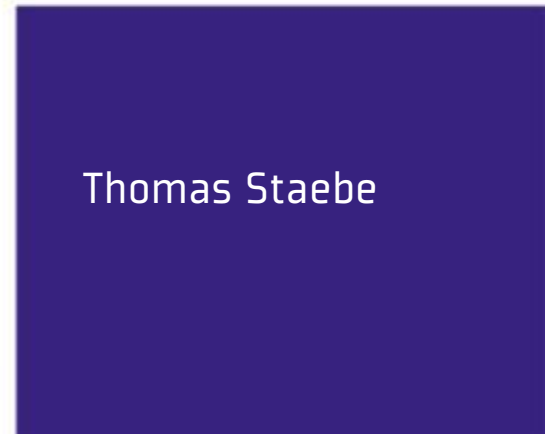
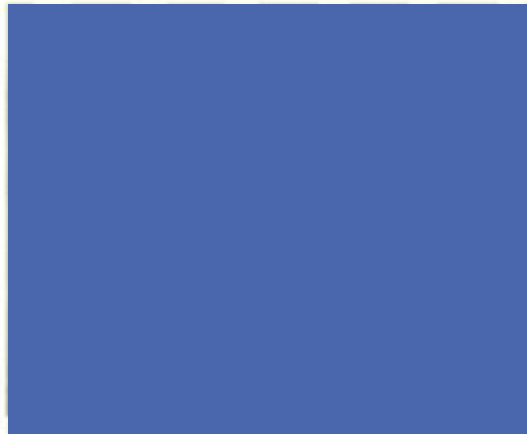
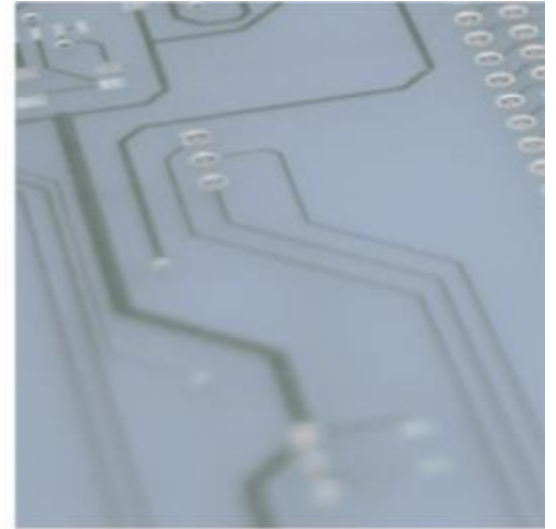
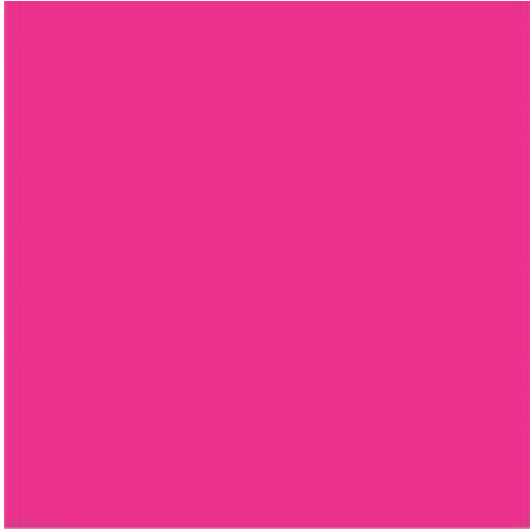


# CanEasy and LIN



- LIN Basics
- Create a LIN configuration in CanEasy
- Import / Export of a LIN configuration
- Playing around with LIN



# What is LIN?

- LIN means „Local Interconnect Network“
  - Used to transfer a low amount of data
  - Often utilized for comfort features, e.g. controlling switches, illumination elements, servomotors
  - The hardware requirements are low
  - A UART or SCI interface is sufficient
-

- The payload is packed into messages (called frames)
  - Every message has its own ID
  - A message is received from all participants
  - If a received message is relevant or not is decided on the help of the ID of the received message
-

# Master Slave Principle

- The communication on the LIN bus is organized according to the “Master Slave Principle”
- The master starts the communication to one or more slaves
- If a slave receives a message it is associated to it has to react according to this message
- Only one slave is allowed to answer to a master request  
[Exception: “Event-Triggered-Frames”]
- A slave is not allowed to send data on its own

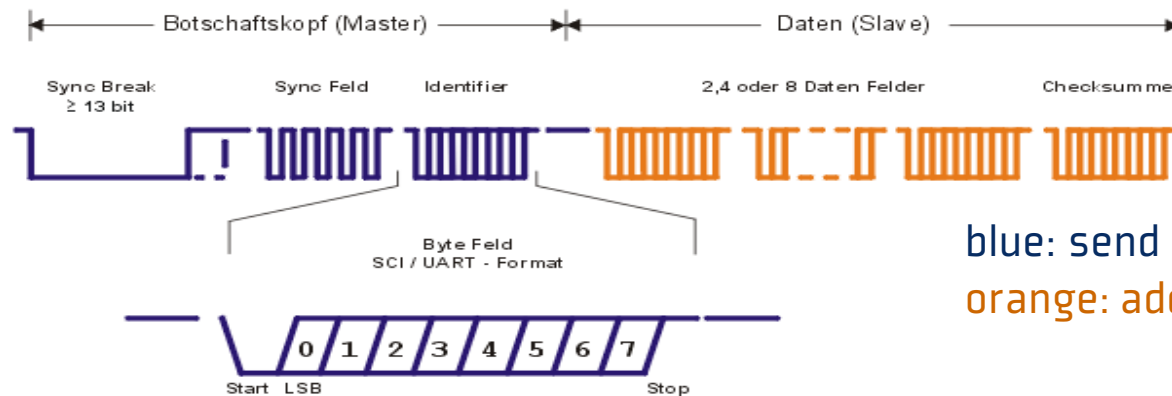


Message Name	Delay [ms]
--------------	------------

Message01	20
Message02	20
Message03	20
Message02	20
Message04	20
Message02	20

- The Scheduler is part of the Master.
- It controls which message has to be triggered, to be sent on the bus, at a specific point of time.
- Its configuration can be thought as a table which contains the message name and the duration [delay] of each message.
- Depending on the system state different schedule tables can be active.
- If the master detects collision for an Event-Triggered frame, then it starts a separate conflict dissolve table which requests all frames assigned to the Event-Triggered frame

# Structure of LIN Messages



- **Synch Break:** signalizes the beginning of a message (dependent on LIN-Standard)
- **Synch Field:** speed adaption from the slaves to the speed of the master (1 Byte)
- **Identifier:** Message identifier (1 Byte: 6 bit ID, 2 bit Checksum)
- **Data Fields:** payload (0-8 Byte).
- **Checksum:** Checksum for detecting transmission errors (1Byte) [dependent on LIN-Standard]

- **Unconditional-Frame**  
Standard message, transmitting of data.
  - **Sporadic-Frame**  
Special „Unconditional-Frame“, is not send cyclic, the master insert these messages into an empty slot of the scheduler table
  - **Event-Triggered-Frame**  
It is allowed that more that one slave answers to these messages. If this is the case and the master detects a collision, it requests one answer for every slave separately which can answer to this “Event -Triggered -Frame.
  - **Diagnostic-Frame**  
Messages with ID **0x3C** and **0x3D**, are used for diagnosis purposes. It is not allowed to transmit “normal” data with this messages.
  - **Reserved-Frame**  
Message IDs which are not allowed to be used [**0x3E**, **0x3F**].
-

# LIN 1.3 vs LIN 2.0

- **Lin 1.3**
    - Unconditional-Frame
    - Event-Triggered-Frame
    - Diagnostic-Frame
    - LIN Description File (LDF)
  
  - **LIN 2.0**
    - Sporadic-Frame
    - Enhanced checksum
    - Mandatory node configuration commands
    - Node Capability Language specification (NCF)
-

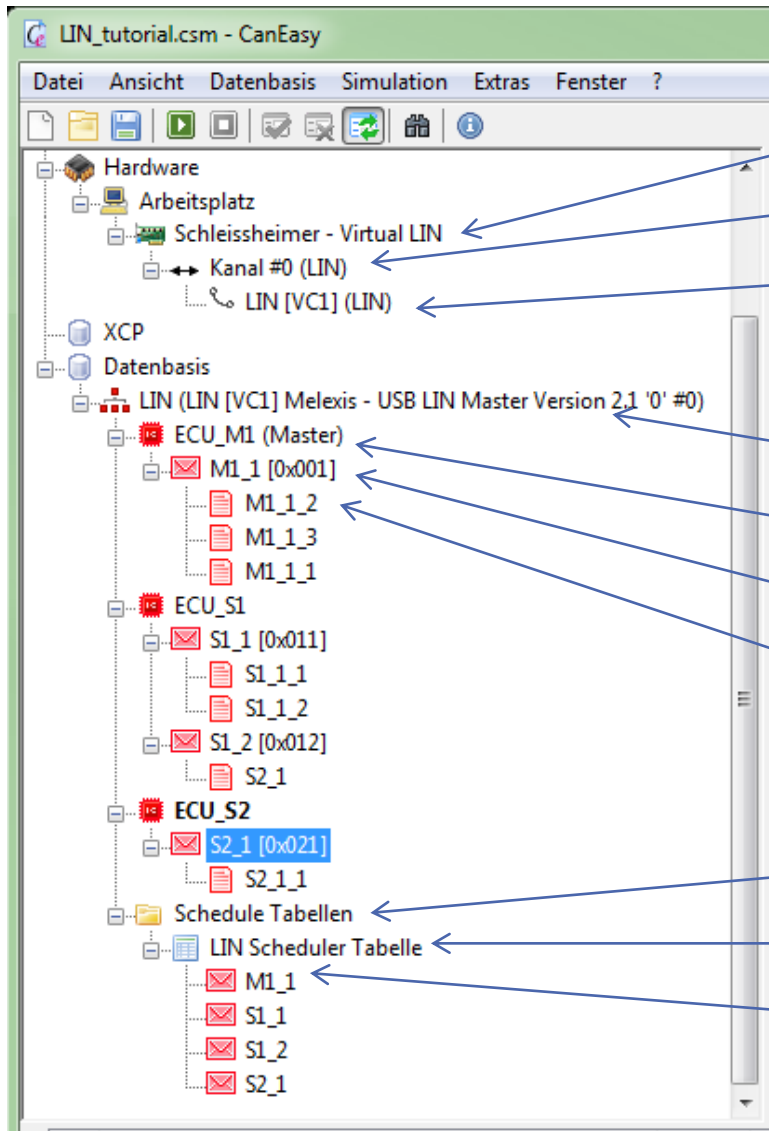
## Pros

- The communication on a LIN bus is deterministic, because of everything is controlled centralized by the master.

## Cons

- If the master is out of order, the whole bus breaks down.
  - The master has to know all the slaves which are part of the system to be able to talk to them.
-

# LIN modelling in CanEasy



Hardware  
Channel  
associated (LIN)bus

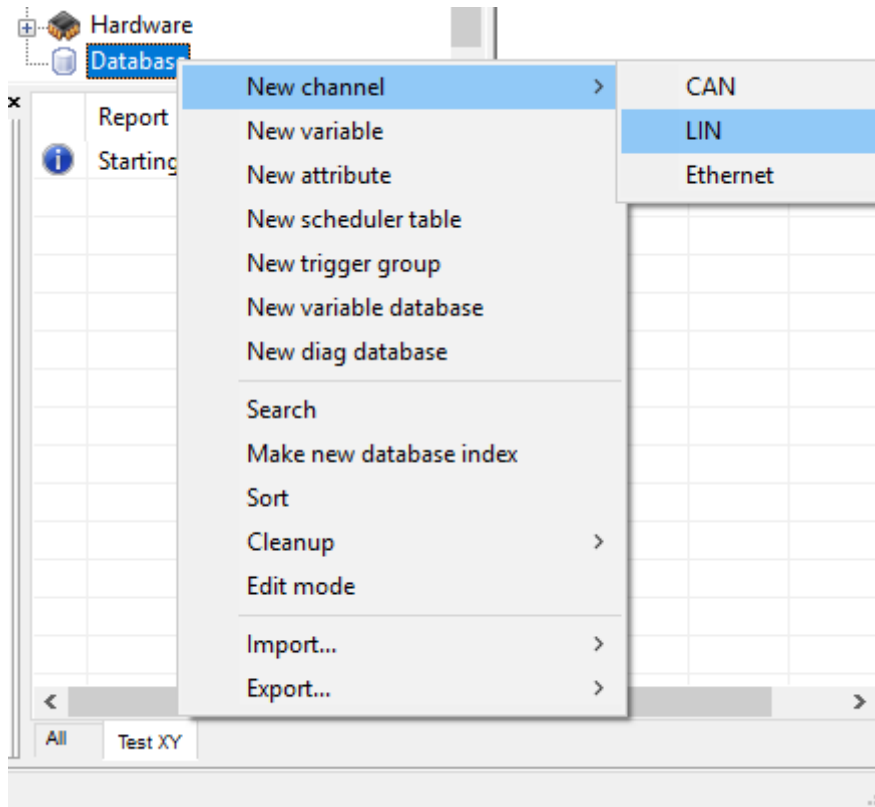
LIN bus  
ECUs  
Messages  
Signals

Scheduler  
Table  
Table entries

# Creation of a LIN configuration

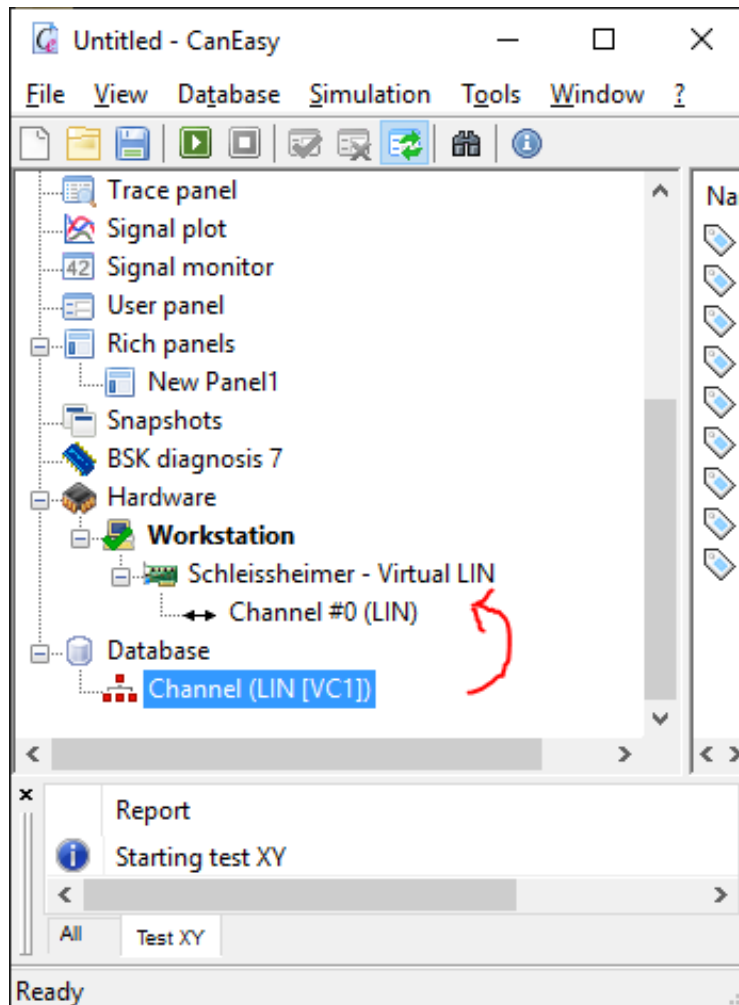
- There are two possibilities to create a LIN configuration in CanEasy
    1. **Manuel configuration**  
Busses, ECUs, Messages and Signals can be configured by hand
    2. **Import LDF or NCF files**  
CanEasy can import LDF and NCF files and automatically generates Busses, ECUs, Messages and Signals
-

# Create a LIN bus



1. Right click "Database"
2. Click new channel
3. Click LIN
4. Enter a name

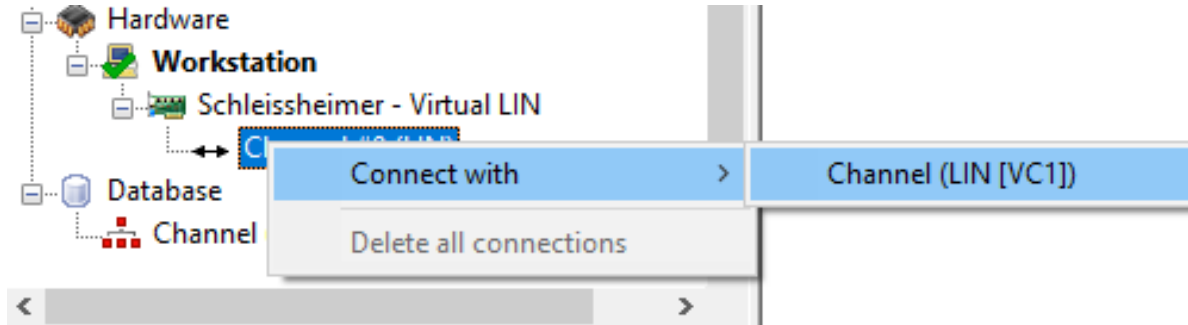
# Attach a (LIN)bus to hardware (1)



Drag n' Drop the bus onto  
a hardware channel

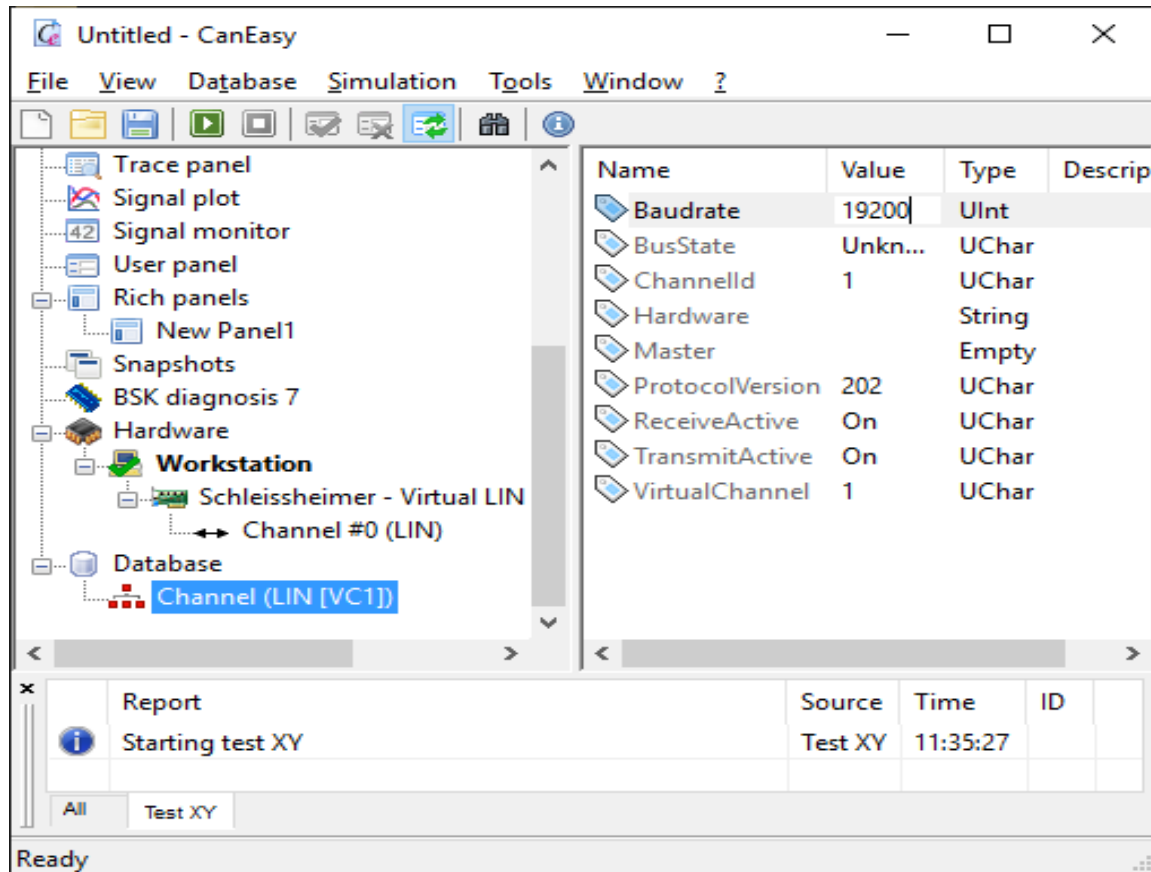
or...

