toolchain
- Quick Start Guide -

designation:	**toolchain – Quick Start Guide -**
version date:	**01.02.17**

Table of Contents

Introduction

This manual is especially for users who have little or no experience in dealing with CAN bus systems. In addition, the handling of the miunske Toolchain is explained by means of a simple project. The toolchain is provided free of charge and can be downloaded from miunske website at https://miunske.com/download

When the toolchain is started for the first time, a sample project is generated, which contains all configurable miunske devices. This project is the basis for this quick guide.

Detailed explanations of the individual product-specific functions can be accessed via the menu “HELP -> “Instructions”.

The readout and writing of the miunske CAN products can be realized with a PCAN-Peak USB / PCI adapter or CANfox-Sontheim USB adapter. The necessary drivers are provided by the installation wizard of the toolchain. No other software is needed to read and write the miunske CAN products.

This document explains the communication via PCAN Explorer and the Sontheim CAN Explorer.

Information on the Peak products can be found on the manufacturer's website Http://www.peak-system.com.

Information on Sontheim products can be found on the manufacturer's website at https://www.sontheim-industrie-elektronik.de.

1 CAN Communication (strong Simplified)

CAN devices can send and receive data. **Simultaneous sending and receiving is not possible**, exceptions are devices that have two or more CAN transceivers and are located in the same CAN network.

To communicate in a CAN network, each message has a unique identifier (ID). It is not possible that two or more CAN devices can send a message with the same identifier at the same time.

If a device supports the sending of several different CAN messages, these also require unique identifiers. For example, an I/O module can send the status of its analog and digital inputs in two different messages in different update rates, so these CAN messages also require different IDs.

All devices, in a CAN bus, must be configured to the same baud rate (transfer speed) in order to communicate with one another at all.

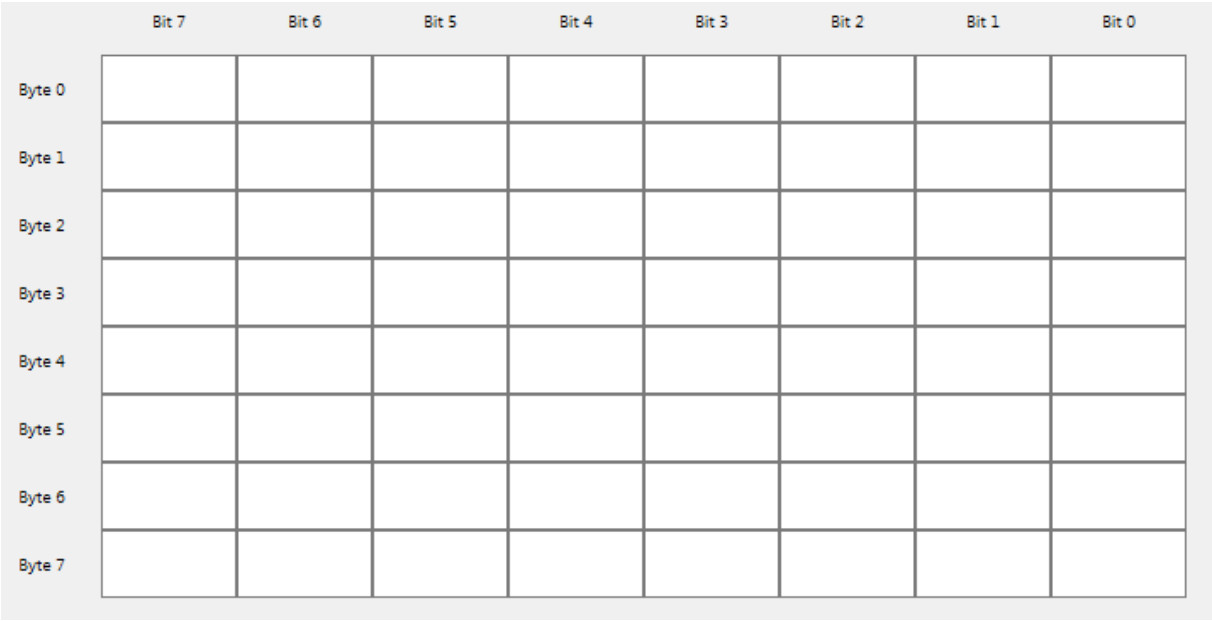
A CAN network consists of at least two components. (Example: CAN I/O module and keyboard)

In order to be able to communicate between two or more devices, a termination of 120 Ohm must be available at the two most widely distributed CAN nodes, between CAN-H and CAN-L.

1.1 CAN Message

Within a CAN network the messages (ID and message) are displayed in the **hexadecimal system** (0x ... = hexadecimal notation).

A message always consists of the ID and the data. The data is composed of a maximum of 8 bytes of 8 bits each. The count of bits and bytes starts at 0.



Messages, within a CAN network, can be displayed and monitored using tools (e.g.: PCAN-View, CAN Explorer, etc.).

The messages are always structured according to the following scheme. The data length can vary. For example, only 2 bytes can be transmitted.

ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7

Example of a CAN message

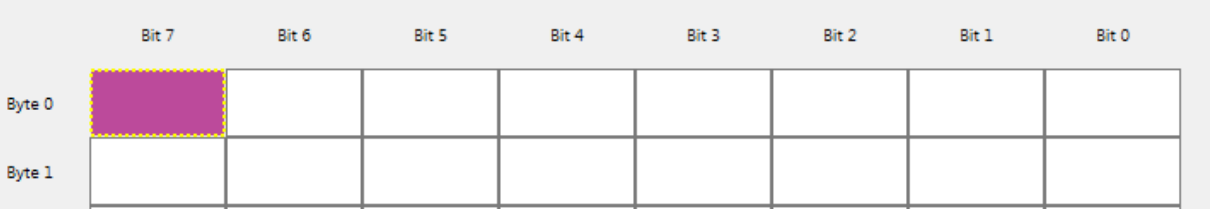
0x666 00 00 00 00 00 00 00 00
[ID][8 Byte Data]
[Message]

Each byte in the CAN network is displayed as follows.

bit position	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
value	8	4	2	1	8	4	2	1

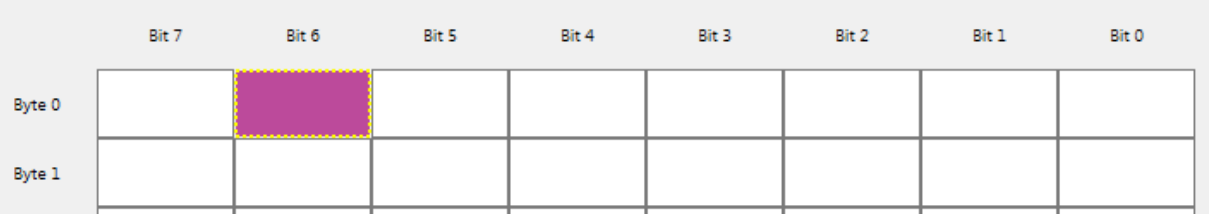
Example: Bit position und message

Bit position 7 is set in byte 0.



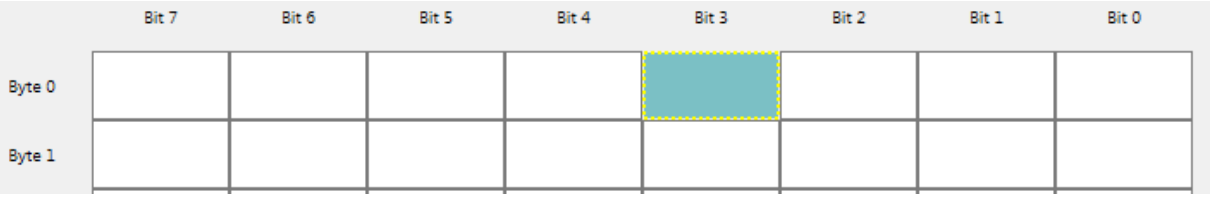
CAN message: 0x666 80 00 00 00 00 00 00 00

Bit position 6 is set in byte 0.



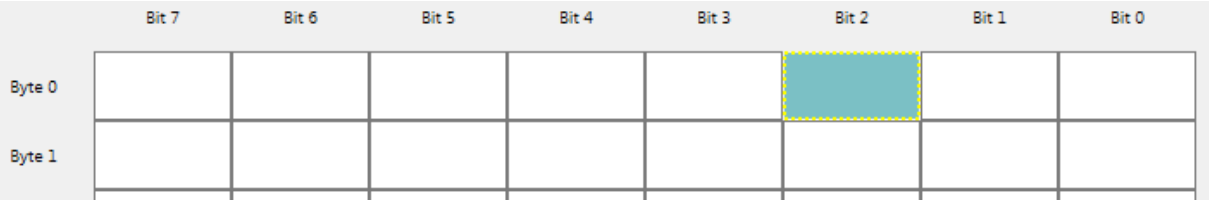
CAN message: 0x666 40 00 00 00 00 00 00 00

Bit position 3 is set in byte 0.



CAN message: 0x666 08 00 00 00 00 00 00 00

Bit position 2 is set in byte 0.



CAN message: 0x666 04 00 00 00 00 00 00 00

2 The Miunske-toolchain

2.1 Short explanation of the terms

workspace ->	Is a collection of project maps. A workspace can contain several solutions.
solutions ->	Projects are managed in a solution. A solution could, for example, be a customer and the projects underneath are different vehicles with different equipment and functions.
project ->	In a project, the miunske products required or used for your project are managed.
device ->	Miunske products that are needed for this project and can be configured using the toolchain. (e.g.: CAN keyboards, CAN I/O modules, etc.)
symbolfile ->	All CAN buses with the associated CAN IDs / messages are managed in a symbolfile. These can be provided with corresponding names. This allows a fast and unambiguous assignment between signals and functionality / parameters.
high activ ->	It is changed from logic 0 to logic 1.
low activ ->	It is changed from logic 1 to logic 0.

2.2 Creating an example project

In the following section, a project is created in the toolchain. The project is assigned to a device and corresponding parameters are set.

2.2.1 Creating a solution

Via the menu "FILE -> New -> Solution" a new project folder is created in the workplace. Location and name can be selected freely. It is displayed as a tree structure in the Workplace window.

2.2.2 Create a new project in the solution

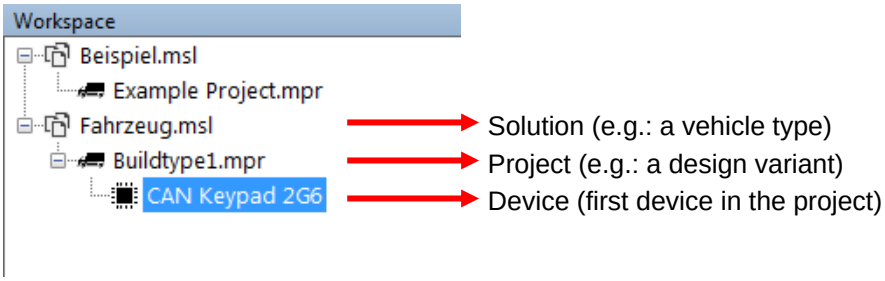
Select "FILE -> New -> Project". When a new project is created, the location and the name can also be freely selected. It is recommended to create a new folder in the file system.

