LAUTERBACH
DEVELOPMENT TOOLS

LIN Bus Protocol Analyzer

Release 02.2024

MANUAL

LIN Bus Protocol Analyzer

LIN Bus Protocol Analyzer

Version 04-Mar-2024

General Function

LIN PROTOanalysis requires one of the following hardware configurations:

- PowerIntegrator with single-ended Probe
- PowerTrace-II / PowerTrace-III with single-ended Probe
- PowerProbe

Two target connections have to be done:

- LIN signal
- GND

ATTENTION:

The signal voltage of the LIN bus can have values up to 10.8 V which exceeds the maximal allowed voltage of the TRACE32 probes. Precautions have to be taken to avoid probe damages. In case of overvoltage the probes must not be connected direct to the target.

The PROTOanalyzer decodes the whole transmission, considering LIN bus specification version 2.0 and 2.1 (http://www.lin-subbus.org).

The results of the Protocol Analysis are time correlated to other TRACE32 tools like a program flow trace. This way there is a close linkage between LIN activity and program-execution and vice versa.

Features

The LIN PROTOanalyzer supports LIN version 2.0 as well as 2.1 including the associated formats of the LIN Description File (LDF).

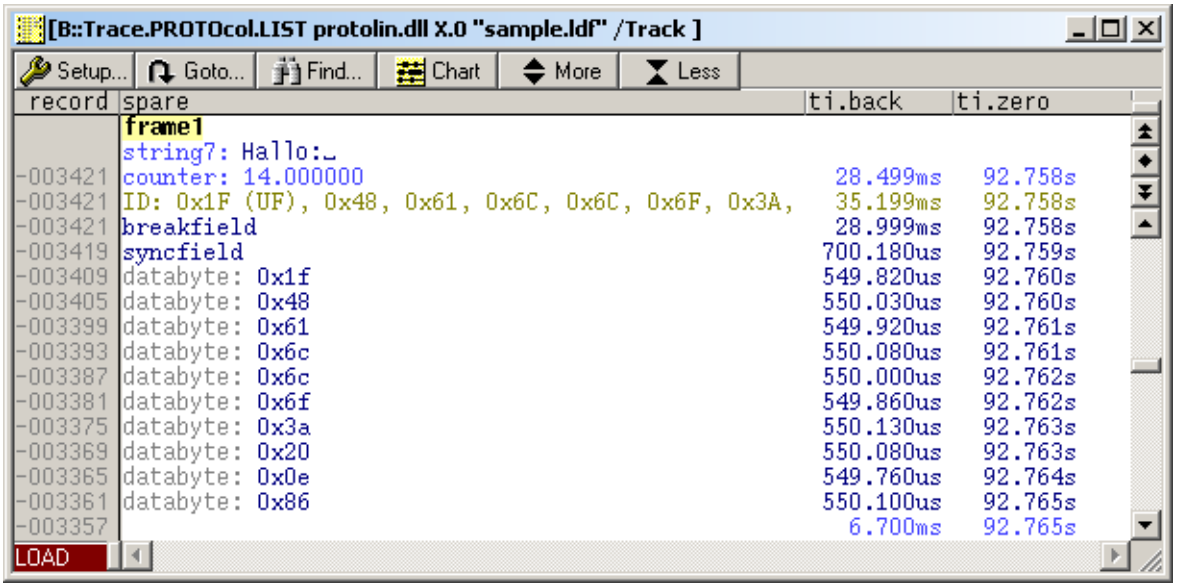
After successful parsing the LDF file and processing the input data, all detected LIN frames are displayed by their defined names. Included signals and their encoding is displayed as well if they are defined in the LDF file.

PDU packets are detected and displayed by the addressed slave node name. Also configuration services are detected and interpreted (see **"Restrictions of Configuration Services"** (linbus_app.pdf)).

Errors at frame level (wrong checksum value, missing databytes, wrong PID etc.), at byte level (wrong value of stopbit, undetermined bit value) and at PDU level (wrong PDU sequence, configuration service unpredictable etc.) are detected and displayed.

The chart display shows payloads of correctly detected frames with their defined names. It also shows erroneous frames and unknown bus traffic.