# Attach a (LIN)bus to hardware (2)



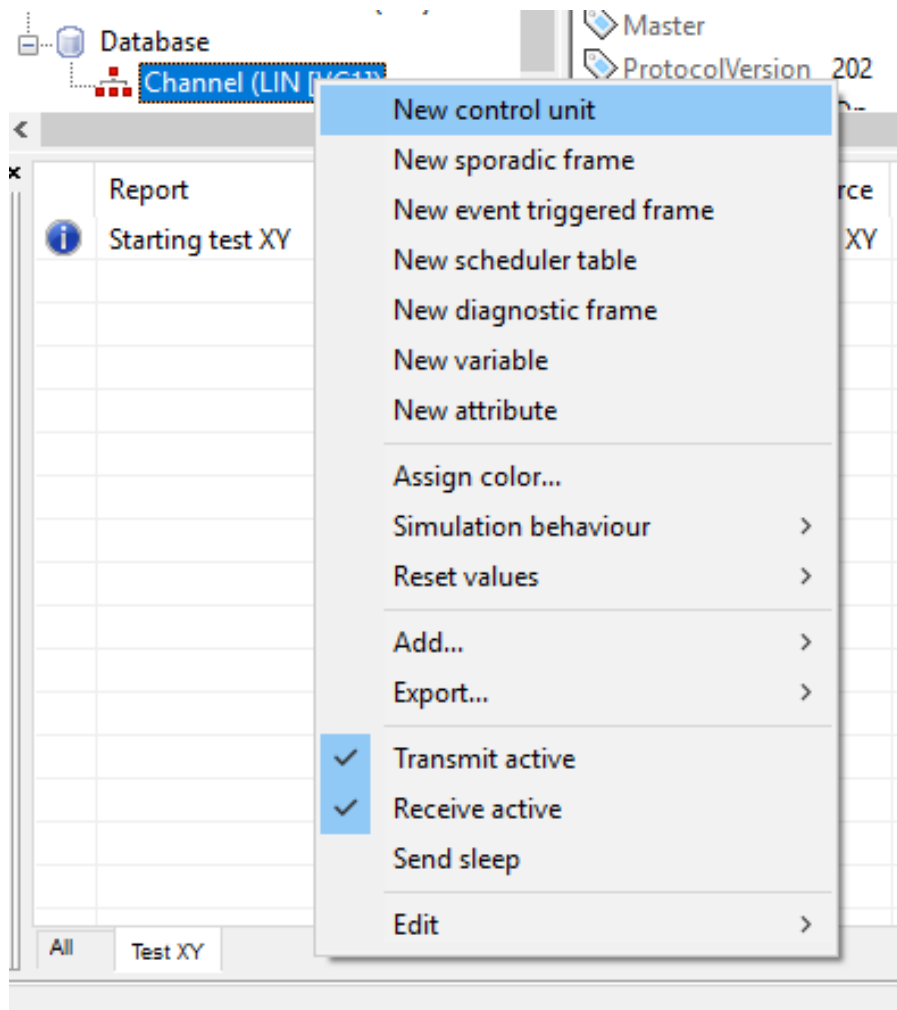
1. Right click on a hardware channel
2. Click on "connect to"
3. Click on the bus you would like to connect

# Configure a LIN bus



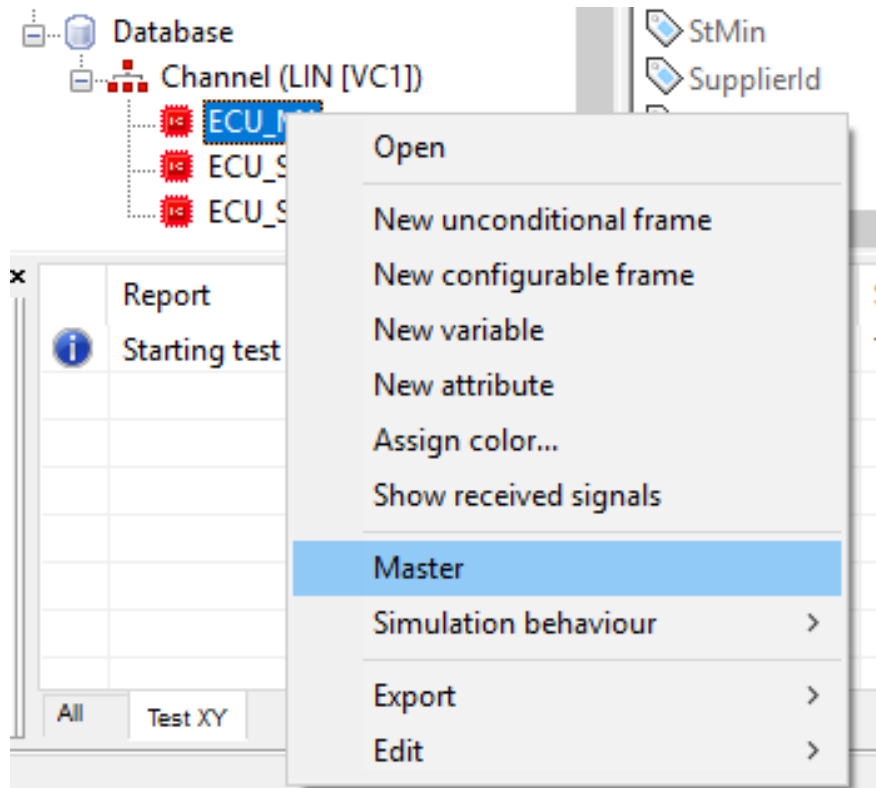
- Edit the parameter on the right side  
[at least the Baudrate]

# Create a ECU

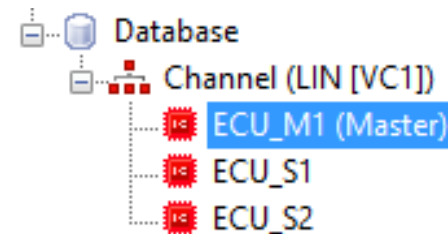


1. right click (LIN) bus
2. Click "new ECU"
3. Enter a name

# Master / Slave Configuration

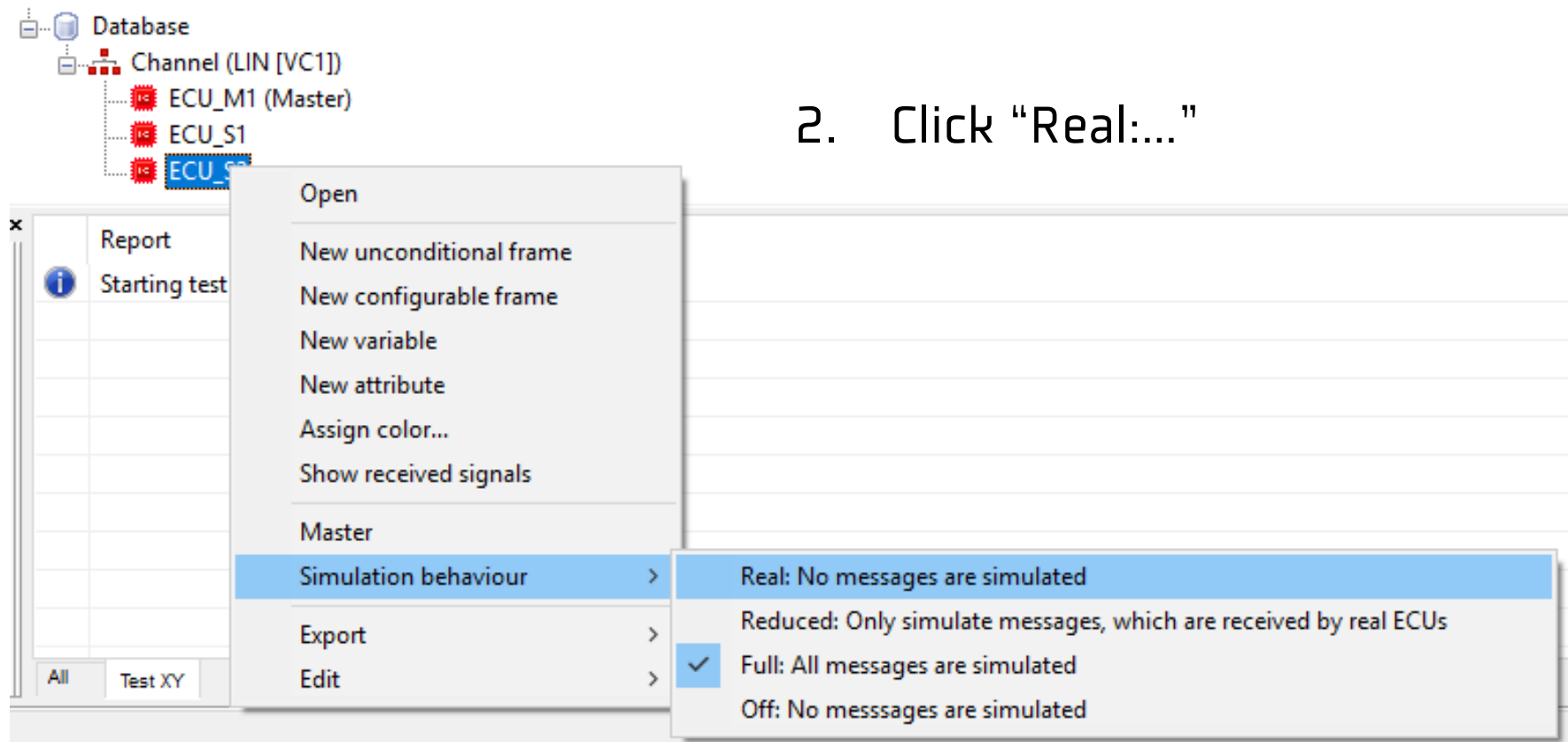


1. Right click on the ECU which should become the Master
2. Click "Master"

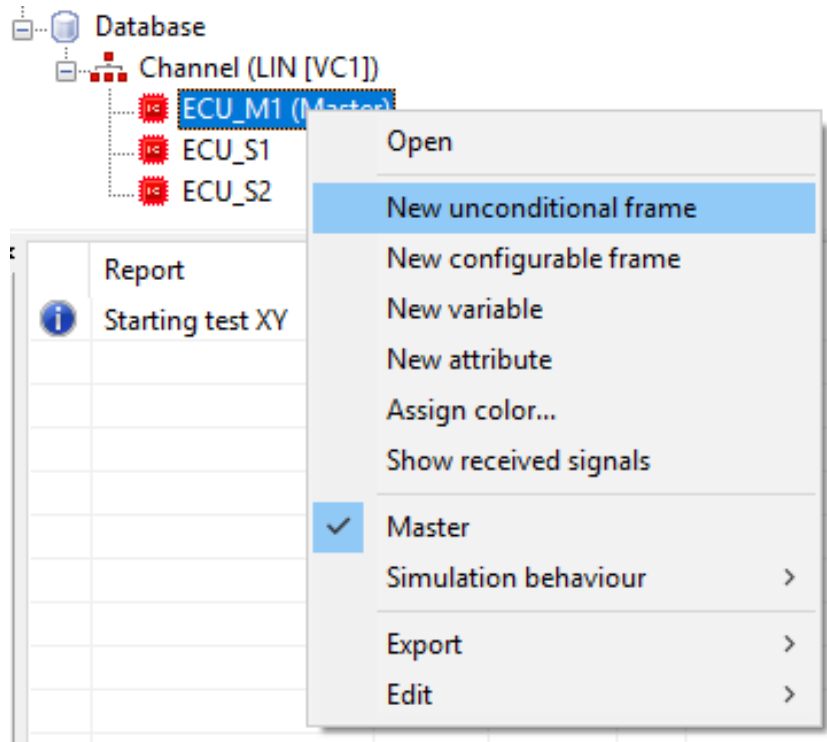


# Real / Simulated

1. Right click on the ECU which should not be simulated
2. Click "Real:..."

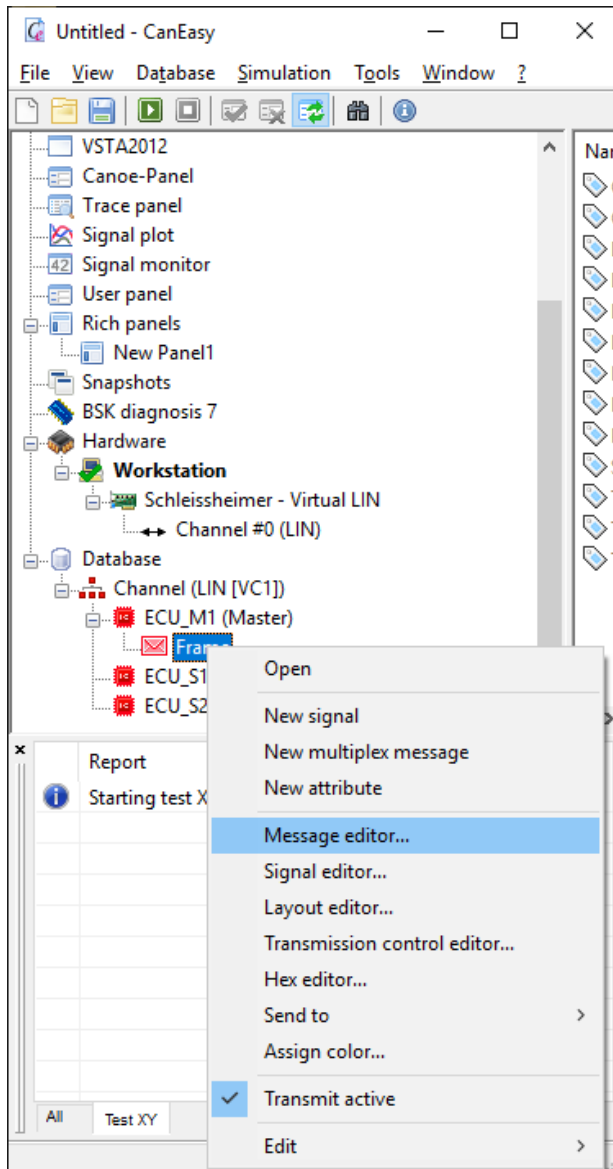


# Create a Message



1. Right click on the ECU which should obtain a new message
2. Click on „new Unconditional-Frame“
3. Enter name

# Configure Message (1)



- Edit the parameter on the right side

or

1. Right click on the message which should be configured
2. Click on „Message Editor...”

# Configure Message (2)

Message editor

Bus name:  Control unit:

Name	Identifier	Cycle time	Description
Frame	0x000	0	

Display

Name:  Description:

Attributes

Identifier:  ☐ STD ☐ XTD

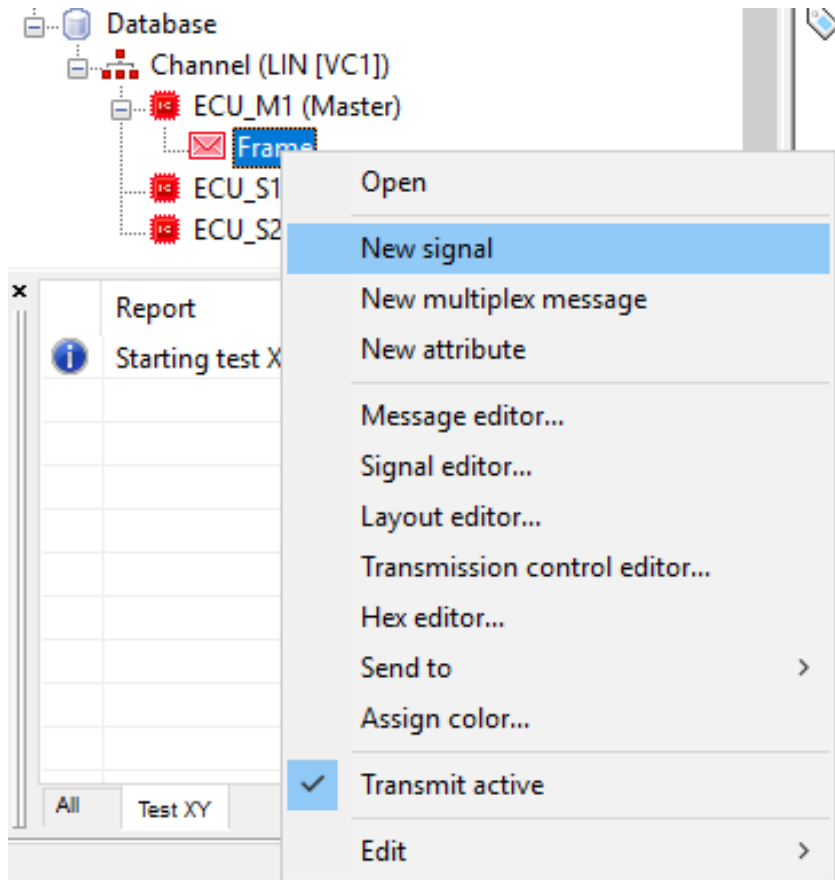
Data bytes:  Byte(s)

Checksum:

Transmission control

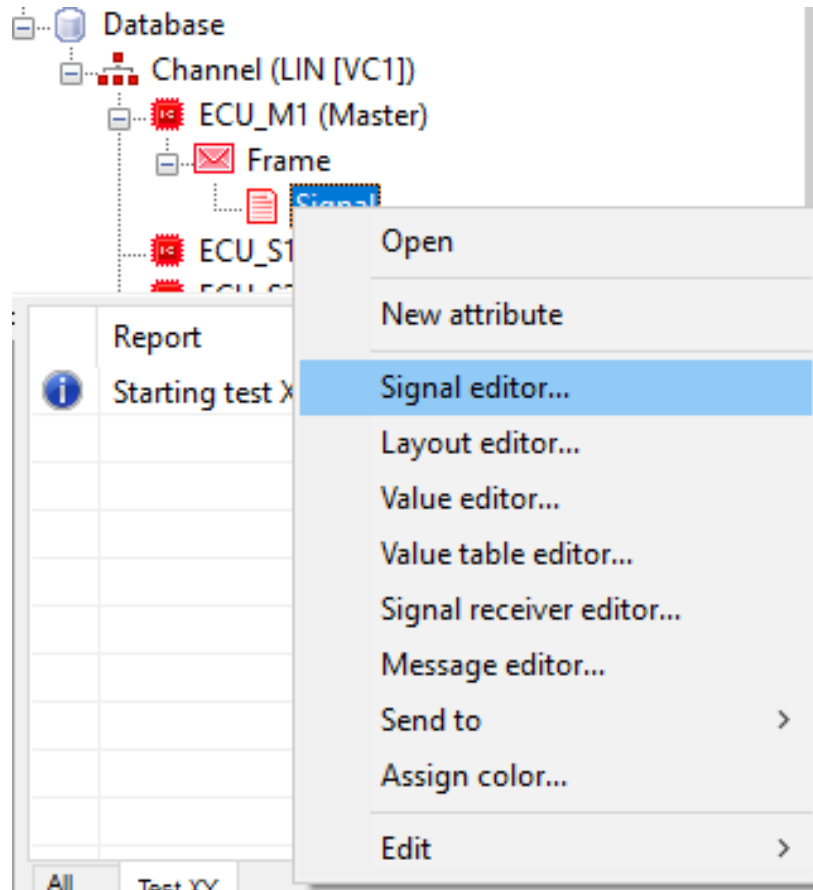
1. Set message Identifier
  2. Set number of data bytes
  3. Set type of checksum
- You can add new messages by clicking on the „New“ button.