Note: By default, the symbolfile of the "Example Project" is used, so it is not necessary to create your own symbolfile. If, however, a new CAN bus or a new symbol line is to be created for the

project, this can be read in the "Symbol Manager" document. ("HELP -> Instructions -> Symbol Manager")

2.2.3 Adding a device (CAN keyboard 2G6)

Via the menu "FILE -> New -> Device" the desired device can be selected and added to the project. In the example the device CAN keyboard 2G -> CAN keyboard 2G6 is added to the project.



3 Example of minimum parameterization CAN keyboard 2G6

First, the device (CAN keyboard 2G6) is selected in the project. In the middle part of the toolchain the settings for the keyboard are displayed. (For detailed explanation see "HELP -> Instructions -> CAN KEYBOARD 2G")

3.1 Definition of switching or display elements

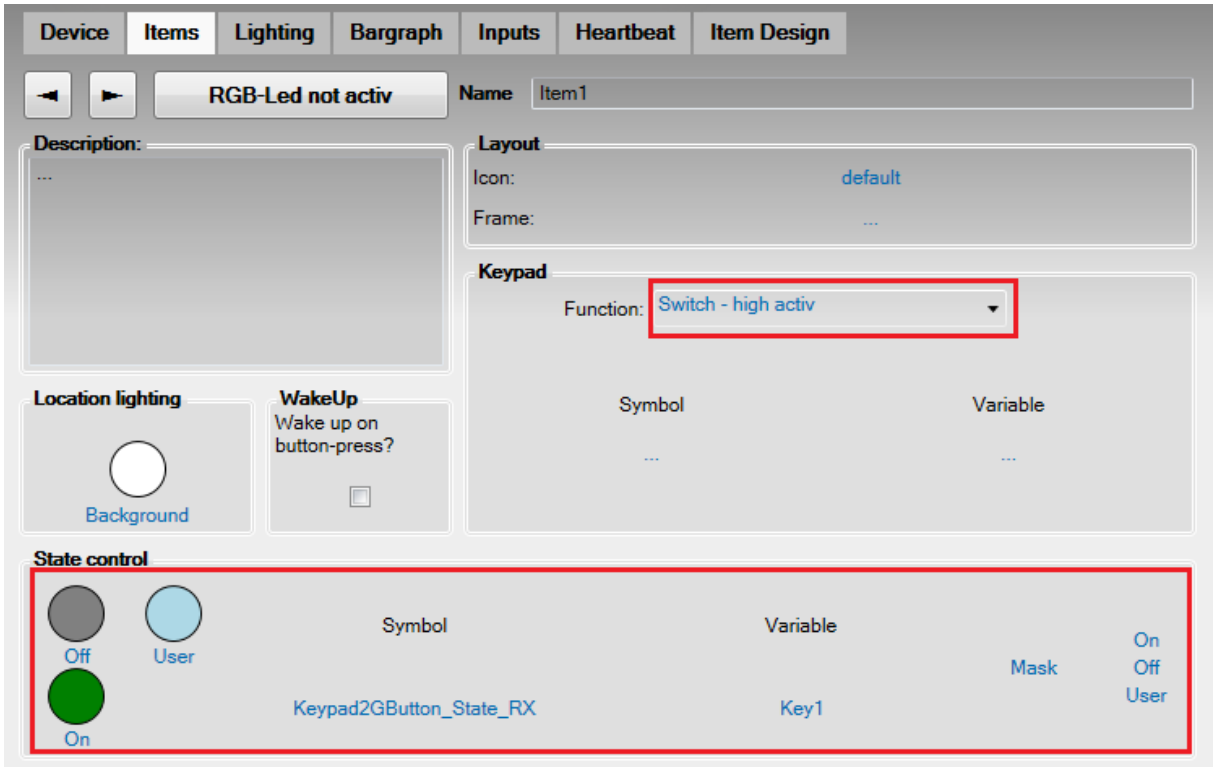
The settings for the individual switching or display elements are made under the tab fields. By selecting the field to be parameterized on the keyboard layout, the settings can be changed.

Example:

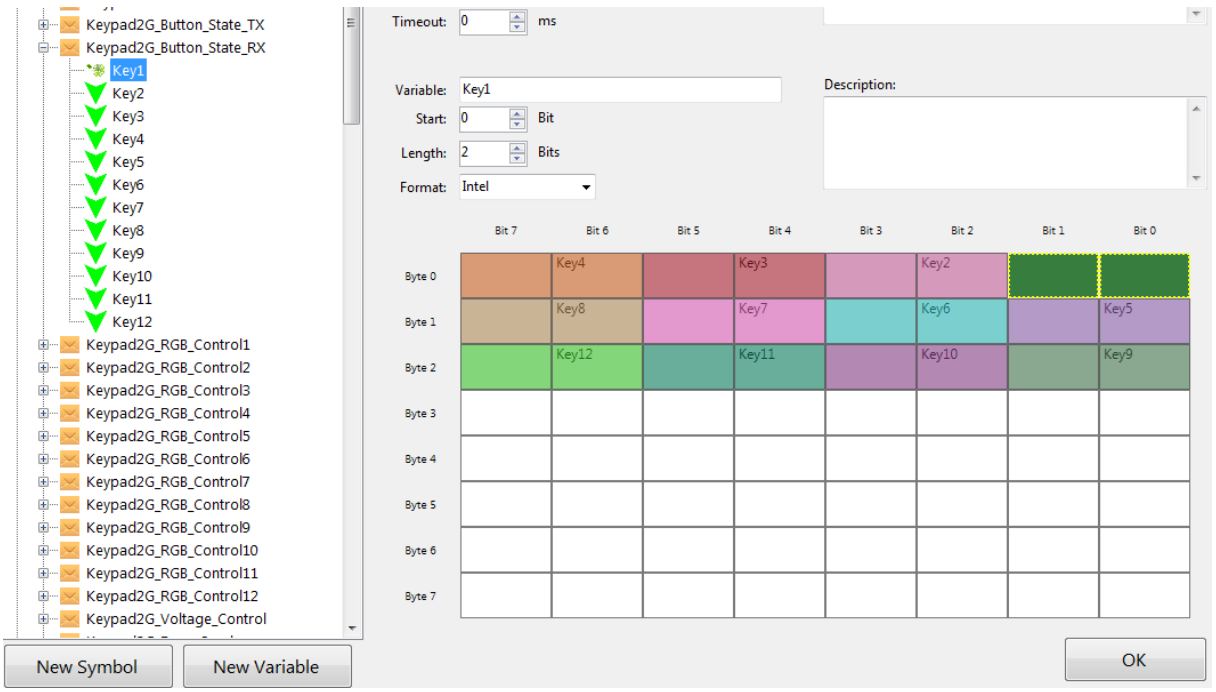
Key 1 is to be used as a pure display.

