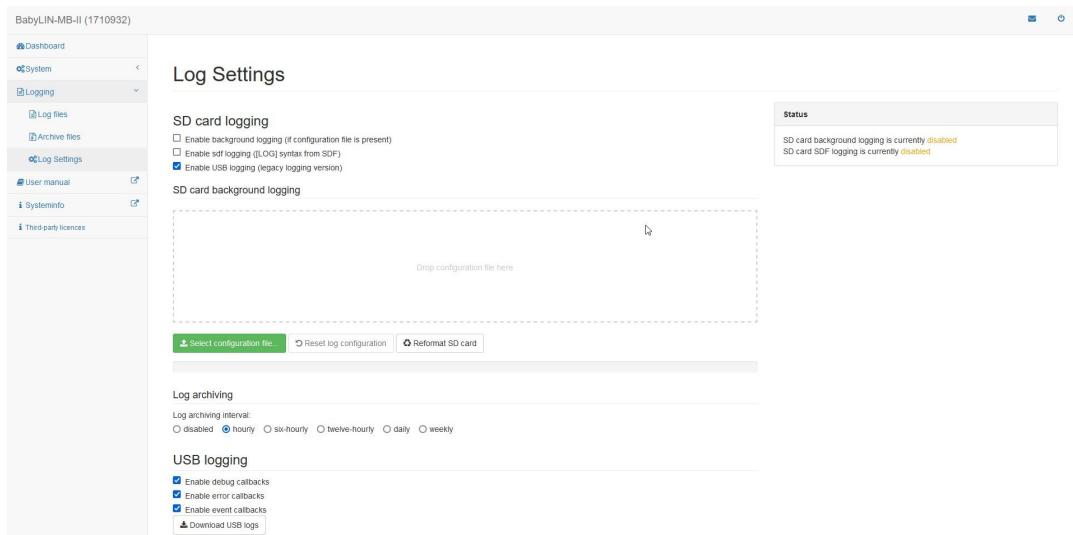
After opening the Log file View, you can now add a logger. Adding the logger opens the Logging Configuration. Existing LOG configurations can be loaded or new ones can be created.



You can customise the logging of bus communication according to your requirements. You can select which data should be tracked and how this data should be saved in the next step.

7.6.4 Log Settings via Webinterface

In the Log Settings Menu of the web interface, the extensive logging settings of the Baby-LIN-MB-II can be found. By uploading a configuration file, all settings regarding the type, format and content of the logging can be set. Furthermore, it is possible to activate/deactivate certain logging settings such as background logging or SDF logging.



With the button "Reformat SD card" all stored log files on the SD card can be deleted. If the button is greyed out, the log configuration must first be reset. Normally, no log files need to be deleted by hand. The setting options for the log archiving intervals determine how often the log files are overwritten. If no specific settings are made, older log files are automatically removed from the SD card, when the storage capacity exceeds 40% of the SD card size or when 50k log files are stored.

- 50k log files reached: the oldest 20% of the log files are deleted.
- 40% of SD card size reached: 5% of oldest log files are deleted

7.6.4.1 Debug format

This format is a proprietary human readable ASCII format. It contains debug information about the action the Baby-LIN-MB-II has executed. Its interpretation is difficult for customers. Therefore it should be sent to the Lipowsky Industrie-Elektronik GmbH. Check chapter "Support information" for more information.

7.6.5 Log data targets

7.6.5.1 Overview

The Baby-LIN-MB-II can write log data to the following targets:

Target	Description
"SimpleMenu"	The logging data can be viewed and written into a file by using the "SimpleMenu".
"USB mass storage device"	The logging data are stored as a file on a connected USB mass storage device.
"UDP push logger connection"	The logging data are sent as UDP packets to a custom UDP server.
"Debug connection"	A TCP server, that will send debug data to all connected clients.

7.6.5.2 SimpleMenu

The SimpleMenu can show the bus communication of the Baby-LIN-MB-II in the frame monitor. Additionally that communication can be written into log files in the "Binary format".

7.6.5.3 USB mass storage device

The log data are stored as a file on any connected USB mass storage device connected to the USB host interface. The folder in which the log files are stored is BL-MB-01\log\ . The filename will contain the date, time and channel the log file was started on.

The Baby-LIN-MB-II creates one file for each channel and one global file. All filenames contain a timestamp.



Advice

The following files would be created by a Baby-LIN-MB-II with six LIN channels.

```

USB:\BL-MB-01\log\channel_0_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_1_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_2_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_3_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_4_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_5_lowlvllog_01-01-2016_02-08-07.txt
USB:\BL-MB-01\log\channel_global_lowlvllog_01-01-2016_02-08-07.txt
    
```

The channel files contain the host commands and responses as well as frame information from one channel. The global file contains the host commands and responses of all channels, but no frame informations.



Warning

The USB mass storage device has to be formatted as FAT16 or FAT32.



Warning

The maximum current output of the USB host interface is 500 mA. USB mass storage devices, that require higher currents may not work.

Check chapter "X3 - USB host" for more information.

7.6.5.4 UDP push logger connection

The Baby-LIN-MB-II implements a UDP client that sends logging data to a UDP server. To configure the IP address and UDP port of the target server configure the "Configuration variables".

7.6.5.5 Debug connection

The Baby-LIN-MB-II implements a TCP server on port 2709, that will send log data to any connected client. These log data should only be used for debugging purposes. They are not convenient for logging the actions on any bus. Check chapter "Debug format" for more information.

To receive these log data a TCP connection has to be established to the TCP port 2709. The Baby-LIN-MB-Tool has a logger that can be configured to receive the debug data and log the into a file. Check chapter "Baby-LIN-MB-Tool" for more information.

7.6.6 Log data formats

7.6.6.1 Format overview

The Baby-LIN-MB-II is writing log files in the following formats:

Target	Description	Frame size
Binary format	This is the default format of the current firmware of the Baby-LIN-MB-II. This format uses a proprietary binary data format to store the data. It is optimized for speed and a low file size. Files in this format can be converted into other formats using the "LogViewer".	24 Bytes
ASCII format	This format is a human readable ASCII format, that can be processed by some third party products.	88 Bytes
Debug format	This format is a human readable ASCII format, that is mainly used for debugging purposes. It does not contain any bus information.	

7.6.6.2 Binary format

This format uses a proprietary binary data format to store the log data. It is optimized for speed and a low file size. This file can be viewed, edited and converted into other formats using the "LogViewer".

7.6.6.3 ASCII format

This format is a human readable ASCII format. It consists of a header, the logged frame data and comments. It can be processed by many third party products. The ASCII format has the following structure:

```
date Fri May 12 13:38:13 2017
base hex timestamps absolute
internal events logged
// version HARP-4 V.1.43 Build1
```

2.788508 Li	11 Tx 1 00	checksum = ff	CSM = classic
2.788508 Li	11 Tx 1 00	checksum = ff	CSM = classic
3.288498 Li	12 Rx 8 43 a1 16 d0 a7 53 29 00	checksum = 10	CSM = classic
3.538493 Li	13 Rx 0	NodeResponseMissing	
3.788488 Li	11 Tx 1 01	checksum = fe	CSM = classic
4.288531 Li	12 Rx 8 4d a6 19 d3 a9 54 2a 00	checksum = f6	CSM = classic
4.538525 Li	13 Rx 0	NodeResponseMissing	
4.788519 Li	11 Tx 1 02	checksum = fd	CSM = classic
5.288509 Li	12 Rx 8 56 ab 1d d5 ab 55 2a 00	checksum = df	CSM = classic
5.538504 Li	13 Rx 0	NodeResponseMissing	
5.788499 Li	11 Tx 1 03	checksum = fc	CSM = classic
6.288489 Li	12 Rx 8 60 b0 20 d8 ad 56 2b 00	checksum = c6	CSM = classic
6.538536 Li	13 Rx 0	NodeResponseMissing	
6.788531 Li	11 Tx 1 04	checksum = fb	CSM = classic
7.288521 Li	12 Rx 8 6a b5 23 da af 57 2b 00	checksum = af	
7.538515 Li	13 Rx 0	NodeResponseMissing	
7.788510 Li	11 Tx 1 05	checksum = fa	CSM = classic
8.288500 Li	12 Rx 8 74 ba 27 dd b1 58 2c 00	checksum = 95	CSM = classic
8.538495 Li	13 Rx 0	NodeResponseMissing	
8.788490 Li	11 Tx 1 06	checksum = f9	CSM = classic
9.288532 Li	12 Rx 8 7e bf 2a df b3 59 2c 00	checksum = 7e	CSM = classic
9.538527 Li	13 Rx 0	NodeResponseMissing	
9.788522 Li	11 Tx 1 07	checksum = f8	CSM = classic
10.288511 Li	12 Rx 8 88 c4 2d e2 b5 5a 2d 00	checksum = 65	CSM = classic
10.538506 Li	13 Rx 0	NodeResponseMissing	
10.788501 Li	11 Tx 1 08	checksum = f7	CSM = classic

The frame line components are explained using the first frame:

TarComponentget	Description
2.788508	This is a timestamp. It represents the seconds since the Baby-LIN-MB-II was powered on.
Li	A bus identifier
11	The ID of the frame
Tx	The direction of the frame
1	The length of the frame
00	All data bytes of the frame
checksum = ff	The checksum of the frame.
CSM = classic	The checksum type used by the frame

7.7 ASCII host command protocol

7.7.1 Syntax

The ASCII host command protocol is an ASCII based protocol to communicate with the Baby-LIN-MB-II. This protocol can be used via the "X10 - Ethernet" or "X7 - RS-232" connectors. This protocol is mainly used for the communication with a PLC, but other devices able to process ASCII communication can be used as well, e.g. PCs.

A command always starts with a colon : (ASCII code: 0x3A) followed by the name of the command. Depending on the command a list of parameters can follow, that are separated by a blank (ASCII code: 0x20). The command is always terminated with a carriage return (ASCII code: 0x0D).

```
:MacroExec 0 1 1 5000<CR>
```

The result also starts with a colon and is terminated with a carriage return. In between is the result of the command

: 0<CR>

Numbers can be entered as decimal or hexadecimal values:

Number format	Description	Example
Decimal	Decimal numbers can be written without formatting characters.	42
Hexadecimal	Hexadecimal numbers can be written by adding a leading H character (ASCII code: 0x48).	2AH

Signals can always be referenced by the signal index or the signal name:

Number format	Description	Example
Signal index	Signal indices can be written without formatting characters.	15
Signale name	Signal names can be written by adding a leading ! character (ASCII code: 0x21).	!Ignation

7.7.2 API modes

The Baby-LIN-MB-II supports different API modes to allow efficient processing as well as backwards compatibility.

API mode value	0	1	2
API mode name	Immediate API	CmdDone API	Compatibility API
History	The Baby-LIN-MB only supported one LINBus channel. Since no parallel processing was required, a command blocked until it finished. After the reception of a response the next command could be send by the client, e.g. a PLC. The immediate API simulates this mode, even though it is not optimal to handle multiple channels.	The Baby-LIN-MBII introduced multiple channels. The blocking commands prevented the efficient operation of multiple channels. This is now possible with the introduction of none blocking commands. Since a command may return, before it has finished, the execution state has to be polled.	Over time certain differences between the Immediate API and the command handling of the Baby-LIN-MBwere found. Since many new customers used the Baby-LIN-MB-II
Usage	Use this mode, if your application was developed for a Baby-LIN-MB-II and you do not want to handle execution states of commands and busy tokens.	Use this mode, if your application was developed for a Baby-LIN-MB-II and you want to handle multiple channels efficiently.	Use this mode, if your application was developed for a Baby-LINMB and you want to replace the device with a Baby-LIN-MB-II.
Immediate execution	✓	Only some commands return immediately.	✓
Busy tokens		✓	
Compatible with the Baby-LIN-MB	Most commands, but not all.		✓

The differences between the Immediate API and the Compatibility API are minimal, but could break a working Baby-LIN-MB application. The following commands are affected:

- "Command Version"
- "Command ReadById"

- "Command ReadByldCompare"

The Baby-LIN-MB-II starts by default in the Immediate API.

7.7.3 CmdDone API

Since the Baby-LIN-MB-II supports more than one channel, the Immediate API and Compatibility API can not be used efficiently. Commands executed in these API modes will block and not return until the command is finished. A parallel control of multiple channels can only be achieved by the CmdDone API.

Using this API mode, a command can either return a result immediately or it returns a token. This token then has to be queried to check if a command is finished and to finally receive the result. That way commands can be send to multiple channels and the execution happens parallel.

The token response can be identified by a leading "T" character in the response.

Direction	Command / Response	Description
Client → Baby-LIN-MB-II	:<Command Par1 ... ParX><CR>	A command is sent to the Baby-LIN-MB-II
Baby-LIN-MB-II → Client	:T<token><CR>	The Baby-LIN-MB-II starts the execution and returns a token.
Client → Baby-LIN-MB-II	:CmdDone <token><CR>	The client queries the execution state of the command.
Baby-LIN-MB-II → Client	:B<CR>	The Baby-LIN-MB-II indicates, that is still busy executing the command.
Client → Baby-LIN-MB-II	:CmdDone <token><CR>	The client queries the execution state of the command.
Baby-LIN-MB-II → Client	:XXX<CR>	The Baby-LIN-MB-II returns the result, since the execution is finished. XXX : result if the command succeeded. YYY : error code if the command failed.
	:@YYY<CR>	

The Baby-LIN-MB-II starts by default in the Immediate API. To efficiently communicate with multiple channels, the CmdDone API must be activated.

Then one command per channel can be executed parallel.

Direction	Command / Response	Description
Client → Baby-LIN-MB-II	:MacroExec 0 1 1 5000<CR>	A command is sent to the first channel.
Baby-LIN-MB-II → Client	:T4711<CR>	The Baby-LIN-MB-II starts the execution and returns a token.
Client → Baby-LIN-MB-II	:MacroExec 1 1 1 5000<CR>	A command is sent to the second channel.
Baby-LIN-MB-II → Client	:T4713<CR>	The Baby-LIN-MB-II starts the execution and returns a token.
Client → Baby-LIN-MB-II	:CmdDone 4711<CR>	The state of the command execution for the first command is queried.
Baby-LIN-MB-II → Client	:B<CR>	The Baby-LIN-MB-II indicates it is still busy executing the command.
Client → Baby-LIN-MB-II	:CmdDone 4712<CR>	The state of the command execution for the second command is queried.
Baby-LIN-MB-II → Client	:0<CR>	The second command finished and succeeded.
Client → Baby-LIN-MB-II	:CmdDone 4713<CR>	The state of the command execution for the third command is queried.
Baby-LIN-MB-II → Client	:@12<CR>	The third command finished and failed.
Client → Baby-LIN-MB-II	:CmdDone 4711<CR>	The state of the command execution for the first command is queried.
Baby-LIN-MB-II → Client	:0<CR>	The first command finished and succeeded.

7.7.4 TCP connections: Single and multi socket

The current firmware of the Baby-LIN-MB-II allows two different connection modes:

The single socket mode uses a single TCP connection to control all channels of the Baby-LIN-MB-II. The target channel of channel specific commands is identified by a channel parameter in the command.

The multi socket mode uses an individual TCP connection to control each channel individually. The target channel of channel specific commands is identified by the TCP port of the connection. The channel parameter of the channel specific commands must be omitted.



Warning

The switch from one connection mode to the other requires changes in the command parameters, since the multi socket mode does not use the channel parameter.

The following table sums up the properties of the connection modes:

Property	Signal socket connection	Mult socket connection
TCP ports and channel access	10002for all channels.	10003 and folloing: <ul style="list-style-type: none"> • 10003: LIN 1 • 10004: LIN 2 • 10005: LIN 3 • 10006: LIN 4 • 10007: LIN 5 • 10008: LIN 6 • 10009: CAN-HS • 10010: CAN-FD 1 • 10011: CAN-FD 2



Advice

The default single socket TCP port can be changed in the settings. The multi socket TCP ports are always the nine following ports. Check chapter "Registry variables" for more information.

Property	Signal socket connection	Mult socket connection
API mode	<ul style="list-style-type: none"> • 0: Immediate API • 1 : CmdDone API • 2 : Campatibility API If multiple channels should be controlled simultaneously, the CmdDone API must be used. Check chapter "API modes" for more information.	<ul style="list-style-type: none"> •0: Immediate API • 2 : Campatibility API Since each channel has its own connection, there is no need for the CmdDone API.
Channel parameter	All commands, that target a certain channel, have an explicit channel parameter.	The channel parameter of the channel specific commands must be omitted.

7.7.5 Quick reference

The following table gives you an overview over the available commands. Not all commands are available for all versions of the device. Check the "Supported by" columns. Some commands do not require a channel, while others do. Check the "Required channels" column:

- Empty: No channel is required. The command applies to the device as a whole.
- LIN: This command only works with a LIN channel.
- CAN: This command only works with a CAN channel.
- LIN, CAN: This command works with a LIN or CAN channel.