# Create a Signal



1. Right click on the Message which should obtain a new signal
2. Click on „new Signal“
3. Enter name

# Configure Signal (1)



- Edit the parameter on the right side

or

1. Right click on the Signal which should be configured
2. Click on „Signal Editor...”

# Configure Signal (2)

Signal editor

Bus name: Channel Control unit: ECU\_M1 Message: Frame

Name	Description	Byte order	Startbit	Length	M
Signal		Int	00	08	

View

Name: Signal Display: Slider

Description:

Scale

Minimum: Offset: Unit

Maximum: Factor:

Attributes

☐ Multiplex

Value type: Unsigned

Default value:

Startbit: 0 Length: 8

Byte order: Intel

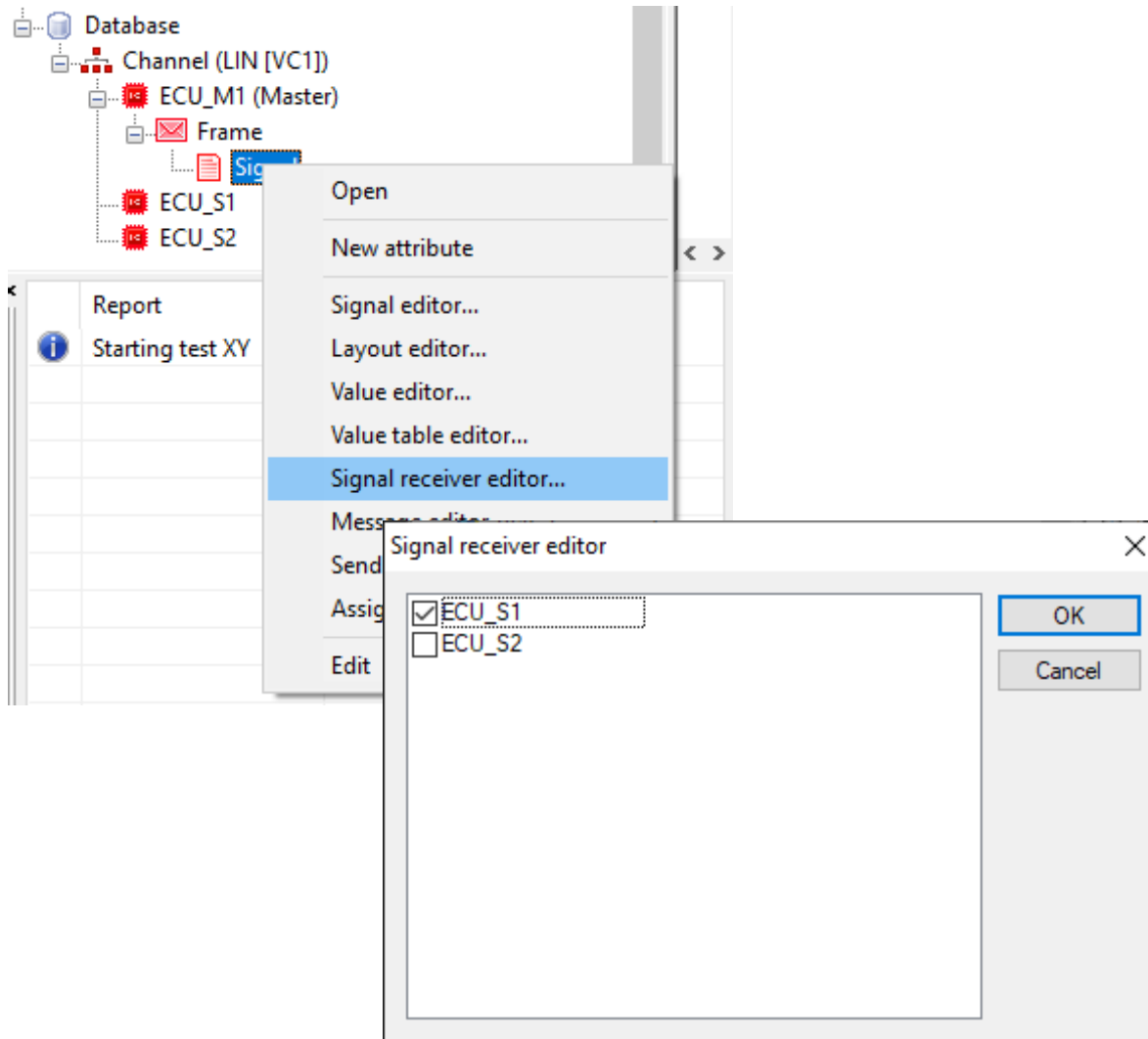
Value table

None

New Delete OK Cancel

- Select the signal which should be edited
- Set the startbit of the signal inside the message
- Set the length of the signal (in bits)
- Configure the scaling of the signal to transform it into a phys. value (if necessary).

# Configure Receiver of Signals (1)



1. Right click on a **signal** which should be received
2. Click on "Signal receiver editor..."
3. Select the ECUs which should receive this signal

or ...

# Configure Receiver of Signals (2)

Untitled - CanEasy

File View Database Simulation Tools Window ?

Hardware

- Workstation
  - Schleissheimer - Virtual LIN
    - Channel #0 (LIN)

Database

- Channel (LIN [VC1])
  - ECU\_M1 (Master)
    - Frame
    - Signal
  - ECU\_S1
    - Frame
  - ECU\_S2
    - Frame
    - Frame1

Report

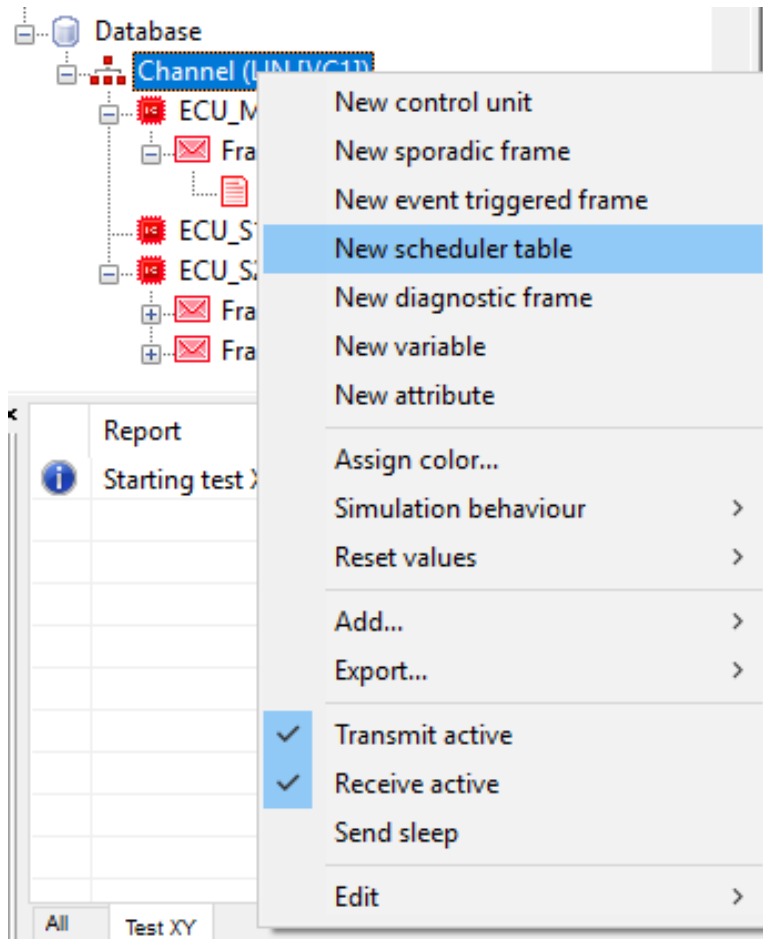
Starting test XY

Received signals from ECU\_M1

- ECU\_S2
  - Frame
    - Signal
  - Frame1
    - Signal

1. Right click on a **ECU** which should receive signals
2. Click on "Show Receive signals"
3. Drag n' drop everything which is received by the ECU into the window.  
[signals, messages or whole ECUs]

# Create a Scheduler



- Right click on a LIN bus
- Click on  
“new Scheduler Table”
- Enter Name

# Configure Scheduler Table Behavior

The screenshot shows the CanEasy software interface. On the left, a tree view displays the project structure: Channel #0 (LIN) > Database > Channel (LIN [VC1]) > ECU\_M1 (Master) > Frame > Signal > ECU\_S1 > ECU\_S2 > Frame > Frame1 > Schedule tables > LIN Scheduler Table. The 'LIN Scheduler Table' is selected and highlighted in blue. On the right, a table displays the configuration parameters for the selected table:

Name	Value	Type	Description
ActivationCounter	0	UChar	
ActivationTime	0	UShort	
Active	True	UChar	
Duration	0	UShort	
EndOfTableMode	NoAc...	UChar	
Repeat	0	UChar	
StartWithSimulation	True	UChar	
StopWithSimulation	True	UChar	

At the bottom, there is a 'Report' tab and a 'Test XY' button. The status bar at the very bottom indicates 'Ready'.

The behavior of a scheduler table can be configured on the right side.

## StartWithSimulation

- The scheduler table is activated automatically after the "**ActivationTime**" has been expired after the simulation was started.

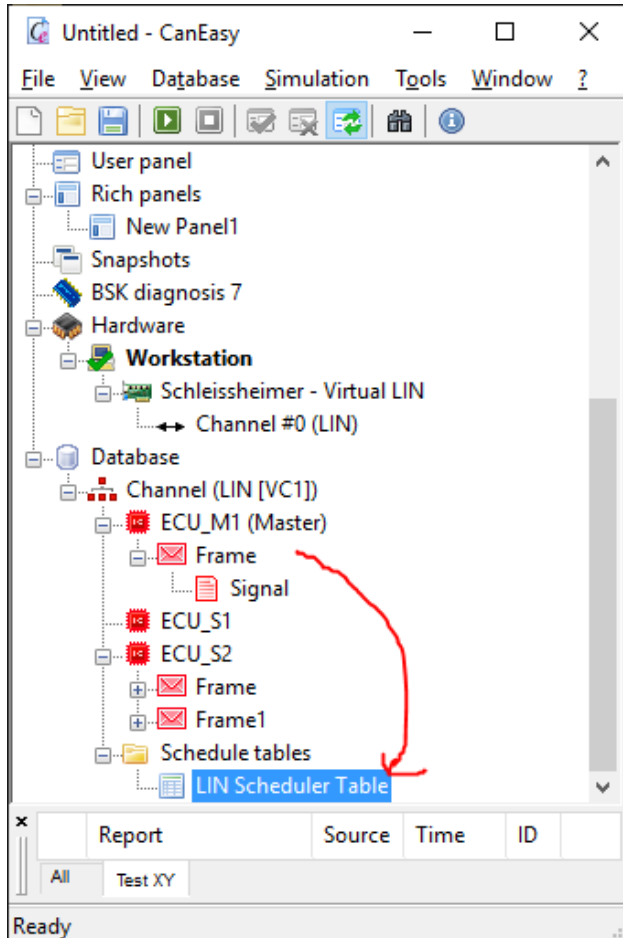
## StopWithSimulation

- The scheduler table is stopped automatically after the simulation was stopped.

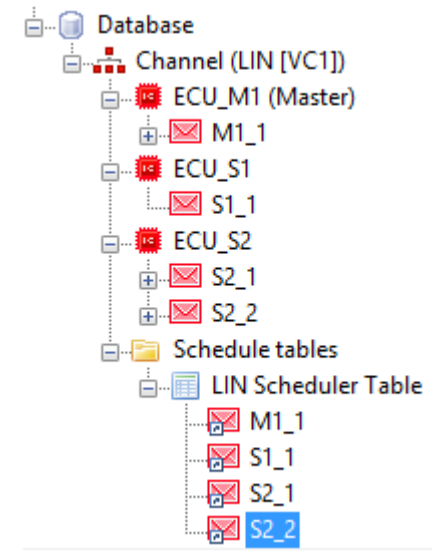
## EndOfTableMode

- *NoAction*: Nothing will happen when the processing of the table has finished
  - *Repeat*: The table will be repeated after the processing of the table has finished
-

# Add Messages to Scheduler Table



- Drag n' drop the message which should be added into the scheduler table



# Configure Message Duration

The screenshot shows the CanEasy software interface with the 'Database' tab selected. The left sidebar displays a hierarchical tree structure of the database. The right pane shows the configuration details for the selected message, 'M1\_1'.

**Database Structure:**

- Schleissheimer - Virtual LIN
  - Channel #0 (LIN)
    - Database
      - Channel (LIN [VC1])
        - ECU\_M1 (Master)
          - M1\_1
          - ECU\_S1
            - S1\_1
          - ECU\_S2
            - S2\_1
            - S2\_2
          - Schedule tables
            - LIN Scheduler Table
              - M1\_1
              - S1\_1
              - S2\_1
              - S2\_2

**Message Configuration Table:**

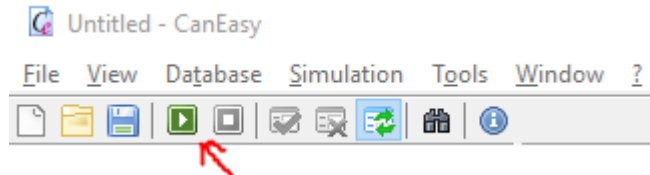
Name	Value	Type	Description
Active	True	UChar	
Duration	140	UShort	
ExecuterMode	Repla...	UChar	
Reference	//DB/...	String	
Repeat	0	UChar	

The bottom status bar shows 'Ready' and a 'Test XY' button.

- Configure the duration of the message, to specify when the next message can be sent on the right side.

# Start Simulation

- Click onto the green “Start” button



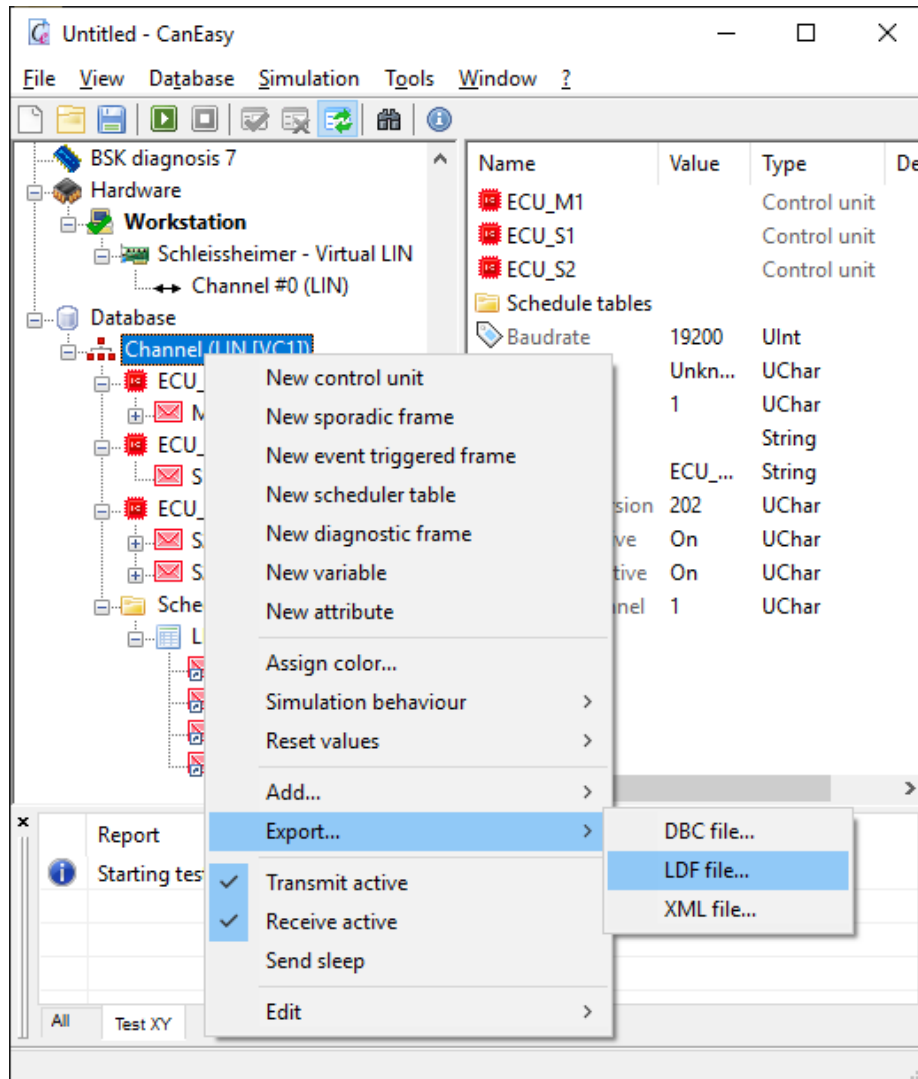
all

- On the trace we can see which messages are sent and the data of the answer from the slaves.
- Because of “ECU\_S2” was set to “Real” its messages data are not simulated by CanEasy.

With this files the configuration of a complete LIN system or individual ECUs can be stored and exchanged.