Proceed:

Button 1 is selected in the keyboard layout. The function "no function" is selected in the "Keypad" area. Pressing the key does not send a CAN message. In order to use the button as a display element, colors are assigned to the various states via the "State control" area.



Finally, the CAN message is defined, with which the color change of the RGB LED is to take place.



The desired CAN message for the status change is assigned by selecting a symbol in the Drive state. To do this, select "..." and the Symbol Manager is opened.

Here you can find all defined symbols and variables, which are only to be selected. If the desired message / ID, with the corresponding information does not exist, a new

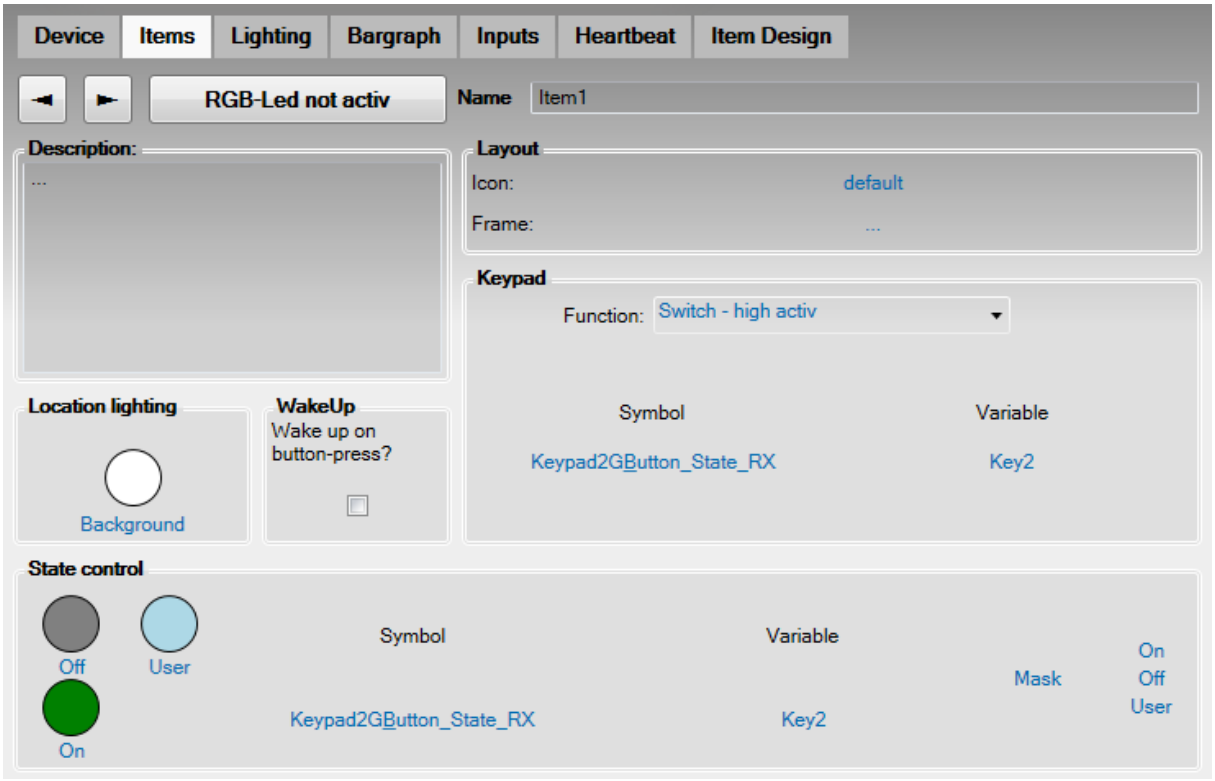
message (new symbol) can be created and configured with the desired information (new variable). Confirm the selection with "OK".

"Keypad2GButton_State_RX" with the variable "Key 1" has been selected as the symbol. The status change is now controlled by the CAN message with ID 0x12002 (bit positions 1 and 2 in byte 0).

Key 2 is to be used as a switch - High active

Proceed:

Button 2 is selected in the keyboard layout. The "Switch - high active" function is selected in the "Keypad" area. The CAN message is then assigned to the field, which indicates a change in the actuation (ON / OFF) on the CAN bus. The control of the RGB LED is independent of the set function.

