ModusToolbox™ LIN Configurator user guide

LIN Configurator version 2.0

About this document

A newer revision of this document may be available on the web here.

Scope and purpose

The Local Interconnected Network (LIN) Configurator is a stand-alone tool included with the ModusToolbox™ software. LIN is a low-cost serial communication standard used for various electronic components inside automobiles.

Intended audience

This document helps application developers understand how to use the LIN Configurator as part of creating a ModusToolbox™ application.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus
Italics	Denotes file names and paths.
`Monospace`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
File > New	Indicates that a cascading sub-menu opens when you select a menu item

Abbreviations and definitions

The following define the abbreviations and terms used in this document:

- API – application programming interface
- LIN – local interconnected network
- LDF – LIN Description File
- NCF – Node Capability File describes a LIN slave node as seen from the LIN bus
- NAD – network address
- MRF – multiple reference frame
- SRF – single reference frame

Reference documents

Refer to the following documents for more information as needed:

- VS Code for ModusToolbox™ user guide
- Eclipse IDE for ModusToolbox™ user guide
- ModusToolbox™ tools package user guide
- Device Configurator user guide
- Device datasheets
- Device technical reference manuals
- LIN Middleware API reference guide

Table of contents

About this document . . . 1

Table of contents . . . 2

1 Overview . . . 3

1.1 Supported middleware . . . 3

2 Launch the LIN Configurator . . . 4

2.1 make command . . . 4

2.2 VS Code and Eclipse IDE . . . 4

2.3 Executable (GUI) . . . 4

2.4 Executable (CLI) . . . 4

3 Quick start . . . 5

4 Code generation . . . 6

5 GUI description . . . 7

5.1 Toolbar . . . 7

5.2 Multi-Instance support . . . 8

5.3 Parameter subtabs . . . 8

6 Parameters description . . . 9

6.1 General subtab . . . 9

6.2 Frames subtab . . . 10

6.3 Signals subtab . . . 12

6.4 Transport Layer subtab . . . 15

6.5 Configuration Services subtab . . . 17

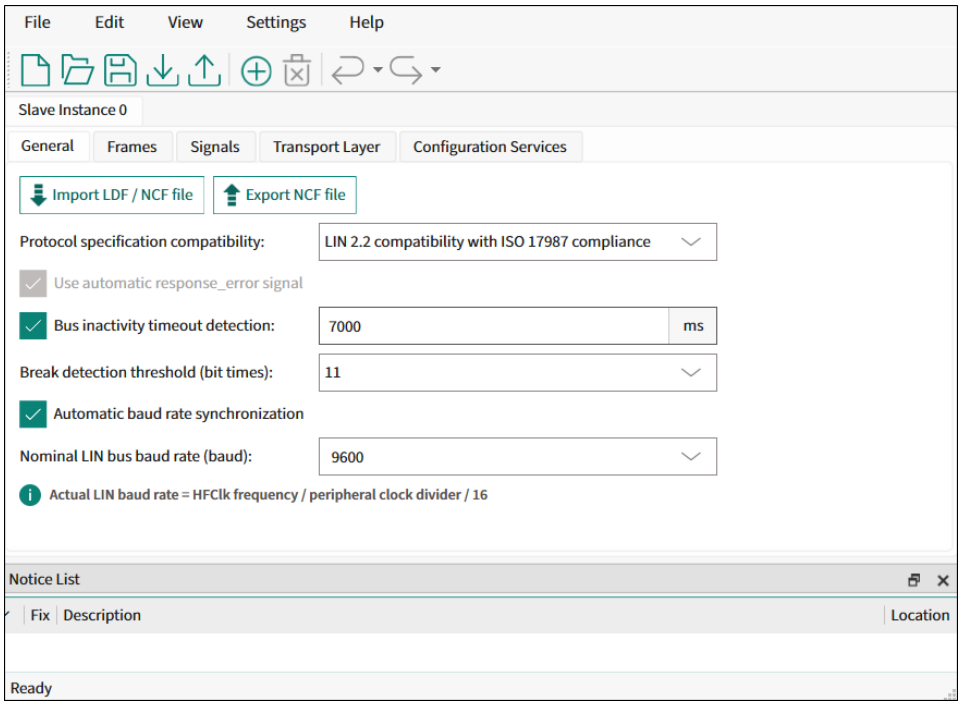
7 Version changes . . . 19

Revision history . . . 20

Disclaimer . . . 21

1 Overview

The Local Interconnected Network (LIN) Configurator is used to generate build-time configuration for the LIN middleware. The tool is supported on Windows, Linux, and macOS.

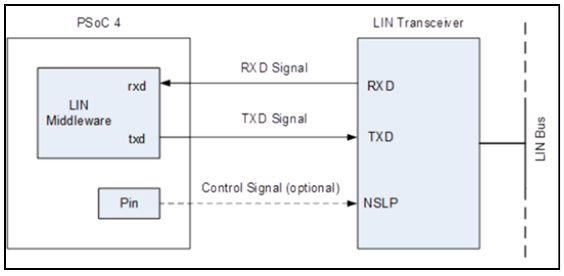


1.1 Supported middleware

The LIN Configurator middleware uses configuration setups created in the LIN Configurator GUI or with external applications.