Configuration services ('LIN Node configuration and Identification Specification', http://www.lin-subbus.org) are supported with restriction that under certain circumstances (intended) side effects could not be determined. E.g.: If configuration service 'Conditional change NAD' is used with an identifier that could not be determined using the data specified in the LDF.
Please refer to **"Restrictions of Configuration Services"** (linbus_app.pdf).



There are three different levels of display. With the buttons "More" and "Less" you can deactivate/activate the lower levels. Level 3, which displays the frame name and defined signals, can't be deactivated.

How to use the PROTOanalyzer

The PROTOanalyzer requires a LIN specific file (‘protolin.dll’) which contains the algorithm to analyze and display the recorded LIN transmission.

The source code and the ‘protolin.dll’ file are part of the installation CD. The code matches the TRACE32 Protocol Analysis interface. It is open for user modifications and therefore could be changed to fit special needs.
Please refer to the ‘doxygen’ documentation inside the source code (http://www.doxygen.org/index.html).

Literature on TRACE32 installation CD:

- **protocol_app.pdf**
- **protocol_user.pdf**

Format:	*<trace>*.**PROTOcol.LIST** *<proto_dll> <channel> <ldf>* [*<enc_display>*] [*<composite_config>* [*<jitter_sync_field_div>* [*<max_spike_time>*]]]
<proto_dll>:	file name of DLL including path
<channel>:	input channel of LIN signal
<ldf>:	file name of LinDescriptionFile (string)
<enc_display>:	**SHOWnoenc** or **HIDEnoenc** (optional, default=SHOWnoenc)
<composite_config>:	composite configuration name (optional, string)
<jitter_sync_field_div>:	value controls break/sync detection (optional, integer, default=200.)
<maxs_pike_time>:	time parameter for spike detection (optional, default = 1 µs)

Example command line:

```
Probe.PROTOcol.LIST .\analysis\protoLIN.dll x.0 "sample.ldf" show "" 100.
1us
```

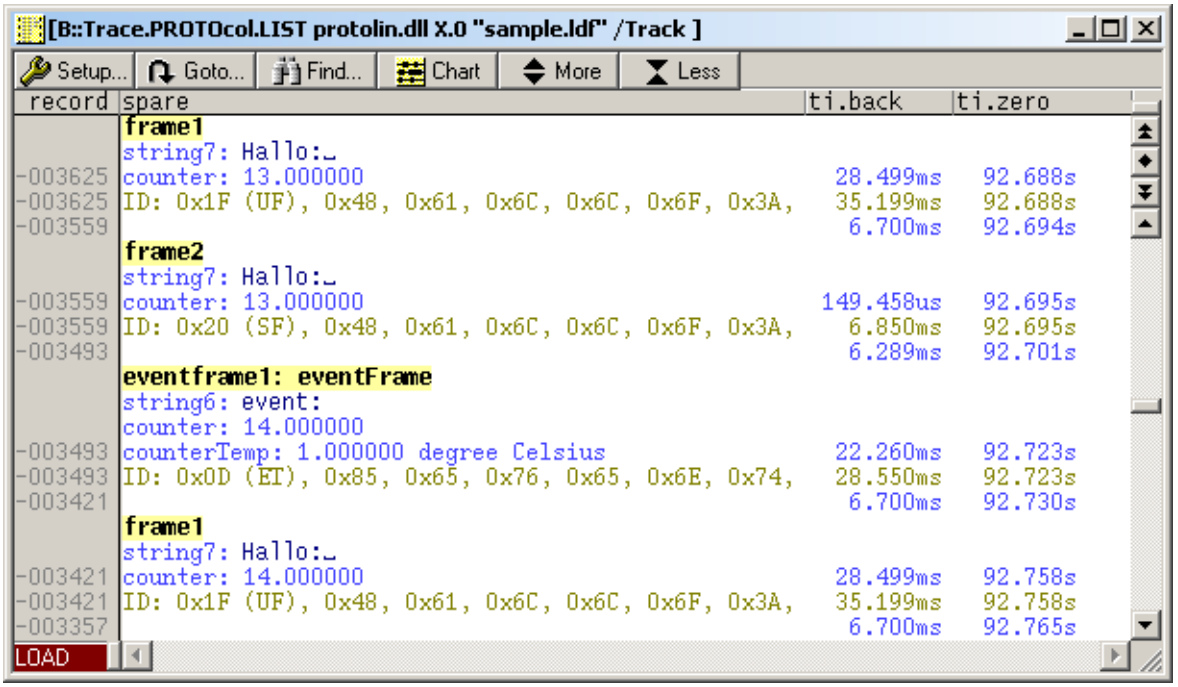
- *<ldf>* specify the LIN Description File relatively to the current directory. If a LDF is unavailable, you can also specify an empty string. Although only raw data bytes will be displayed in this case