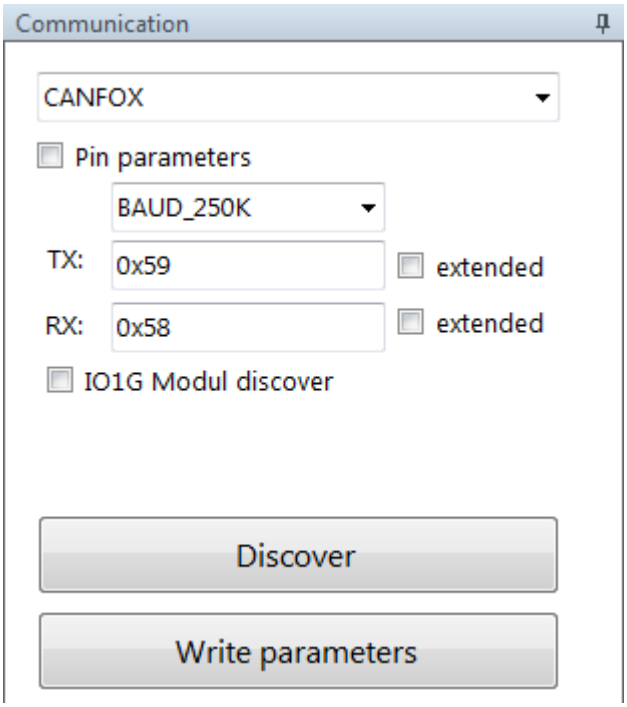


The change of the RGB LED is sent by the switching subscriber via CAN message. In order for the RGB LED to change and display the status, this is previously parameterized to the corresponding CAN message. This is done analogously to the example for the application as a pure display field.

3.2 Write the parameterization to the device

Once the settings for the parameterization of the device have been completed, these must be transferred to the device.

Interface and baud rate must be adapted according to the device configuration. (Default baud rate of the miunske CAN products is 250 kBit/s, except the CAN I/O modules 1G these have a baud rate of 500 kBit/s)



With the "Write Parameters" button in the Communication window, the configuration is transferred to the device.

If the data has been successfully transferred to the keyboard, a corresponding message is displayed. This is also signaled if the transmission is faulty or aborted. The progress of a transfer can be tracked at the bottom left of the toolchain window. In addition, information about the current activities of the toolchain and its communication is printed in text form to the "Output" window.

3.3 Test of the parameterization

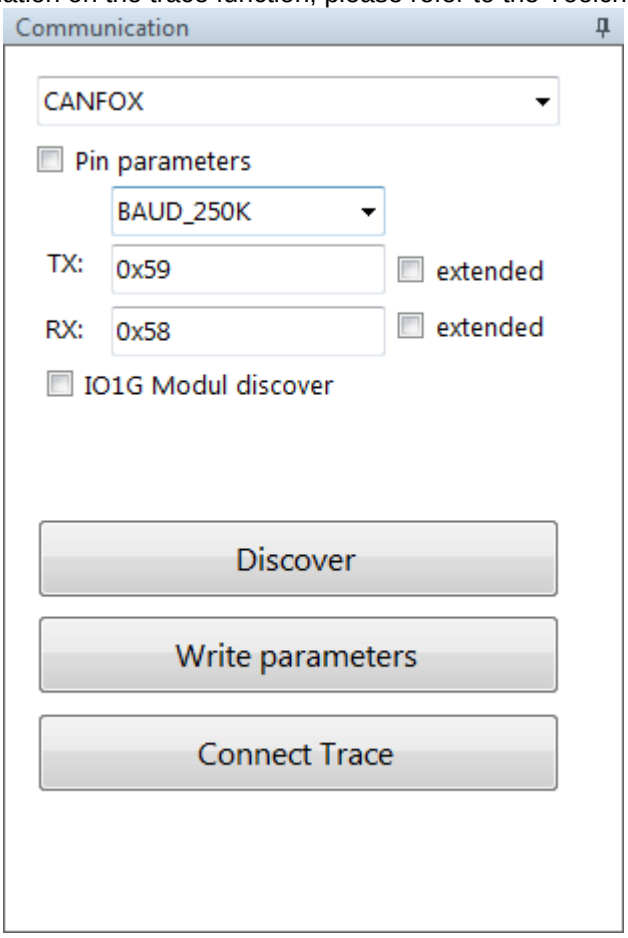
If the parameters have been successfully transmitted, pressing key 2 causes this actuation to be displayed in the previously set CAN message.

Since at least two devices in the same network (CAN bus) are required for the CAN communication, the reading and sending can be simulated with a CANFOX (Sontheim CAN Explorer) or a PEAK (PCAN Explorer) interface.

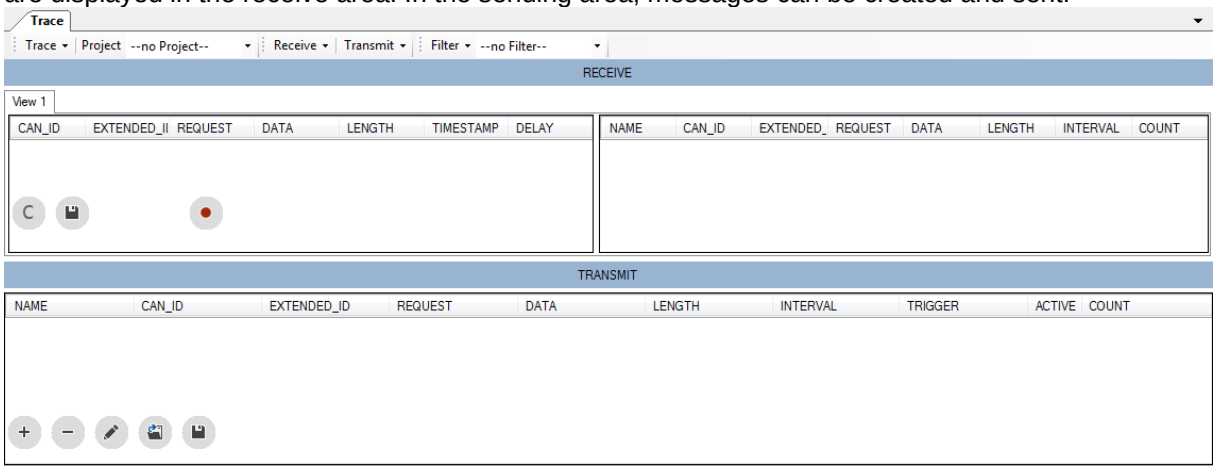
3.3.1 Example toolchain Trace function

The trace function of the Toolchain makes it possible to send and receive messages in a CAN network without further tools. It is available starting with Toolchain version 4.0.