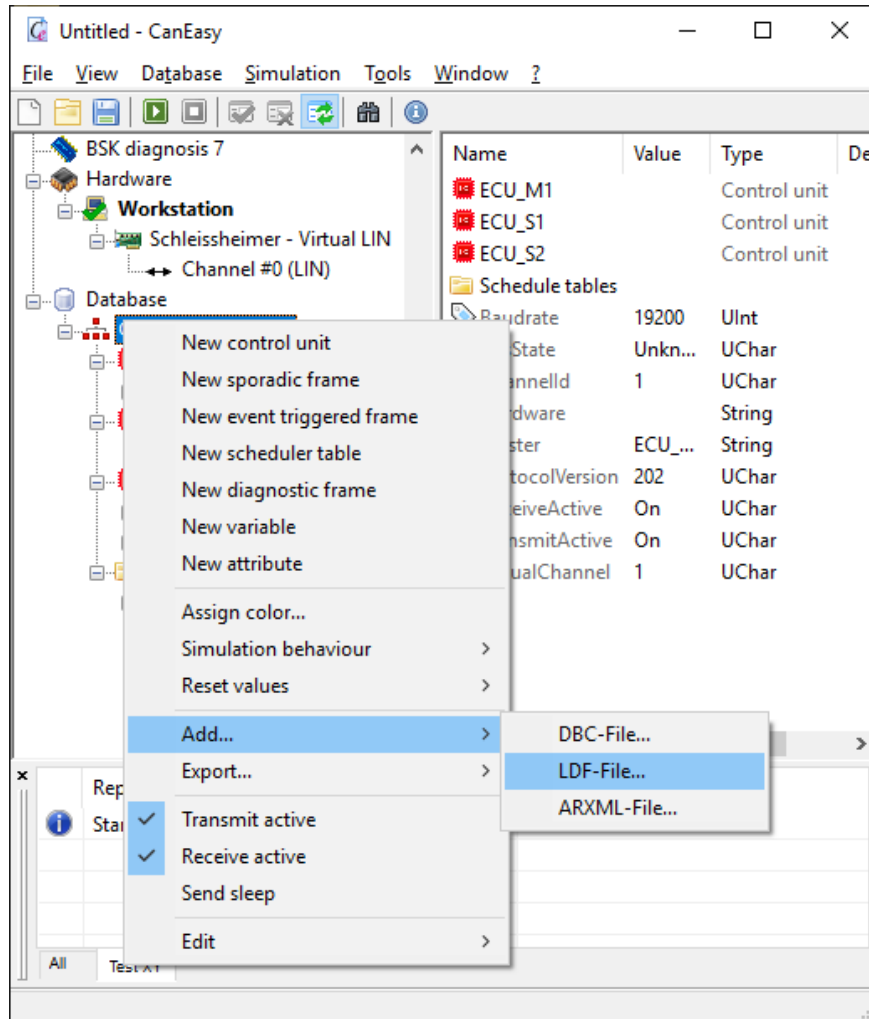
- **LIN Description File (LDF)**  
Describes the whole LIN bus system,  
all slaves and the master.
  - **Node Capability File (NCF)**  
Describes only one slaves.
-

# Export a LDF-File



- Right click on LIN bus
- Click on „Export...”
- Click on „LDF-File...”
- Enter path and file name and press “Save” in file dialog

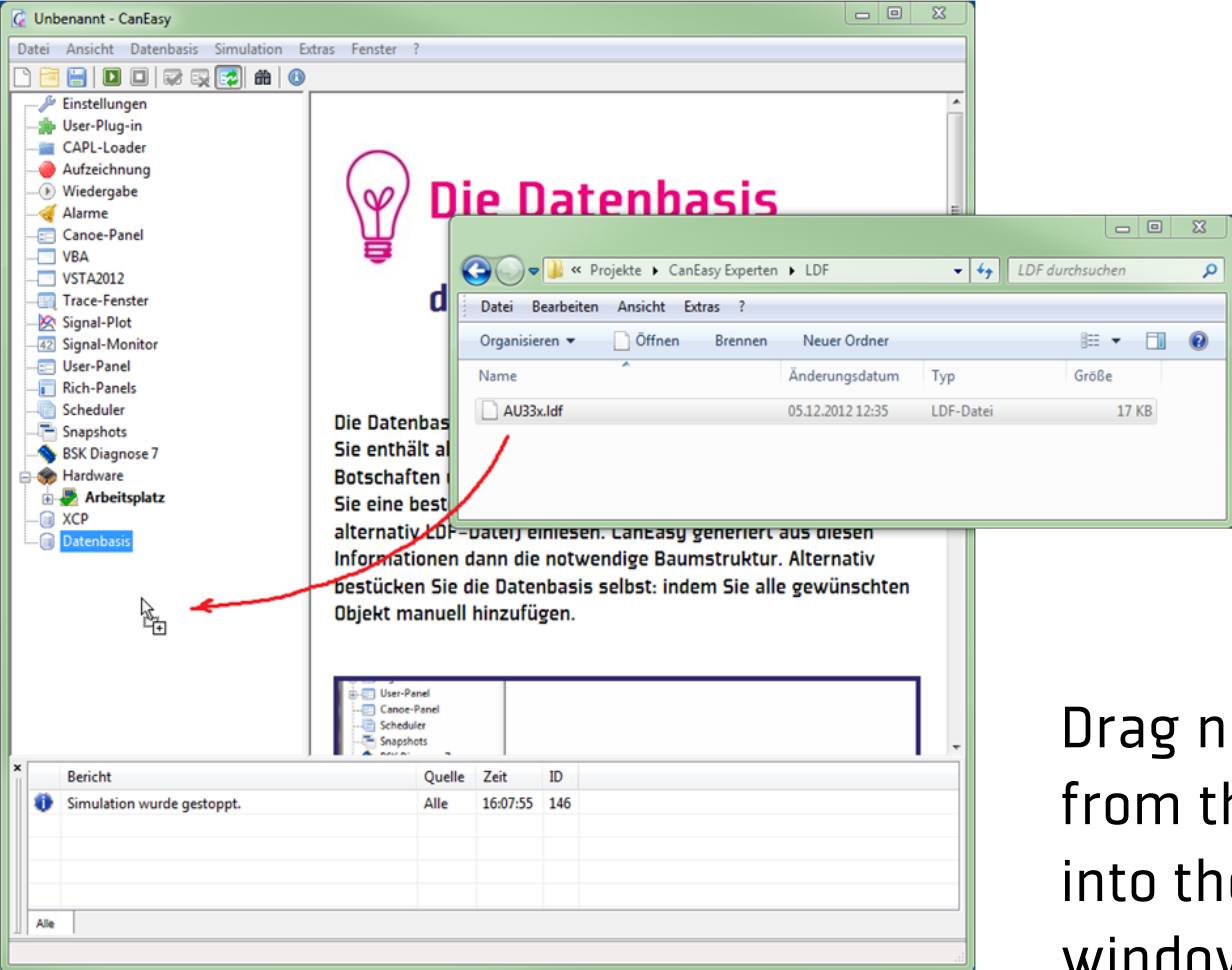
# Import a LDF-File



- Right click on "Database"
- Click on "Import..."
- Click on "LDF-File..."
- Select LDF-File in file dialog

or...

# Import by Drag n' Drop



**Die Datenbasis**

Die Datenbasis enthält alle Botschaften, die Sie eine best... alternativ LDF-Datei einlesen. CanEasy generiert aus diesen Informationen dann die notwendige Baumstruktur. Alternativ bestücken Sie die Datenbasis selbst: indem Sie alle gewünschten Objekt manuell hinzufügen.

File Explorer: Projekte > CanEasy Experten > LDF

Name	Änderungsdatum	Typ	Größe
AU33x.ldf	05.12.2012 12:35	LDF-Datei	17 KB

CanEasy Log:

Bericht	Quelle	Zeit	ID
Simulation wurde gestoppt.	Alle	16:07:55	146

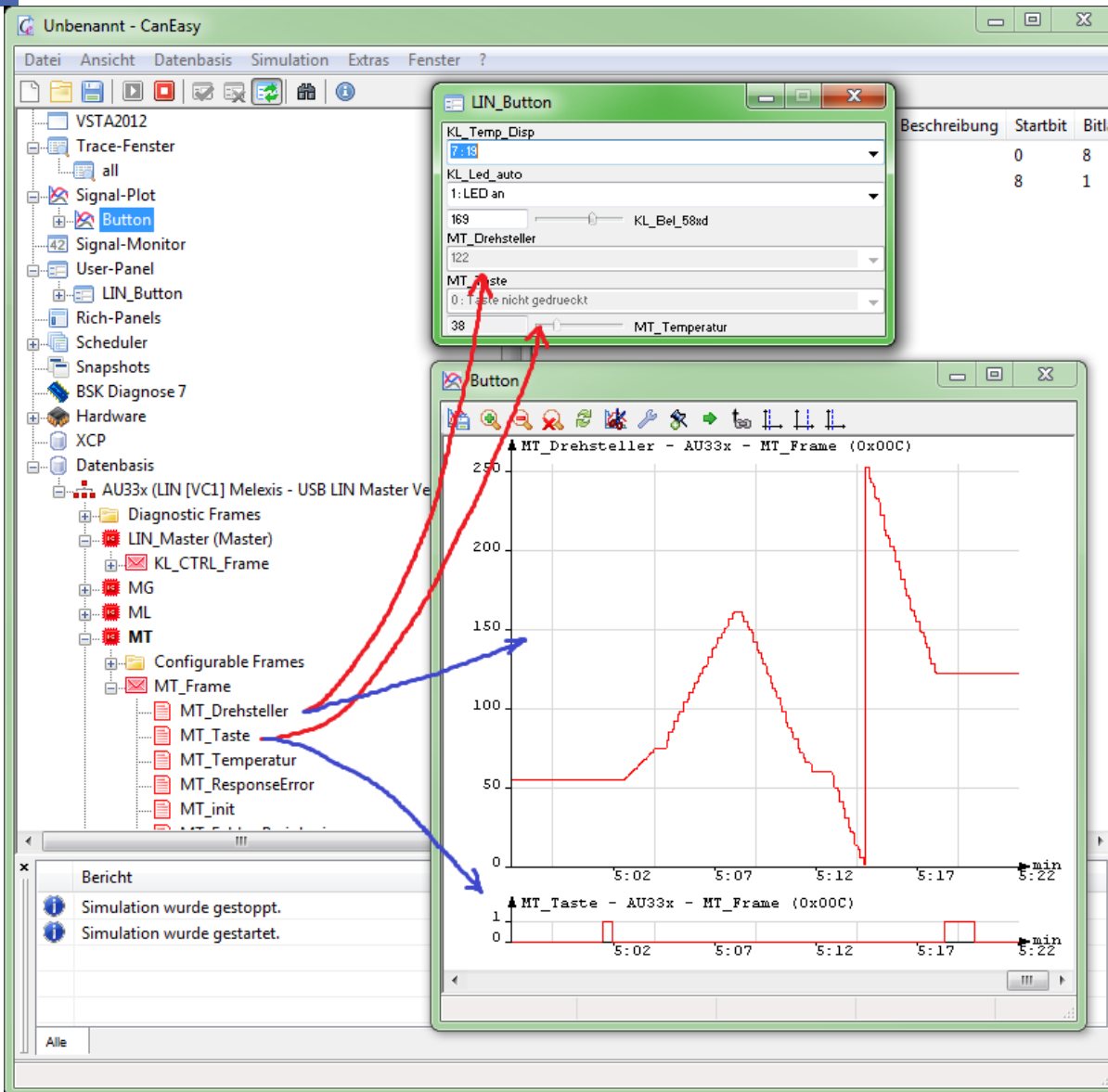
Drag n' drop the LDF-File from the File Explorer into the CanEasy window



## Let's try to bring the “LIN-Button” to life

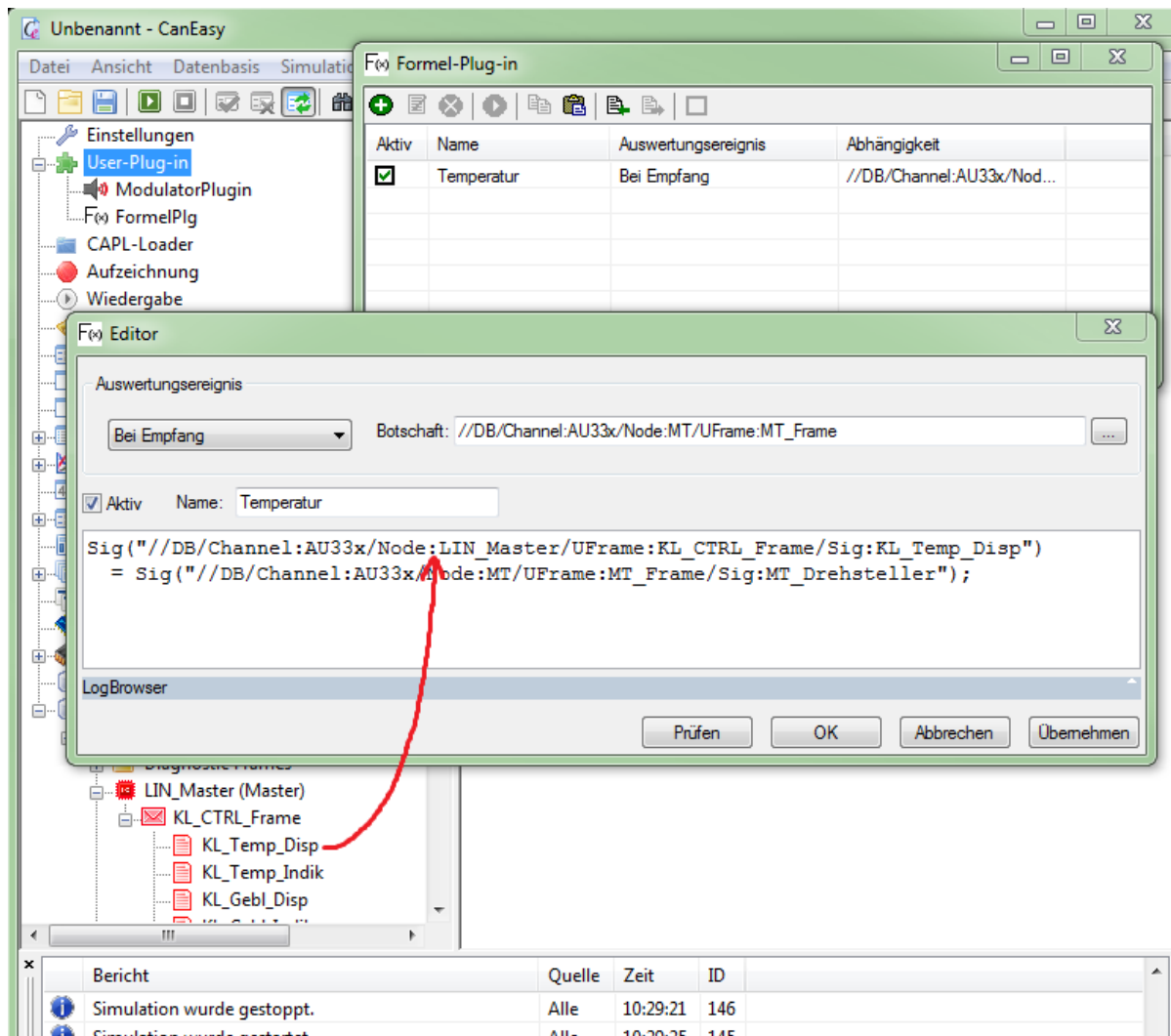
- Import LDF file „AU33x.ldf“
- Set ECU „MT“ to real
- Link the bus to a hardware channel
- Start Simulation
- **MT/MT\_Frame/MT\_Drehsteller:**  
contains the position of the incremental encoder
- **MT/MT\_Frame/MT\_Taste:**  
contains the status if button is pressed or not
- **LIN\_Master/KL\_CTRL\_Frame/KL\_Temp\_Displ:**  
selects a temperature value for the display on top of the LIN-Button.
- **LIN\_Master/KL\_CTRL\_Frame/KL\_Bel\_58xd:**  
illumination brightness of the display on top of the LIN-Button.
- **LIN\_Master/KL\_CTRL\_Frame/KL\_Led\_auto:**  
Switching on/off of the “Auto” LED in the display on top of the LIN-Button.

# Analyzing the interesting Signals



- Create:
  - Signal-Plot
  - User-Panel
- Drag n' drop the interesting signals into the plot and the user panel
- Play around with the Button e.g. press is, rotate the ring, and change the signals from the master

# Change Displ. Val. by rotating the Ring



- Load the "User-Plug-in / Formel Plug-in"
- Configure the activation event
- Write a formula which copies the value from the ring to display value

# Switching on/off the "Auto" LED

F Editor

Evaluation event

Pre transmit Message: //DB/Channel:Channel/Node:MC/UFrame:IC

☒ Active Name: Auto-LED

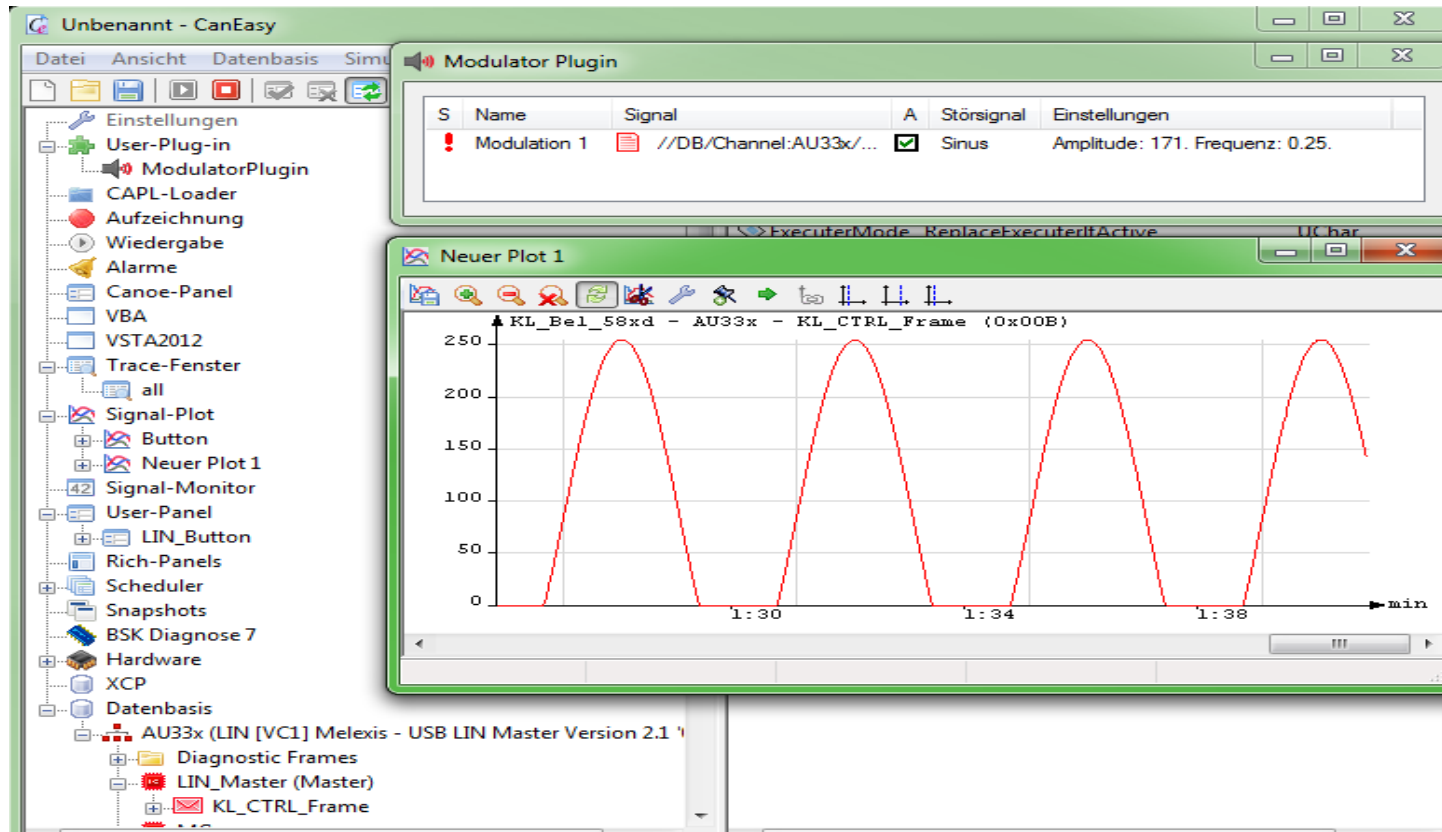
```
TasterCur = Sig("//DB/Channel:Channel/Node:MC/UFrame:IC/Sig:MT_Taste");  
HighFlanke = TasterAlt == 0 && TasterCur == 1;  
  
Sig("//DB/Channel:Channel/Node:MC/UFrame:IC/Sig:KL_Led_auto")  
  = Sig("//DB/Channel:Channel/Node:MC/UFrame:IC/Sig:KL_Led_auto") + HighFlanke;  
  
TasterAlt = TasterCur;
```