The following parameters are optional and are needed only to optimize functionality:

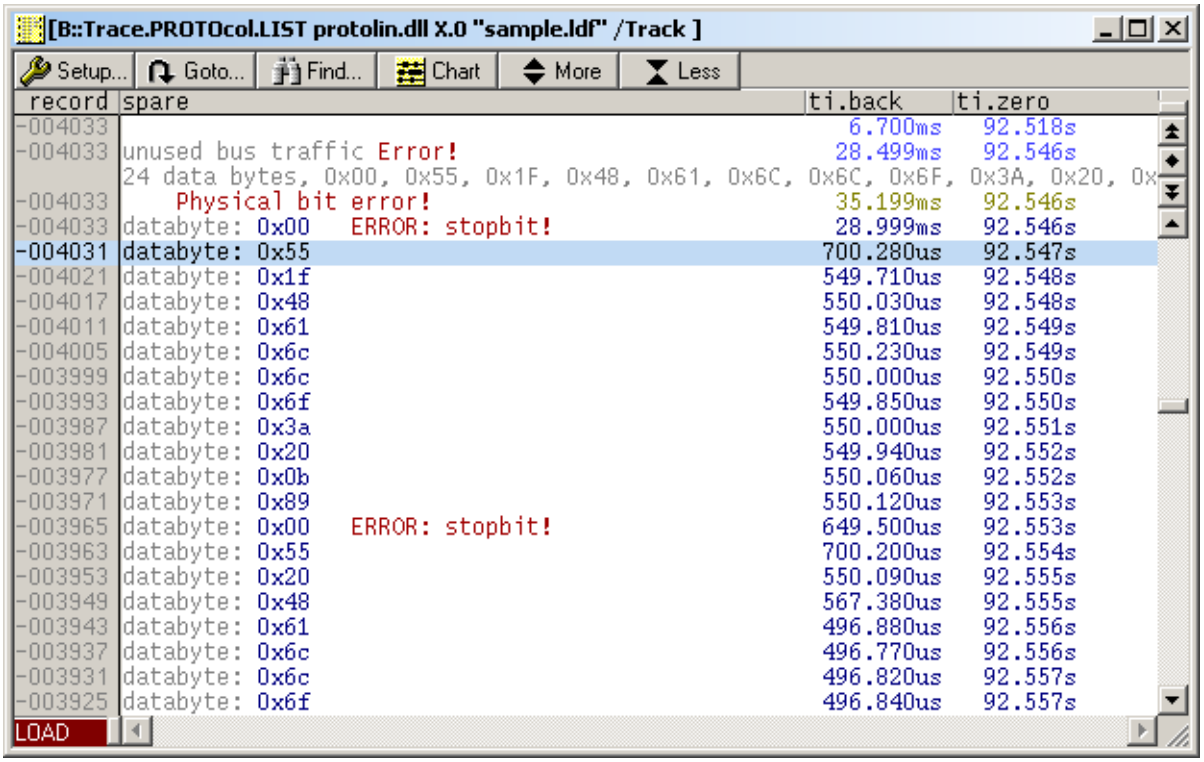
- *<enc_display>* specifies whether signal without defined encoding should be displayed as “signalname: no encoding defined!”. Using option “HIDEnoenc” allows you to show only signals which are defined in the LDF.
- *<composite_config>* allows to select a “composite configuration” inside the LDF. If you don’t need this feature please specify an empty string.
- *<jitter_sync_field_div>* is used to bypass some inaccuracy detecting levels of the LIN signal. To detect break/sync fields correctly the time differences between falling edges of the sync field are compared. These differences shall differ only to 0.5% which corresponds (with 1/*<jitter_sync_field_div>*) to *<jitter_sync_field_div>* = 200. If some break/sync fields are not detected correctly but detected as two bytes with value 0x00 (causing a stopbit error) and value 0x55 you can try to decrease *<jitter_sync_field_div>*. (see also second screenshot)
- *<max_spike_time>* is used for spike detection. After each (detected) break/sync field, the threshold for spike detection is recalculated to be 2/16 bit time. Every signal level connected shorter than the threshold value is considered to be a spike. With *<max_spike_time>* you can set the start value of this threshold.