The LIN middleware implements a LIN 2.2 and ISO17987-compliant slave node on PSOC™ 4 devices.

Hardware interface between PSOC™ 4 and LIN Bus



Name	Version	Link
LIN Middleware library	2.0	https://infineon.github.io/lin/html/index.html

2 Launch the LIN Configurator

There are several ways to launch the LIN Configurator, and those ways depend on the application, and how you use the various tools.

2.1 make command

As described in the ModusToolbox ™ tools package user guide "ModusToolbox™ build system" chapter, you can run numerous make commands in the application directory, such as launching the LIN Configurator. After you have created a ModusToolbox™ application, navigate to the application directory and type the following command in the appropriate bash terminal window:

```
make lin-configurator
```

This command opens the LIN Configurator GUI for the specific application in which you are working.

2.2 VS Code and Eclipse IDE

VS Code and Eclipse have tools to launch the LIN Configurator from within an open application. Refer to the applicable user guide for more details:

- VS Code for ModusToolbox™ user guide
- Eclipse IDE for ModusToolbox™ user guide

2.3 Executable (GUI)

You can launch the LIN Configurator GUI by running its executable as appropriate for your operating system (for example, double-click it or select it using the Windows **Start** menu). By default, the configurator is installed here:

<install_dir>/ModusToolbox/tools_<version>/lin-configurator<version>

When launched this way, the LIN Configurator opens without any frames or signals configured (see Overview). You can either open a specific configuration file or create a new one. See **Menus** (under GUI description) for more information.

Launching the Configurator this way is not recommended as part of application development.

2.4 Executable (CLI)

The LIN Configurator executable can be run from the command line and it has a "cli" version of the executable as well. Running the executable from the command line can be useful as part of batch files or shell scripts to re-generate the source code based on the latest configuration settings. The exit code for the executable is zero if the operation is successful, or non-zero if the operation encounters an error. For more information about the command-line options, run the executable using the `-h` option.

3 Quick start

This section provides a simple workflow for how to use the LIN Configurator.

1. Launch the LIN Configurator .
2. Configure parameters as required.
3. Save the configuration. See Code generation.
4. Use the generated structures from *mtbcfg_lin.h* and *mtbcfg_lin.c* files as input parameters for functions in your application.

4 Code generation

The LIN Configurator generates header (*mtbcfg_lin.h*) and source (*mtbcfg_lin.c*) files that contain relevant firmware used by the LIN middleware configuration and operation. The generated .h and .c files are located in the *GeneratedSource* folder next to the *. *mtblin* file which contains the user configuration.

Refer to the LIN Middleware Library for more information about this code.

5 GUI description

The LIN Configurator GUI contains menus, tabs and subtabs to configure the LIN settings and a Notice List to provide indications.

Menus

File

- **New** – Creates a new file with new configuration.
- **Open...** – Opens the configuration file.
- **Close** – Closes the configuration file.
- **Save** – Saves the existing configuration.
- **Save As...** – Saves the existing file under a different name.
- **Open in System Explorer** – Opens your computer's file explorer tool to the folder that contains the *.modus file.
- **Import...** – Imports a specified configuration file.
- **Export...** – Exports the current configuration file into a specified file.
- **Recent files** – Shows recent files that you can open directly.
- **Exit** – Closes the Configurator.

Edit

- **Add new instance** – You can start a new instance by either clicking this button or the same button on the Multi-instance support.
- **Delete current instance** – You can remove the current instance by either clicking this button or the same button on the Multi-instance support.
- **Undo** – Undoes the last action or sequence of actions.
- **Redo** – Redoes the last undone action or sequence of undone actions.

View

- **Notice List** – Shows/hides Notice List. The pane displays by default.
- **Toolbar** – Shows/hides the toolbar.
- **Reset View** – Resets the view to the default.

Settings

- **Analytics...** – Opens the "Analytics Permission" dialog

Help

- **View Help** – Opens this document.
- **About LIN Configurator** – Opens the About box for version information, with links to open https://www.infineon.com and the current session log file.

Notice List

The **Notice List** pane combines notices (errors, warnings, tasks, and notes) from many places in the configuration into a centralized list. If a notice shows a location, you can double-click that entry to display the parameter causing the error or warning.

For more information about the Notice List, refer to the Device Configurator user guide.

5.1 Toolbar

The toolbar contains common commands from the **File** and **Edit** menus. Use the check box under the **View** menu to show or hide the toolbar. See GUI description for details.



5.2 Multi-Instance support

The LIN Configurator supports multiple slave instances to configure different settings for each instance. By default, the Configurator contains only Slave Instance 0. The Configurator includes commands to add and remove instances.