Command (and alias)	Description	MB	MB-II	Required channels	Link
Version	Return the firmware version of the device.	✓	✓		"Command Version"
SetApiMode	Switch between immediate and CmdDone API.		✓		"Command SetApiMode"
CmdDone	Query command token in CmdDone API.		✓		"Command CmdDone"
B	Set the baudrate of the serial connection.	✓			"Command B"
LoadSdf	Load SDFFile	✓	✓	LIN,CAN	"Command LoadSdf"
CurrentSdf	Check if any SDFFile was loaded to a channel.		✓	LIN,CAN	"Command CurrentSdf"
Start LinStart	Start the simulation.	✓	✓	LIN,CAN	"Command Start" "Command LinStart"
Stop StopStart	Start the simulation.	✓	✓	LIN,CAN	"Command Stop" "Command LinStop"
LinSchedule	Select and run a LIN schedule.	✓	✓	LIN	"Command LinSchedule"
SchedMode	Select a schedule operation mode.		✓	LIN	"Command SchedMode"
RdSignal LinRdSignal	Read a signal value.	✓	✓	LIN,CAN	"Command RdSignal" "Command LinRdSignal"
WrSignal LinWrSignal	Write a signal value.	✓	✓	LIN,CAN	"Command WrSignal" "Command LinWrSignal"
VarRead	Reads multiple signal values.		✓	LIN,CAN	"Command VarRead"
VarWrite	Writes multiple signal values.		✓	LIN,CAN	"Command VarWrite"
FormatSignals	Format signal values as strings.		✓	LIN,CAN	"Command FormatSignals"
LinMstReq	Issue a Master Request.	✓	✓	LIN	"Command LinMstReq"
LinSlvResp	Fetch a value from the Slave response.	✓	✓	LIN	"Command LinMstReq"
LinState	Read the LIN power supply state.	✓	✓	LIN	"Command LinState"
MacroExec	Execute a SDF macro	✓	✓	LIN,CAN	"Command MacroExec"
Diag22	Execute the UDS diagnostic service 22 "Read data by identifier".	✓	✓	LIN	"Command Diag22"
RTCWrite	Write the date and time of the RTC.	✓			"Command RTCWrite"

Command (and alias)	Description	MB	MB-II	Required channels	Link
RtcById	Execute the LIN node configuration service "Read by identifier".		✓	LIN	Command ById
ReadByIdCompare	Execute the LIN node configuration service "Read by identifier similar".		✓	LIN	"Command ReadByIdCompare"
WaitSignal	Wait for a specified signal value.	✓	✓	LIN,CAN	"Command WaitSignal"
SeqRun	Execute a sequence from the SDF.	✓	✓	LIN,CAN	"Command SeqRun"
SeqExec	Execute a sequence from the SDF. (Deprecated: Use SeqRun)	✓	✓	LIN	"Command SeqExec"
Delay	Wait a specified time.		✓		"Command Delay"
DigOut	Set digital output		✓		"Command DigOut"
DigIn	Set digital input		✓		"Command DigIn"
SetConfigVar	Set a config variable.		✓		"Command SetConfigVar"
GetConfigVar	Get a config variable		✓		"Command GetConfigVar"
LicenceInstall	Install a license		✓		"Command LicenceInstall"

7.8 Commando List

7.8.1 Command Version

The `Version` command can be used to query the firmware version of the Baby-LIN-MB-II.



Tip

This command should always be the first command in a communication for the following reasons:

- This command can be used to test the connection. It will always respond, as long as the connection is established, no matter what hardware or software components you have installed or want to use.
- The firmware version is one of the most important questions, we will ask, if you encounter any problems and need support.

Return value	Example
The version of the firmware is returned.	:1.5.4
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Typ	Example	Description
Command	:Version	Query the firmware version.
Response	:1.5.4	The firmware version of the Baby-LIN-MB-II.



Version incompatibility

API mode		Version composition
0	Immediate API	1.5.4
1	CmdDone API	1.5.4
2	Compatibility API	V.1.5

7.8.2 Command SetApiMode


Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The `SetApiMode` command can be used to switch between the different API modes. Check chapter "API modes" for more information.

Parameter	Description								
1	This parameter will set the API mode. This decides if command calls are blocking or not. Check chapter "API modes" for more information.								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Immediate API</td> </tr> <tr> <td>1</td> <td>CmdDone API</td> </tr> <tr> <td>2</td> <td>Compatibility API</td> </tr> </tbody> </table>	Value	Description	0	Immediate API	1	CmdDone API	2	Compatibility API
Value	Description								
0	Immediate API								
1	CmdDone API								
2	Compatibility API								

Return value	Example
: 0 is returned, if the mode was set.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	:SetApiMode 0	Set mode to the immediate API.
Response	: 0	The API was set to the immediate API.

7.8.3 Command CMDDone


Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The `CmdDone` command is required for the `CmdDone` API. If a command returns a token, it must be queried with this `CmdDone` command. The response will either contain the command's result or a busy result.

Check chapter "API modes" for more information.

Parameter	Description
1	This parameter is the token, a previous command returned. With this token the <code>CmdDone</code> command can check, if the previous command is finished. Check chapter "API modes" for more information.

Return value	Example
The busy response :B is returned, if the queried command has not yet finished. Try to send the <code>CmdDone</code> again some time later.	: B
The return value of the queried command is returned. This can also be an error.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	:SetApiMode 1	Set mode to the immediate API.
Response	:0	The API was set to the immediate API.
Command	:version	Request firmware version.
Response	:T4711	Received a token to request the command state.
Command	:CmdDone 4711	Check if command is finished.
Response	:B	The command is still executed.
Command	:CmdDone 4711	Check if command is finished.
Response	:1.2.3	The command is finished and the commands result is returned.

7.8.4 Command B



Warning

This command is only available for the Baby-LIN-MB. This command was intentionally not implemented for the Baby-LIN-MB-II. The baud rate of the Baby-LIN-MB-II can be changed using the "Web interface".

The B command changes the baud rate from the default value of 9600 Baud to the specified speed.



Advice

The response to this command is sent with the current baud rate. After the Baby-LIN-MB has sent the response, it will switch to the new baud rate.

Parameter	Description
1	This parameter is the designated baud rate. The minimum baud rate is 4800. The maximum baud rate is 115000.

Return value	Example
:0 if the new baud rate will be set.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:B 38400	Set the baud rate to 38.400 Baud.
Response	:0	The baud rate will be set for the next command to 38.400 Baud. This response is sent with the old baud rate.

7.8.5 Command CurrentSdf



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The CurrentSdf command can be used to check if a SDF file is loaded to a channel.

Parameter	Description
1	This parameter is the index of the channel, that should be checked, if a SDFFile is loaded. This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.
2	The name of a SDFFile, that exist in the internal memory of the Baby-LIN-MB-II.

Return value	Example
The name of the SDFFile is returned, if a SDFFile is loaded.	: 0
:@30 is returned, if no SDFFile is loaded.	:@30
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:CurrentSdf 0	Check if a SDFFile is loaded to the channel with index 0.
Response	:@30	No SDFFile was loaded to the channel with index 0.
Command	:SetApiMode 0	Load a SDFFile to the channel with index 0.
Response	: 0	The SDFFile was loaded successfully to the channel with index 0.
Command	:CurrentSdf 0	Check if a SDFFile is loaded to the channel with index 0.
Response	:test.sdf	The name of the SDFFile, that was loaded, is returned.




7.8.6 Command Start



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The Start command can be used to start the simulation for a specific channel. The channel index is passed in the first parameter. If the channel is a LIN channel, a schedule index can be passed using the optional second parameter.

Parameter	Description
1	<p>This parameter is the index of the channel, on which the simulation should be started.</p>  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p>
2	<p>This parameter is the index of a schedule, that should be started with the simulation.</p>  <p>Advice If no parameter is passed, the first schedule is started automatically.</p>  <p>Warning This parameter can only be used on LIN channels. For CAN channels the second parameter will be ignored.</p>

Return value	Example
The name of the SDFFile is returned, if a SDFFile is loaded.	:0
:0 is returned, if the simulation was started successfully.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:Start 1 2	The simulation for the channel with index 1 is started and the schedule with index 2 is started.
Response	:0	The simulation was started successfully.

7.8.7 Command LinStart

The LinStart command is an old alias for the "Command Start".


7.8.8 Command Stop



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The Stop command can be used to stop the simulation for a specific channel. The channel index is passed in the first parameter.

Parameter	Description
1	<p>This parameter is the index of the channel, on which the simulation should be started.</p>  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p>

Return value	Example
The name of the SDFFile is returned, if a SDFFile is loaded.	:0
:0 is returned, if the simulation was started successfully.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:Stop 1	The simulation for the channel with index 1 is stopped.
Response	:0	The simulation was stopped successfully.

7.8.9 Commando LinStop

The LinStop command is an old alias for the "Command Stop".


7.8.10 Commando LinSchedule



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The LinSchedule command can be used to start a schedule for a specific channel. The channel index is passed in the first parameter. The schedule index is passed in the second parameter.

Parameter	Description
1	<p>This parameter is the index of the channel, on which the schedule should be started.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>
2	This parameter is the index of a schedule, that should be started.

Return value	Example
The name of the SDFFile is returned, if a SDFFile is loaded.	:0
:0 is returned, if the simulation was started successfully.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:LinSchedule 1 2	The schedule with index 2 is started on the channel with index 1.
Response	:0	The schedule was started successfully.

7.8.11 Commando SchedMode



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.


The SchedMode command can be used to set the operation mode of a schedule.

After loading a SDFFile, all schedule tables are initiated with the cyclic schedule mode by default. A schedule table will keep it's mode until a new mode is explicitly set or the SDFFile is reloaded. So typically, the schedule mode for a particular table is given only once in the beginning.



Tip

The SchedMode command only sets the mode of the schedule. It will not start or queue the schedule. The schedule has to be started using "Command LinSchedule".

Parameter	Description
1	<p>This parameter is the index of the channel, on which the schedule should be started.</p> <div style="background-color: yellow; padding: 5px; border: 1px solid black;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>
2	This parameter is the index of a schedule, that should be started.
3	This parameter is the operation mode for the specified schedule. Check the table below for the different modes and allowed values.

Mode value	Mode	Cyclic	Schedule switch	Description
0	Cyclic (Default)	✓	Immediately	In this mode the schedule table will be processed cyclically. It will be repeated until another schedule is requested. If a switch to another schedule table is requested, it will be performed immediately. This mode is the default mode and will be used when no SchedMode command has been issued for a schedule table.
1	Single run		After the schedule is finished	In this mode the schedule table will be processed once and completely. If no other schedule is requested, no schedule will be processed subsequently. If a switch to another schedule is requested, it will be queued and not executed before the current Single run schedule is finished. It is possible to queue up to 32 schedules.
2	Exit on complete	✓	After the schedule is finished	In this mode the schedule table will be processed cyclically. It will always finish and be repeated until another schedule is requested. If a switch to another schedule table is requested, it will be queued and not executed before the current Exit on complete schedule is finished. It is possible to queue up to 32 schedules.

Return value	Example
: 0 is returned, if the mode was set successfully.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	:SchedMode 0 1 1	Sets the single run mode for the second schedule of the first channel 0. The schedule is not started.
Response	:0	The mode was successfully set.
Command	:SchedMode 0 2 1	Sets the single run mode for the third schedule of the first channel. The schedule is not started.
Response	:0	The mode was successfully set.
Command	:SchedMode 0 0 0	Sets the cyclic mode for the first schedule of the first channel 0. The schedule is not started.
Response	:0	The mode was successfully set.
Command	:Schedule 0 1	Start the second schedule in the single run mode. The schedule will immediately start, since no other schedule was running.
Response	:0	The mode was successfully started.
Command	:Schedule 0 2	Enqueue the third schedule in the single run mode. This schedule will not start, until the second schedule has finished.
Response	:0	The mode was successfully enqueued..
Command	:Schedule 0 0	Enqueue the third schedule in the cyclic mode. This schedule will not start, until the third schedule has finished. This schedule will run until another schedule is started.
Response	:0	The mode was successfully enqueued..


Attention



The queue for the schedule tables can contain a maximum of 32 entries. The user has to make sure, not to overfill the queue.


Tip

The system variable "@@SYSBUSSTATE" can be used to check, if a schedule is currently running.

7.8.12 Command RdSignal

The `RdSignal` command can be used to read the value of signals for a specific channel. The channel index is passed in the first and the signals in the following parameter.

Parameter	Description									
1	<p>This parameter is the index of the channel, on which the signal value should be read.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>									
2...17	<p>This parameter identifies the signal, whose value should be read.</p> <div style="background-color: #008080; color: white; padding: 5px;"> <p>Tip</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Signal property</th> <th>Description</th> <th>Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td><code>:LinRdSignal 0 12</code></td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td><code>:RdSignal 0 !SignalName</code></td> </tr> </tbody> </table> </div> <div style="background-color: yellow; padding: 5px; margin-top: 10px;">  <p>Warning</p> <p>The maximum number of signals, that can be passed, is 16.</p> </div>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>
Signal property	Description	Example								
Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>								
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>								

This command will try to read the signal values directly from the bus. If they do not appear on the bus, an error is returned. The timeout depends on the number of signals, that should be read.

For one signal, the timeout is 300 ms. For each additional signal, 200 ms are added. So if four signals should be read, the maximum time that is waited is 900 ms. If not all signals appeared on the bus within this time, an error is returned.

Return value	Example
The signal values are returned as list of values, if all could be read. If any signal could not be read, an error will occur.	<code>:2 -314 17</code>
A default error may be returned. Check chapter "Return codes" for more information.	<code>:@1</code>

Type	Example	Description
Command	<code>:RdSignal 1 2 !KeepAlive</code>	The signals with index 2 and the name <code>KeepAlive</code> are read from the channel with index 1.
Response	<code>:15 20</code>	All signals could be read.

7.8.13 Command `LinRdSignal`

The `LinRdSignal` command is an old alias for the "Command `RdSignal`".

7.8.14 Command `WrSignal`

The `WrSignal` command can be used to write the value of a signal for a specific channel.


Attention

Only those signals can be written, that are published by a node, that is simulated by the Baby-LINMB- II. For CAN signals the frame has to be simulated as well.

Parameter	Description									
1	<p>This parameter is the index of the channel, on which the signal value should be written.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>									
2	<p>This parameter identifies the signal, whose value should be written.</p> <div style="background-color: #008080; color: white; padding: 5px;"> <p>Tip</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: #cccccc;">Signal property</th> <th style="background-color: #cccccc;">Description</th> <th style="background-color: #cccccc;">Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td><code>:LinRdSignal 0 12</code></td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td><code>:RdSignal 0 !SignalName</code></td> </tr> </tbody> </table> </div>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>
Signal property	Description	Example								
Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>								
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>								
3	The value that should be written to the the signal.									

Return value	Example
:0 is returned, if the simulation was started successfully.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	<code>:WrSignal 1 !KeepAlive 12</code>	The value 12 should be written to the signal with the name <code>KeepAlive</code> on the channel with index 1.
Response	:0	The new value was written to the signal successfully..

7.8.15 Command LinWrSignal


The `LinWrSignal` command is an old alias for the "Command `WrSignal`".

7.8.16 Command VarRead


Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The `VarRead` command is used to read multiple signal values at once. One signal will be used as starting signal. Starting with this signal, the values of each following signal will be returned. The format of the returned values can be configured.

Parameter	Description										
1	<p>This parameter is the index of the channel, whose signals should be written.</p> <div style="background-color: yellow; padding: 5px; border: 1px solid black;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>										
2	<p>This parameter identifies the mode. The mode defines how the values will be returned.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Mode</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The values will be returned as single integers encoded as decimal values separated by space characters.</td> </tr> <tr> <td>1</td> <td>The values will be returned as single integers encoded as hexadecimal values separated by space characters.</td> </tr> <tr> <td>2</td> <td>The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned by escaping them. The ' character and the hexadecimal value will be used. A carriage return character would be represented as "\0D". If the length parameter is 0, values will be read until the first signal value is 0.</td> </tr> <tr> <td>3</td> <td>The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned as '.' characters. If the length parameter is 0, values will be read until the first signal value is 0.</td> </tr> </tbody> </table>	Mode	Description	0	The values will be returned as single integers encoded as decimal values separated by space characters.	1	The values will be returned as single integers encoded as hexadecimal values separated by space characters.	2	The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned by escaping them. The ' character and the hexadecimal value will be used. A carriage return character would be represented as "\0D". If the length parameter is 0, values will be read until the first signal value is 0.	3	The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned as '.' characters. If the length parameter is 0, values will be read until the first signal value is 0.
Mode	Description										
0	The values will be returned as single integers encoded as decimal values separated by space characters.										
1	The values will be returned as single integers encoded as hexadecimal values separated by space characters.										
2	The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned by escaping them. The ' character and the hexadecimal value will be used. A carriage return character would be represented as "\0D". If the length parameter is 0, values will be read until the first signal value is 0.										
3	The values will be returned as characters of a single string. Each signal will be evaluated as byte and encoded as ASCII character. Certain characters will be returned as '.' characters. If the length parameter is 0, values will be read until the first signal value is 0.										
3	<p>This parameter identifies the starting signal of the array.</p> <div style="background-color: #008080; color: white; padding: 5px; border: 1px solid black;"> <p>Tip</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%; border-collapse: collapse; background-color: white; color: black;"> <thead> <tr> <th style="width: 20%;">Signal property</th> <th style="width: 50%;">Description</th> <th style="width: 30%;">Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td>:LinRdSignal 0 12</td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td>:RdSignal 0 !SignalName</td> </tr> </tbody> </table> </div>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	:LinRdSignal 0 12	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	:RdSignal 0 !SignalName	
Signal property	Description	Example									
Signal index	The signal index can be used simply by passing the index number.	:LinRdSignal 0 12									
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	:RdSignal 0 !SignalName									
4	<p>This length parameter identifies the number of signals, that will be read.</p>										

In modes 2 and 3 certain characters will be escaped or replaced.

These are all characters smaller than 32 (0x20, Space) and larger than 126 (0x7E, ~).

Return value	Example
A list of decimal encoded integers may be returned, if the mode is 0.	:1 2 3
A list of hexadecimal encoded integers may be returned, if the mode is 1.	:AB CD EF
A string containing escaped characters may be returned, if the mode is 2.	:new\0Dline
A string containing a lot of '.' characters may be returned, if the mode is 3.	:new.line
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	<code>:VarRead 0 0 !SIG1 4</code>	The values of 4 signals starting with SIG1 will be returned as list of decimal integers.
Response	<code>:1 2 3 4</code>	The 4 values are: 1, 2, 3, 4
Command	<code>:VarRead 0 1 !SIG1 5</code>	The values of 5 signals starting with SIG1 will be returned as list of decimal integers.
Response	<code>:67 89 AB CD EF</code>	The 5 values are: 103, 137, 171, 205, 239
Command	<code>:VarRead 0 2 !SIG1 0</code>	The values of the signals starting with SIG1 will be returned as string with escaped characters for escaped characters. The end of the string is defined by the first signal with value 0.
Response	<code>:new\0Dline</code>	8 characters are returned, because the ninth signal value was a 0. The fourth character is an escaped carriage return character.
Command	<code>:VarRead 0 3 !SIG1 8</code>	The values of 8 signals starting with SIG1 will be returned as string with '.' for escaped characters.
Response	<code>:new.line</code>	8 characters are returned. The fourth character was a carriage return character and therefore is returned as '.'.