Example Screenshots

This screenshot shows level 3 and level 2 analysis.



This screenshot shows an example of wrong break/sync field detection => decrease *<jitter_sync_field>*.



Explanation of LIST Display

In case of empty or wrong output, please check the messages of the LDF parser. These are displayed in the message AREA window.

At startup only display level 3 is active. With the buttons “More” and “Less” display level 1 and 2 can be enabled/disabled.

Byte Level (Level 1)

At this level all data bytes, break fields and sync fields are displayed independently from any frame interpretation.

Error Messages

“ERROR: stopbit!”: Indicates the detection of a wrong stopbit value. This could happen if a previous break/sync field sets a new baudrate but was not detected by the PROTOAnalyzer or if the current break/sync field was wrongly discarded. To avoid the later case please refer to *<jitter_sync_field>* in **“Further Information on Parameters”** (linbus_app.pdf).

“ERROR: undef bit value!“: The bit value could not be determined because the logic value toggled between earliest bit sampling time and latest bit sampling time.

Frame Level (Level 2)

At this level data bytes of frames are bundled and the following verifications are done:
- correct parity bits of PID field
- frame with this ID defined
- frame complete
- correct checksum

In case of an event triggered frame this verifications are also done:
- correct parity bits of PID of triggered frame (first data byte)
- this ID corresponds to a triggered frame (of this event triggered frame)

The two characters enclosed in parentheses after the ID display indicate the frame type:

- “NA“ = frame type not available
- “UC“ = unconditional frame (normal frame)
- “SF” = sporadic frame
- “ET” = event triggered frame
- “DF” = diagnostic frame

If data bytes after the end of frame (or after the ID field if no frame is defined with this ID) were detected, these are bundled and displayed as “unused bus traffic”.

Error Messages

"Physical bit error!": Error in byte level.

"PID error!": Parity bits doesn't match ID.

"Frame incomplete, x databyte(s) + checksum are missing!": Their are missing x databytes and the checksum byte before the next break field or respectively before the end of recorded traffic.

"ID not defined!": ID is (currently) not assigned with a frame name. Please check LDF and previous configuration services (PDU level) which could reassign IDs.

"Checksum error!": Checksum byte has wrong value. This could happen because of some wrong detected databyte or the wrong checksum type is assumed. The later could be caused by a misconfigured LDF (publisher and subscriber nodes of 'Signals' definition).

"Event triggered frame has no triggered frame defined! ": No triggered frame is defined in LDF inside the 'Event_triggered_frames' definition of this frame.

"Event triggered frame PID (x) error!": Parity bits don't match ID.

"Frame ID (0x%02hhX) not added to event triggered frame!": Frame with ID is corresponds not to a triggered frame of this event triggered frame.

PDU Level (Level 3)

The main purpose of this level is to show the frame names, to collect the frames which corresponds to one PDU and to display the "header" of a PDU as well as its side effects in case of a configuration service.

If traffic without frame basis was detected it is displayed as "unused bus traffic".

Error Messages

"Error!": Some error occurred in previous levels.

"PDU sequence error!": This PDU was received without a context.

"PDU sequence error at frame x of PDU!": The xth frame of a PDU was the last correctly received.

Messages after Configuration Services

"not defined": This diagnostic request/response is not implemented in PROTOanalyzer.

It is possible that the traffic later on will be interpreted wrongly!

"negative response": A negative response (to last request) was detected.

“unknown side effect": The PROTOanalyzer detects a positive request. (with a side effect), but the responding node could not be reconstructed

It is possible that the traffic later on will be interpreted wrongly!

“unexpected response (no request detected)": No request detected which causes this response.

It is possible that the traffic later on will be interpreted wrongly!

“(SF, x bytes)”: This is a single frame with x valid databytes.

"(FF, x bytes)": This is the first frame of a multi frame PDU with x valid databytes.