5.3 Parameter subtabs

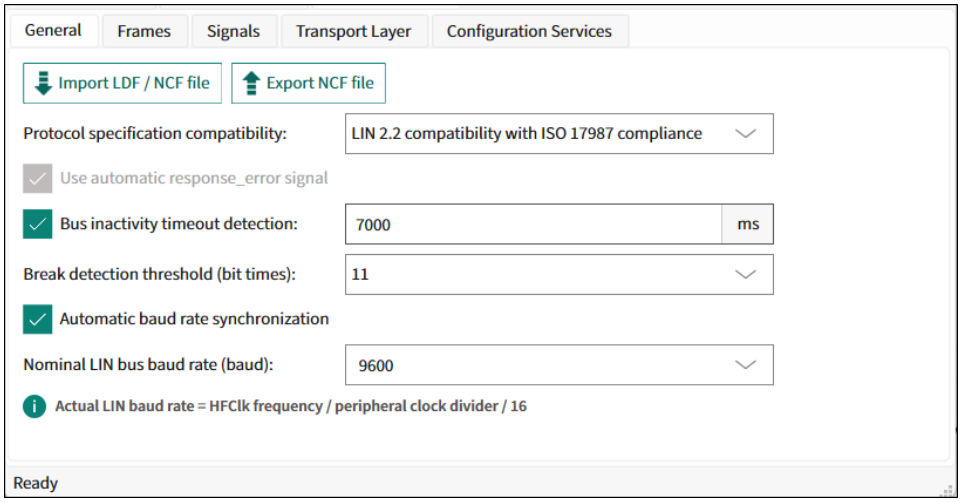
Each slave instance contains the following subtabs to configure various settings:

- Parameters description
- Frames subtab
- Signals subtab
- Transport Layer subtab
- Configuration Services subtab

6 Parameters description

6.1 General subtab

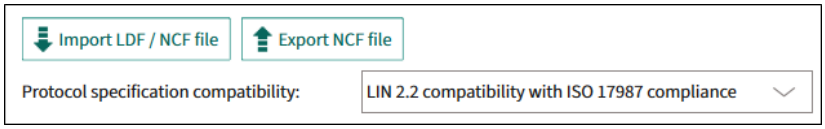
The **General** subtab allows general configuration of the LIN resource. The subtab looks as follows:



6.1.1 General subtab parameters

Slave instance properties

The LIN Configurator supports importing NCF and LDF files to configure the LIN slave node and export the current configuration into an NCF file. Also, the Configurator automatically configures signals based on LDF/NCF files supplied by the user.



Protocol specification compatibility

Supports the functionality to select from the drop-down menu:

- LIN 2.2 compatibility with ISO 17987 compliance
- LIN 2.2 compatibility without ISO 17987 compliance

Use automatic response_error signal

Enables the response_error signal option (response error notification to the LIN master).

Select this option to add the “RE” signal on the Unplaced Signals pane on the **Signals** subtab.

Bus inactivity timeout detection

Select this option to automatically detect the LIN bus inactivity threshold with a free-running timer. The valid range is 4000 ms to 10000 ms.

Break detection threshold (bit times)

Select this option to configure a slave-node break detection threshold. The default value is 11 dominant local-slave bit times. It can be set to either 10 or 11 or 12.

***Note**: For the P4000S, P4100S, P4100S Plus, and PSOC™ 4500S series, the actual threshold can be up to one bit less than the specified threshold. For example, if the value of the Break detection threshold option is 11, the actual threshold will be from 10 to 11 bits.*

Automatic baud rate synchronization

Enabled by default. You can disable this option.

- Enabled – The middleware measures the exact baud rate of the bus from the sync byte field of each LIN frame header.
- Disabled – Instead of measuring, the middleware receives the sync byte field as a 0x55 data byte.

***Note**: Select this option for successful communication within the cluster if a slave's clock variance is greater than ±1.5 percent.*

***Note**: The baud rate synchronization is only used to match one nominal LIN bus baud rate chosen for any given LIN cluster. It is not used to synchronize to any valid baud rate.*

Nominal LIN Bus Baud Rate

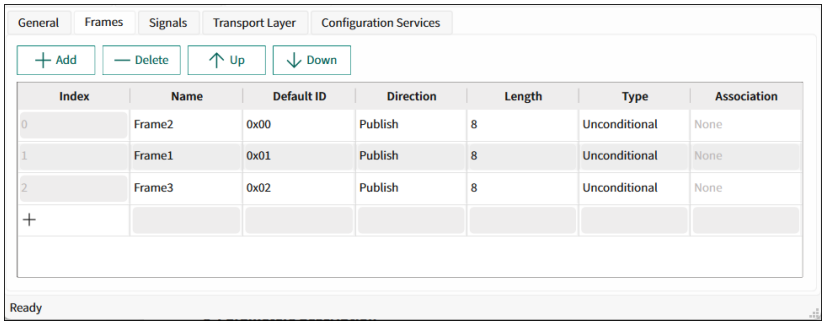
Enter the **Nominal LIN bus baud rate (baud)** at which this slave node must operate. The valid range is 1000 to 20000 baud. The values in the drop-down list are 2400, 9600, 10417, and 19200. Also, you can enter any value between 1000 and 20000 here.

The Configurator shows the formula to calculate the actual baud rate:

Actual LIN baud rate = HFClk frequency / peripheral clock divider / 16)

6.2 Frames subtab

The **Frames** subtab is used to configure how the LIN slave responds to PID values sent by the master on the bus. The **Frames** subtab contains a toolbar to navigate frames and frame configuration table.



***Note**: The **Frames** subtab does not show diagnostic frames.*

6.2.1 Frames toolbar

The **Frames** subtab toolbar contains four buttons.