LogBrowser

Check OK Cancel Apply

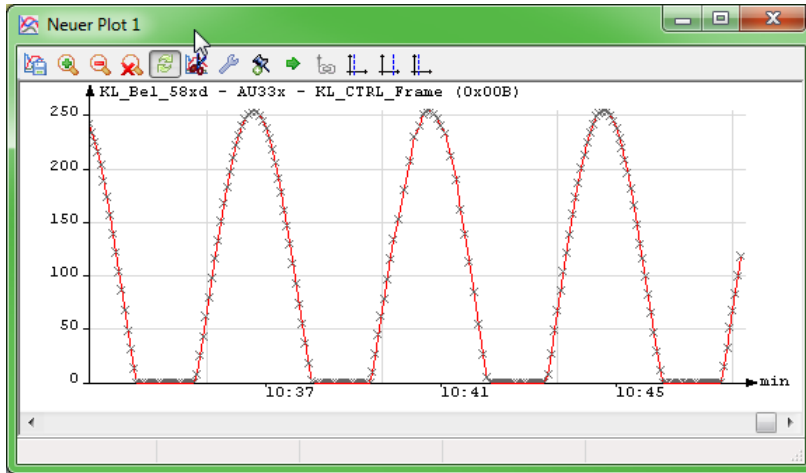
- Add a new formula
- The old value of „MT\_Taste“ is used to generate a rising edge
- This edge is added to „KL\_Led\_auto“ to change the state for the „Auto“ LED

# Animating the Display Illumination

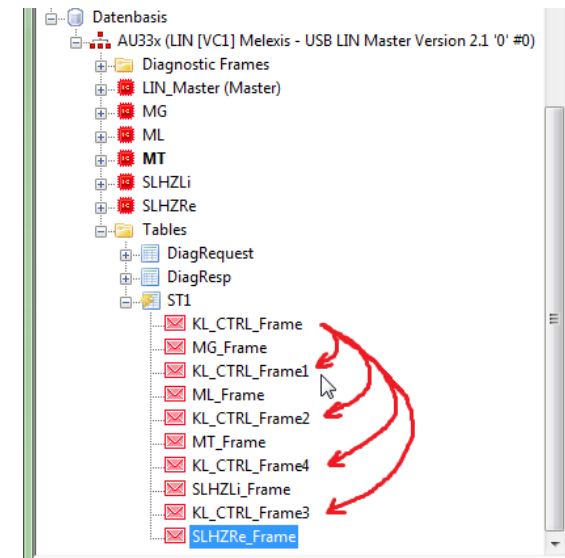
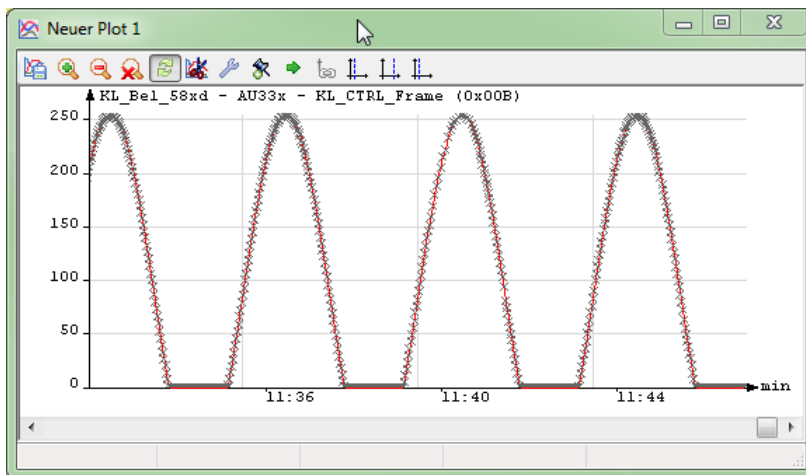


- Load the "User-Plug-in / Modulator Plugin"
- Drag n' Drop the signal "LIN\_Master/KL\_CTRL\_Frame/KL\_Bel\_58xd" into the Modulator Plugin.
- Set "Start Value" = "84", "Amplitude" = "171", "Frequency" = "0.25"

# Enhance the Animation Smoothness



Switch on the sample points in the plot



Add the signal "" several times to the Scheduling Table to increase the number of transmissions per scheduler cycle.

- LIN Basics
    - Master/Slave, Scheduler
    - Message structure, Message types
  - Create a LIN configuration in CanEasy
    - Bus, ECU, Messages, Signals
    - Connecting to LIN Hardware
    - Scheduler Table
  - Import / Export of a LIN configuration
    - LDF, NCF
  - Using CanEasy to analyse a LIN System
    - User Panels, Trace, Plot
  - Using CanEasy to simulate ECU functionalities
    - Formula Plug-in, Modulator Plug-in
-

# Advanced LIN Functionalities & CanEasy

- Event Triggered Frames
- Sporadic Frame
- Configuration



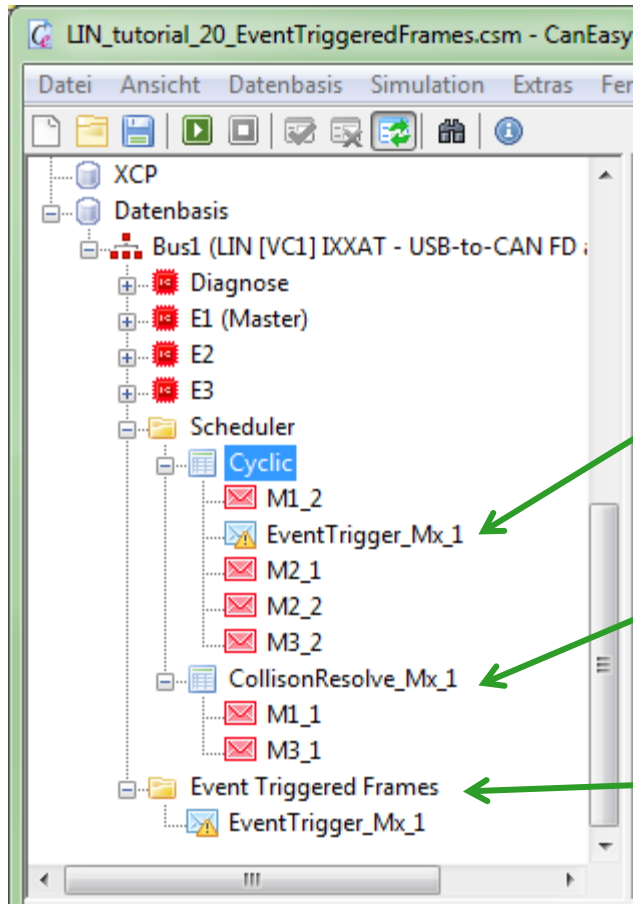
+



# Event Triggered Frames

- More than one message are assigned to an Event Triggered Frame
  - The master requests the Event Triggered Frame and all the slaves, where the data for this message have been changed answer at the same time to the request.
  - **No answer:** no data has been changed
  - **One answer:** the data for one slave has been changed, the message can be received without problems.
  - **Multiple answers:** if the master detect check sum errors, then more than one slave answer to the request. Now the master starts a special scheduler table dissolve the conflict. In this table all messages are contained which are assigned to the Event Triggered Frame.
-

# Event Triggered Frames in CanEasy



A Event Triggered Frame appears at three positions:

- Trigger for the Event Triggered Frame in a (cyclic) Scheduler table
- Collision Resolving Table with all the messages which will be requested separately
- Definition of the Event Triggered Frame

# Configure the Conflict Dissolve Table

LIN\_tutorial\_20\_EventTriggeredFrames.csm - CanEasy

Menü: Datei Ansicht Datenbasis Simulation Extras Fenster ?

**Datenbasis**

- Bus1 (LIN [VC1] Schleissheimer - Virtual I...)
  - Diagnose
    - E1 (Master)
      - M1\_1 [0x011]
      - M1\_2 [0x012]
    - E2
      - M2\_1 [0x021]
      - M2\_2 [0x022]
    - E3
      - M3\_1 [0x031]
      - M3\_2 [0x032]
  - Scheduler
    - Configuration
    - Cyclic
    - CollisionResolve\_Mx\_1**
    - M2\_1
    - M3\_1
  - Event
    - Event Triggered Frames
      - EventTrigger\_Mx\_1

**Table:**

Name	Beschreibung	Wert	Typ
M1_1			
M3_1			
ActivationCounter		0	UChar
ActivationMode		Manual	UChar
ActivationTime		0	UShort
Active		True	UChar
Duration		0	UShort
EndOfTableMode		NoAction	UChar
NextTableRef			String
Repeat		0	UChar

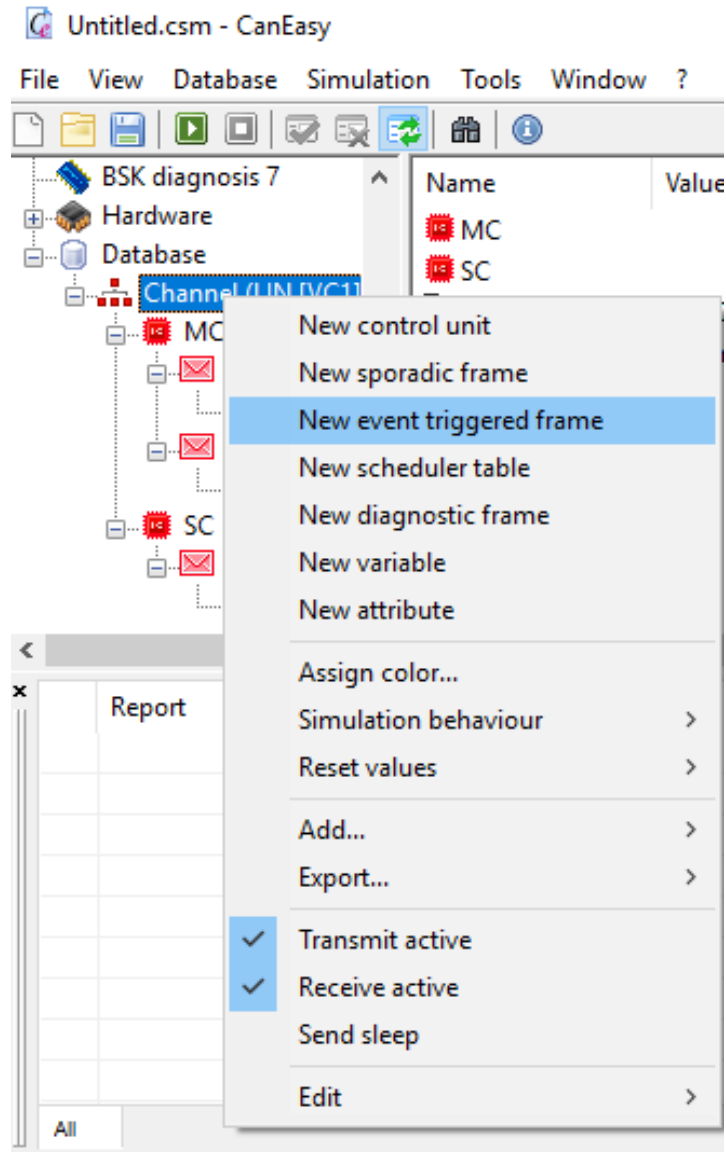
**Bericht**

Quelle	Zeit	ID

Alle Scheduler

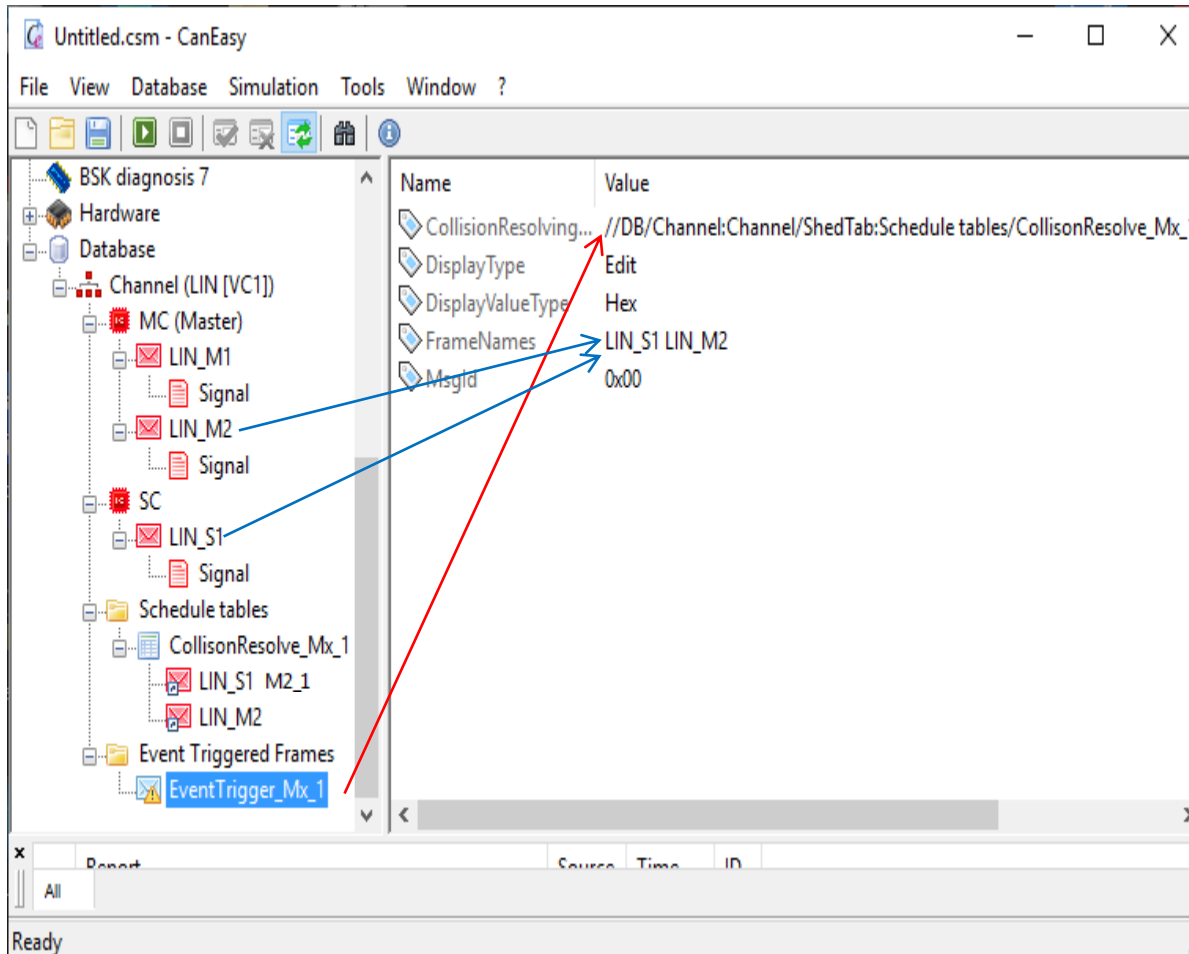
- Add a new Scheduler Table
- Add all message to the Collision Dissolve Table which are assigned to the Event Triggered Frame
- Activation Mode should be "Manual"

# Add an Event Triggered Frames



- Right Click on a bus
- Click on "New Event-Triggered-Frame"
- Enter name

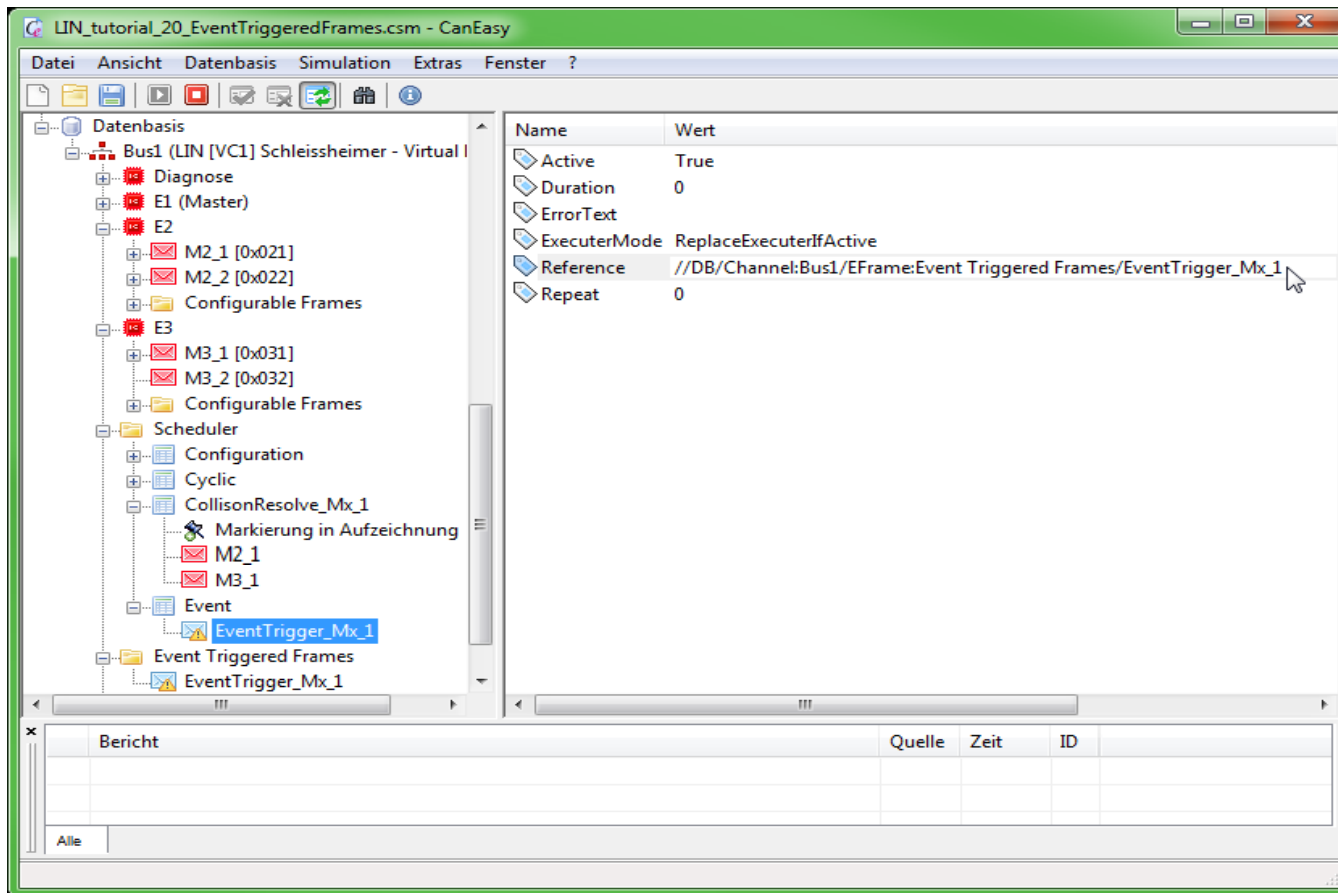
# Configure Event Triggered Frames



The following can be configured on the right:

- Name of the Collision Resolving Table
- Messages are assigned to this Event Triggered Frame
- Message ID of the Event Triggered Frame

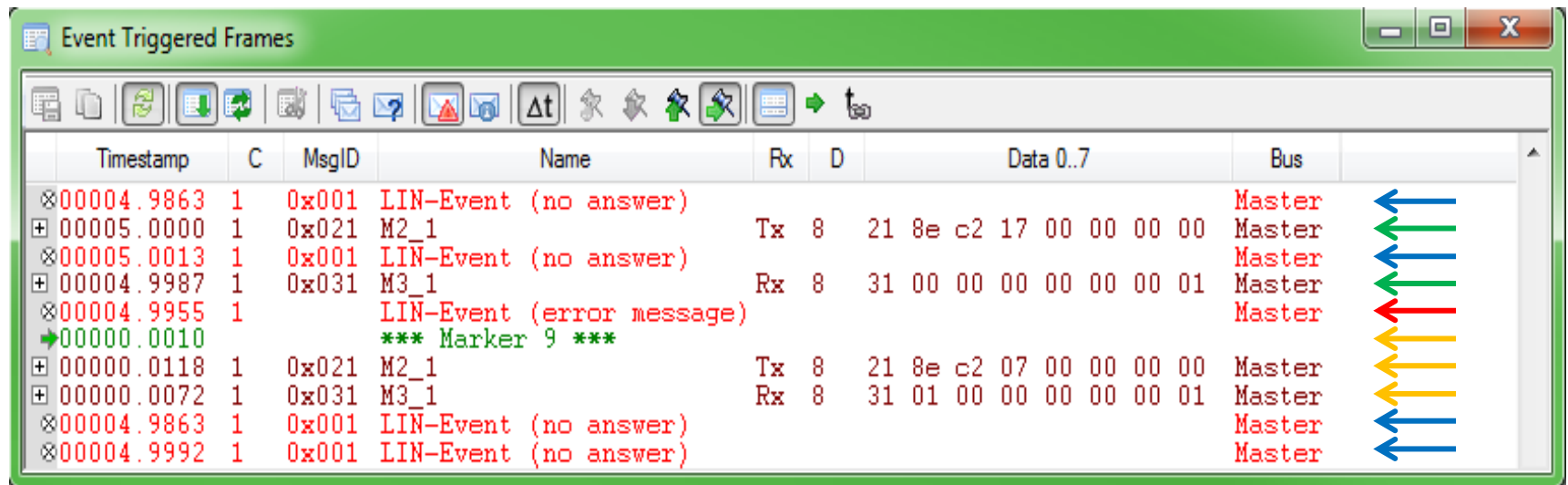
# Add the Event Triggered Frame to a Scheduler Table



- Add a "Transmit Request" to a Scheduler Table
- Insert the path to the Event Triggered Frame for the "Reference" attribute.

# Trace for a Event Triggered Frames

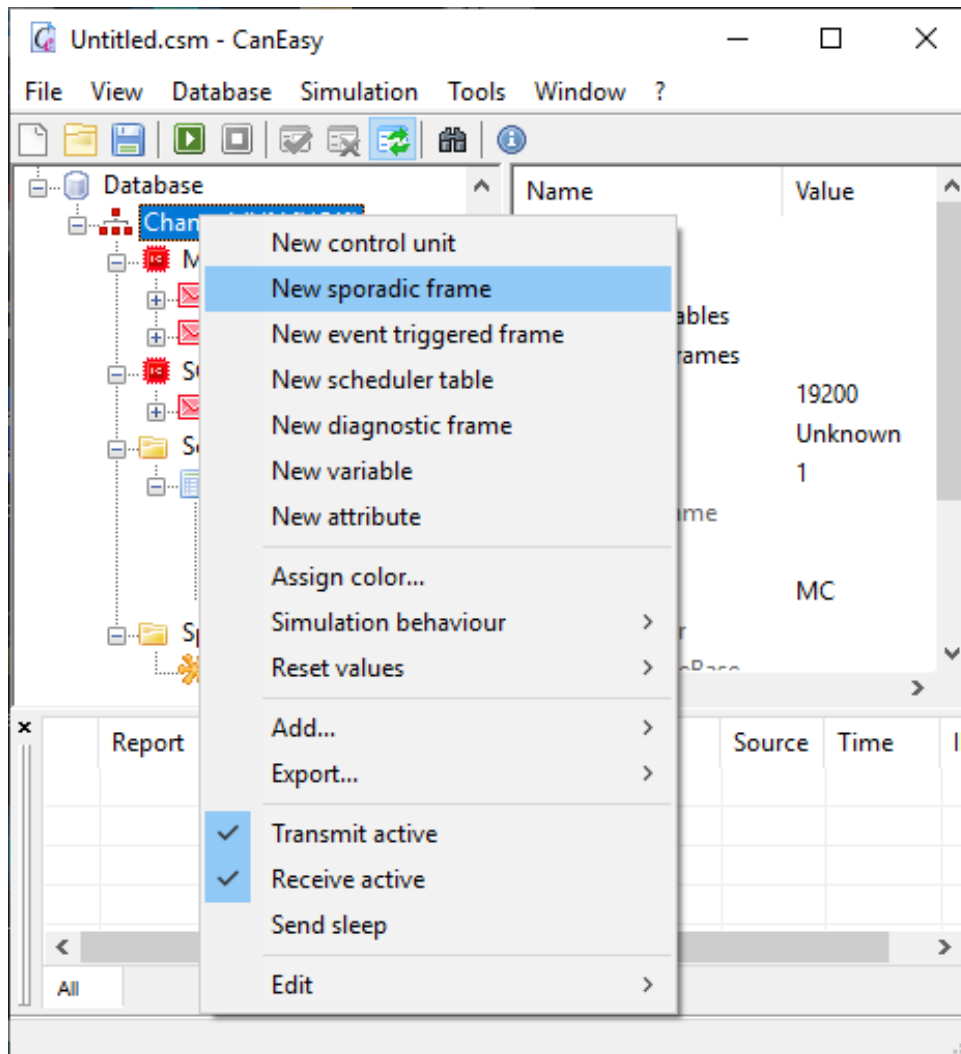
After starting the simulation and using real hardware the behavior of the Event Triggered Frames can be seen in the trace



Timestamp	C	MsgID	Name	Rx	D	Data 0..7	Bus
00004.9863	1	0x001	LIN-Event (no answer)				Master
00005.0000	1	0x021	M2_1	Tx	8	21 8e c2 17 00 00 00 00	Master
00005.0013	1	0x001	LIN-Event (no answer)				Master
00004.9987	1	0x031	M3_1	Rx	8	31 00 00 00 00 00 00 01	Master
00004.9955	1		LIN-Event (error message)				Master
00000.0010			*** Marker 9 ***				
00000.0118	1	0x021	M2_1	Tx	8	21 8e c2 07 00 00 00 00	Master
00000.0072	1	0x031	M3_1	Rx	8	31 01 00 00 00 00 00 01	Master
00004.9863	1	0x001	LIN-Event (no answer)				Master
00004.9992	1	0x001	LIN-Event (no answer)				Master

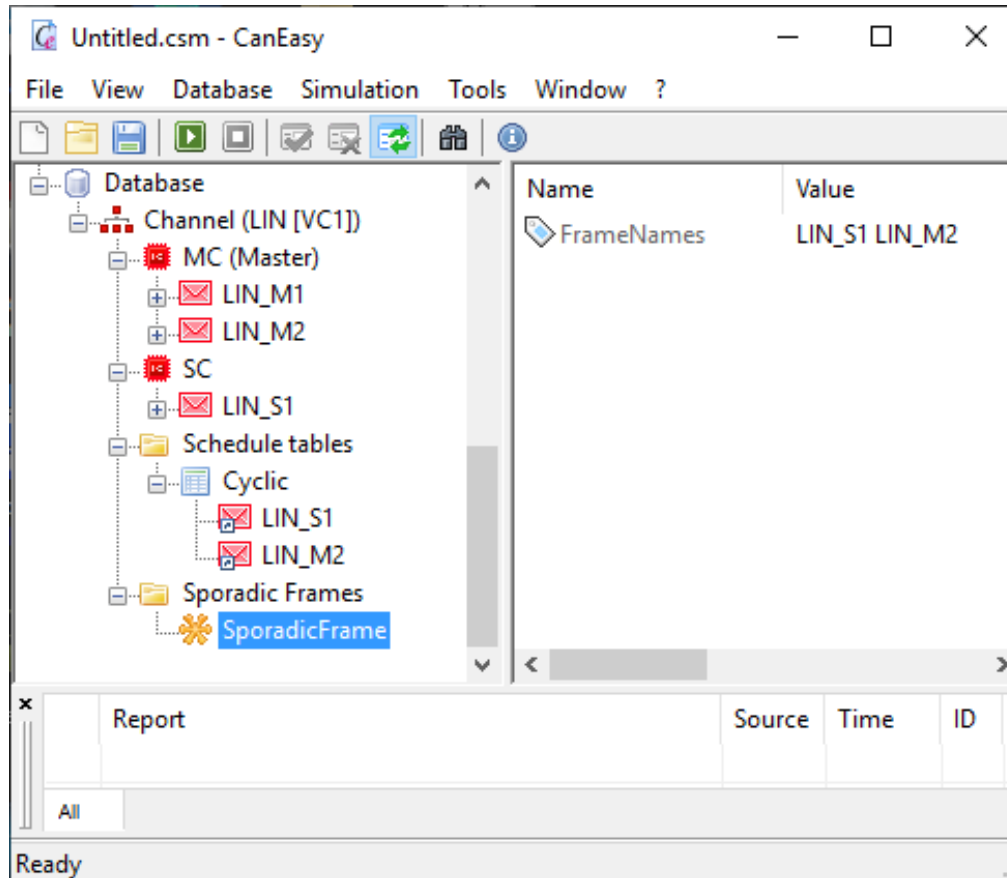
- No answer, because of no signal has been changed
- Event Triggered Frame without collision
- Event Triggered Frame with collision
- Conflict Dissolve Table [\*\*\* Marker \*\*\*]

# Add a Sporadic Frames



- Right click on the LIN bus
- Click on "New Sporadic-Frame"
- Enter name

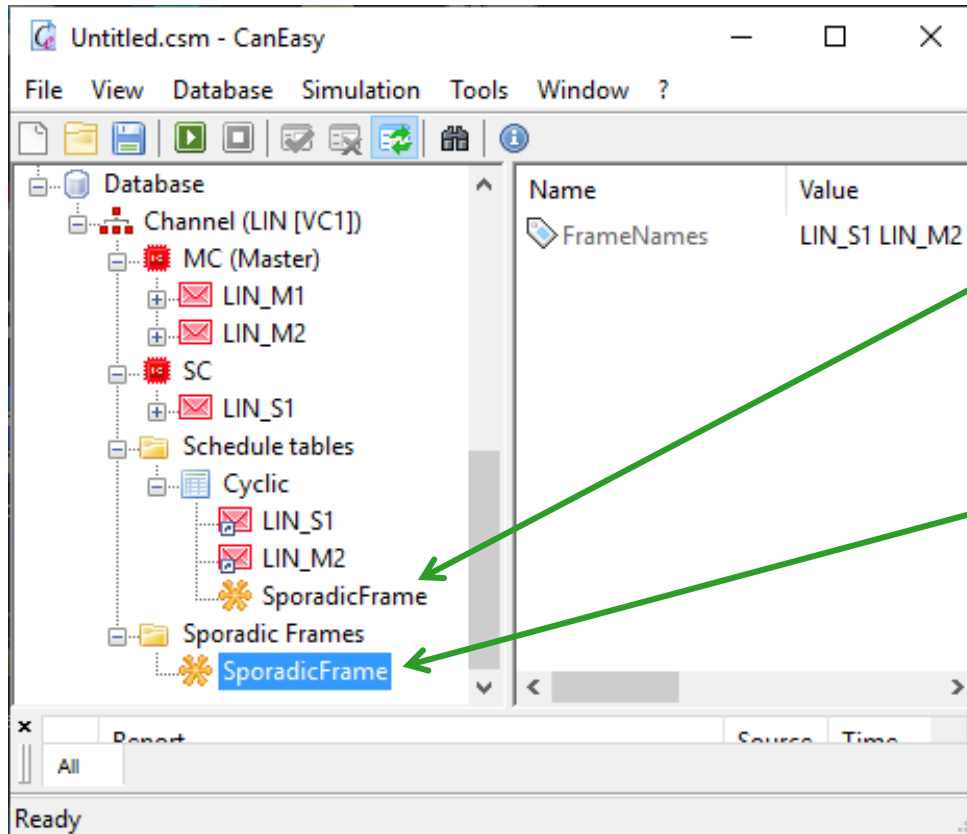
# Configure a Sporadic Frames



A Sporadic Frame can be configured on the right side:

- Enter the names (not the path) of the frames which should belong the Sporadic Frame at "**FrameNames**"

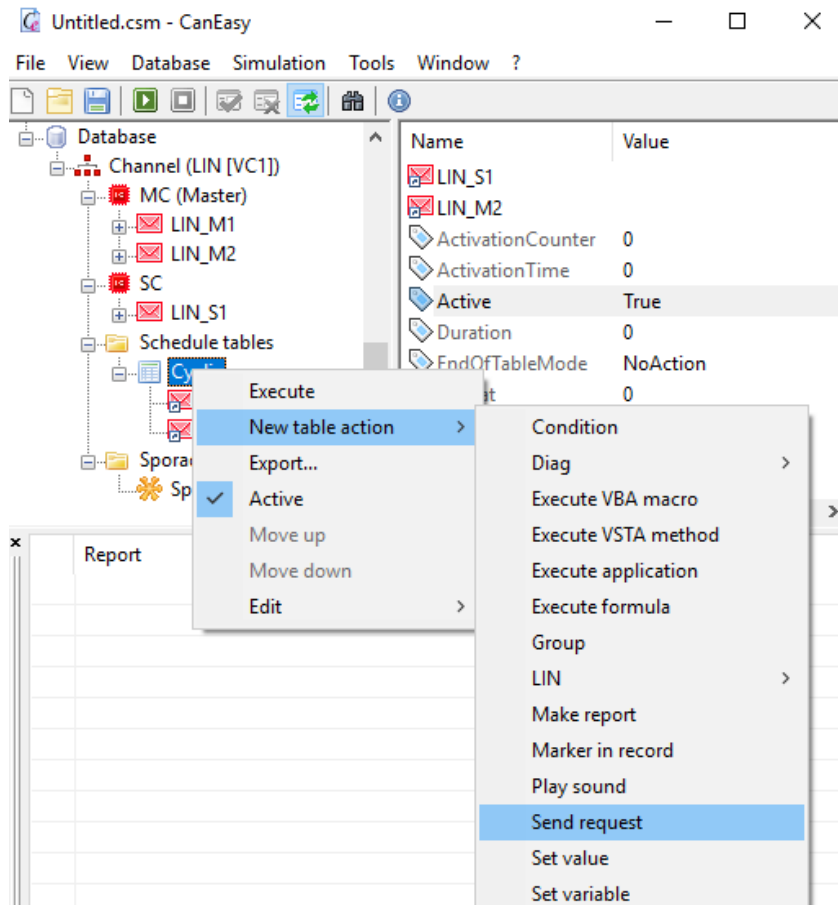
# Sporadic Frames



A Sporadic Frame appears at two positions in the CanEasy Tree:

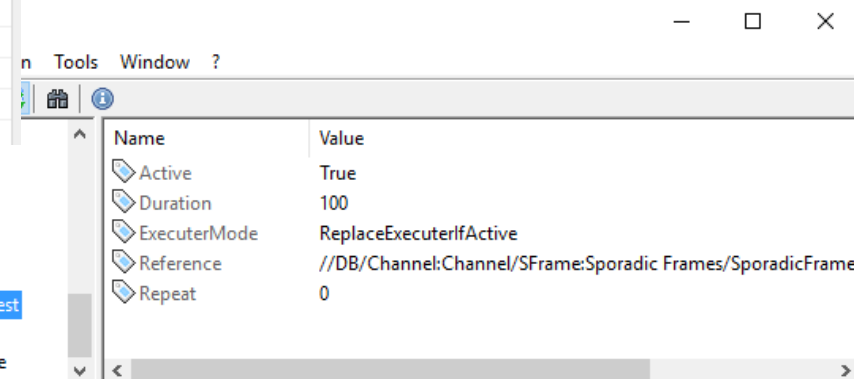
- Trigger for the Sporadic Frame in a (cyclic) Scheduler table
- Definition of the Sporadic Frame