7.8.17 Command VarWrite



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The `VarWrite` command is used to write multiple signal values at once. One signal will be used as starting signal. Starting with this signal, the values of each following signal will be set. The format of the passed values can be configured.

Parameter	Description															
1	<p>This parameter is the index of the channel, whose signals should be written.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>															
2	<p>This parameter identifies the mode. The mode defines how the values can be passed.</p> <table border="1"> <thead> <tr> <th>Mode</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The value for each signal is passed as an individual value. If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.</td> </tr> <tr> <td>1</td> <td>A single string is passed. Each character is assigned to a single signal. The string uses the ASCII encoding. Non printable characters can be passed by escaping them. Just use the 'character and add the hexadecimal value of the character. A carriage return character would be represented as "\0D". If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.</td> </tr> <tr> <td>2</td> <td>A single value can be passed. That value is assigned to all signals. The number of target signals is defined by the length parameter. If the value is larger than the signal, then the value is cut.</td> </tr> </tbody> </table>	Mode	Description	0	The value for each signal is passed as an individual value. If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.	1	A single string is passed. Each character is assigned to a single signal. The string uses the ASCII encoding. Non printable characters can be passed by escaping them. Just use the 'character and add the hexadecimal value of the character. A carriage return character would be represented as "\0D". If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.	2	A single value can be passed. That value is assigned to all signals. The number of target signals is defined by the length parameter. If the value is larger than the signal, then the value is cut.							
Mode	Description															
0	The value for each signal is passed as an individual value. If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.															
1	A single string is passed. Each character is assigned to a single signal. The string uses the ASCII encoding. Non printable characters can be passed by escaping them. Just use the 'character and add the hexadecimal value of the character. A carriage return character would be represented as "\0D". If the value is larger than the signal, then the value is cut. The maximum number of values, that can be set at once, is 400.															
2	A single value can be passed. That value is assigned to all signals. The number of target signals is defined by the length parameter. If the value is larger than the signal, then the value is cut.															
3	<p>This parameter identifies the starting signal of the array.</p> <div style="background-color: #008080; color: white; padding: 5px;"> <p>Tip</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: #cccccc;">Signal property</th> <th style="background-color: #cccccc;">Description</th> <th style="background-color: #cccccc;">Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td>:LinRdSignal 0 12</td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td>:RdSignal 0 !SignalName</td> </tr> </tbody> </table> </div>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	:LinRdSignal 0 12	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	:RdSignal 0 !SignalName						
Signal property	Description	Example														
Signal index	The signal index can be used simply by passing the index number.	:LinRdSignal 0 12														
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	:RdSignal 0 !SignalName														
4...	<p>The following parameters depend on the mode.</p> <table border="1"> <thead> <tr> <th>Mode</th> <th>Parameter</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>4...n</td> <td>The values, that should be individually assigned to the target signals.</td> </tr> <tr> <td>1</td> <td>4</td> <td>The string, whose characters should be individually assigned to the target signals.</td> </tr> <tr> <td>2</td> <td>4</td> <td>The length of the array. The target value will be assigned to this number of signals.</td> </tr> <tr> <td>2</td> <td>5</td> <td>The target value, that will be assigned to each signal.</td> </tr> </tbody> </table>	Mode	Parameter	Description	0	4...n	The values, that should be individually assigned to the target signals.	1	4	The string, whose characters should be individually assigned to the target signals.	2	4	The length of the array. The target value will be assigned to this number of signals.	2	5	The target value, that will be assigned to each signal.
Mode	Parameter	Description														
0	4...n	The values, that should be individually assigned to the target signals.														
1	4	The string, whose characters should be individually assigned to the target signals.														
2	4	The length of the array. The target value will be assigned to this number of signals.														
2	5	The target value, that will be assigned to each signal.														

Return value	Example
: 0 is returned, if the values were written.	:@0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	<code>:VarWrite 0 0 !SIG1 1 127 FFh</code>	The values 1, 127 and 255 will be assigned to the 3 signals starting with SIG1.
Response	<code>:0</code>	The new values were written successfully.
Command	<code>:VarWrite 0 1 !SIG1 "abcde"</code>	The values 97, 98, 99, 100 and 101 are assigned to the 5 signals starting with SIG1 .
Response	<code>:0</code>	The new values were written successfully.
Command	<code>:VarWrite 0 2 !SIG1 7 FFh</code>	The value 255 will be assigned to all 7 signals starting with SIG1.
Response	<code>:0</code>	The new values were written successfully.

7.8.18 Command FormatSignals




Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The `FormatSignals` command is used to format one or multiple signals as text. One signal will be used as starting signal. Starting with this signal, the bytes of each signal will be evaluated as one continuous byte stream.

If the length of a signal is not a multiple of 8 bits, the missing bits will be filled with 0s. Therefore each signal will be interpreted as a 1-8 Byte wide value.

Parameter	Description									
1	<p>This parameter is the index of the channel, whose signals should be formatted.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>									
2	<p>This parameter identifies the starting signal, from whose value a string should be created.</p> <div style="background-color: #008080; color: white; padding: 5px;"> <p>Tip</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: #cccccc;">Signal property</th> <th style="background-color: #cccccc;">Description</th> <th style="background-color: #cccccc;">Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td><code>:LinRdSignal 0 12</code></td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td><code>:RdSignal 0 !SignalName</code></td> </tr> </tbody> </table> </div>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>
Signal property	Description	Example								
Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>								
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>								
3	The number of signals, that should be evaluated.									

Parameter	Description																		
4	This optional parameter selects the mode, what kind of string should be created.																		
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>This is the default case, if this optional parameter is not set. In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. Check mode 7 for an inverted byte order.</td> </tr> <tr> <td>1</td> <td>In this mode each signal byte is formatted as hexadecimal value. The bytes will be concatenated with an additional space character between each byte.</td> </tr> <tr> <td>2</td> <td>In this mode each signal byte is formatted as decimal value. The bytes will be concatenated with an additional space character between each byte.</td> </tr> <tr> <td>3</td> <td>In this mode each signal byte is formatted as BCD coded value. The nibble order is high nibble first (0x1234 => "1234").</td> </tr> <tr> <td>4</td> <td>In this mode each signal byte is formatted as BCD coded value. The nibble order is low nibble first (0x1234 => "4321").</td> </tr> <tr> <td>5</td> <td>In this mode all bytes of the signal are formatted as single decimal value. Signed signals will be formatted with sign.</td> </tr> <tr> <td>6</td> <td>In this mode all bytes of the signal are formatted as single hexadecimal value. Signed signals will be formatted without sign.</td> </tr> <tr> <td>7</td> <td>In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. This mode is like mode 0 with inverted byte order.</td> </tr> </tbody> </table>	Value	Description	0	This is the default case, if this optional parameter is not set. In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. Check mode 7 for an inverted byte order.	1	In this mode each signal byte is formatted as hexadecimal value. The bytes will be concatenated with an additional space character between each byte.	2	In this mode each signal byte is formatted as decimal value. The bytes will be concatenated with an additional space character between each byte.	3	In this mode each signal byte is formatted as BCD coded value. The nibble order is high nibble first (0x1234 => "1234").	4	In this mode each signal byte is formatted as BCD coded value. The nibble order is low nibble first (0x1234 => "4321").	5	In this mode all bytes of the signal are formatted as single decimal value. Signed signals will be formatted with sign.	6	In this mode all bytes of the signal are formatted as single hexadecimal value. Signed signals will be formatted without sign.	7	In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. This mode is like mode 0 with inverted byte order.
	Value	Description																	
	0	This is the default case, if this optional parameter is not set. In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. Check mode 7 for an inverted byte order.																	
	1	In this mode each signal byte is formatted as hexadecimal value. The bytes will be concatenated with an additional space character between each byte.																	
	2	In this mode each signal byte is formatted as decimal value. The bytes will be concatenated with an additional space character between each byte.																	
	3	In this mode each signal byte is formatted as BCD coded value. The nibble order is high nibble first (0x1234 => "1234").																	
	4	In this mode each signal byte is formatted as BCD coded value. The nibble order is low nibble first (0x1234 => "4321").																	
	5	In this mode all bytes of the signal are formatted as single decimal value. Signed signals will be formatted with sign.																	
6	In this mode all bytes of the signal are formatted as single hexadecimal value. Signed signals will be formatted without sign.																		
7	In this mode each signal byte is interpreted as ASCII coded character. The characters will be concatenated without additional space characters. This mode is like mode 0 with inverted byte order.																		
The bytes will be concatenated with an additional space character between each byte. If the value of a nibble is higher than 9, the value is not a valid BCD value and the command will therefore return an error.																			
5	This optional parameter defines, if the processing will stop as soon as a byte is 0. This is useful for ASCII coded strings with different lengths. If the fourth parameter is 1 or 2 this behaviour might be wanted.																		
	<table border="1"> <thead> <tr> <th>value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The processing will stop as soon as a byte is 0.</td> </tr> <tr> <td>1</td> <td>This is the default case, if this optional parameter is not set. The processing will not stop as soon as a byte is 0.</td> </tr> </tbody> </table>	value	Description	0	The processing will stop as soon as a byte is 0.	1	This is the default case, if this optional parameter is not set. The processing will not stop as soon as a byte is 0.												
	value	Description																	
	0	The processing will stop as soon as a byte is 0.																	
1	This is the default case, if this optional parameter is not set. The processing will not stop as soon as a byte is 0.																		

Return value	Example
The formatted string if no error occurred.	:Test
:@14 is returned, if a nibble is not a valid BCD encoded nibble.	:@14
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:FormatSignals 0 !SIG1 2 0	The 2 signals starting with SIG1 are formatted as ASCII coded string.
Response	:HalloWelt	The ASCII coded string is returned.
Command	:FormatSignals 0 !SIG1 2 1	The 2 signals starting with SIG1 are formatted as hexadecimal coded bytes.
Response	:48 61 6c 6c 6f 57 65 6c 74	The hexadecimal coded bytes are returned.
Command	:FormatSignals 0 !SIG1 2 2	The 2 signals starting with SIG1 are formatted as decimal coded bytes.
Response	:72 97 108 108 111 87 101 108 116	The decimal coded bytes are returned.


7.8.19 Command LinMstReq



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The `LinMstReq` command can be used to issue a Master Request on the specified channel.

Parameter	Description
1	<p>This parameter is the index of the channel, on which the Master Request should be issued.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>
2...9	These parameters are the 8 data bytes of the Master Request.
10	This is the timeout in ms, that is waited to receive all Slave Responses.
11	This is the number of Slave Responses, that are expected after the Master Request. This parameter is optional. If it is omitted, one Slave Response will be expected. The maximum number of expected Slave Responses is 32.

Return value	Example
The formatted string if no error occurred.	:Test
:0 is returned, if the Master Request was issued. At this point the Master Request is only scheduled and no Slave Responses have been received.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description															
Command	:LinMstReq 0 43H 6H B2H 1H 2H 0H 0H 27H 1000	<p>A Master Request is issued with the following properties:</p> <table border="1"> <thead> <tr> <th>Property</th> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>channel index</td> <td>0</td> <td>The index of the channel</td> </tr> <tr> <td>Data bytes</td> <td>43H 06H B2H 01H 02H 00H 00H 27H</td> <td>The data bytes</td> </tr> <tr> <td>Timeout</td> <td>1000</td> <td>The time within which the Slave Responses are expected.</td> </tr> <tr> <td>Expected Slave Responses</td> <td>1</td> <td>The number of expected Slave Responses. One is the default value, if none is passed.</td> </tr> </tbody> </table>	Property	Value	Description	channel index	0	The index of the channel	Data bytes	43H 06H B2H 01H 02H 00H 00H 27H	The data bytes	Timeout	1000	The time within which the Slave Responses are expected.	Expected Slave Responses	1	The number of expected Slave Responses. One is the default value, if none is passed.
Property	Value	Description															
channel index	0	The index of the channel															
Data bytes	43H 06H B2H 01H 02H 00H 00H 27H	The data bytes															
Timeout	1000	The time within which the Slave Responses are expected.															
Expected Slave Responses	1	The number of expected Slave Responses. One is the default value, if none is passed.															
Response	: 0	The Master Request was issued.															

7.8.20 Command LinSlvResp




Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The `LinSlvResp` command can be used to fetch a value from the Slave Responses from a previous `LinMstReq`. The parameters define the position within the data bytes of all received Slave Responses.

If a Master Request received more than one Slave Response, their data bytes are merged together to one big array. The position and length of the value, you are interested in, related to this array.

Parameter	Description								
1	<p>This parameter is the index of the channel, on which the Master Request was issued, this Slave Response was following.</p>  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p>								
2	This is the starting bit of the value within the Slave Response's data bytes, you are interested in.								
3	This is the length in bits of the value within the Slave Response's data bytes, you are interested in.								
4	<p>This optional parameter lets you choose the format of the response.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>This is the default case, if this optional parameter is not set. The frame data is formatted as 64 bit integer. The maximum number of one frame can be returned.</td> </tr> <tr> <td>1</td> <td>The frame data are returned as hexadecimal coded bytes. The bytes are separated with space characters.</td> </tr> <tr> <td>2</td> <td>The frame data are returned as string. The encoding of the string depends on the sender.</td> </tr> </tbody> </table>	Value	Description	0	This is the default case, if this optional parameter is not set. The frame data is formatted as 64 bit integer. The maximum number of one frame can be returned.	1	The frame data are returned as hexadecimal coded bytes. The bytes are separated with space characters.	2	The frame data are returned as string. The encoding of the string depends on the sender.
Value	Description								
0	This is the default case, if this optional parameter is not set. The frame data is formatted as 64 bit integer. The maximum number of one frame can be returned.								
1	The frame data are returned as hexadecimal coded bytes. The bytes are separated with space characters.								
2	The frame data are returned as string. The encoding of the string depends on the sender.								



Version incompatibility

There are incompatibilities between the Baby-LIN-MB-I and the Baby-LIN-MB-II:

- Baby-LIN-MB-I:

The command always returns :B, if not all data are available and the timeout is not reached yet.

- Baby-LIN-MB-II, API mode 0:

For compatibility reasons the behavior is the same like with the Baby-LIN-MB-I:

The command always returns :B, if not all data are available and the timeout is not reached yet.

- Baby-LIN-MB-II, API mod 1:

Since a command returns :B if it is busy, the LinSlvResp command returns :I (incomplete).

Return value	Example
The value is returned, if the Master Request was executed, all Slave Responses were received and the value could be extracted from the data bytes.	:0
Only in API mode 0: :B is returned, if the Master Request is not finished or not all Slave Responses have been received	:B
Only in API mode 1 : linebreak : I is returned, if the Master Request is not finished or not all Slave Responses have been received.	:I
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:SetApiMode 0	Show example in API mode 0.
Response	:0	The immediate API mode is activated.
Command	:LinMstReq 0 43H 6H B2H 1H 2H 0H 0H 27H 1000 2	The Master Request is scheduled and two Slave Responses are expected.
Response	:0	The immediate API mode is activated.
Command	:LinSlvResp 1 56 24	A value from the slave response is requested.
Response	:B	The Master Request was not yet sent and/or the Slave Responses have not been received completely.
Command	:LinSlvResp 1 56 24	A value from the slave response is requested again.
Response	:16777215	The Master Request was sent and the Slave Responses have been received completely. A 24 bit value was extracted from the data bytes.
Command	:SetApiMode 1	Show example in API mode 1.
Response	:0	The CmdDone API mode is activated.

Type	Example	Description
Command	:LinMstReq 0 43H 6H B2H 1H 2H 0H 0H 27H 1000 2	The Master Request is scheduled and two Slave Responses are expected.
Response	:0	The Master Request was issued.
Command	:LinSlvResp 1 56 24	A value from the slave response is requested.
Response	:I	The Master Request was not yet sent and/or the Slave Responses have not been received completely.
Command	:LinSlvResp 1 56 24	A value from the slave response is requested again.
Response	:16777215	The Master Request was sent and the Slave Responses have been received completely. A 24 bit value was extracted from the data bytes.

7.8.21 Command LinState



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The `LinState` command can be used to query the state of a LIN interface. The channel index is passed in the first parameter.



Advice

If the command is send to a CAN channel, the result will always be :1, which means bus power is supplied.

Parameter	Description
1	<p>This parameter is the index of the channel, on which the schedule should be started.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>

Return value	Example
: 0 is returned, if no LIN power is supplied.	:@0
: 1 is returned, if the LIN power is supplied.	: 1
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:LinState 1	Query the LIN state of the channel with index 1.
Response	:0	The channel is not supplied with bus power.
Command	:LinState 0	Query the LIN state of the channel with index 0.
Response	:1	The channel is supplied with bus power.

7.8.22 Command RdFrameError



Attention

The command is deprecated and is no longer supported as of firmware version V1.14.17. With the SDFV3 function to send a frame via blocking inject can be better checked whether a frame could be sent successfully.

The `RdFrameError` command can be used to query the error code that occurred the last time the specified frame was sent. The channel index is passed in the first and the frame ID in the second parameter.





Parameter	Description
1	<p>This parameter is the index of the channel, on which the error code should be queried.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>
2	This parameter identifies the frame, whose error code should be read.

Return value	Example
No error has occurred after the frame was sent.	:0
2: Read data from BUS does not match send data	:2
3: Framing error in data reception	:3
4: Checksum failed	:4
5: Data timed out (incomplete message reception)	:5
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:RdFrameError 1 2	The error code, that occurred after the frame with ID 2 was last sent on the channel with index 1.
Response	:0	No error occurred.

7.8.23 Command MacroExec

The `MacroExec` command is used to start the execution of a macro on the specified channel.