- **Add** – You can start a new frame on the table by either clicking the **Add** button or the **+** button at the end of the **Index** column.

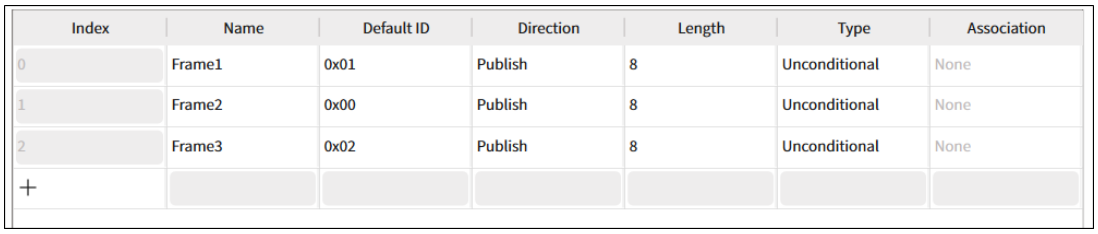
- **Delete** – Removes the currently selected frame from the table. The index number fields are changed accordingly.

  ***Note**:* *If a frame is deleted on this subtab, any signal configured on the **Signals** subtab and placed into it is relocated onto the **Unplaced Signals** pane.*

- **Up / Down** – Reorder the **Index** number values for each frame.

6.2.2 Frames configuration table

The table contains eight columns.



Index	Name	Default ID	Direction	Length	Type	Association
0	Frame1	0x01	Publish	8	Unconditional	None
1	Frame2	0x00	Publish	8	Unconditional	None
2	Frame3	0x02	Publish	8	Unconditional	None
+						

- **Index** – Displays the ordering number of every used frame. These numbers cannot be directly modified.

  You can start a new row by clicking the **Add**or **+** button at the end of the **Index** column.
- **Name** – Displays the unique name of each frame. To change the name, click it.
- **Default ID** – Defines the frame ID, which the frame will use before any configuration request by the master. Note that the LIN master can reconfigure frame IDs at run time. Enter a value from 0x00 to 0x3B in the hex format into these cells.
- **Direction** – Provides two drop-down options for each frame to select the direction to send the data for the frame:
  - **Publish** – Data transmission.
  - **Subscribe** – Data reception.
- **Length** – Defines how many bytes are received or sent for each frame. The valid range is 1 - 8.
- **Type** – Provides two drop-down options for each frame to select the frame type:
  - **Unconditional**
  - **Event-triggered**

  For more details, refer to **Association**.

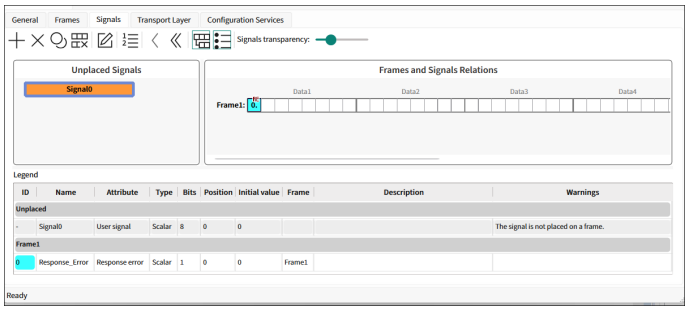
  ***Note**:* *You cannot choose the **Event-triggered** for a **Subscribe** frame. If you change this cell **Event-triggered** to **Unconditional**, you must change the value of this frame to "None" in the **Association** column, if its name displays in any cells in that column.*

- **Association** – Used to relate unconditional frames with event-triggered frames. Allows the selection of the frame name of any unconditional frames that are not already associated with an event-triggered frame.

  An event-triggered frame must have at least one conditional frame associated with it. The valid values for this setting are the names of any existing unassociated unconditional frames. Only one unconditional frame can be associated with an event-triggered frame. As a result, when one of these cells has the name of an unconditional frame in it, this unconditional frame will no longer display as an option in any other row.

6.3 Signals subtab

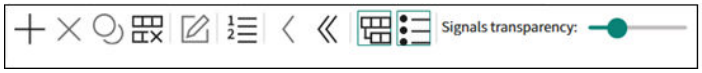
The **Signals** subtab is used to specify the parameters of signals added into the LIN frames.



The subtab toolbar and panes – Unplaced Signals, Frames and Signals Relations, Legend – are described in the following sections.

6.3.1 Signals toolbar

The Signals toolbar contains buttons to configure and navigate signals:



- **Add signal** – Puts a new signal on the Unplaced Signals pane. The Signal Properties dialog displays after clicking this button.
- **Delete signal** – Removes the selected signal. A warning will display first.
- **Clone signal** – Duplicates the selected signal.

  ***Note**:* *If the duplicated and original signals have the same name, a warning about the parameters and location of such signals will display.*

- **Delete all signals** – Removes all signals. A warning will display first.
- **Signal properties** – The Signal Properties dialog of the selected signal displays after clicking this button.
- **Renumber signals' IDs** – Reorders the signals index values.
- **Move signal to unplaced** – Relocates the selected signal to the Unplaced Signals pane.
- **Move all signals to unplaced** – Relocates all signals to the Unplaced Signals pane.
- **Show/hide event-triggered frames** – Shows or hides event-triggered frames.
- **Show/hide Legend** – Shows or hides the Legend pane.
- **Signals transparency** – This slider sets the transparency for signals graphics.