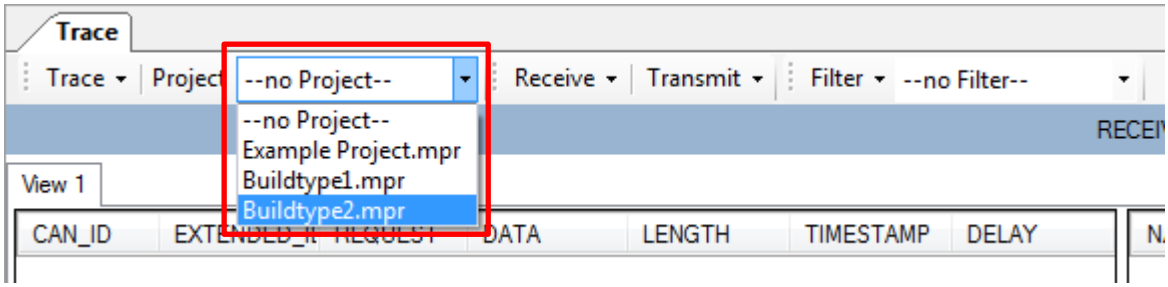
The trace is accessed using the "Connect Trace" button in the Communication window or "VIEW". The settings for the interface and the baud rate are accepted according to the settings in the Communication window. As long as the trace function is active, no parameters can be read or written with devices. If the trace is to be left, it must be terminated via the "Disconnect Trace" button.
For further information on the trace function, please refer to the Toolchain manual.



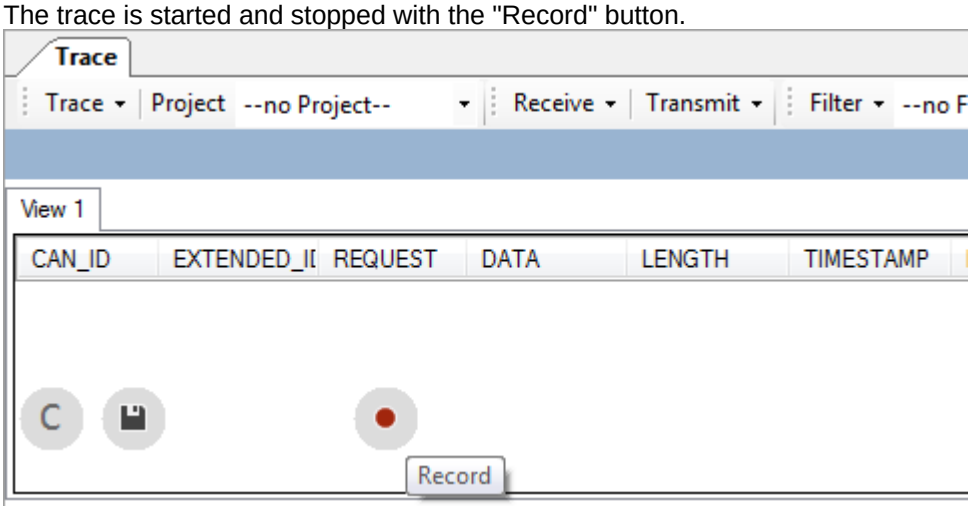
The trace window is divided into the **RECEIVE** and **TRANSMIT** areas. All received messages are displayed in the receive area. In the sending area, messages can be created and sent.



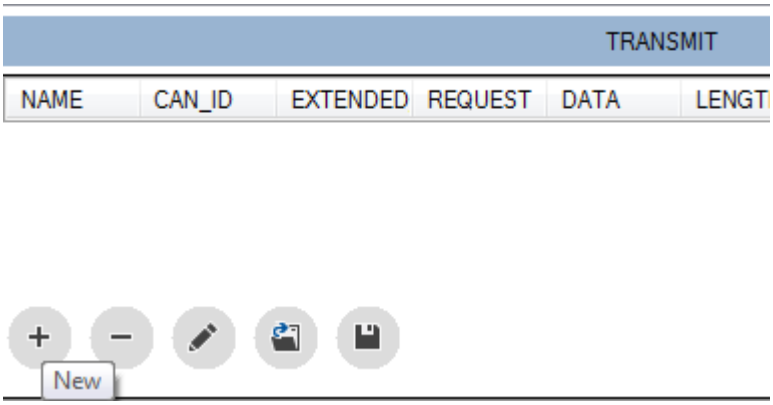
The receive area is in turn divided into two areas. On the right, all messages are displayed in chronological order. In the left pane, the received messages are summarized. The number and the interval are determined for each message.
A trace can be assigned directly to a project. This allows the received and sent messages to be matched with the messages stored in the Symbol Manager. The assignment of a project takes place via the selection list Project in the trace menu. All projects of the current toolchain workspace are displayed.



The trace is started and stopped with the "Record" button.