Parameter	Description						
1	<p>This parameter is the index of the channel, that the macro should be executed on.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>						
2	The index of the macro, that should be executed.						
3	<p>This optional parameter configures, if the call just starts the macro and then instantly returns or if this call is blocking and will not return until the macro was finished.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The macro is just started and the command returns immediately.</td> </tr> <tr> <td>1</td> <td>The command does not return, until the macro finished its execution.</td> </tr> </tbody> </table> <div style="background-color: #008080; color: white; padding: 5px; margin-top: 10px;">  <p>Tip</p> <p>If you want to evaluate the result of a macro, the call must be blocking, until the macro has finished.</p> </div> <div style="background-color: yellow; padding: 5px; margin-top: 10px;">  <p>Warning</p> <p>If no value is passed for this parameter, the default value of 0 is used. Therefore the command will return immediately.</p> </div>	Value	Description	0	The macro is just started and the command returns immediately.	1	The command does not return, until the macro finished its execution.
Value	Description						
0	The macro is just started and the command returns immediately.						
1	The command does not return, until the macro finished its execution.						
4	<p>This optional parameter defines the maximum timeout in Milliseconds, that the command will be blocked. After this time the command will return with a <code>:@5</code>.</p> <div style="background-color: yellow; padding: 5px; margin-top: 10px;">  <p>Warning</p> <p>If no value is passed for this parameter, the default value of 1000 is used. Therefore the timeout is 1 second.</p> </div>						
5...14	These optional parameters will be passed as parameters to the called macro.						

Return value	Example
: 0 if the command was not blocking and the macro was started successfully or the command was blocking and the macro returned 0.	: 0
If the command was blocking, any value can be returned as result of the macro.	: 314
If the command was blocking, failures of the macro are returned with an offset of 50000. So any error between 50000 and 115535 are error results from the macro.	: @50698
If the command was blocking and the timeout was exceeded, a timeout error is returned.	: @5
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	:MacroExec 0 1	The macro with index 1 on the first channel will be started. The command will return immediately.
Response	: 0	The macro was started successfully.
Command	:MacroExec 1 3 1	The macro with index 3 on the second channel will be executed. The command will be blocked until the macro is finished
Response	: 7	The macro was executed and finished successfully with a return value of 7.
Command	:MacroExec 2 5 1 100	The macro with index 5 on the third channel will be executed. The command will be blocked until the macro is finished.
Response	: 5	The macro was started, but did not finish within the given timeout time.
Command	:MacroExec 3 7 1 100 15 16383	The macro with index 7 on the fourth channel will be executed. The command will be blocked until the macro is finished. The first parameter of the macro will be set to 15. The second parameter of the macro will be set to 16383.
Response		

7.8.24 Command Diag22



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.

The `Diag22` command can be used to execute the UDS diagnostic service 0x22 Read Data by Identifier on a specific node. Although the UDS diagnostic service 0x22 features the query of multiple values with one call, this command allows the query of just one value at each call.




Warning

To execute this UDS diagnostic service a diagnostic schedule with MRQ (Frame ID: 0x3C) and SRP (Frame ID: 0x3D) frames must exist in the underlying LDF.


Warning

The diagnostic schedule will keep running even after the command has finished. If another schedule is required, it has to be started manually using the "Command LinSchedule".

Parameter	Description
1	<p>This parameter is the index of the channel, on which the UDS diagnostic service should be executed.</p>  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p>
2	This parameter is the index of a schedule, that contains the MRQ and SRP frames.
3	This parameter is the NAD of the node.
4	This parameter is the 16 Bit service ID.

Return value	Example
: 0 if the command was not blocking and the macro was started successfully or the command was blocking and the macro returned 0.	: 0
The response will contain the ASCII data extracted from the multiple response frames coming from the slave.	: 3C8959537
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	:DIAG22 0 1 10 065EH	The request with the service ID 0x065E (in this example the software part number) is sent to the node with the NAD 10 on channel 0. Therefor the schedule with index 1 will be started.
Response	: 3C8959537	The answer as ASCII data (in this example the software part number).

7.8.25 Command RTCWrite


Warning

This command is only available for compatibility reasons for the Baby-LIN-MB-II. The function will always return :0 but not trigger any actions.

The `RTCWrite` command will not trigger any actions. It is only available due to compatibility reasons. The real-time clock is set using the "Web interface".

Parameter	Description
1	This parameter is the day of the month of the date. Valid values are: 1 . . . 31.
2	This parameter is the month of the date. Valid values are: 1 . . . 12.
3	This parameter is the year of the date. Valid values are: 2013 . . . 2200.
4	This parameter is the hour of the time. Valid values are: 0 . . . 23.
5	This parameter is the minute of the time. Valid values are: 0 . . . 59.
6	This parameter is the second of the time. Valid values are: 0 . . . 59.

Return value	Example
:0 is always returned. Even though the real-time clock was not set.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:RTCWrite 24 12 2017 17 30 00	The real-time clock will not be changed. This command will do nothing and return a success.
Response	:0	The real-time clock was not set. A success was only returned for compatibility reasons.

7.8.26 Command RTCRead

The RTCRead command can be used to read the date and time of the real-time clock.

Return value	Example
The date and time of the real-time clock.	:24 12 2017 17 30 00
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:RTCRead	Read the date and time of the real-time clock.
Response	:24 12 2017 17 30 00	<p>The successful command returns the date and time with the following list:</p> <ul style="list-style-type: none"> • The day of the month of the date. • The month of the date. • The year of the date. • The hour of the time. • The minute of the time. • The second of the time.

7.8.27 Command ReadById



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.


Tip

This command is deprecated. The protocols in the SDF file are more powerful and more flexible. Check chapter "Custom Protocols" for more information. The output of the response data can be formatted with the "Command FormatSignals".

The `ReadById` command can be used to read node configuration and identification using the "Read by identifier" service from the LIN standard diagnostic.

The 8 data bytes of the Master Request always have the following structure:


Data bytes	D1	D2	D3	D4	D5	D6	D7	D8
Description	NAD	PCI	SID	Identifier	Supplier ID LSB	Supplier ID MSB	Funktion ID LSB	Functio ID MSB
Constant values		0x06	0xB2					
Values set via System Variables	@@Diag NodeId				@@Diag SupplierId		@@Diag FunctionId	
Values set via parameters				Second Parameter				


Additionally a schedule will be executed, that is defined by the system variable `@@ScheduleDiag`. This schedule should contain the MRQ and SRP diagnostic frames.

If the diagnostic schedule, NAD, supplier ID or function ID system variables are not set within the SDF, they will receive default values:

System variable	Description	Default value if not explicitly set
<code>@@ScheduleDiag</code>	Diagnostic schedule	0
<code>@@DiagNodeId</code>	NAD	0x7F
<code>@@DiagSupplierId</code>	Supplier ID	0x7FFF
<code>@@DiagFunctionId</code>	Function ID	0xFFFF

The default values of `@@DiagNodeId`, `@@DiagSupplierId` and `@@DiagFunctionId` are wildcards and address every node.

Parameter	Description																		
1	<p>This parameter is the index of the channel, that the SDFile should be loaded to.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>																		
2	<p>This parameter is the identifier used in the request. Certain identifiers are already defined in the LIN specification:</p> <table border="1"> <thead> <tr> <th>Identifier</th> <th>Description</th> <th>Length of response</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>LIN product identification</td> <td>RSID + 5</td> </tr> <tr> <td>1</td> <td>Serial number</td> <td>RSID + 4</td> </tr> <tr> <td>2...31</td> <td>Reserved</td> <td></td> </tr> <tr> <td>32...63</td> <td>user defined</td> <td>User defined</td> </tr> <tr> <td>64...255</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Identifier	Description	Length of response	0	LIN product identification	RSID + 5	1	Serial number	RSID + 4	2...31	Reserved		32...63	user defined	User defined	64...255	Reserved	
Identifier	Description	Length of response																	
0	LIN product identification	RSID + 5																	
1	Serial number	RSID + 4																	
2...31	Reserved																		
32...63	user defined	User defined																	
64...255	Reserved																		
3	<p>This parameter defines how the response data has to be interpreted. The following values are possible:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Read response as bytes as binary array.</td> </tr> </tbody> </table>	Value	Description	0	Read response as bytes as binary array.														
Value	Description																		
0	Read response as bytes as binary array.																		
4	<p>This parameter defines, how the response data should be formatted and returned by the Baby-LIN-MB-II. The following values are possible:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The response will consist of the length of the data and the data bytes itself coded as hexadecimal values.</td> </tr> <tr> <td>1</td> <td>The response will consist of the length of the data and the data bytes itself coded as decimal values.</td> </tr> <tr> <td>2</td> <td>The response will consist of a single string. The data bytes are interpreted as ASCII characters.</td> </tr> <tr> <td>3</td> <td>The response will consist of a single decimal number. The data bytes will be interpreted as BCD coded bytes. The LSB is interpreted first.</td> </tr> </tbody> </table>	Value	Description	0	The response will consist of the length of the data and the data bytes itself coded as hexadecimal values.	1	The response will consist of the length of the data and the data bytes itself coded as decimal values.	2	The response will consist of a single string. The data bytes are interpreted as ASCII characters.	3	The response will consist of a single decimal number. The data bytes will be interpreted as BCD coded bytes. The LSB is interpreted first.								
Value	Description																		
0	The response will consist of the length of the data and the data bytes itself coded as hexadecimal values.																		
1	The response will consist of the length of the data and the data bytes itself coded as decimal values.																		
2	The response will consist of a single string. The data bytes are interpreted as ASCII characters.																		
3	The response will consist of a single decimal number. The data bytes will be interpreted as BCD coded bytes. The LSB is interpreted first.																		

Return value	Example
The interpreted data bytes if request was successfully.	:2 FF FF
A default error may be returned. Check chapter "Return codes" for more information.	:@1
If the slave responded with a NRC, the error also includes the first two bytes after the NRC.	:@31 FF FF
<div style="background-color: orange; padding: 5px;">  <p>Version incompatibility This result type is only returned when the Compatibility API is active. If not, only the error code is returned. Check chapter "API modes" for more information.</p> </div>	

Type	Example	Description
Command	:ReadById 0 1 0 0	The serial number is requested. The output format should be hexadecimal bytes.
Response	:5 F2 87 08 01 03	5 data bytes are returned. The request was successful, since the RSID is 0xF2. The 4 data bytes form the serial number.

7.8.28 Command ReadByIdCompare



Warning

This command is only available on LIN channels. If it is executed on a CAN channel, an error will be returned.



Tip

This command is deprecated. The protocols in the SDFFile are more powerful and more flexible. Check chapter "Custom Protocols" for more information.

The `ReadByIdCompare` command can be used to read node configuration and identification using the "Read by identifier" service from the LIN standard diagnostic and compare them with data stored in the SDFFile.

The 8 data bytes of the Master Request always have the following structure:

Data bytes	D1	D2	D3	D4	D5	D6	D7	D8
Description	NAD	PCI	SID	Identifier	Supplier ID LSB	Supplier ID MSB	Funkction ID LSB	Functio ID MSB
Constant values		0x06	0xB2					
Values set via System Variables	@@Diag NodeId				@@Diag SupplierId		@@Diag FunctionId	
Values set via parameters				Second Parameter				

Additionally a schedule will be executed, that is defined by the system variable `@@ScheduleDiag`. This schedule should contain the MRQ and SRP diagnostic frames.

If the diagnostic schedule, NAD, supplier ID or function ID system variables are not set within the SDF, they will receive default values:

System variable	Description	Default value if not explicitly set
<code>@@ScheduleDiag</code>	Diagnostic schedule	0
<code>@@DiagNodeId</code>	NAD	0x7F
<code>@@DiagSupplierId</code>	Supplier ID	0x7FFF
<code>@@DiagFunctionId</code>	Function ID	0xFFFF

The default values of `@@DiagNodeId`, `@@DiagSupplierId` and `@@DiagFunctionId` are wildcards and address every node.

The response will be compared with a compare block, that is defined in the SDFFile. A compare block always consist of the start index, the length and the data bytes, the response is compared with. The result will only show a match, if all data bytes from the compare block match the data from the response.

Up to 8 data blocks can be defined. The following examples show the structure of the compare blocks:

System variables	Description
<code>@@CmpBlockXStartPos</code>	This variable defines the index of the first byte within the data bytes in the response, that should be compared. X identifies the the compare block.
<code>@@CmpBlockXLen</code>	This variable defines the number of bytes, that should be compared. X identifies the compare block.
<code>@@CmpBlockXDataY</code>	This variable defines the value of a byte within the comapre block. X identifies the compare block. Y identifies the data byte index.

The following example shows the definition of two compare blocks:

System variable name	Initial value	Block identifier	Description
@@CmpBlock1StartPos	1	1	The comparison will start at the byte with index 1, which is the second byte.
@@CmpBlock1Len	4		The compare block consists of 4 data bytes.
@@CmpBlock1Data0	0x87		The response data will be compared with: 0x87 0x08 0x01 0x03
@@CmpBlock1Data1	0x08		
@@CmpBlock1Data2	0x01		
@@CmpBlock1Data3	0x03		
@@CmpBlock2StartPos	2	2	The comparison will start at the byte with index 2, which is the third byte.
@@CmpBlock2Len	6		The compare block consists of 6 data bytes.
@@CmpBlock2Data0	0x03		The response data will be compared with: 0x03 0x5D 0x3A 0x38 0x12 0x9F
@@CmpBlock2Data1	0x5D		
@@CmpBlock2Data2	0x3A		
@@CmpBlock2Data3	0x38		
@@CmpBlock2Data4	0x12		
@@CmpBlock2Data5	0x9F		

Parameter	Description																		
1	<p>This parameter is the index of the channel, that the SDFile should be loaded to.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>																		
2	<p>This parameter is the identifier used in the request. Certain identifiers are already defined in the LIN specification:</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Identifier</th> <th>Description</th> <th>Length of response</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>LIN product identification</td> <td>RSID + 5</td> </tr> <tr> <td>1</td> <td>Serial number</td> <td>RSID + 4</td> </tr> <tr> <td>2...31</td> <td>Reserved</td> <td></td> </tr> <tr> <td>32...63</td> <td>user defined</td> <td>User defined</td> </tr> <tr> <td>64...255</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Identifier	Description	Length of response	0	LIN product identification	RSID + 5	1	Serial number	RSID + 4	2...31	Reserved		32...63	user defined	User defined	64...255	Reserved	
Identifier	Description	Length of response																	
0	LIN product identification	RSID + 5																	
1	Serial number	RSID + 4																	
2...31	Reserved																		
32...63	user defined	User defined																	
64...255	Reserved																		
3	The identifier of the compare block in the SDFile.																		

Return value	Example
: 0 if the response and the compare block are equal.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1
<p>If the slave responded with a NRC, the error also includes the first two bytes after the NRC.</p> <div style="background-color: orange; padding: 5px;"> <p>Version incompatibility</p> <p>This result type is only returned when the Compatibility API is active. If not, only the error code is returned. Check chapter "API modes" for more information.</p> </div>	: @31 FF FF

Type	Example	Description
Command	<code>:ReadByIdCompare 0 1 3</code>	The serial number is requested. The data bytes in the response are compared with block 3.
Response	<code>:0</code>	The data bytes in the response match the data in block 3.

Response 0 if the response data matches the compare values. The `ReadByIdCompare` command is available since firmware version V. 2.20.

7.8.29 Command WaitSignal

The `WaitSignal` command can be used to wait until a signal equals a certain target value or a timeout is reached.

Parameter	Description									
1	<p>This parameter is the index of the channel, on which the signal value should be queried.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>									
2	<p>This parameter identifies the signal, whose value should be compared.</p> <p>To identify a signal the following alternatives are possible:</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Signal property</th> <th>Description</th> <th>Example</th> </tr> </thead> <tbody> <tr> <td>Signal index</td> <td>The signal index can be used simply by passing the index number.</td> <td><code>:LinRdSignal 0 12</code></td> </tr> <tr> <td>Signal name</td> <td>The signal name can be used by putting an exclamation mark directly in front of the name.</td> <td><code>:RdSignal 0 !SignalName</code></td> </tr> </tbody> </table>	Signal property	Description	Example	Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>	Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>
Signal property	Description	Example								
Signal index	The signal index can be used simply by passing the index number.	<code>:LinRdSignal 0 12</code>								
Signal name	The signal name can be used by putting an exclamation mark directly in front of the name.	<code>:RdSignal 0 !SignalName</code>								
3	This parameter must always be "=".									
4	This parameter is the target value, that the signal value needs to equal.									
5	This parameter is the timeout value in Milliseconds. If this time is passed and the signal has not equaled the target value, the command will return.									

Return value	Example
<code>:0</code> is returned, if the signal equals the target value.	<code>0:</code>
<code>:@16</code> is returned, if the signal did not equal the target value within the given timeout.	<code>:@16</code>
A default error may be returned. Check chapter "Return codes" for more information.	<code>:@1</code>

Type	Example	Description
Command	<code>:WaitSignal 0 !test = 5 3000</code>	This call will block until the signal value of test equals the target value 5 or 3 seconds have passed.
Response	<code>:@16</code>	The timeout was reached. The signal value never equaled the target value within this time period.
Command	<code>:WaitSignal 0 !test = 2 2000</code>	This call will block until the signal value of test equals the target value 2 or 2 seconds have passed.
Response	<code>:0</code>	The signal value equaled the target value.

7.8.30 Command SeqRun

The `SeqRun` command can be used to start a sequence. A sequence is a list of commands, that are stored in the SDFFile. Check chapter "Sequences" for more information.

Parameter	Description								
1	<p>This parameter is the index of the channel, on which the sequence should run on. All channel indices within the sequence are replaced, except the special # notation is used.</p> <div style="background-color: yellow; padding: 5px;"> <p>Warning</p> <p>This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>								
2	This parameter is the index of the sequence.								
3	<p>This optional parameter changes the value, that is returned.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>None</td> <td>If the all commands of the sequence returned expected values, the return value is 0.</td> </tr> <tr> <td>0</td> <td></td> </tr> <tr> <td>1</td> <td>If the all commands of the sequence returned expected values, the return value is the response of the last command.</td> </tr> </tbody> </table>	Value	Description	None	If the all commands of the sequence returned expected values, the return value is 0.	0		1	If the all commands of the sequence returned expected values, the return value is the response of the last command.
Value	Description								
None	If the all commands of the sequence returned expected values, the return value is 0.								
0									
1	If the all commands of the sequence returned expected values, the return value is the response of the last command.								