6.3.2 Unplaced Signals

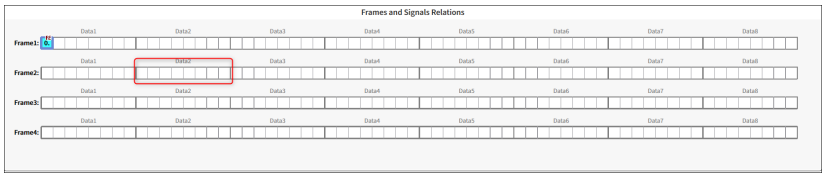
The Unplaced Signals graphical pane serves as temporary to store added signals. Signals can be moved back and forth between the Unplaced Signals pane and the Frames and Signals Relations pane.

***Note**:* *If a frame is deleted on the **Frames** subtab, any signals added to it (configured with the **Signals** subtab) are moved into the Unplaced Signals pane.*

6.3.3 Frames and Signal Relations

The Frames and Signal Relations graphical pane of the **Signals** subtab displays interactive graphics of the configured frames and signals.

Each frame graphic represents a frame added (configured) on the **Frames** subtab.

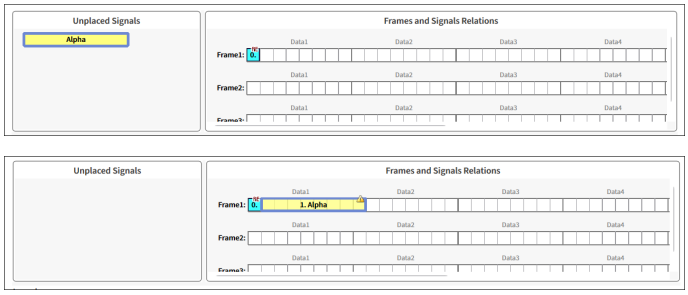


One frame length can be any from 1 to 8 data bytes. Each byte contains eight 1-bit cells: Data1 through Data8. The total number of frames cannot exceed 60. The total size of all frames is limited to 256 bytes.

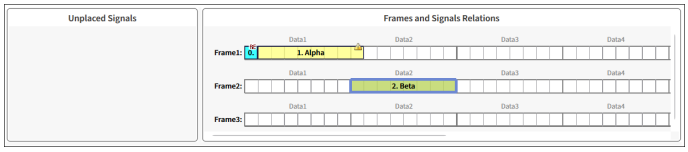
Each signal graphic represents one signal defined for the LIN slave. The graphic for a signal appears as a solid bar. A signal is placed onto the frames using the drag-and-drop. Placed signals occupy bits or bytes of the frames. Clicking a signal symbol selects that signal. Double-clicking a signal symbol opens the Signal Properties dialog.

6.3.4 Graphical Interaction between Unplaced Signals and Frame and Signals panes

On the interactive graphical panes of the **Signals** subtab, you can select signals by clicking them, then, drag and drop them onto the frames:

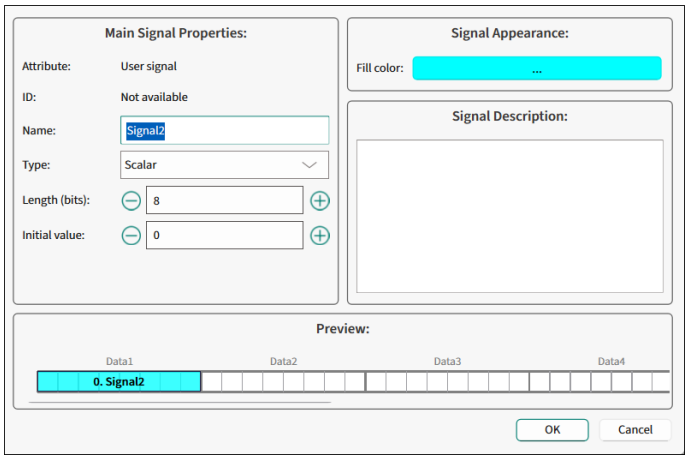


The graphical panes may look as follows depending on the selected signals' parameters and graphic color:



6.3.5 Signal Properties

The "Signal Properties" dialog contains various parameters to configure signals.

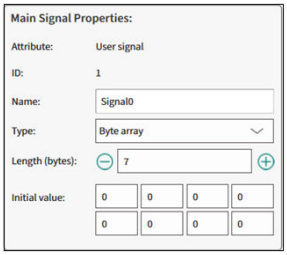


The "Signal Properties" dialog displays if you click the **Add** button on the toolbar to configure and add a new signal or double-click an already configured and added signal's symbol to edit its configuration.