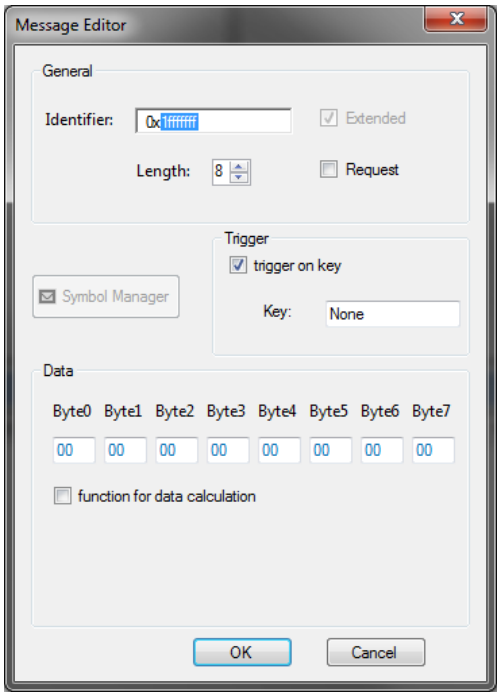
You can add, delete, and edit messages in the **SEND** section. The resulting send list can be saved or reloaded.



Messages can only be sent and received by pressing the "Record" button after starting the trace. All messages from the CAN network are displayed.

In the **SEND** view, you can create messages using the "+" (New) button. The message editor is opened and the corresponding message is edited. With the "OK" button, the message is accepted and added to the current send list.

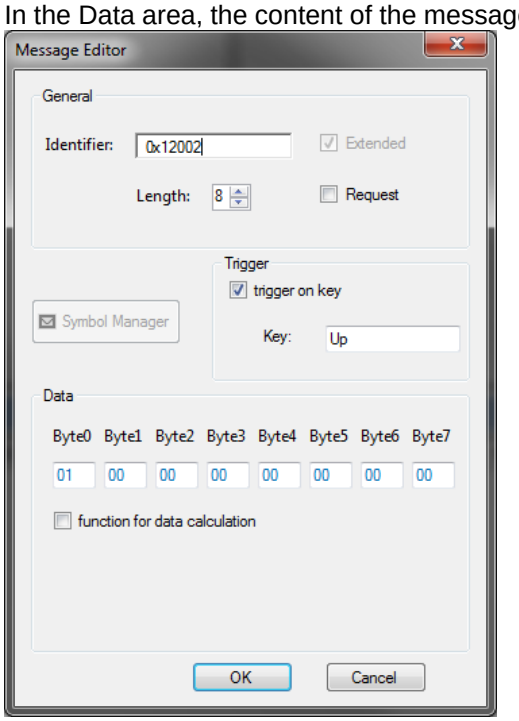
Note: If the trace has been linked to a project, the messages can be loaded directly from the Symbol Manager.



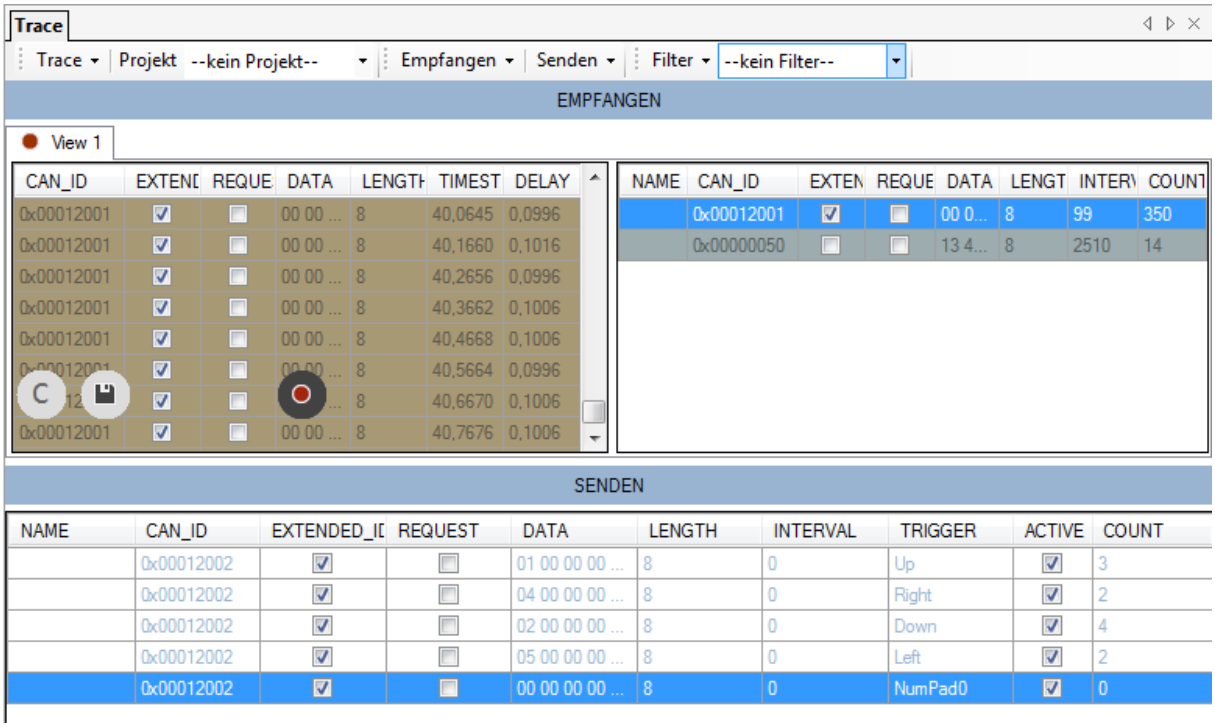
In the message editor, the message ID (identifier) and its length are defined first. The identifier is stored accordingly for the state control, as set in the toolchain. The "Extended" checkbox is set automatically for the corresponding Extended IDs (29-Bit).

A key on the keyboard can be assigned to the message in the "Trigger" area. To do this, the "Key" field is activated and the desired key on the keyboard is pressed.
If the checkbox "trigger on key" is deactivated, a time interval is expected in milliseconds. In this case, the message would be sent cyclically.

In the Data area, the content of the message is set for each byte.