Return value	Example	Description
Command	:SeqRun 0 1	The sequence with index 1 is started on channel 0.
Response	:0	All commands of the sequence were executed successfully.
Command	:SeqRun 1 4	The sequence with index 4 is started on channel 1.
Response	:103	The third command of the schedule returned not an expected value.
Command	:SeqRun 0 2 1	The sequence with index 2 is started on channel 0.
Response	:15	All commands of the sequence were executed successfully and the result of the last command was 15.
Command	:SeqRun 2 3 1	The sequence with index 3 is started on channel 2.
Response	:@101	The first command of the schedule returned not an expected value.

7.8.31 Command SeqExec



Version incompatibility

The SeqExec is deprecated and should not be used. Use "Command SeqRun" instead.

The SeqExec is missing the first parameter of "Command SeqRun", the channel index. The remaining parameters and the behaviour is the same.

7.8.32 Command Delay



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The Delay command can be used to create a delay of a given time.

Parameter	Description
1	The time in Milliseconds, that this command will take to execute.


Return value	Example
:0 returned, when the delay time elapsed.	0:
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:Delay 1000	This command will take 1 second to finish.
Response	:0	The 1 second has passed.

7.8.33 Command DigOut

The DigOut command can be used to set the digital outputs of the Baby-LIN-MB-II.

Check chapter "X9 - LIN, CAN and IO" for more information.

Parameter	Description																		
1	This parameter selects the digital output.																		
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Switch port (X9-24, X9-12)</td> </tr> <tr> <td>1</td> <td>Power switch (X9-22)</td> </tr> <tr> <td>2</td> <td>MIF-DIO-IO-4 (X9-19)</td> </tr> <tr> <td>3</td> <td>MIF-DIO-IO-5 (X9-6)</td> </tr> <tr> <td>4</td> <td>MIF-DIO-IO-6 (X9-18)</td> </tr> <tr> <td>5</td> <td>MIF-DIO-IO-1 (X9-5)</td> </tr> <tr> <td>6</td> <td>MIF-DIO-IO-2 (X9-4)</td> </tr> <tr> <td>7</td> <td>MIF-DIO-IO-3 (X9-7)</td> </tr> </tbody> </table>	Value	Description	0	Switch port (X9-24, X9-12)	1	Power switch (X9-22)	2	MIF-DIO-IO-4 (X9-19)	3	MIF-DIO-IO-5 (X9-6)	4	MIF-DIO-IO-6 (X9-18)	5	MIF-DIO-IO-1 (X9-5)	6	MIF-DIO-IO-2 (X9-4)	7	MIF-DIO-IO-3 (X9-7)
	Value	Description																	
	0	Switch port (X9-24, X9-12)																	
	1	Power switch (X9-22)																	
	2	MIF-DIO-IO-4 (X9-19)																	
	3	MIF-DIO-IO-5 (X9-6)																	
	4	MIF-DIO-IO-6 (X9-18)																	
	5	MIF-DIO-IO-1 (X9-5)																	
	6	MIF-DIO-IO-2 (X9-4)																	
7	MIF-DIO-IO-3 (X9-7)																		
 Warning These outputs are only available if a MIF-DIO is installed. Check chapter "MIF extension modules" for more information.																			
2	This parameter selects the target state for the digital output.																		
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> <th>Switch port X9-24, X9-12</th> <th>Power switch X9-22</th> <th>MIF-DIO-IO-X</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The output will be set to the low state.</td> <td>X9-24 and X9-12 are disconnected.</td> <td>X9-22 is floating.</td> <td>The pin ist floating</td> </tr> <tr> <td>1</td> <td>The output will be set to the high state.</td> <td>X9-24 and X9-12 are connected.</td> <td>X9-22 has the level of the LINSupply (X9-9).</td> <td>The pin has the level of MIF-DIOGND (X9-17)</td> </tr> </tbody> </table>	Value	Description	Switch port X9-24, X9-12	Power switch X9-22	MIF-DIO-IO-X	0	The output will be set to the low state.	X9-24 and X9-12 are disconnected.	X9-22 is floating.	The pin ist floating	1	The output will be set to the high state.	X9-24 and X9-12 are connected.	X9-22 has the level of the LINSupply (X9-9).	The pin has the level of MIF-DIOGND (X9-17)			
	Value	Description	Switch port X9-24, X9-12	Power switch X9-22	MIF-DIO-IO-X														
0	The output will be set to the low state.	X9-24 and X9-12 are disconnected.	X9-22 is floating.	The pin ist floating															
1	The output will be set to the high state.	X9-24 and X9-12 are connected.	X9-22 has the level of the LINSupply (X9-9).	The pin has the level of MIF-DIOGND (X9-17)															

Return value	Example
:0 is returned, if the output was set successfully.	:0
A default error may be returned. Check chapter "Return codes" for more information.	:@1

Type	Example	Description
Command	:DigOut 0 0	Set the switch port to disconnected
Response	:0	The output was successfully set to the low state
Command	:Digout 1 1	Set the power switch to the level of the LIN-Supply
Response	:0	The output was successfully set to the high state



Warning
 Some plugins allow the inversion of the output states. If the digital outputs do not react like expected please check the configuration variables of the installed plugins.

7.8.34 Command DigIn




Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The DigIn command can be used to evaluate the digital inputs of the Baby-LIN-MB-II.

Check chapter "X9 - LIN, CAN and IO" for more information.

Parameter	Description																
1	<p>This parameter selects the digital input.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>DigIn (X9-13, X9-25))</td> </tr> <tr> <td>1</td> <td>MIF-DIO-IN-1 (X9-5)</td> </tr> <tr> <td>2</td> <td>MIF-DIO-IN-2 (X9-4)</td> </tr> <tr> <td>3</td> <td>MIF-DIO-IN-3 (X9-7)</td> </tr> <tr> <td>4</td> <td>MIF-DIO-IO-4 (X9-19)</td> </tr> <tr> <td>5</td> <td>MIF-DIO-IO-5 (X9-6)</td> </tr> <tr> <td>6</td> <td>MIF-DIO-IO-6 (X9-18)</td> </tr> </tbody> </table>	Value	Description	0	DigIn (X9-13, X9-25))	1	MIF-DIO-IN-1 (X9-5)	2	MIF-DIO-IN-2 (X9-4)	3	MIF-DIO-IN-3 (X9-7)	4	MIF-DIO-IO-4 (X9-19)	5	MIF-DIO-IO-5 (X9-6)	6	MIF-DIO-IO-6 (X9-18)
Value	Description																
0	DigIn (X9-13, X9-25))																
1	MIF-DIO-IN-1 (X9-5)																
2	MIF-DIO-IN-2 (X9-4)																
3	MIF-DIO-IN-3 (X9-7)																
4	MIF-DIO-IO-4 (X9-19)																
5	MIF-DIO-IO-5 (X9-6)																
6	MIF-DIO-IO-6 (X9-18)																
	 <p>Warning These inputs are only available if a MIF-DIO is installed. Check chapter "MIF extension modules" for more information.</p>																

Return value	Example	Description
0: is returned, if the input is in low state.	: 0	
: 1 is returned, if the input is in high state.	: 1	
A default error may be returned. Check chapter "Return codes" for more information.	: @1	

Type	Example	Description
Command	DigIn 0	get the state of the DigIn
Response	: 0	The input was evaluated successfully and it is in low state
Command	DigIn 4	Get teh state of the MIF-DIO-IO-4.
Response	: 1	The input was evaluated successfully and ist is in high state


7.8.35 Command SetConfigVar



Warning

This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The SetConfigVar command can be used to set the value of the configuration variables of the Baby-LIN-MB-II. Check chapter "Configuration variables" for more information.

Parameter	Description
1	This parameter selects the configuration variable by the name.
2	<p>This parameter selects the target value for the configuration variable. Depending on the type, different values are possible.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning This channel parameter is only used, if you use the single socket connection. If you instead are using multi socket connections, this channel parameter must be omitted. Check chapter "TCP connections: Single and multi socket" for more information.</p> </div>

Return value	Example
: 0 is returned, if the configuration variable was set successfully.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	<code>:SetConfigVar autostart.mode 1</code>	Set the value of the configuration variable autostart.mode to value 1.
Response	: 0	The configuration variable was successfully set.
Command	<code>:SetConfigVar web.support.mail.subject "Support Mail"</code>	Set the value of the configuration variable web.support.mail.subject to value Support Mail.
Response	: 0	The configuration variable was successfully set.

7.8.36 Command LicenceInstall




Warning
This command is only available for the Baby-LIN-MB-II. The older Baby-LIN-MB does not support it.

The LicenceInstall command can be used to install an activation code to activate new features.



Warning
After a new activation code is installed, the Baby-LIN-MB-II must be restarted, before the activation code is active.

Parameter	Description
1	<p>This parameter is the activation code, you want to install. Check chapter "Handle voucher and activation codes" for more information.</p> <div style="background-color: yellow; padding: 5px;">  <p>Warning The activation code must be framed with quote signs.</p> </div>

Return value	Example
0: is returned, if the activation code was installed successfully.	: 0
A default error may be returned. Check chapter "Return codes" for more information.	: @1

Type	Example	Description
Command	<code>:LicenceInstall "YKDFE TEA3F Q1FZ1 BJCFB IPM10 B"</code>	Install the activation code YKDFE TEA3F Q1FZ1 BJCFB IPM10 B.
Response	0	The activation code was installed successfully.

7.9 Return codes

If a command was not understood or parameters were invalid or any specific error condition on the LIN Bus side causes a command to fail, the system will respond with a error code. All error codes start with the leading @ character. The error code with ID 1 would be returned as: @1

Error code	Example	Error code	Example
1	Unknown command	60	Could not retrieve section infos
2	Invalid Parameter	61	Section could not be downloaded
3	Parameter out of range	70	Configuration variable not found
4	Parameter missing	71	Configuration variable could not be set
5	Command Timeout (incomplete command)	71	Configuration variable could not be received
6	File not found	80	Could not receive pin state
7	File load error	81	Invalid mode
8	SDF download to MIF-LIN failed	82	License code invalid
9	Internal MIF-LIN operation failed	83	Maximum reached
10	Error Signal look up failed	84	Transport protocol error: P2 timing elapsed
11	Node Timeout (no answer from LIN Bus)	99	Critical error. Please contact us: "Support information"
12	LIN bus supply missing	100 - 255	Error in sequence. Return value - 100 = the sequence step that returned an error
13	Channel unknown not activated	256	Power missing, when trying to establish EOL session
14	Data supplied by nodes is invalid or unknown	257	Eol request was responded by negative response
15	Command rejected due to missing prerequisite	260	Slave did not stop reporting Command in progress
16	Application function was not executed within given timeout	267	Slave did not respond on MasterRequest
17	Command failed due to settings in provided SDF	268	Power turned off, after being present in phase eol_initialisation
19	Error occurred during SDF parsing	280	Angle sent in Write OW Offset out of range range set by @LimitForOWOffset != 0
20	A SDF download is pending	300 - 399	Parameter error. If the first parameter is invalid, 301 is returned. If the second parameter is invalid, 302 is returned.
30	No SDF loaded	400	USB mass storage device is missing
31	DTL negative response	401	USB mass storage device could not be mounted
32	DTL Processing	402	An escape sequence was incomplete
33	DTL invalid data	403	An escape sequence was illegal
34	DTL invalid Response length	404	An invalid character was entered: ' ' (0x20), '' (0x22), ' ' (0x7E)
35	Invalid NAD	405	Writing to the log file failed
40	Wrong password	406	Accessing the log path failed
41	Access denied	420	The DTI response date ID was not valid
42	Access level is not sufficient	430	A bus error occurred
50	Command length	431	A schedule index was invalid
		432	An internal library failure occurred

Error code	Example	Error code	Example
512 ... 768	A NRC (negative response code) other than 0x78 occurred. The error code is the NRC code with an offset of 512.	4000	An internal error occurred during the startup of the Baby-LIN-MB-II
790	PLugin inti failed	4001	A plugin could not be opened
791	Plugin configuration data invalid	4002	The initialization function of a plugin could not be found
792	Could not lookup command reference	4003	The initialization function of a plugin failed
793	Could not lookup frame	4004	The command handler could not be found
794	Error in diagnostic schedule	4005	The CmdDone handler could not be found
800	Invalid sequence index in SeqExec command	10000	Plugin init failed
801	Commend parser error in sequence	10001	Plugin init failed twice
802	Not expected value in sequence	10002	Plugin init RTV register failed
803	Plugin command not immediate	50000 ... 50255	Custom macro result error
810	Invalid macro index in MacroExec command	70000 ...	BabyLIN device error offset
811	Invalid macro (e.g. empty macro)	-100001	Internal resource allocation problem. Maybe out of memory/handles/etc..
812	Macro operation refused	-100002	Specified handle invalid
813	Macro still running	-100003	There is no connection open
815	Same macro still running	-100004	Serial port could not be opened or closed
816	Other Macro still running	-100005	Baby-LIN command syntax error
817	Macro start failed	-100006	Baby-LIN does not answer within in timeout
818	Macro has never been executed	-100007	Unable to open file
819	Macro exited with an unhandled execption	-100008	Wrong parameter given to function
820	Invalid signal width	-100009	No datat available upon request
840	Unknown mode	-100010	No SDFfile was loaded previously
960	Error in BCD coded data from node	-100011	Internal message fformat error
961	Invalid compare block used (no definition in SDF file)	-100012	The given signal_nr or name does not exist in loaded SDFfile
962	Compare block mismatch	-100013	The signal chosen is a scalar, but an array function was called
998	Missing parameters for array type parameter	-100014	The signal chosen is an array, but an scalar function was called
999	Memory exhausted (Heap Error)	-100015	The SDFfile is unsupported by connected Baby-LIN due to insufficient firmware version
1000 ... 1999	Internal error	-100016	The given signal has no encoding
2000	Host response error	-100017	The given buffer is too small
2001	Double command	-100018	There is no additional answer data present from last sendCommand-call
2002	Parsing of sub command failed	-100019	Additional data with given index/name not present
3001	PMDM: Buspower is on	-100020	Device supports no Channels
3002	PMDM: Buspower is off	-100021	Unknown command passed to sendCommand
3003	PMDM: Flash init	-100022	A sendCommand message time out
3004	PMDM: Flash write	-100023	SDFfile can not be loaded to a the device due to incompatibility (incompatible SDFV3 to SDF-V2 device)
3005	PMDM: Flash read back	-100024	Value passed as a SDFfile handle is not valid
3006	PMDM:Flash channel locked	-100025	SDFfile can not be unloaded as the SDFfile is in use on a device
3007	PMDM: Flash channel unprepared	-100026	Can not execute command because SDFfile download is in progress

The following error codes are redirected from code, that is also used within the "Baby-LIN-DLL". Therefore the following error codes can occur. Please contact us, if you experience one of the following codes. Check chapter "Support information" for more information.

Error code	Example
0	Operation successful
-100027	Function can not be executed due to wrong mode or configuration
-100094	The number of parameters is not valid for this method
-100095	The value could not be read
-100096	One of the parameters is invalid
-100097	The property has no getter for the type
-100098	The property has no setter for the type
-100099	The value given was not set
-100100 ... -100200	The path parameter given to one of the BLC_UnifiedAccess functions could not be found. The index of that key is the return value --100100. This index is 0 based.
-100201	The ISO-TP service is supposed to send a request but has no request data
-100202	During the reception of the response or the request a frame timeout occurred
-100203	A frame send by the library was not echoed by the BabyLIN device within a timeout. You might have to do a disframe/mon_on with that FrameID.
-100204	The response was not received within timeout_response milliseconds. Maybe the request is malformed?
-100205	A flow-control frame send by the library was not echoed by the BabyLIN device within a timeout. You might have to do a disframe/mon_on with that FrameID.
-100206	The flow-control state reported by the target is not one of the known states
-100207	The flow-control state was "wait"(0x1) in more than max_flow_wait flow-control frames
-100208	The flow-control state was "overflow"(0x2)
-100209	The flow-control was not issued by the other node
-100210	The data for a frame to send can not be put into a frame with the specified frame length.
-101100 ... -101200	The path parameter given to one of the BLC_UnifiedAccess functions could not be resolved. The index of the object, that could not be found, is the return value.