Main Signal Properties

- **Attribute** – The attribute value cannot be modified. The field displays the following values depending on the various signal configuration:
  - **User signal** – The default value.
  - **Response error** – Enable the **Use automatic response_error_signal option** on the **General** subtab. To display it on the "Signal Properties" dialog, click on the "RE" symbol and click the Signal properties button on the **Signals** subtab toolbar.
- **ID** – The ordering number of a signal placed on a frame. Cannot be directly modified.
- **Name** – The default signal name is Signal0, where "0" is equal to the signal index number. The signal name can be changed but it must contain only letters, digits, and underscores.
- **Type** – There are two types of signals:
  - **Scalar** – A signal of 1- to 16-bit length
  - **Byte array** – A signal of 1- to 8-byte length.

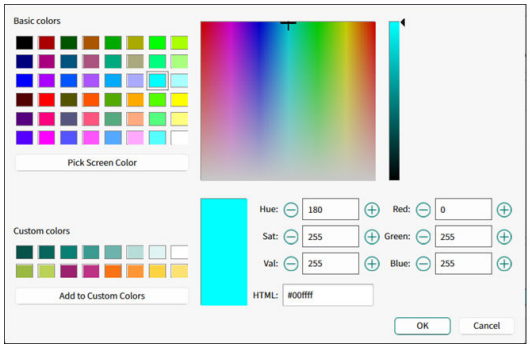
    If the Byte array type is selected, the Initial value parameter displays as many fields for entering values as there are bytes in the Length parameter:



- **Length (bits)** – The length of a signal: 1-16 bits for Scalar signals; 1- 8 bytes for Byte array signals.
- **Initial value** – The initial value must be entered in the decimal format.

Signal Appearance

Fill color – Clicking the color bar displays the dialog where to select the signal graphic color and its parameters.



Signal Description

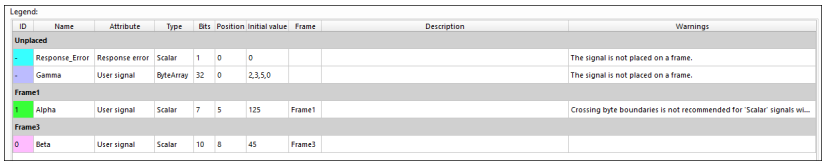
Any signal-related description or other information entered by the user.

Preview

The Preview graphical pane shows what the signal will look like when it is added.

Legend

The Legend pane describes the properties of configured signals.

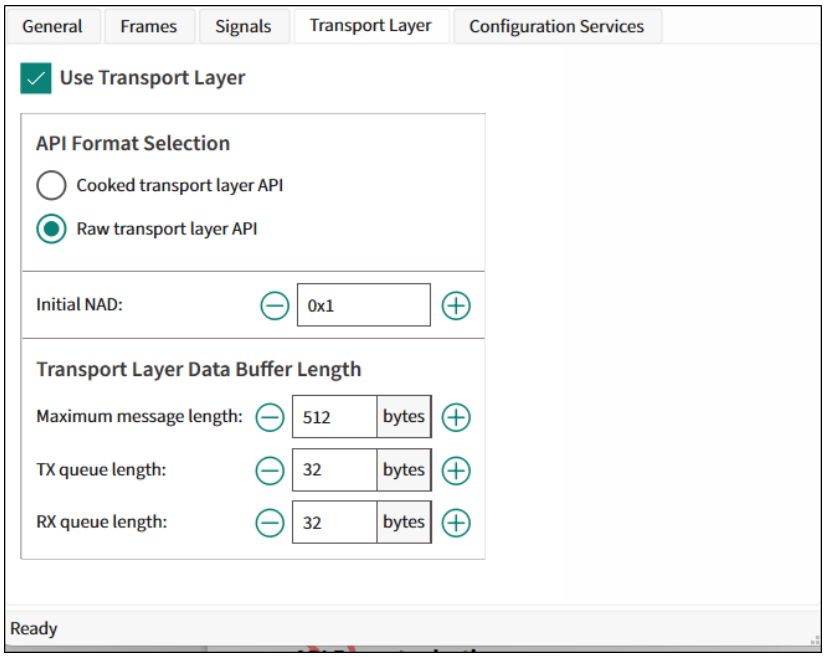


The descriptions of the Legend pane columns – ID, Name, Attribute, Type, Bits (Length (bits)), Initial Value, Signal Description – are available on the **Main Signal Properties** pane of the "Signal Properties" dialog as they are properties of the same configured signals. Also, the Legend pane consists of the following columns:

- **Position** – Indicates the start bit where a signal is located on a frame. The range is 0-63.
- **Frame** – The name of the frame where a signal is located.
- **Warnings** – These are notices related to the configured signals.

6.4 Transport Layer subtab

The **Transport Layer** subtab contains parameters to configure the API and data buffer length.



***Note**: The transport layer itself defines transportation of data contained in one or more frames. See the LIN 2.2 or ISO 17987 specification for detailed information on the Transport Layer.*

Use Transport Layer

- If you select this option, a warning displays to ensure that the values entered on the **Configuration Services** subtab are correct.
- If you deselect this option, a warning displays that the **Automatic Configuration Request Handling** option will also be disabled.

API Format selection

This section is used to select the format for the Transport Layer API functions:

- **Cooked transport layer API** – Used to send and receive Transport Layer messages. It is the user's responsibility to ensure that only one API function is used for each message.
- **RAW transport layer API** – Used to send or receive each frame that makes up a Transport Layer message using one API function call for each frame.