# Add Sporadic Frames to Scheduler



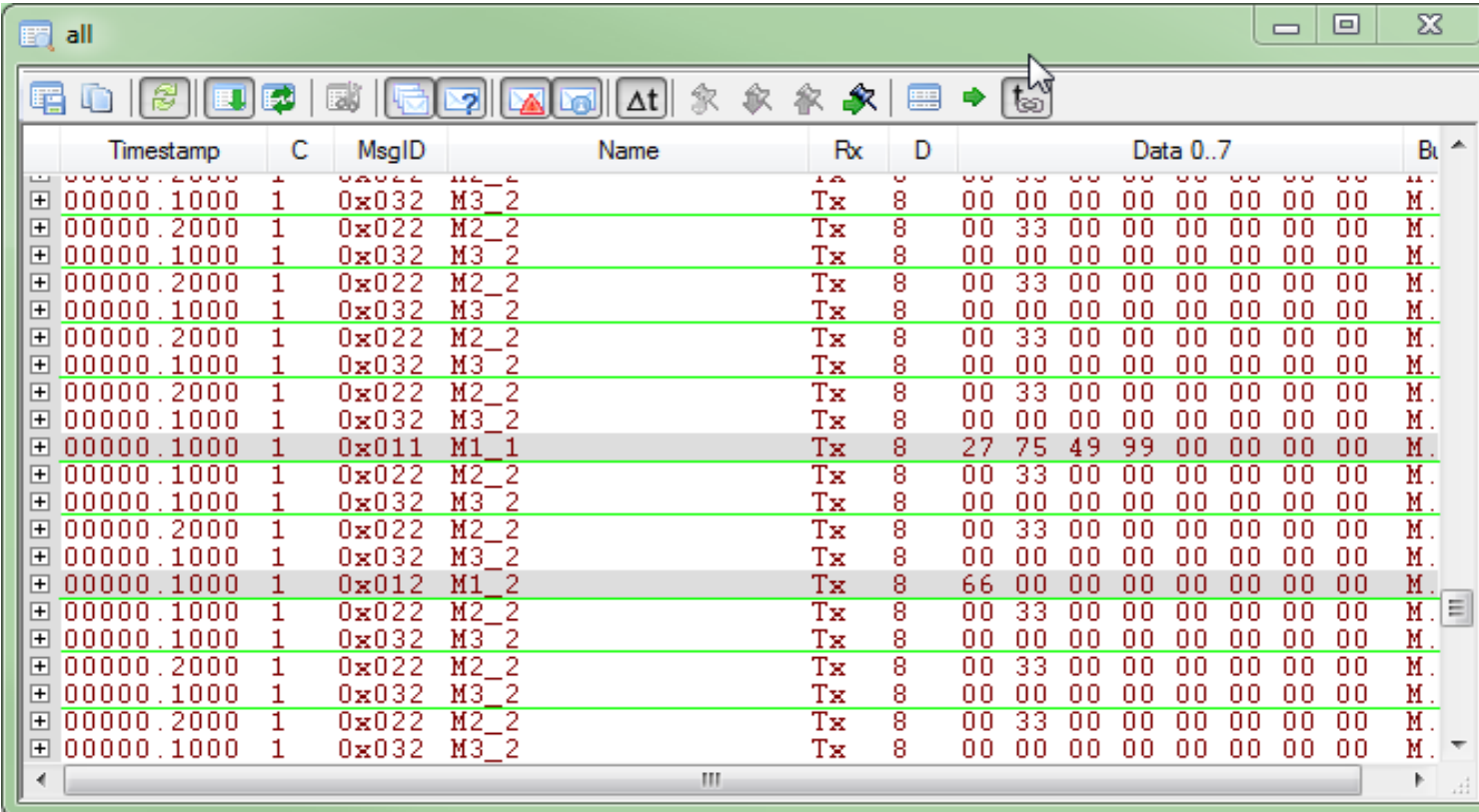
- Set the path to the Sporadic Frame as **Reference**

- Right click on the scheduler table where the Sporadic Frame should be send with
- Click on "New Table action"
- Click on "Send request"
- Enter name



# Trace of Sporadic Frames

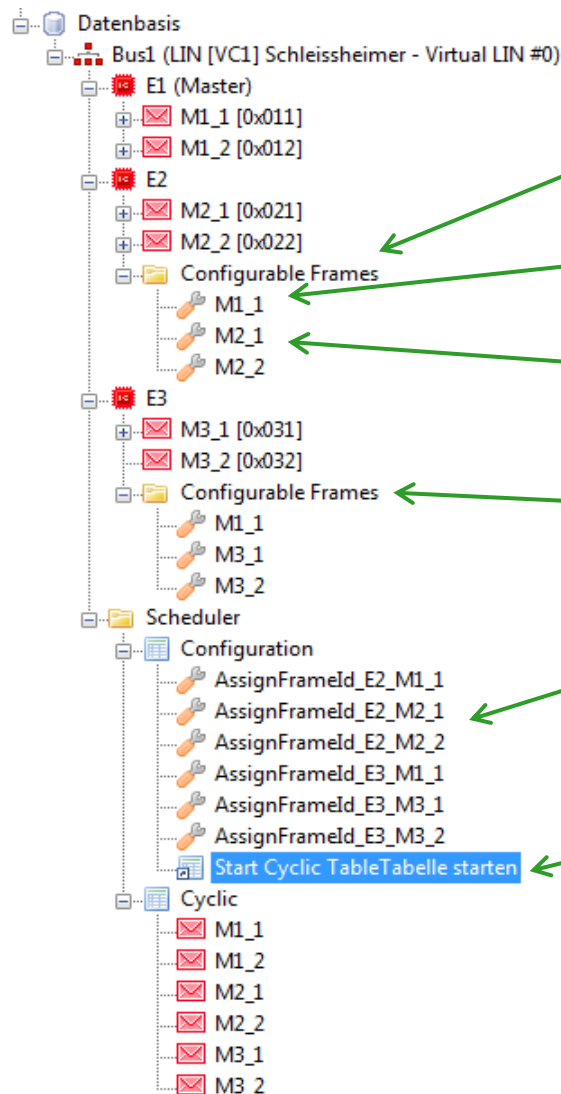
- Start the simulation and have a look into the trace
- If nothing has changed the slot of the Sporadic Frame remains empty [time to "M2\_2" is 200 ms]
- Every time a signal of a frame which belongs to the Sporadic Frame is changed it is send in the slot of the Sporadic Frame



	Timestamp	C	MsgID	Name	Rx	D	Data 0..7	Bu
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.1000	1	0x011	M1_1	Tx	8	27 75 49 99 00 00 00 00	M.
+	00000.1000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.1000	1	0x012	M1_2	Tx	8	66 00 00 00 00 00 00 00	M.
+	00000.1000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.
+	00000.2000	1	0x022	M2_2	Tx	8	00 33 00 00 00 00 00 00	M.
+	00000.1000	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	M.

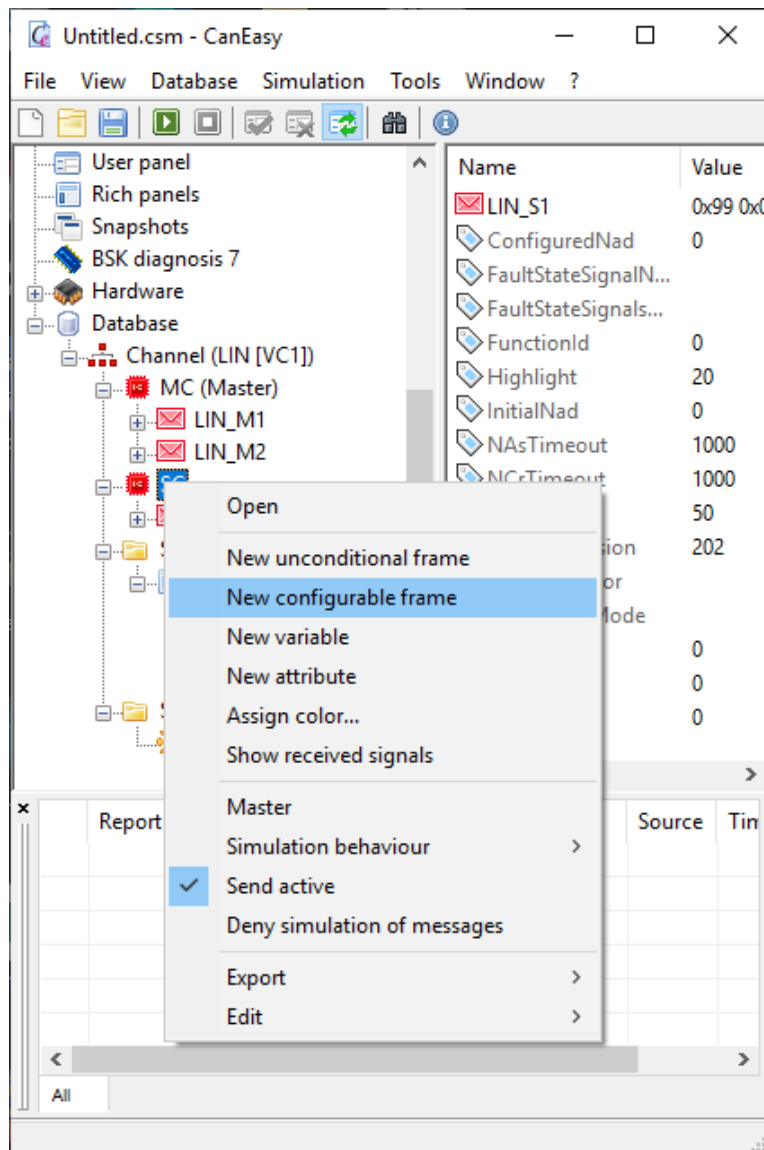
- Configurable Frames are used to configure the message IDs of the ECUs individually
- An ECU can be inserted into a LIN system without any pre-configuration
- The LIN Master ECU assigns the message IDs to all the slave ECUs at system startup

# Configurable Frames in CanEasy



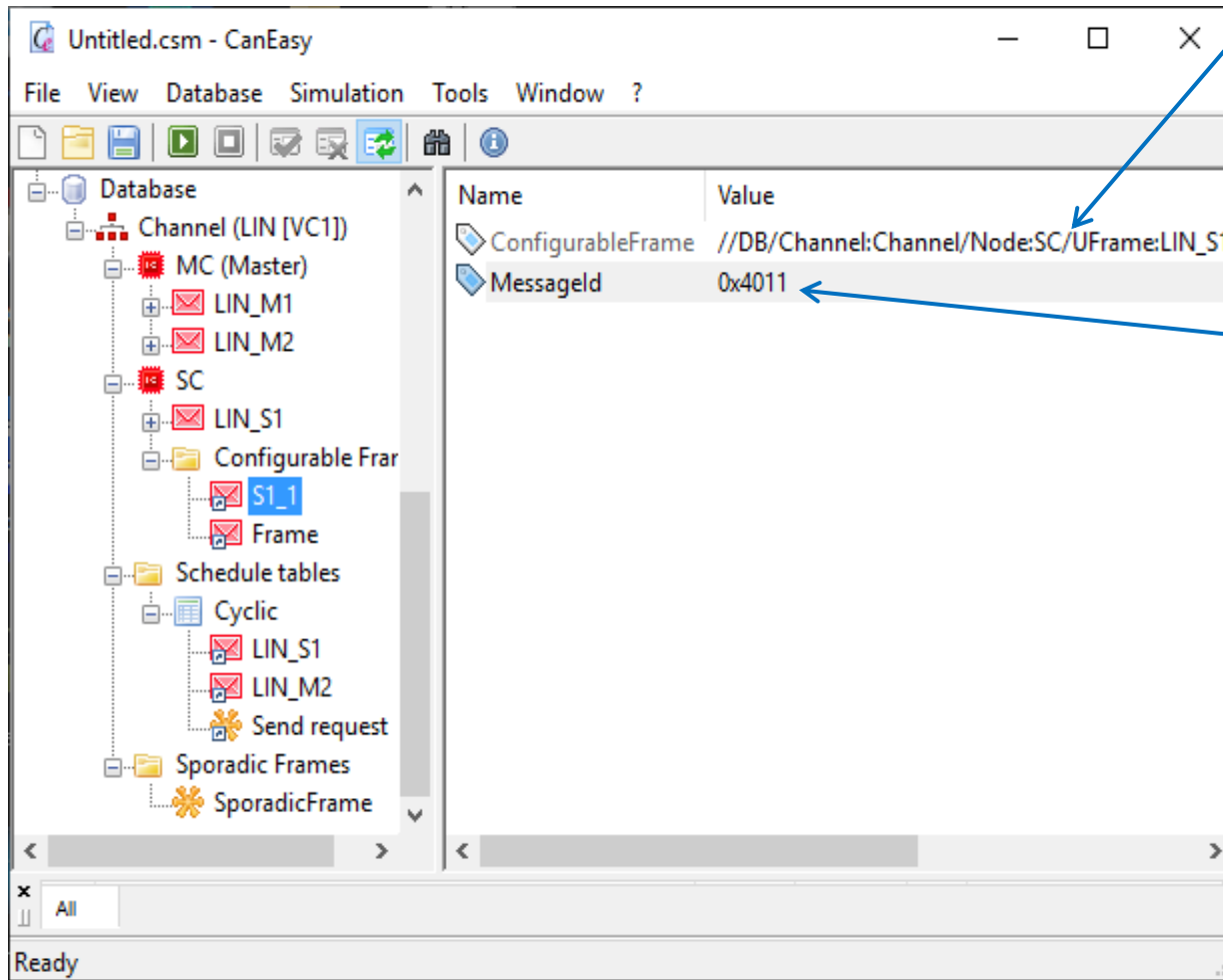
- Configurable Frames for ECU "E3"
- Configurable Master Frame for ECU "E2"
- Configurable Slave Frames for ECU "E2"
- Configurable Frames for ECU "E3"
- Schedule Table for sending the AssignFrameId messages
- Starting the "normal" cyclic Scheduler Table after the frame IDs for the configurable frames are set.

# Define configurable Frames



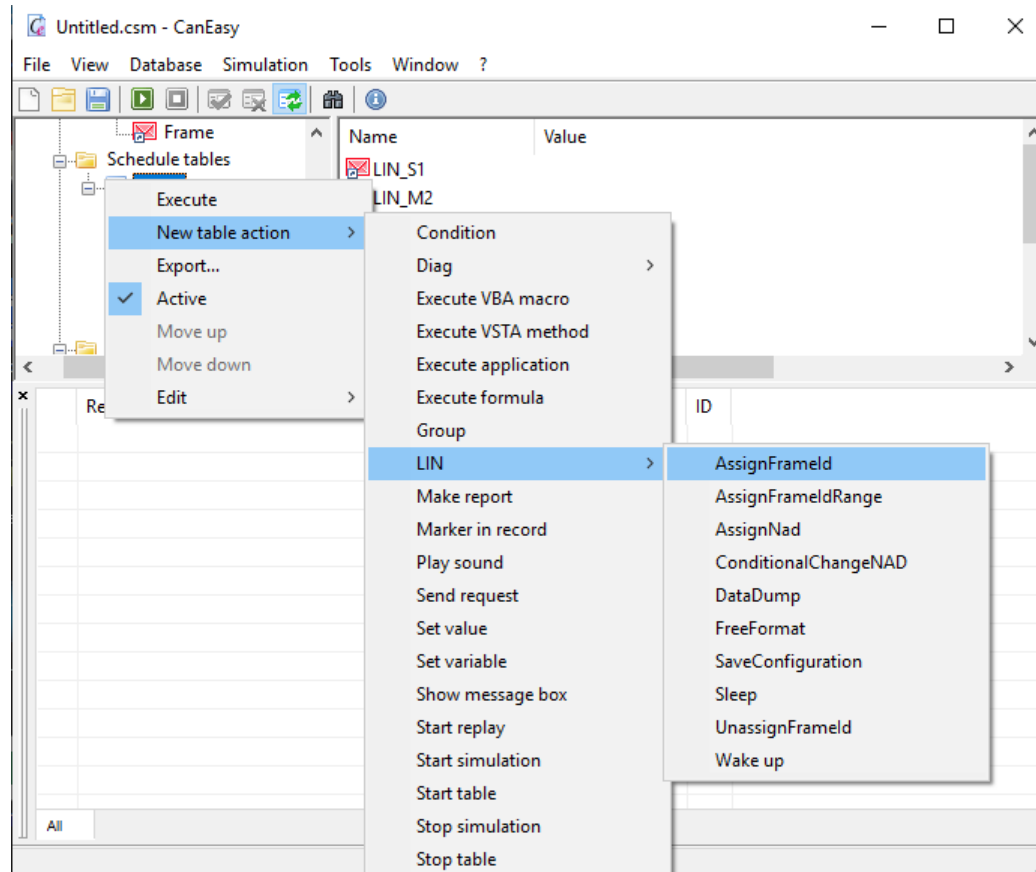
- Right click an ECU
- Click on "New Config-Frame"
- Enter the name of the message which should be configurable

# Configure configurable Frames



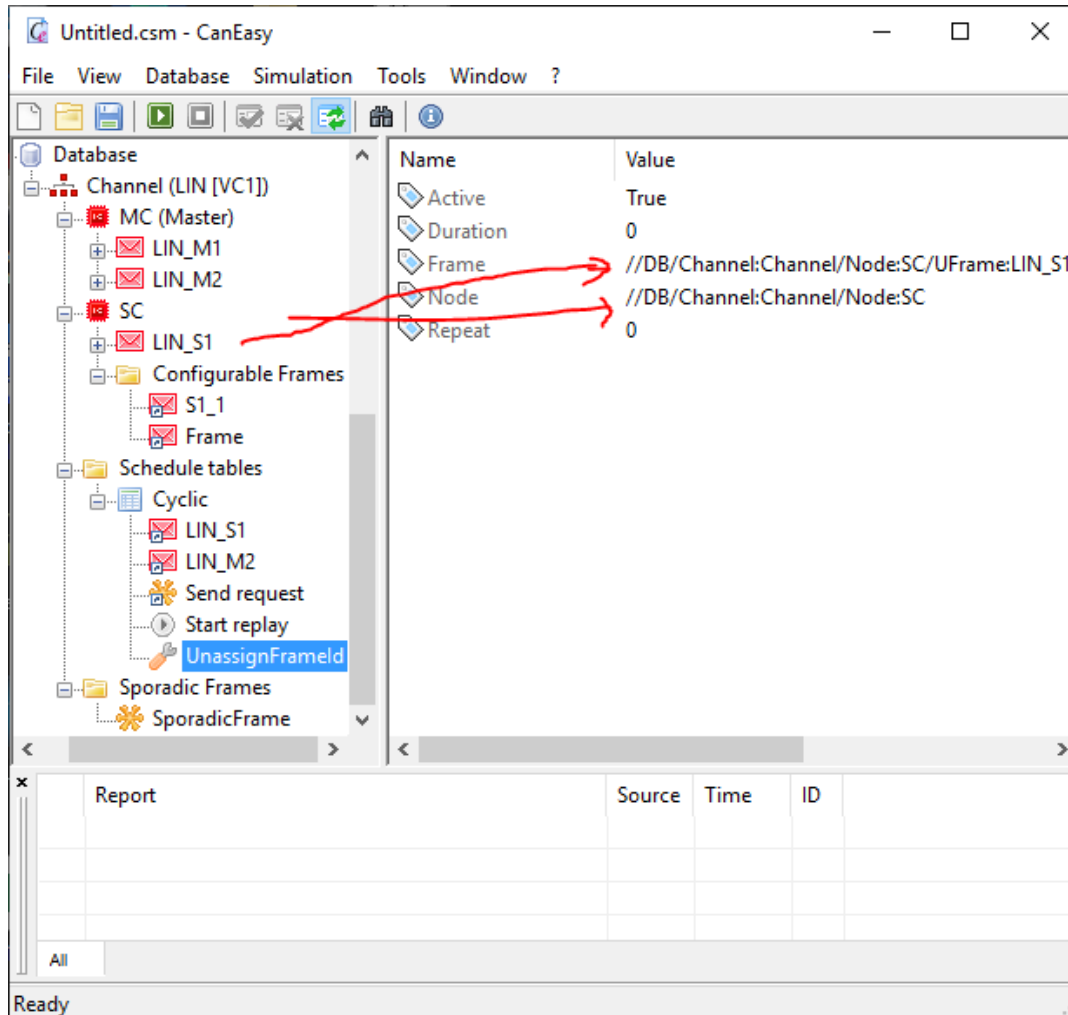
- The link to the message is automatically set
- The "MessageId" of the hardcoded internal message ID, can be configured on the right side

# Schedule Table for "Assign Frame ID"



- Add a new Scheduler Table [e.g. "Configuration"]
- Right click on table "Configuration"
- Click "New Entry / LIN / AssignFrameId"
- Enter name