8 Workflow

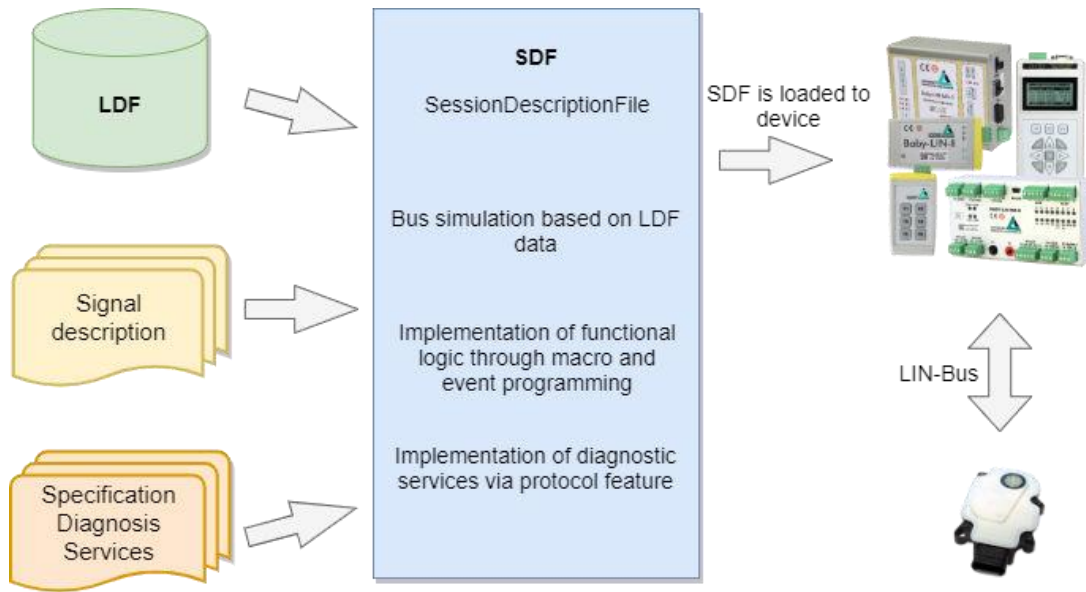
In this chapter we will show you how the workflow looks like in a typical LIN use case. For this purpose, we will introduce the following components to you:

- LDF
- Signal description
- Specification Diagnosis Services

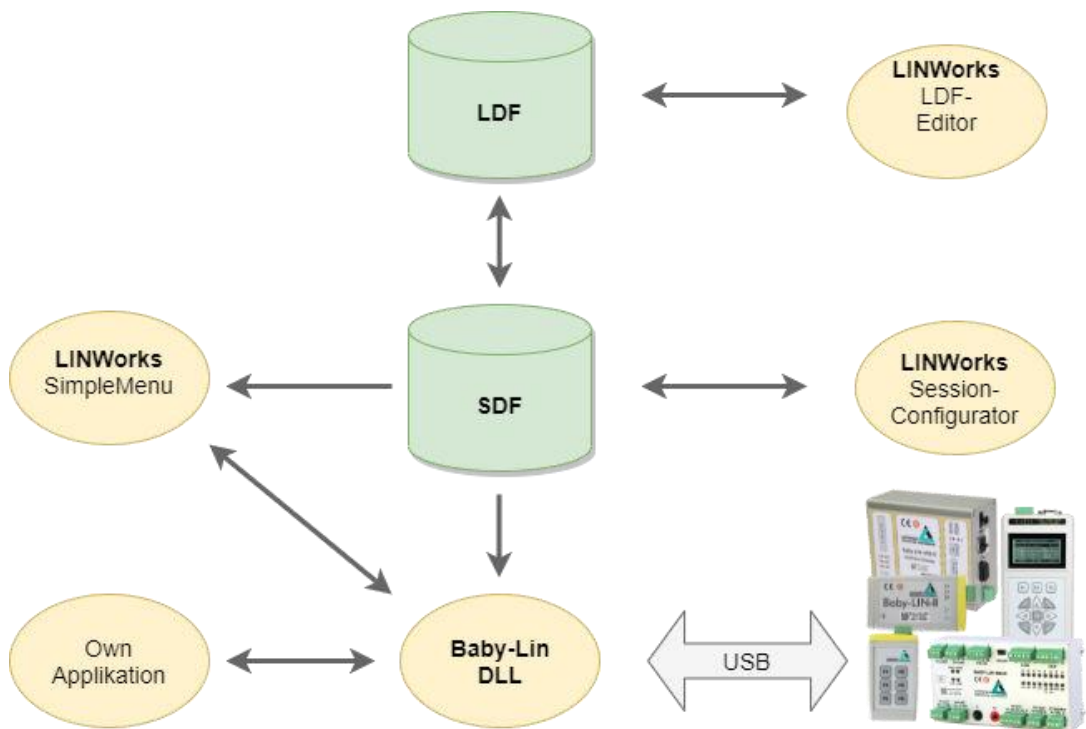
From this information, the SessionDescriptionFile (SDF) can be created. The SDF is the linchpin in LINWorks-based applications.

8.1 Overview

The following graphic shows the typical workflow of a LIN-based application with our Baby-LIN-Device.



This diagram shows how the individual LINWorks software applications are linked to each other.



8.2 Getting started

8.2.1 Introduction

This getting started guide will show you how to create your Lin application using the information from the LDF and the signal descriptions. In the following, you will learn how to create an LDF and integrate it into the SDF. Furthermore, the Unifeid Diagnostic Services will be introduced. After you have successfully created the SDF, the Baby-LIN-MB-II can be operated in standalone mode, LIN bus data can be logged, or macros can be defined for autostart.



Advice

This guide assumes you are using a Microsoft Windows operating system.

8.2.2 Installation

Before you can start using the Baby-LIN-MB-II you have to install several components of the LINWorks software.

If you have not already downloaded the LINWorks software, please download it now from our website: Link: www.lipowsky.de/downloads



Advice

Check chapter "Downloads" for more information.

The following components are required for this getting started guide:

- Baby-LIN driver
- SessionConf
- SimpleMenu
- LDFEdit

8.3 LDF

LDF (LinDescriptionFile) has been developed by the LIN Consortium, in which various parties such as car manufacturers, suppliers and tool suppliers were involved. This means that the LDF specification is not dependent on a single manufacturer and can be used universally. The Format and syntax of the LDF are described in the LIN specification.

Each LIN bus has its own LDF, which collects all the properties of this specific bus in one document. This includes which nodes are present on the bus, which frames are defined and according to which scheme they are to be emulated.

8.3.1 LDF Example

The following example shows the LDF of a windscreen wiper motor.

```
LDF header                LIN_description_file ;
                             LIN_language_version = "1.3" ;
                             LIN_speed = 19.200 kbps ;

Node section              Nodes {
                             Master:MasterECU,1.0000 ms,0.1000 ms ;
                             Slaves:Slave1Motor,Slave2Sensor;
                             }

                             { MessageCounter:8,0x00,MasterECU,Slave1Motor,Slave2Sensor;

                             Ignition:1,0x0,MasterECU,Slave1Motor,Slave2Sensor;
                             WiperSpeed:3,0x0,MasterECU,Slave1Motor;
                             Temperature:8,0xFF,MasterECU,Slave1Motor,Slave2Sensor;
                             WiperActive:1,0x0,Slave1Motor,MasterECU;
                             ParkPosition:1,0x0,Slave1Motor,MasterECU;
                             CycleCounter:16,0x00,Slave1Motor,MasterECU;
                             StatusSensor:8,0x00,Slave2Sensor,MasterECU;
                             ValueSensor:8,0x00,Slave2Sensor,MasterECU;
                             }

Frame section            Frames {
                             MasterCmd:0x10,MasterECU,4{MessageCounter,0;
                             Ignition,8; WiperSpeed,9; Temperature,16; }
                             MotorFrame:0x20,Slave1Motor,4{ WiperActive,0;
                             ParkPosition,1; CycleCounter,16; }
                             SensorFrame:0x30,Slave2Sensor,2StatusSensor,0; ValueSensor,8;
                             }

Schedule table          Schedule_tables {
                             Table1 { MasterCmd delay 20.0000 ms ;
                             MotorFrame delay 20.0000 ms ;
                             SensorFrame delay 20.0000 ms ;}
                             }
```


Signal section

Signals

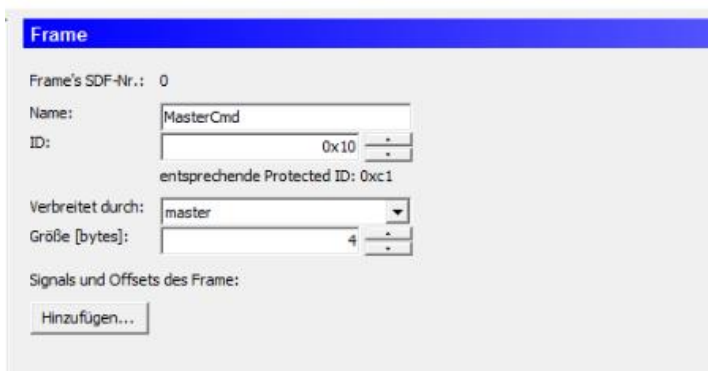
Signal encoding section

```
Signal_encoding_types {
EncodingSpeed { logical_value,0x00,"Off" ;
logical_value,0x01,"Speed1" ;
logical_value,0x02,"Speed2" ;
logical_value,0x03,"Interval" ;}
EncodingTemp { physical_value,0,253,0.8,- 35,"degrees C" ;
logical_value,0xFE,"Signal not supported" ;
logical_value,0xFF,"Signal not available" ;}
}

Signal_representation
{ EncodingSpeed:WiperSpee
d;
EncodingTemp:Temperature;
}
```

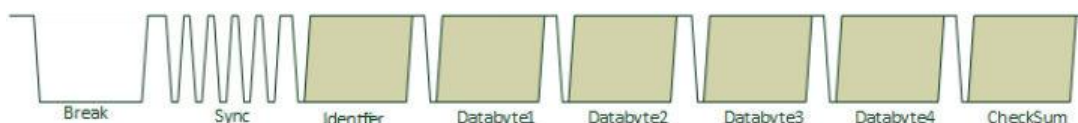
8.3.2 LIN application frames

With the information from an LDF, you can assign all frames that appear on the bus to your publisher using the PID. You can also interpret the data regarding the signals it contains.



```
Frames{
MasterCmd:0x10,MasterECU,4 {
MessageCounter,0;
Ignition,8;
WiperSpeed,9;
Temperature,16; }
}
```

The frame is structured as shown in the following graphic. The frame defined in the LDF is recognised with the identifier with ID = 0x10 and the signals can be mapped from the 4 databytes.



8.3.2.1 Protected LIN identifier

The Frame Id is 8 Bit in size, where by the upper 2 bits are used as parity bits. So only 6 bits remains to represent the effective frame identifier. This

makes a range of 64 different frame id's.

Paritybit P1 (ID.7) ID.1^ID.3^ID.4^ID.5	Paritybit P0 (ID.6) !(ID.0^ID.1^ID.2^ID.4)	Identifier Bits ID.5 - ID.0 0...63
--	---	---------------------------------------

Id dec	Id hex	PID	Id dec	Id Hex	PID	Id dec	Id hex	PID	Id dec	Id hex	PID
0	0x00	0x80	16	0x10	0x50	32	0x20	0x20	48	0x30	0xF0
1	0x01	0xc1	17	0x11	0x11	33	0x21	0x61	49	0x31	0xB1
2	0x02	0x42	18	0x12	0x92	34	0x22	0xE2	50	0x32	0x32
3	0x03	0x03	19	0x13	0xD3	35	0x23	0xA3	51	0x33	0x73
4	0x04	0xc4	20	0x14	0x14	36	0x24	0x64	52	0x34	0xB4
5	0x05	0x85	21	0x15	0x55	37	0x25	0x25	53	0x35	0xF5
6	0x06	0x06	22	0x16	0xD6	38	0x26	0xA6	54	0x36	0x76
7	0x07	0x47	23	0x17	0x97	39	0x27	0xE7	55	0x37	0x37
8	0x08	0x08	24	0x18	0xD8	40	0x28	0xA8	56	0x38	0x78
9	0x09	0x49	25	0x19	0x99	41	0x29	0xE9	57	0x39	0x39
10	0x0A	0xCA	26	0x1A	0x1A	42	0x2A	0x6A	58	0x3A	0xBA
11	0x0B	0x8B	27	0x1B	0x5B	43	0x2B	0x2B	59	0x3B	0xFB
12	0x0C	0x4C	28	0x1C	0x9C	44	0x2C	0xEC	60	0x3C	0x3C
13	0x0D	0x0D	29	0x1D	0xDD	45	0x2D	0xAD	61	0x3D	0x7D
14	0x0E	0x8E	30	0x1E	0x5E	46	0x2E	0x2E	62	0x3E	0xFE
15	0x0F	0xCF	31	0x1F	0x1F	47	0x2F	0x6F	63	0x3F	0xBF



Advice

Note that the following IDs are reserved for protocol extensions and diagnostic and configuration data:

- 60 (0x3C) and 61 (0x3D) are used to carry diagnostic and configuration data.
- 62 (0x3E) and 63 (0x3F) are reserved for future protocol enhancements.

8.3.3 LIN Scheduling

The order in which the frames are sent to the LIN bus is defined in a so-called Schedule Table. One or more Schedule Table(s) are defined in each LDF.

Each table entry describes a frame by its LDF name and a delay time, which is the time that is made available to the frame on the bus.



A Schedule Table is always selected as active and is executed by the master. The master places the corresponding frame headers on the bus and the publisher assigned to this frame places the corresponding data section + checksum on the bus.

Only the master can switch the Schedule Table. Thus the master application determines which frames appear on the bus in which time sequence.

8.3.4 LIN Diagnostic frames

Diagnostic frames are a pair of MasterRequest (0x3c) and SlaveResponse (0x3D) frames. Used to send information that is not described in the LDF.

0x3C MasterRequest:

Request Data define the node and the requested action.



0x3D SlaveResponse:

Data generated by the addressed slave; content depends on request



The Master Request and Slave Response have special properties:

- They are always 8 bytes long and always use the Classic Checksum.
- No static mapping of frame data to signals; frame(s) are containers for transporting generic data.
- Request and response data can consist of more than 8 data bytes.

The MasterRequest - SlaveResponse mechanism can be used to transmit a wide variety of data because it is a universal transport mechanism. A main application is the diagnosis and End of Line (EOL) configuration of nodes.

In the field there is a whole range of different protocols, depending on the vehicle and ECU manufacturer:

- A lot of proprietary diagnostics or EOL protocols
- DTL based protocols (Diagnostic Transport Layer)
- Keyword 2000 Protocol (ISO 14230 -1 to 4)
- UDS (Unified Diagnostic Services) (ISO 14229-1:2013)

8.4 Session Description File (SDF)

8.4.1 How to create a LIN application

1. Requirement



A LIN node (slave) and a suitable LDF file are available. An application is to be implemented in which a simulated LIN master allows the node to be operated in a certain way.

2. Requirement



However, the information in the LDF is usually not sufficient. The LDF describes the access and interpretation of the signals, but the LDF does not describe the functional logic behind these signals. Therefore you need an additional signal description which describes the functional logic of the signals.

3. Requirement



If the task also requires diagnostic communication, a specification of the diagnostic services supported by the nodes is also required. In the LDF, only the frames with the respective data bytes are defined, but not their meaning.

These requirements can then be defined and edited together in a Session Description file (SDF).

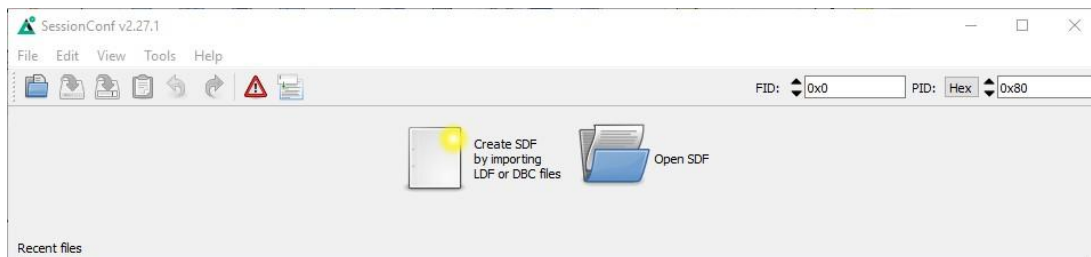
8.4.2 Introduction

The Session Description file (SDF) contains the bus simulation based on the LDF data. The logic of the individual frames and signals can be programmed by macros and events. In addition to the LDF LIN schedule, further diagnostic services can be implemented in the SDF via protocols.

This makes the SDF the central working point of all LINWorks applications.

8.4.3 Create a SDF

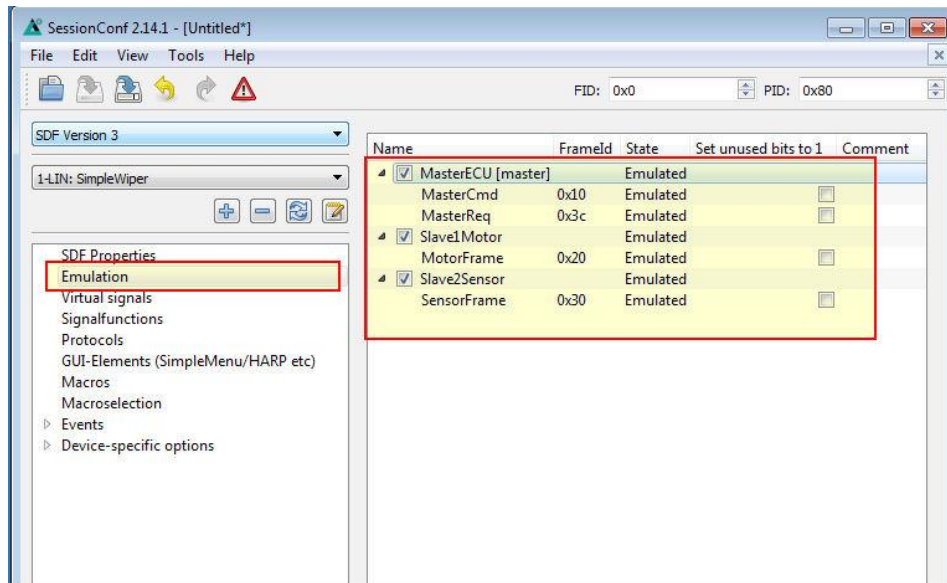
The SessionConf software application is used to create and edit the SDF. For this purpose, an existing LDF is imported.



8.4.4 Common Setup

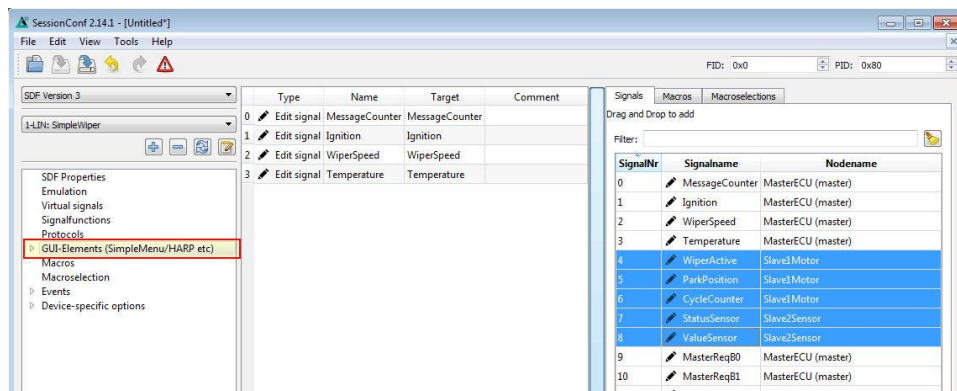
8.4.4.1 Emulation

Select Emulation in the navigation menu on the left. Here you can select which nodes you want to be simulated by the Baby-LIN-MB-II. If you only want to monitor the LIN-Bus, select nothing.