Initial NAD

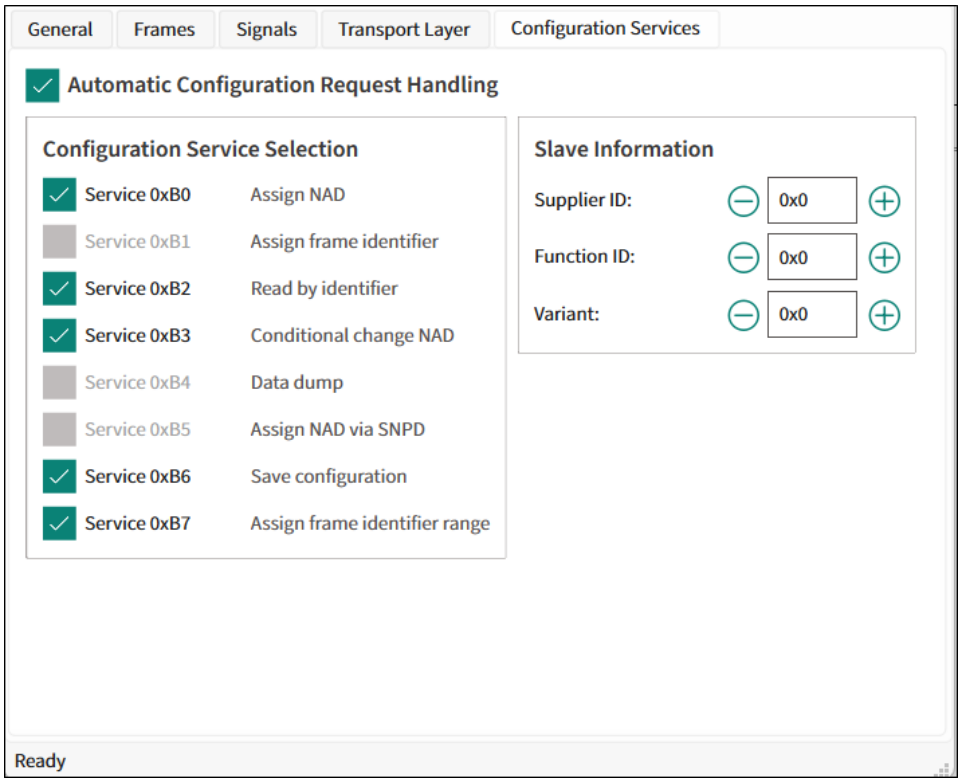
Used to select the Network Address (NAD) of the slave node. Each slave node has an initial NAD list. The NAD is used in MRF and SRF frames to address one particular slave node in a cluster. Note that this field is used to select the initial NAD for the node. The NAD of a slave node can change at run time.

Transport Layer Data Buffer Length

- **Maximum message length (bytes)** – Used to select the maximum Transport Layer message length that this slave node supports. The minimum value is 6. The LIN Configurator only supports Transport Layer messages with lengths up to 4095 bytes.
- **TX queue length / RX queue length (bytes)** – Applicable when the Raw Transport Layer API format is selected. The LIN Configurator supports queue lengths from 8 to 2048 with 8-byte steps. The default size of each queue is 32 bytes.

6.5 Configuration Services subtab

The **Configuration Services** subtab contains a check box list of eight services and the slave parameters.



Configuration Service selection

The LIN Configurator supports eight configuration service requests (0xB0 to 0xB7) to select for automatic handling:

- **Service 0xB0** – Assign NAD – An optional service per LIN 2.2 and ISO17987 specifications. This request is used to assign a new NAD value to the slave node.
- **Service 0xB1** – Assign frame identifier – The obsolete service per LIN 2.2 and ISO 17987 specifications.
- **Service 0xB2** – Read by identifier – A mandatory service per LIN 2.2 and ISO 17987 specifications. This request is used to allow the LIN master to read the slave's identification information (Supplier ID, Function ID, and Variant). This middleware only supports the LIN Product Identification version of this request.
- **Service 0xB3** – Conditional change NAD – An optional service per LIN 2.2 specifications. Also, this service is deprecated (optional) in ISO 17987-3.
- **Service 0xB4** – Data dump – The optional service per LIN 2.2 and ISO 17987 specifications, not supported by this middleware.
- **Service 0xB5** – Assign NAD via SNPD – This service is not supported per LIN 2.2 and ISO 17987 specifications.
- **Service 0xB6** – Save configuration – An optional service request per LIN 2.2 and 17987 specifications.
- **Service 0xB7** – Assign frame identifier range – A mandatory configuration service per LIN 2.2 and ISO 17987-3 specifications. This service allows the LIN master to change the volatile frame PID values for the slave's frames.

Slave information

Select the Automatic Configuration Request Handling option and three fields become available to enter values used in the configuration service request:

- **Supplier ID** – A 16-bit value but its valid range is 0x0000 to 0x7FFE.

***Note**: The supplier ID default value is 0x00C3.*

- **Function ID** – A 16-bit value, its valid range is 0x0000 to 0xFFFE.
- **Variant** – An 8-bit value, its valid range is 0x00 to 0xFF.