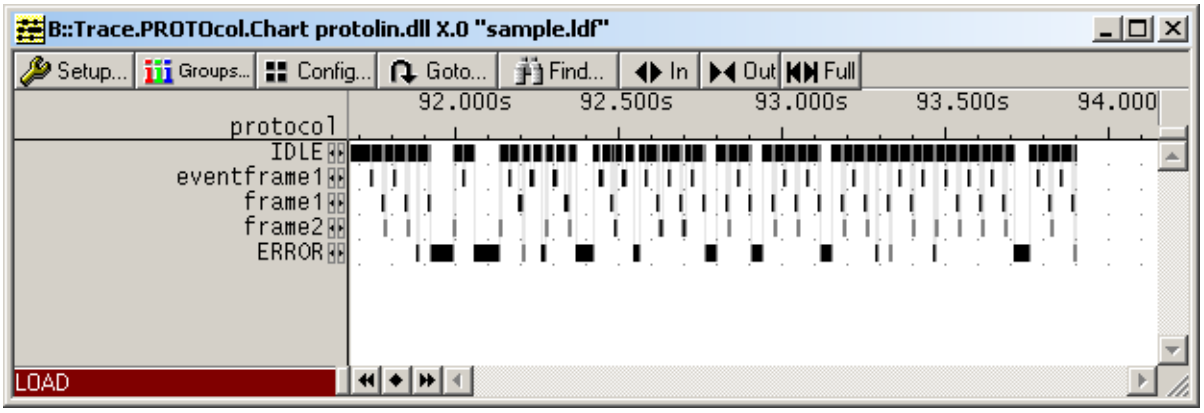
"(CF, no. x)”: This is a consecutive frame with number x which was detected out of sequence of a PDU.

“PCI code unknown!”: The PCI field has some unknown bit combination.

Format	*<trace>*.**PROTOcol.CHART** *<protodll> <channel> <ldf>* [*<encdisplay>*] [*<jittersyncfielddiv>* [*<maxspiketime>*]]

The command format has the same structure as above. Only the BaseCommand is different.

This visualization shows the duration of every transmission, when the bus is in idle state as well as the duration of frames with errors and unknown bus traffic. It's very useful for checking the traced transmissions for any errors or rare frames.



<trace>.STATistic Statistic visualization

Format	*<trace>*.**PROTOcol.STATistic** *<protodll> <channel> <ldf>* [*<encdisplay>*] [*<jittersyncfielddiv>* [*<maxspiketime>*]]

The command format has the same structure as above. Only the BaseCommand is different.

This visualization shows the share of every transmission type. It's very useful for checking the transmission timing. This way critical timings can be found easily.

Restrictions of Configuration Services

- ‘Conditional change NAD’ can only be determined if the data byte which is used to specify the slave is stored in the LDF.

  This is the case with databytes extracted from:

  - LIN Product Identification (Supplier ID, Function ID, Variant)
  - Message ID 1..16 (MessageID, PID) (in case of a version 2.0 slave node)

  This is **not** the case for databytes extracted from:

  - Serial Number
  - User-defined Interpretations

  In the special case that the extracted databyte could not determined but only one slave node matches the NAD and their was a positive response, this node is guessed to be the node addressed by this service.

- If any side effect could not determined (e.g because some prior NAD changing service have not been interpreted and therefore the service uses an “unknown” NAD from protocol analyzer point of view) the message “unknown side effect” is displayed.
- ‘Assign NAD via SNPD’ is not supported.

Restriction of LDF Parser

Generally every part of the LDF which is not needed by PROTOanalyzer is skipped and not interpreted. If a whole block (e.g. ‘Schedule_tables’) is skipped no syntax checking is done.

Skipped/not Interpreted Parts of LDF

Skipped global definitions:

- ‘Channel_name’
- ‘Schedule_tables’
- ‘Signal_groups’ (deprecated since version 2.0)
- ‘dynamic_frames’

Skipped within ‘Node_attributes’ definition:

- ‘response_error’
- ‘fault_state_signals’
- ‘P2_min’
- ‘ST_min’
- ‘N_As_timeout’
- ‘N_Cr_timeout’

Skipped within ‘Sporadic_frames’ definition:

- ‘sporadic_frame_name’

Skipped within ‘Event_triggered_frames’ definition:

- ‘collision_resolving_schedule_table’