8.4.4.2 GUI-Elements

Select GUI-Elements in the navigation menu on the left. Here you can add signals you want to monitor.



Advice

There are other ways to monitor frames and signals, but this is a good and configurable starting point.

8.4.4.3 Virtual signals

Virtual signals can store values just like bus signals, but they do not appear on the bus. They can be used for many different tasks like:

- Temporary values, like counters
- Operands and results from calculations
- Store constants
- etc.

The size of a virtual signal can be set to 1...64 bits. important for use in the protocol feature.

Each signal has a default value that is set when the SDF is loaded.

Name	Length	Initial Value (decimal)	Initial Value (hexadecimal)	Initial Value (ASCII)	Reset on BUS start	Signed	
@SYSBUSSTATE	32	0	0x0		<input type="checkbox"/>	<input type="checkbox"/>	Gets the state of the LIN- or CAN-Bus.
int8	32	0	0x0		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
int16	16	0	0x0		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
int32	32	0	0x0		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
int64	64	0	0x0		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
repetitions	32	0	0x0		<input type="checkbox"/>	<input type="checkbox"/>	
runtime	32	0	0x0		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
sync	1	0	0x0		<input type="checkbox"/>	<input type="checkbox"/>	
failure	16	0	0x0		<input type="checkbox"/>	<input type="checkbox"/>	

8.4.4.4 System signals

System signals are virtual signals with reserved names. When a system signal is applied, a virtual signal is created at the same time and linked to a specific behaviour.

In this way, you can access timer, input and output resources and system information.



Advice
For more information and a list of all available system signals, please check the chapter "System variables".

8.4.4.5 Macros

Macros are used to combine multiple operations into a sequence. Macros can be started by events or, can also be called from other macros in the sense of a Goto or Gosub. The DLL API calls a macro with the macro_execute command.

Label	Condition	Command	Comment
0		Print on Debug report: "Macro starts"	
1		Gosub macro "BusStart0"	Macro BusStart is being executed
2		Gosub macro "Example(250, 1000)"	Macro Example is executed and is passed the values 250 and 1000 as parameters.
3		Print on Debug report: "Execution was successful"	

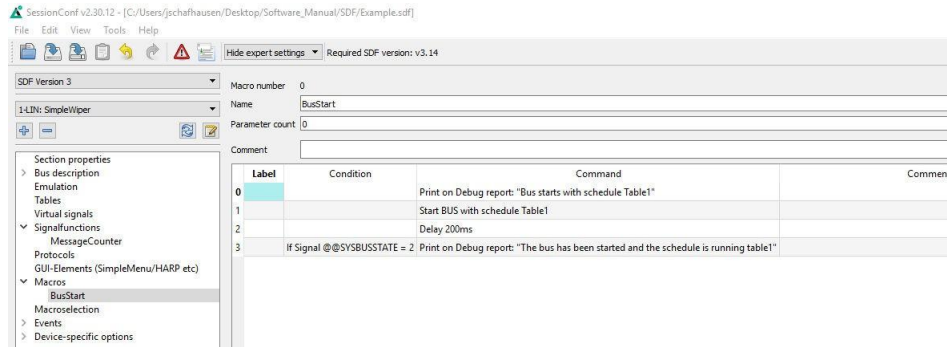
All Macro Commands can use signals from the LDF and signals from the Virtual Signal section like the system signals.

Another important function of the macros is to control the bus. The bus can be started and stopped via macro. Furthermore, the schedule can be selected and the status of the bus can be checked with the help of the system signals.



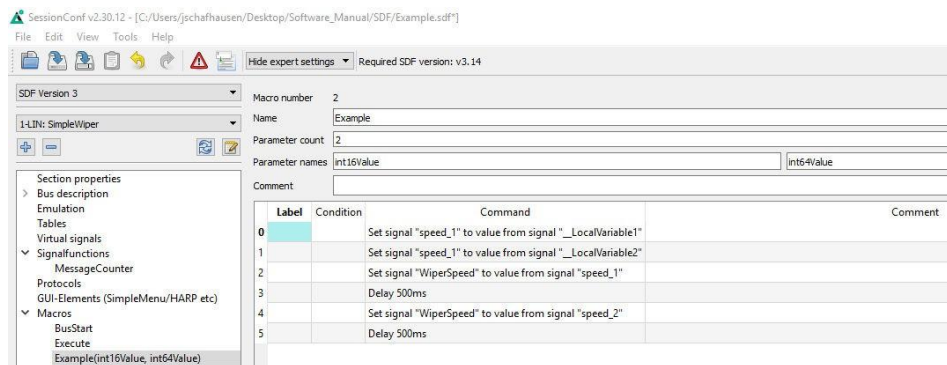
Advice

For more information or if you want to implement an autostart macro, please read the chapter "Configuring an autostart macro".



Each macro always provides 13 local signals:

`_LocalVariable1`, `_LocalVariable2`, ..., `_LocalVariable10`, `_Failure`, `_ResultLastMacroCommand`, `_Return`
 The last 3 provide a mechanism to return values to a callcontext (`_Return`, `_Failure`) or to check the result of a previous macro command. The signals `_LocalVariableX` can be used e.g. as temporary variables in a macro.



A macro can receive up to 10 parameters when called. In the macro definition, you can give these parameters names, which are then displayed on the left in the menu tree in brackets after the macro name. The parameters end up in the signals `_LocalVariable1...10` of the called. If no parameters or less than 10 parameters are passed, the remaining `_LocalVariableX` signals receive the value 0.

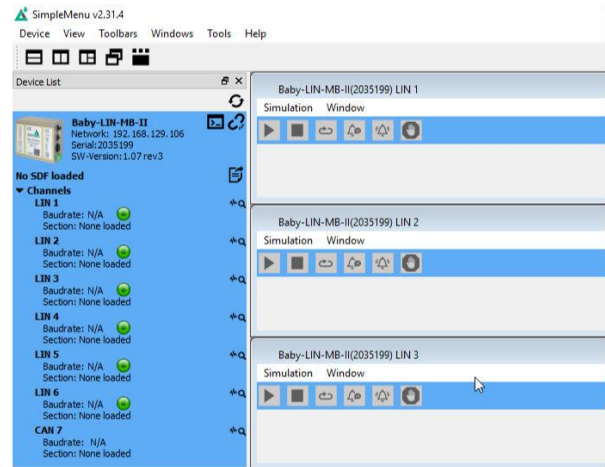
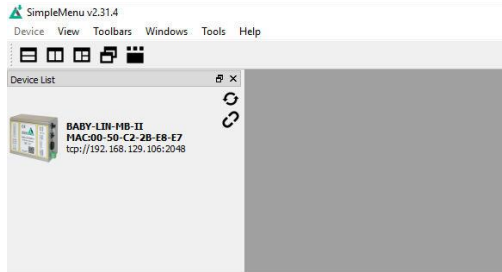
8.4.5 Example SDF

You can download the GettingStarted_Example SDF in the download area on our website under the following link.

Link: www.lipowsky.de/downloads

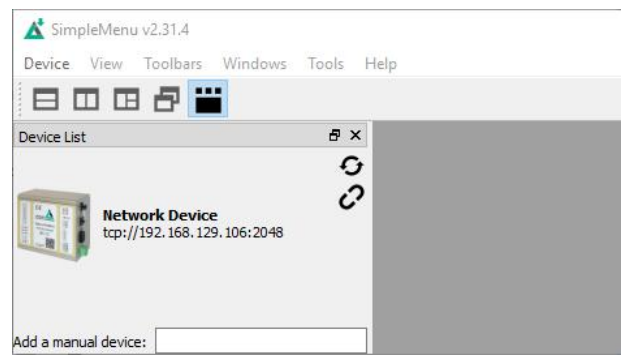
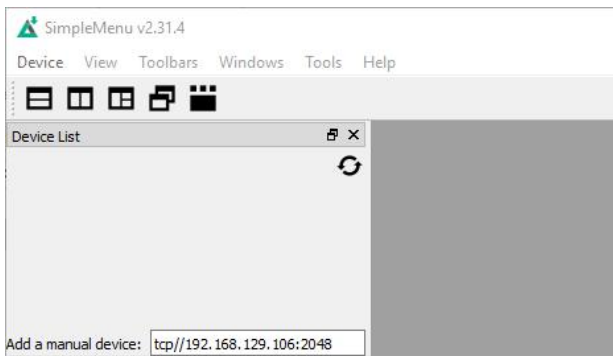
8.4.6 Start the bus communication

Start the SimpleMenu. You should be able to find your Baby-LIN-MB-II in the device list on the left. Click the connect button and then load the SDF you created earlier.

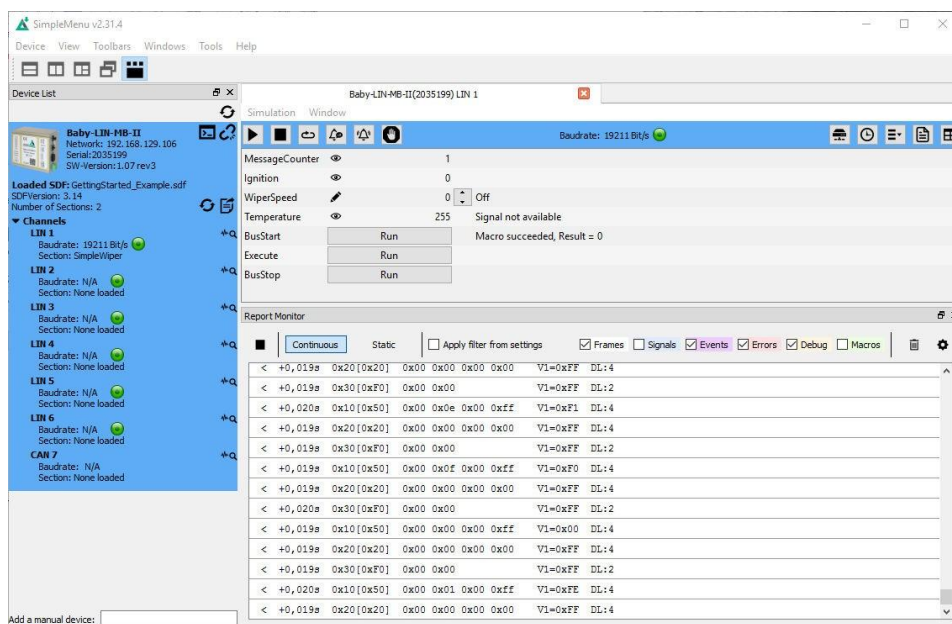


It may happen that you cannot find your Baby-LIN-MB-II in the Device list. This is because the device search in the network works via a UDP broadcast and some switches do not support it. In this case you have to add your Baby-LIN-MB-II manually.

To do this, enter the command `tcp://` in the "Add a manual device" input field, followed by the IP address of the device and the port number. The device will appear in the device list of the SimpleMenu and the connection can be established.



Now you can see the variables you added to monitor. To start the simulation/monitoring click on the start button.



Now you will see the changes of these signals.

8.5 Stand-alone mode and autostart

8.5.1 Enable the stand-alone mode

The Baby-LIN-MB-II is able to operate stand-alone without a PC, PLC or operator. To enable this mode several requirements need to be met:

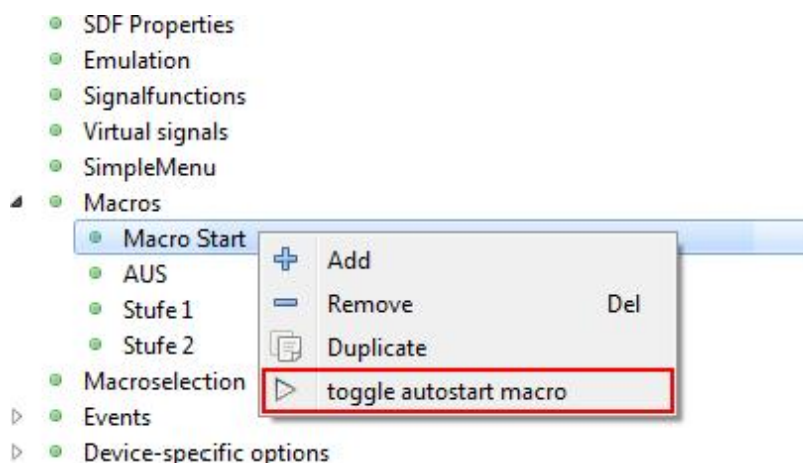
- You need a SDF with a macro that is marked as "autostart" macro.
- The SDF needs to be stored persistently on the Baby-LIN-MB-II.
- The device needs to be configured to automatically start the autostart macro of a persistently stored SDF when powered up.

8.5.2 Configure the autostart macro

You can mark a macro in a SDF as autostart macro. This means that this macro is started automatically when the SDF is loaded. Usually this macro will start the LIN-Bus communication and perform necessary initialisations.

To mark a macro as autostart macro the following steps are necessary.

- Open your SDF using the SessionConf.
- If this SDF does not already have a macro that you want to use as autostart macro please create one.
- Right click on the macro and select "toggle autostart macro". The macro will now have the "[autostart]" marker.



Advice

Please note that only one macro can be marked as autostart macro.



Tip

You probably want to start the LIN-Bus within the macro since it is not automatically started by loading the SDF.

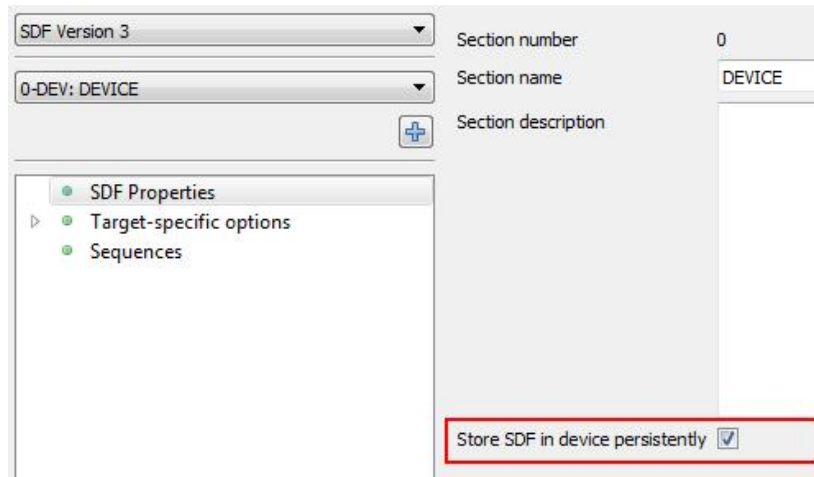
8.5.3 Store a SDF persistently

The following applies to the Baby-LIN-MB-II:

- All SDFs stored on the Baby-LIN-MB-II are saved persistently. There is no need to configure the device or SDF in a special way. The option "Store SDF in device persistently" has no effect for the Baby-LIN-MB-II.

The following applies to all other devices:

- It is possible to store a SDF persistently on the Baby-LIN-MB-II. Therefore a special option in the SDF must be configured:
 - Start the SessionConf and load your SDF.
 - In the left menu click on "SDF Properties".
 - Check the "Store SDF in device persistently" option.



- Save your SDF and reload it to a device using the SimpleMenu.



Advice

The option "Store SDF in device persistently" can always be found in the "SDF Properties". It is independent from the SDF version (SDF v2, SDF v3) or type of section currently selected. This option is global for the whole SDF even though it can be found in every section.

8.5.4 Configure the device to automatically load a SDF and start a macro

The Baby-LIN-MB-II can be configured to automatically start a autostart macro. The Autostart can be triggered if the device is powered on or if the LIN-Bus is powered on. These target-specific options (also known as target configuration) are persistently stored on the Baby-LIN-MB-II.

There are 2 important options that affect the autostart feature:

Target-specific option	Possible values	Description
Autostart	Off	No schedule or macro is started when the autostart is triggered.
	Schedule	The first schedule but not the autostart macro is started when the autostart is triggered.
	Schedule + Macro	The first schedule and the autostart macro are started when the autostart is triggered.
	Macro only	No schedule but the autostart macro is started when the autostart is triggered.
Autostart on LIN Power	Off	An autostart is triggered only when the device is powered on.
	On	An autostart is triggered each time the LIN-Bus is powered on.



Advice

If you want to start another but the first schedule simply select the macro only option and start another than the first schedule from that macro.

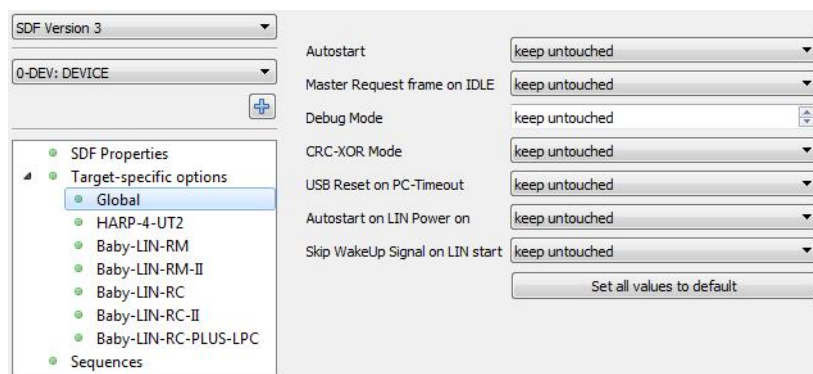

Tip

Triggering the autostart each time the LIN-Bus is powered on is especially helpful for applications in which the node connected to the LIN-Bus is changed periodically, e.g. EOL applications. In such a case, the simulation may be started automatically every time a new node is connected and the LIN-Bus voltage is turned on.