6.5.1 Automatic Configuration Request Handling

All the other options on the subtab are available only if the Automatic Configuration Request Handling option is selected. Deselect this option to handle these requests with your own custom application code. Whenever any automatically-handled request occurs during the LIN bus operation, the corresponding MRF and SRF frames will not be available to the application through the Transport Layer API. If this option is not selected on this subtab, then the corresponding MRF and SRF frames of the configuration service request are received or sent by the application using the Transport Layer API.

7 Version changes

This section lists and describes the changes for each version of this tool.

Version	Change descriptions
1.0	New tool.
1.10	Added Multi-Instance support.
	Added the Import LDF/NCF file and Export NCF file options.
1.20	Changed the device library file from xml to *props.json*.
1.21	Fixed GUI issues, which were truncated label text on the **Transport Layer** and **Configuration Services** tabs.
1.30	Various back-end changes.
1.40	Various back-end changes.
1.50	Various back-end changes.
1.60	Minor back-end changes.
1.70	Minor back-end changes.
1.80	Minor back-end changes.
1.90	Minor back-end changes.
2.0	GUI style update.
	Added dynamic RX and TX buffer size support.
	Updated the default signal placement strategy to better match typical LIN frame configurations.
	Added Analytics collection.

Revision history

Revision	Date	Description
**	2021-04-23	1.0 New tool.
*A	2021-09-22	Updated to version 1.10.
*B	2022-09-28	Updated to version 1.20.
*C	2023-05-16	Updated to version 1.21.
*D	2024-01-23	Updated to version 1.30.
*E	2024-01-24	Minor changes.
*F	2024-09-27	Updated to version 1.40.
*G	2024-11-04	Updated device names "PSoC" to all UPPERCASE "PSOC".
*H	2024-12-05	Updated to version 1.50.
*I	2025-03-21	Updated to version 1.60.
*J	2025-08-13	Updated to version 1.70.
*K	2025-12-11	Updated to version 1.80.
*L	2026-03-21	Updated to version 1.90.
*M	2026-08-28	Updated to version 2.0.

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2026-08-28

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2026 Infineon Technologies AG

All Rights Reserved.

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-jhj1712340789176

Important notice

Products which may also include samples and may be comprised of hardware or software or both ("Product(s)") are sold or provided and delivered by Infineon Technologies AG and its affiliates ("Infineon") subject to the terms and conditions of the frame supply contract or other written agreement(s) executed by a customer and Infineon or, in the absence of the foregoing, the applicable Sales Conditions of Infineon. General terms and conditions of a customer or deviations from applicable Sales Conditions of Infineon shall only be binding for Infineon if and to the extent Infineon has given its express written consent.

For the avoidance of doubt, Infineon disclaims all warranties of non-infringement of third-party rights and implied warranties such as warranties of fitness for a specific use/purpose or merchantability. Infineon shall not be responsible for any information with respect to samples, the application or customer's specific use of any Product or for any examples or typical values given in this document.

If this document is marked as 'Preliminary', 'Target', 'Draft' or 'Proposal', Infineon reserves the right to change all information given in this document at any time without notice. Subject to the development and release of a Product for series supply by Infineon, the technical specifications of the Product are set forth in the relevant datasheet provided by Infineon without such marking.

The data contained in this document is exclusively intended for technically qualified and skilled customer representatives. It is the responsibility of the customer to evaluate the suitability of the Product for the intended application and the customer's specific use and to verify all relevant technical data contained in this document in the intended application and the customer's specific use. The customer is responsible for properly designing, programming, and testing the functionality and safety of the intended application, as well as complying with any legal requirements related to its use.

Unless otherwise explicitly approved by Infineon, Products may not be used in any application where a failure of the Products or any consequences of the use thereof can reasonably be expected to result in personal injury.

However, the foregoing shall not prevent the customer from using any Product in such fields of use that Infineon has explicitly designed and sold it for, provided that the overall responsibility for the application lies with the customer. Infineon expressly reserves the right to use its content for commercial text and data mining (TDM) according to applicable laws, e.g. Section 44b of the German Copyright Act (UrhG). If the Product includes security features:

Because no computing device can be absolutely secure, and despite security measures implemented in the Product, Infineon does not guarantee that the Product will be free from intrusion, data theft or loss, or other breaches ("Security Breaches"), and Infineon shall have no liability arising out of any Security Breaches.

If this document includes or references software:

The software is owned by Infineon under the intellectual property laws and treaties of the United States, Germany, and other countries worldwide. All rights reserved. Therefore, you may use the software only as provided in the software license agreement accompanying the software.

If no software license agreement applies, Infineon hereby grants you a personal, non-exclusive, non-transferable license (without the right to sublicense) under its intellectual property rights in the software (a) for software provided in source code form, to modify and reproduce the software solely for use with Infineon hardware products, only internally within your organization, and (b) to distribute the software in binary code form externally to end users, solely for use on Infineon hardware products. Any other use, reproduction, modification, translation, or compilation of the software is prohibited.

For further information on the Product, technology, delivery terms and conditions, and prices, please contact your nearest Infineon office or visit www.infineon.com.