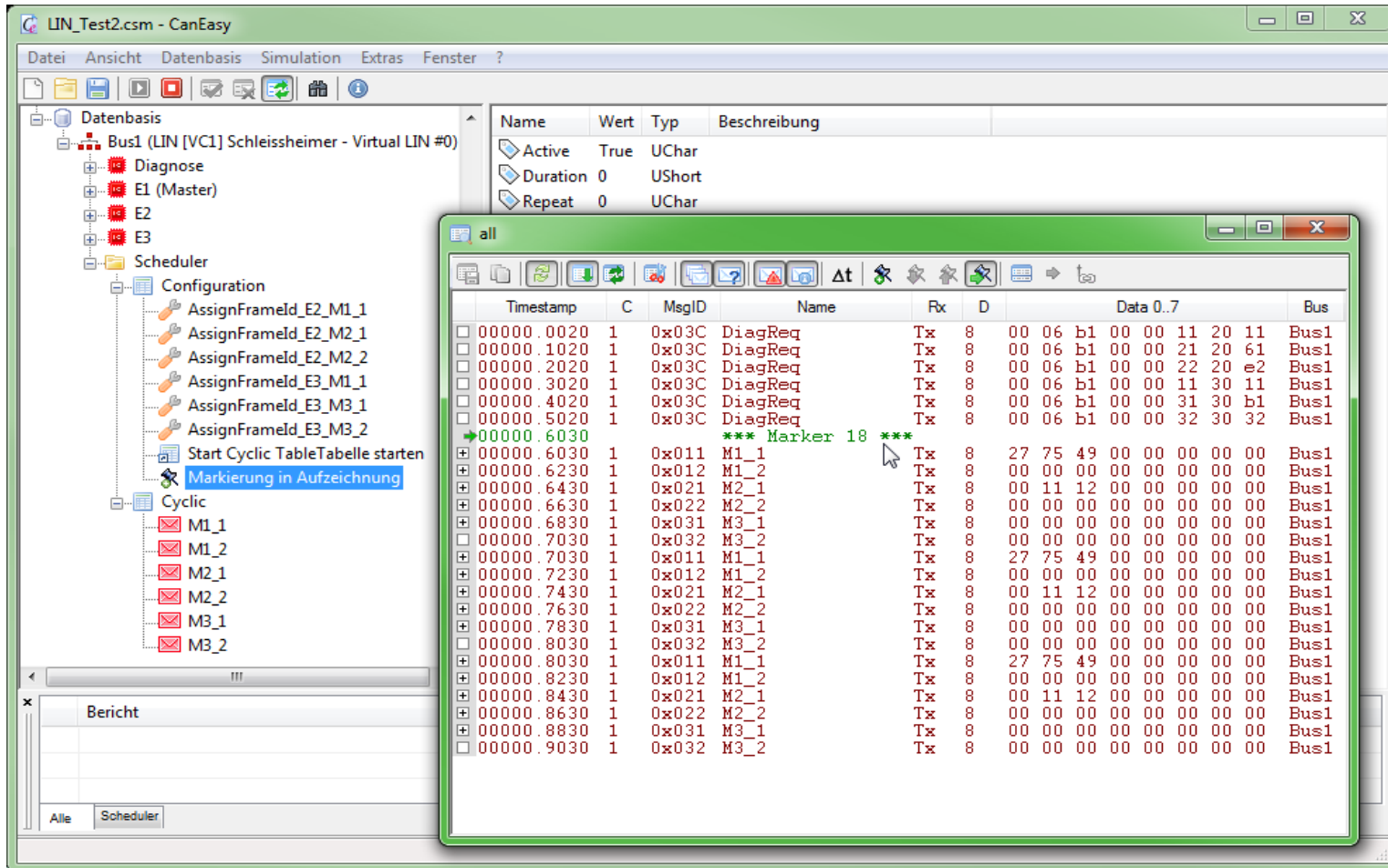
# Configure AssignFrameId message



The AssignFrameId message can be configured on the right side:

- The path of the ECU has to be entered for "Node"
- The path of the Message has to be entered for "Frame"
- The hardcoded internal message ID taken from the „Configurable Frames“ entry
- The PID is calculated with the configured Message ID underneath the ECU
- The NAD is Taken from the ECU

# Trace for Configuration and Cyclic table



The screenshot displays the CanEasy software interface for LIN\_Test2.csm. The left sidebar shows a project tree with a 'Configuration' folder expanded, listing various frame assignments (e.g., AssignFrameId\_E2\_M1\_1) and a 'Cyclic' folder containing M1\_1, M1\_2, M2\_1, M2\_2, M3\_1, and M3\_2. The main window shows a table of configuration parameters with columns: Name, Wert, Typ, and Beschreibung. The 'all' trace window is overlaid, showing a list of messages with columns: Timestamp, C, MsgID, Name, Rx, D, Data 0..7, and Bus. A green arrow points to a message at timestamp 00000.6030, labeled '\*\*\* Marker 18 \*\*\*', which is the end of the configuration table.

Name	Wert	Typ	Beschreibung
Active	True	UChar	
Duration	0	UShort	
Repeat	0	UChar	

Timestamp	C	MsgID	Name	Rx	D	Data 0..7	Bus
00000.0020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 11 20 11	Bus1
00000.1020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 21 20 61	Bus1
00000.2020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 22 20 e2	Bus1
00000.3020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 11 30 11	Bus1
00000.4020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 31 30 b1	Bus1
00000.5020	1	0x03C	DiagReq	Tx	8	00 06 b1 00 00 32 30 32	Bus1
00000.6030	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00	Bus1
00000.6230	1	0x012	M1_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.6430	1	0x021	M2_1	Tx	8	00 11 12 00 00 00 00 00	Bus1
00000.6630	1	0x022	M2_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.6830	1	0x031	M3_1	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.7030	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.7030	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00	Bus1
00000.7230	1	0x012	M1_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.7430	1	0x021	M2_1	Tx	8	00 11 12 00 00 00 00 00	Bus1
00000.7630	1	0x022	M2_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.7830	1	0x031	M3_1	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.8030	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.8030	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00	Bus1
00000.8230	1	0x012	M1_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.8430	1	0x021	M2_1	Tx	8	00 11 12 00 00 00 00 00	Bus1
00000.8630	1	0x022	M2_2	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.8830	1	0x031	M3_1	Tx	8	00 00 00 00 00 00 00 00	Bus1
00000.9030	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00	Bus1

- Because of the „Marker in recording“ action at the end of the „Configuration“ table we can detect the end of the configuration very easily in the trace.

# Assign Frame ID Range

With AssignFrameIdRange up to four message IDs can be assigned at one for one ECU.

LIN\_tutorial\_20\_AssignFrameIdRange.csm - CanEasy

Datenbasis

- Bus1 (LIN [VC1] IXXAT - USB-to-CAN FD au)
  - Diagnose
  - E1 (Master)
  - E2
    - M2\_1 [0x021]
    - M2\_2 [0x022]
    - Configurable Frames
      - M1\_1
      - M2\_1
      - M2\_2
  - E3
    - M3\_1 [0x031]
    - M3\_2 [0x032]
    - Configurable Frames
      - M1\_1
      - M3\_1
      - M3\_2
  - Scheduler
    - Configuration
      - AssignFrameIdRange\_E2
      - AssignFrameIdRange\_E3
      - Start Cyclic TableTabelle starten
      - Markierung in Aufzeichnung

Properties:

Name	Wert	Typ	Beschreibung
Active	True	UChar	
Duration	100	UShort	
ErrorText		String	
Index	0	UChar	
Node	//DB/Channel:Bus1/Node:E2	String	
PID		UChar Array	
Repeat	0	UChar	

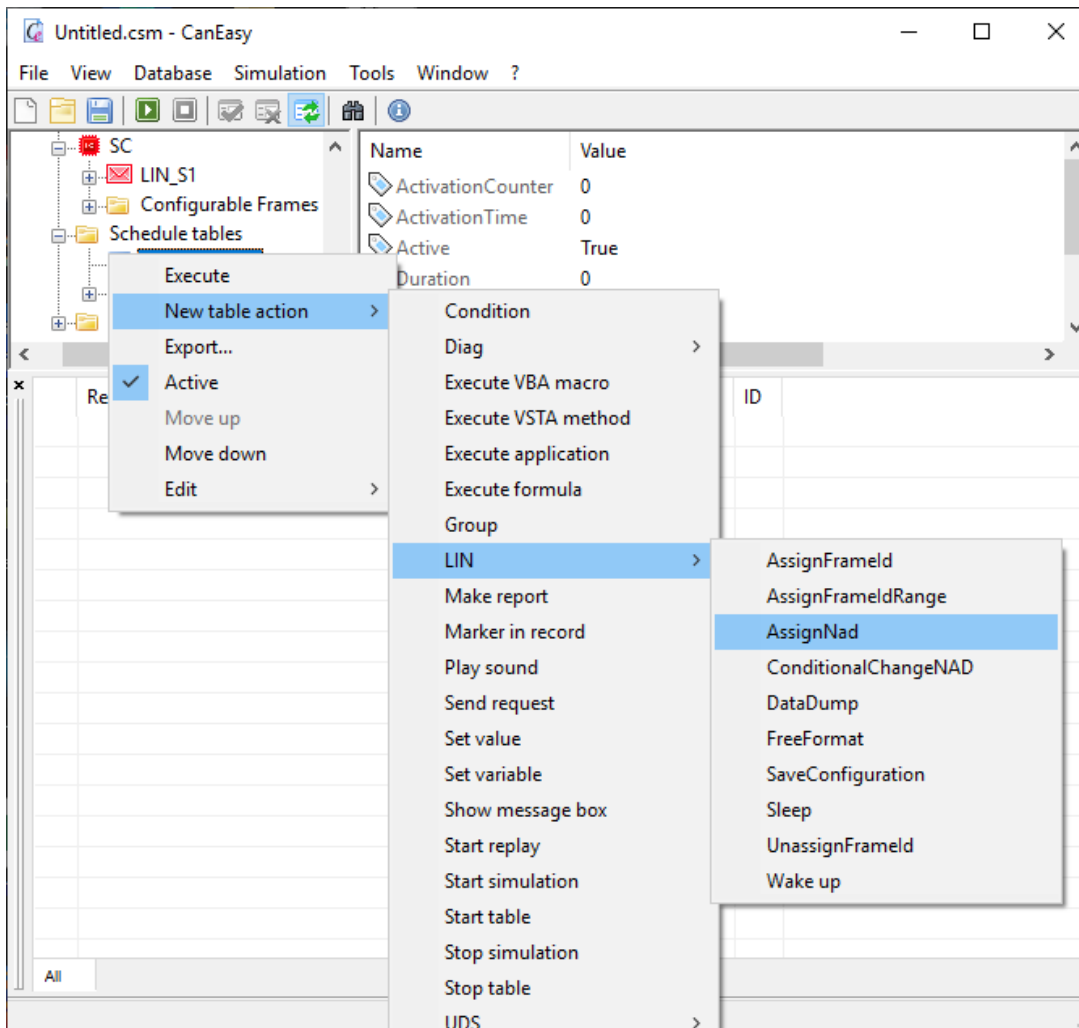
all

timestamp	C	MsgID	Name	Rx	D	Data 0..7
0.0233	1	0x03C	DiagReq	Tx	8	00 06 b7 00 11 61 e2 ff
0.1230	1	0x03C	DiagReq	Tx	8	00 06 b7 00 11 b1 32 ff
0.2160			*** Marke...			
0.2257	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00
0.2460	1	0x012	M1_2	Tx	8	00 00 00 00 00 00 00 00
0.2659	1	0x021	M2_1	Tx	8	00 11 12 00 00 00 00 00
0.2860	1	0x022	M2_2	Tx	8	00 00 00 00 00 00 00 00
0.3059	1	0x031	M3_1	Tx	8	00 00 00 00 00 00 00 00
0.3259	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00
0.3330	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00
0.3459	1	0x012	M1_2	Tx	8	00 00 00 00 00 00 00 00
0.3659	1	0x021	M2_1	Tx	8	00 11 12 00 00 00 00 00
0.3859	1	0x022	M2_2	Tx	8	00 00 00 00 00 00 00 00
0.4059	1	0x031	M3_1	Tx	8	00 00 00 00 00 00 00 00
0.4259	1	0x032	M3_2	Tx	8	00 00 00 00 00 00 00 00
0.4330	1	0x011	M1_1	Tx	8	27 75 49 00 00 00 00 00

- Add „AssignFrameIdRange“ message to a scheduler table
- Set the path to the ECU for “Node”
- Set the index of the first message which should be considered
- The message IDs are set in the order of the messages are contained in the “Configurable Frames” configuration

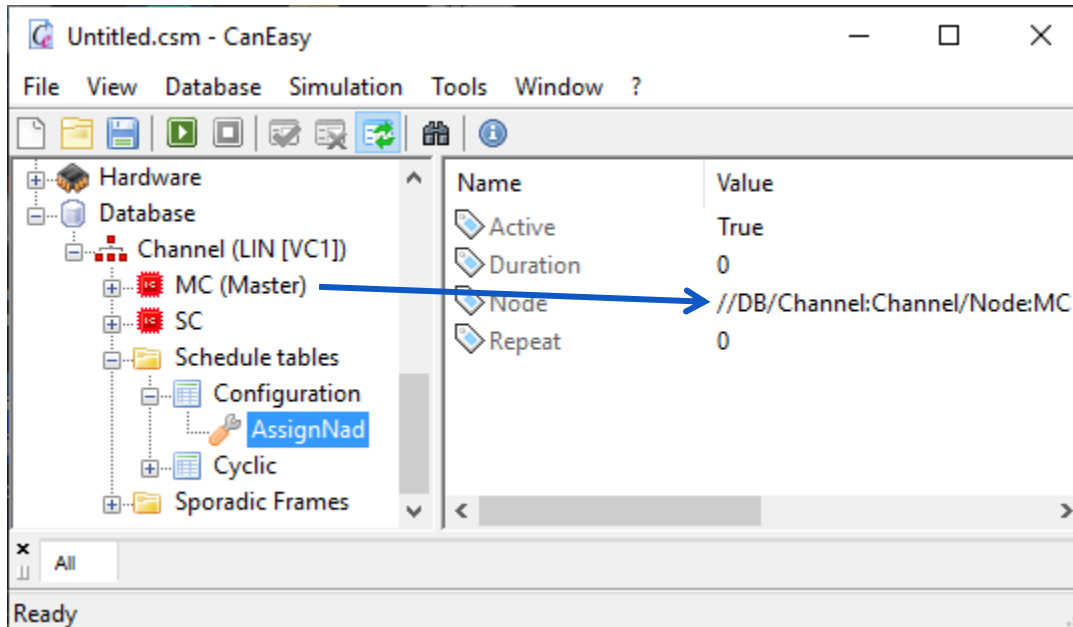
# Assign NAD

With AssignNAD a new Node Address can be assigned to a slave

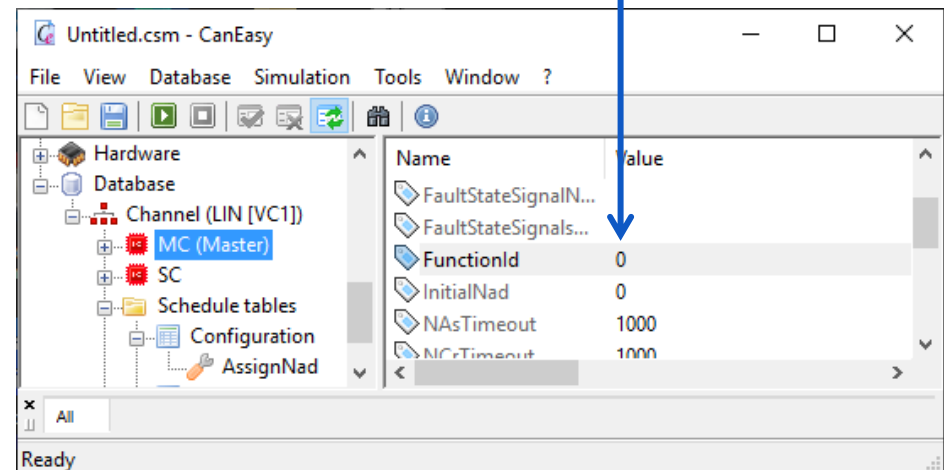


- Right click on a scheduler table
- Click on "New Table action" / "LIN" / "AssignNAD"
- Enter name

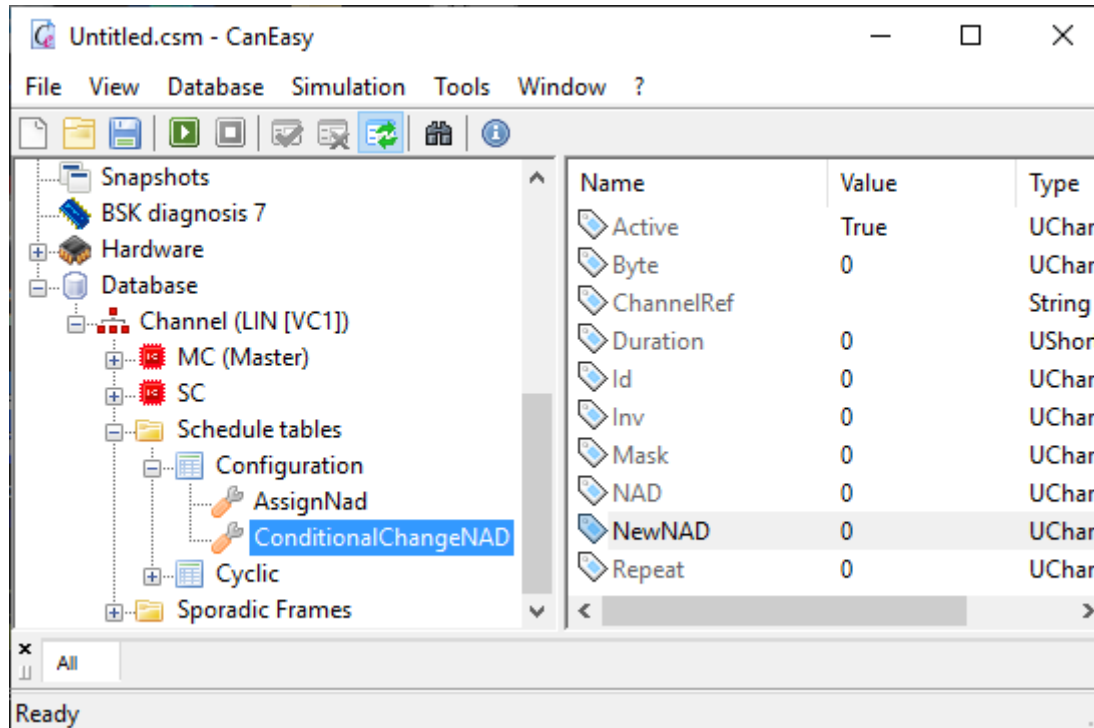
# Configure Assign NAD message



- The only thing which has to be configured is the "Node"
- Enter the path to the ECU as "Node"
- The settings for "Supplier ID" and "Function ID" are taken from the ECU configuration



# Conditional Change NAD



- Right click on a scheduler table
- Click on  
"New Table action" /  
"LIN" /  
"ConditionalChangeNAD"
- Enter name
- Configuration can be done at the right side:  
**NAD** and **NewNAD**  
**Mask** and **Inv**  
**ID** and **Byte**

Thank you for your attention!

:-)



+