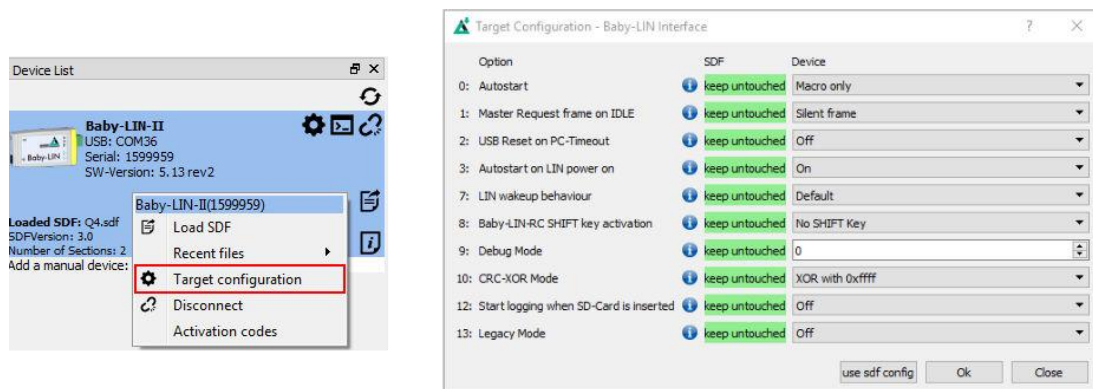

Warning

The option "Autostart on LIN Power on" is not required for the Baby-LIN-MB-II if the LIN voltage is also used as the logic supply.

SDF can store preset values for the target-specific options. These preset values can be found in the device section. These values overwrite the values in the device each time the SDF is uploaded. If a value is set to "keep untouched" the value in the device will not be changed.



The target-specific options can be set using the SimpleMenu. Right click on the connected device and click on "Target configuration". In the following dialog you can change the values in the device. For comparison reasons the preset values of the SDF are shown.



9 LINWorks Software - Overview

The LINWorks is a collection of software to operate the Baby-LIN-MB-II. The complete LINWorks software package is available for download on our website. There you will also find the LINWorks Software Manual, which gives a detailed overview of the individual program and how to work with and create Session Description Files.

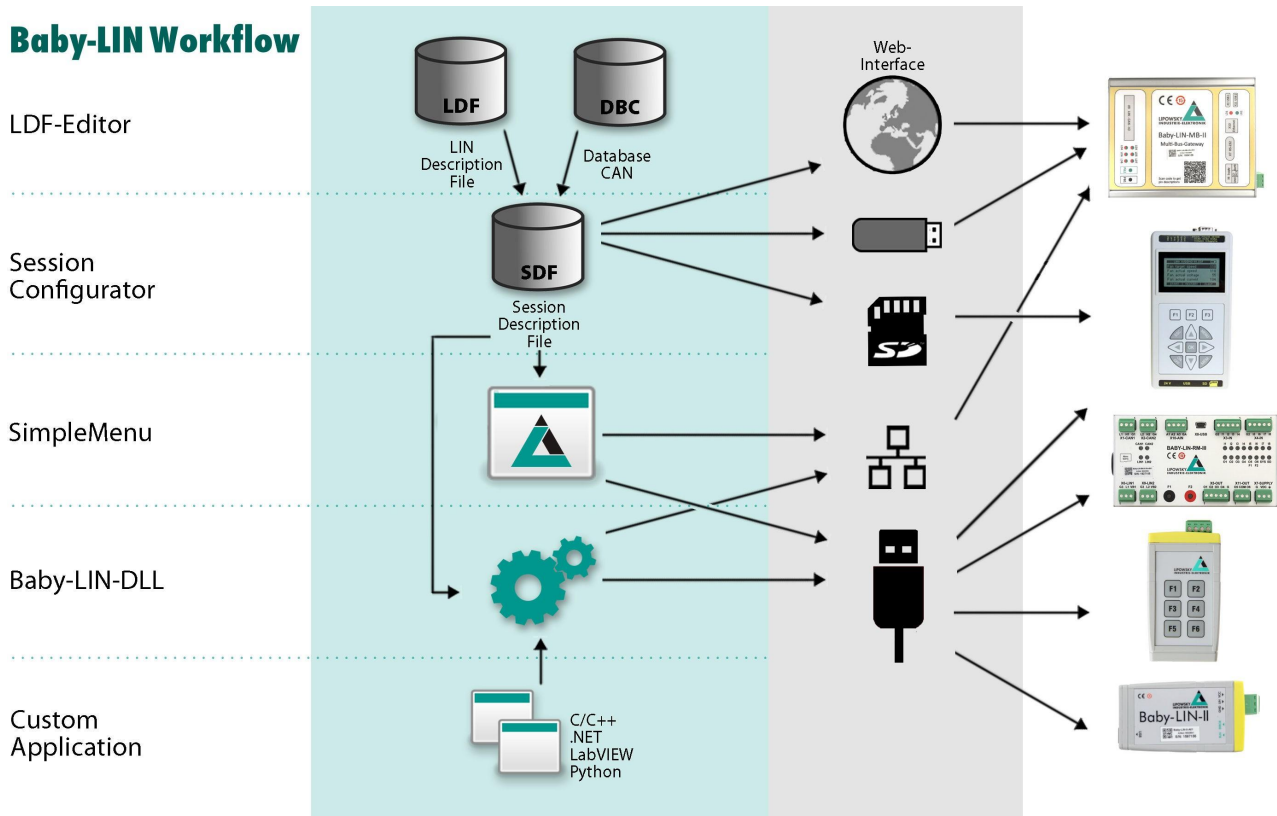
You can download both from the following link: <https://www.lipowsky.de/downloads/>



Component	Archive subfolder	Description
LINWorks Documentation	Documentation	The datasheets show a quick overview about a Baby-LIN product and its features. The user manuals contain the main documentation. The application notes contain some older information, that have not been added to the user manuals yet.
Baby-LIN driver		The Baby-LIN driver is necessary to connect a Baby-LIN-MB-II to a windows PC via USB. The Baby-LIN-MB-II will be available as virtual COM port .
LDFedit	LINWokrs	The LDFedit allows the inspection, creation and edit of a LDFfile (LIN Description File).
SessionConf	LINWokrs	The SessionConf allows the inspection, creation and edit of a SDFfile (Session Description File) and features a file import for LDFfiles (for LIN-Bus simulation). It defines everything needed for a complete simulation of each available bus, e.g. which nodes on each bus are available and which nodes should be simulated by the Baby- LIN-RC-II. Moreover it allows defining an application logic. This programming ability is available for each device out of the box.
SimpleMenu	LINWokrs	The SimpleMenu is used to establish a connection to the Baby-LIN-MB-II and upload SDFfiles, change the device target configuration, control the bus and monitor the frames and signals on the bus. Even without a LDFfile/ SDFfile the bus can be monitored and the frames can be logged.
LogViewer	LINWorks	The LogViewer can show and convert the log files of the Baby-LIN-MB-II as well as the SimpleMenu.
Baby-LIN-DLL	Development	The Baby-LIN-DLL allows customers to create their own application and use all features of the Baby-LIN-MB-II like controlling and monitoring the LIN-Bus interfaces. The Baby-LIN-DLL is a native C/C++ DLL. It is available for Windows, Linux and RaspberryPi . Wrapper for .NET, Python, VB6 and LabView are available. Of course we provide examples for all supported languages.
Serial writer	Tools	The serial writer is used to change the serial number , that is stored within the persistent memory of a Baby-LIN-MB-II . This serial number influences the allocation of the virtual COM port number, the Baby-LIN-MB-II is available under.
BLProg	Tools	The BLProg is used to update the firmware of a Baby- LIN-RC-II. If you download a firmware package from our customer portal (portal.lipowsky.de) a current version of the BLProg will always be included.
Baby-LIN-MB-Tool	Tools	The Baby-LIN-MB-Tool allows to access many features of the Baby-LIN-MB-II. It can help to search and identify Baby-LIN-MB-IIs, change the network configuration and select different modes. Scripts using the ASCII command protocol can be executed, debugged and logged. The simulation mode allows to simulate certain behaviours of the Baby-LIN-MB-II to test custom applications. Additionally the Baby-LIN-MB-Tool features many different logging capabilities.

The following graphic shows how you can use our LINWorks software in connection with our the Baby-LIN-Devices.

Baby-LIN Workflow



10 Migration information

10.1 Migration from Baby-LIN-MB-II to Baby-LIN-MB

All Baby-LIN products of the second generation were designed to be compatible with the first generation. Due to hardware and software changes, the compatibility may be affected in certain areas.

If you have used a Baby-LIN-MB in your environments and applications and now want to replace it with a Baby-LIN-MB-II, the following chapters give you an overview of the topics, you have to consider.



Version incompatibility

Each of the following chapters may decrease the compatibility depending on your application and the way, you use the Baby-LIN-MB-II. Therefore you should check all these chapters very carefully.

10.2 Performance

The Baby-LIN products of the second generation are in common more powerful.

The faster and more powerful CPU executes operations faster and therefore allows more operations in the same time interval. The higher memory allows to download bigger SDF files into the Baby-LIN-MB-II.

The SDF-V3 format allows to use new powerful features within the SDF file.

	Baby-LIN-MB	Baby-LIN-MB-II
CPU	ARM9, 180 MHz	ARM Cortex-A5, 528 MHz
Memory	64 MB SD-RAM	256 MB DDR-RAM
SDF-Version	SDF-V2	SDF-V2 & SDF-V3

10.3 LIN-Bus transceiver



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The used LIN-Bus transceiver has changed. The following table shows you, what properties have changed:

Baby-LIN product	Baby-LIN-MB	Baby-LIN-MB-II
LIN-Bus transceiver	Si9241A	MC33662
Maximum LIN-Bus baud rate	200 kBaud	125 kBaud
Minimum LIN-Bus voltage	3.8 V	6.9 V
Maximum LIN.Bus voltage	36 V	26 V

10.4 Changed connectors



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The Baby-LIN-MB-II features different connectors than the Baby-LIN-MB. This was necessary since the Baby- LIN-MB-II offers a lot more connections. If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you have to change your wiring.

The easier solution is to use the Baby-LIN-MB compatibility adapter. This adapter offers you the connectors of the Baby-LIN-MB.

10.5 SDF file names



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The Baby-LIN-MB-II runs on an embedded Linux system, which gives a lot of opportunities to implement additional network and file system features. This operating system change has an influence on the processing of SDF file names.

The file names are no longer reduced to the 8.3 system. The string before the dot can be longer than 8 characters and the extension is no longer limited to 3 characters.

The Baby-LIN-MB-II processes the file names case sensitive. That means that 2 files looking similar on the Baby-LIN-MB or a PC, could be 2 different files on the Baby-LIN-MB-II.

10.6 Real-time clock

The Baby-LIN-MB-II integrates a hardware-based real-time clock (RTC) with battery backup. This clock is used by the Linux operating system as well as the Baby-LIN-MB-II firmware.



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The command `RTCWrite` was used at the Baby-LIN-MB to set the clock. This clock lost its value each time, the device was not powered. Therefore the clock had to be set with this command. Since the Baby-LIN-MB-II keeps its clock value, even if it is not powered.

For compatibility reasons the command is still available, but it will not change the clock.

10.7 Power on/off behaviour



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The Baby-LIN-MB-II has an integrated UPS (uninterruptible power supply) that allows the save shut down of the system during power fail events or keeps the system running on short power drops. Check chapter "UPS - Uninterruptible power supply" for more information.

The drawback of this feature is the missing possibility to reset the system by cutting the supply voltage for a short moment, which worked on the Baby-LIN-MB.

As a replacement for this method, the system offers a reboot feature, which can be triggered by pressing both buttons PB1 and PB2 together for at least 5 seconds. Check chapter "PB1, PB2 - Push buttons" for more information.

10.8 MB-II LinSivRsp Busy



Version incompatibility

If you want to replace a Baby-LIN-MB with a Baby-LIN-MB-II you should check the following chapter, since this issue reduce the compatibility depending on your application and the way, you use the Baby-LIN-MB-II.

The Baby-LIN-MB-II features a new API for the "ASCII host command protocol". By default it is deactivated. Your old command scripts from the Baby-LIN-MB will run.

If you want to use the new `CmdDone` API, you should check the chapter "API modes" in combination with the "Command LinSivResp". Since the new `CmdDone` API is using the answer `:B` to show, that a command ist still busy, the "Command LinSivResp". will use `:I` as answer instead. Your existing parsing components then must be updated to take that behaviour into account.

10.9 SDF versions: SDF-V3 and SDF-V2

SDFiles contain the configuration to setup all Baby-LIN products. New features are continuously developed and the format therefor changes.

The first generation of Baby-LIN products supported all features up to the main format version 2. This format is called SDF-V2.

With the second generation of Baby-LIN products new features were added to the SDFfile format, that require the higher performance of the new devices. The format with these new features is called SDF-V3.

Since new features are added continuously to the SDFfile format, you should always keep your LINWorks and firmware up to date. Check chapter

"Update philosophy" for more information.

10.9.1 Compatibilities

All Baby-LIN products support the SDF-V2 format.



Attention

The SDF-V3 format must be unlocked on some devices using an optional voucher code..



Advice

If the Baby-LIN product is only needed to replace an old device or extend an old installation, the SDF-V3 option is not required.

The following table gives an overview over all Baby-LIN products and their support for SDF-V2 and SDF-V3:

Baby-LIN product generation	Baby-LIN product	SDF-V2	SDF-V3
Baby-LIN products Generation I	Baby-LIN	✓	
	Baby-LIN-RC	✓	
	Baby-LIN-RM	✓	
	Baby-LIN-MB	✓	
	HARP-4	✓	✓ (optional)
Baby-LIN products Generation II	Baby-LIN-II	✓	✓
	Baby-LIN-RC-II	✓	✓
	Baby-LIN-RM-III	✓	✓
	Baby-LIN-MB-II	✓	✓
	HARP-4	✓	✓

The format of a SDF file is visible and can be changed in the SessionConf. The conversion between the formats has the following rules:

Source format	Target format	Feasibility
SDF-V2	SDF-V3	This is always possible. After the change SDF-V3 features can be used.
SDF-V3	SDF-V2	This conversion is only possible, if no SDF-V3 features are used. If this is the case, the SessionConf will show a list of incompatible elements.

10.9.2 Section

Some of the Baby-LIN products of the second generation feature more than just one LIN-Bus interface. Multiple LIN-Bus interfaces are available as well as the new CAN-Bus interfaces. The content of a SDF-V2 SDF file will be mapped into a LIN section within a SDF-V3 SDF file.

The new CAN section supports the same features as a LIN section with minor changes due to the differences of the LIN- and CAN-Bus.

The new device section is a container for features, that are not specific to a channel, but the device in common. The main feature of the device section is, that it may contain the target configuration for a device. This means, that the target-specific options may be configured simply by loading a SDF file. Check chapter "Target configuration and target-specific options" for more information.

SDF file format	Content
SDF-V2	The description of a single LIN interface. This is equivalent to a LIN section in a SDF-V3 SDF file.
SDF-V2	Multiple sections: <ul style="list-style-type: none"> • A device section. • Any number of LIN sections. A LIN section is based on a LDF file. • Any number of CAN sections. A CAN section is based on a DBC file or a ARXML file.

10.9.3 Target-specific options

The target-specific options are a set of options for a Baby-LIN product. They can be set using the SimpleMenu. Now with SDF-V3 they can also be stored within a SDFFile.

The target-specific options within a SDF-V3 SDFFile can be applied, when it is downloaded:

Download method	Effect
Download using the SimpleMenu	The user will be queried, if the target-specific options from the SDFFile should be applied. The user can change the target-specific options manually.
Download using the Baby-LIN-DLL	The target-specific options are applied automatically.

Check chapter "Target configuration and target-specific options" for more information.

10.9.4 Names

The length of the element names within a SDFFile have been increased.

Element in the SDFFile	Length in a SDF-V2 SDFFile	Length in a SDF-V3 SDFFile
(Section) Description	450	4096
Element names	40	64

10.9.5 Emulation

Each emulated frame in a LIN section can now be configured to set unused bits to 1 instead of 0.

The emulation of CAN section differs slightly from the emulation of a LIN section:

- Each node can be configured to have on of three states:

Node state	Description
Off	The frame is not sent and its reception is not checked. The frame will still be visible to the user.
Monitored	The frame is expected to be received. If the timeout is exceeded, an error will be created.
Emulated	The frame will be sent with the given cycle time.

- Each frame can be configured to have an individual state based on the node state:

Node state	Frame state	Description
Off	Off	The frame is not sent and its reception is not checked.
Monitored	Off	The reception of the frame is checked based on the given timeout.
	Monitored	The reception of the frame is checked based on the given timeout.
Emulated	Off	The frame is not sent and its reception is not checked.
	Emulated	The frame is sent with the given cycle time.

10.9.6 Virtual signals and system variables

The size of virtual signals can now be increased to 64 bits.

The behaviour of the virtual signals on bus start can now be changed. Now it is possible to reset virtual signals, when the bus is resetted.

Virtual signals can now also be interpreted as signed variables.

Element	Length in a SDF-V2 SDFFile	Length in a SDF-V3 SDFFile
Size of virtual signals	1 - 16	1 - 64
Size of system variables	16	32
Reset on Bus start	yes	configurable
Sign	unsigned	configurable

New system variables have been added. Since they are device specific, their usage does not depend on the format of the SDFFile.

10.9.7 Signalfunctions

New AUTOSAR CRCs have been added to the signalfunctions. The CRC is calculated according to the AUTOSAR standard. Both, profile 1 and profile 2 are available.

All CRC signalfunctions have more properties to support custom deviations from the standard CRC calculation.

10.9.8 Macros

New macro commands have been added to improve the programming and reduce the number of required commands.

A conditional macro command execution has been added. Each macro command can now be configured to have a condition, e.g. a comparison between two signals. The macro command is only executed, if the condition is true.

A new feature is the call of a macro as sub macro. The calling macro blocks until the sub macro is finished. By passing arguments and a return value mechanism, the user can now define reusable functions.

With the addition of conditions, functions and new macro commands new powerful sequences can be realised.