Example: CAN message with the identifier 0x12002 with the length 8 bytes is to send the bit position 1 active as high active when pushing the key on the up arrow key in byte 0.
If all required messages have been created successfully, they are displayed in the Send list, in the area **SEND**. With the help of the checkbox in column ACTIV, it is determined whether the message should be sent or not. If the checkbox is active, the message is sent either to the set keypress or the stored time interval.



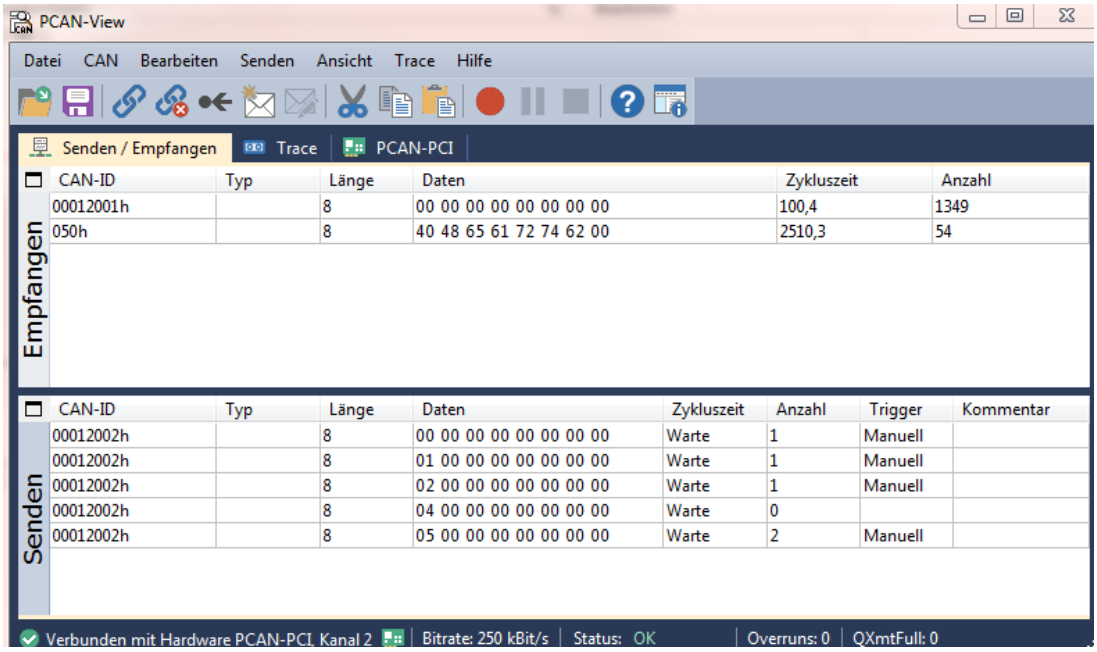
If the key assigned to the message is pressed on the PC keyboard, the parameterized LED of the CAN keyboard changes the status color.

3.3.2 Example PCAN Viewer with a Peak CAN Interface

When the PCAN Viewer is started, the interface and the desired bit rate are queried. The transmission and reception view is then displayed. The key 2, which has been parameterized as a pushbutton or switch, can now be pressed on the keyboard. The CAN ID, the key, and the corresponding bit position are then displayed in the PCAN viewer.

To change the status display, a Can message has to be sent to the keyboard. In the PCAN viewer, a new message is inserted in the "Send" area. The CAN ID is set according to the state control in the toolchain.

The following graphic shows how the RGB LED can be changed using CAN messages.



Double-click the corresponding message to send it to the keyboard. The button then changes the color.

3.3.3 Example CAN Explorer with a CANfox CAN interface

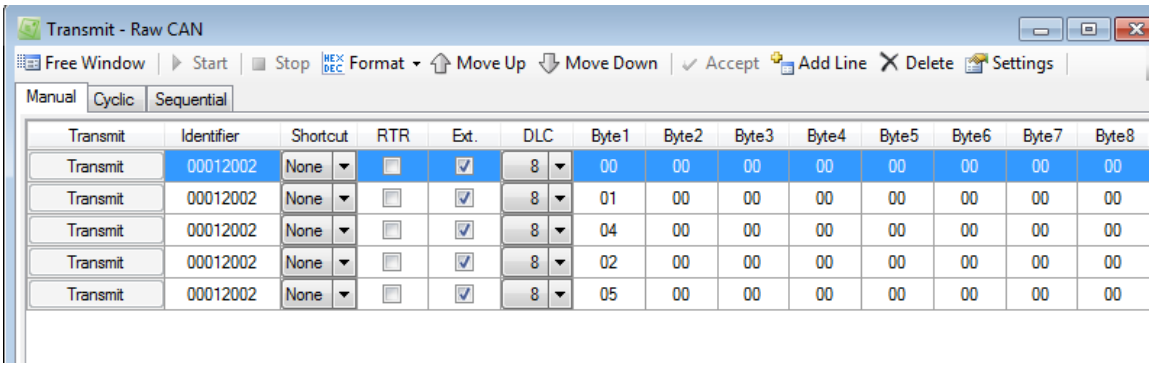
Open CAN Explorer in starter mode. The interface is set to CANFox and the baud rate, according to the keyboard. In the CAN Explorer, these settings must be set for receive and transmit direction.

When the settings are completed, the Play button starts the function. For the data packages to be displayed, the trace view must be opened.

The key 2, which has been parameterized as a pushbutton or switch, can now be pressed on the keyboard. The Can ID of the key and the corresponding bit position are displayed in the CAN Explorer.

To change the status display, a CAN message must now be sent to the keyboard. In PCAN Explorer, a new message is inserted in the "Send" area. The CAN ID is stored accordingly, for the state control, as set in the toolchain.

The graphic shows how the RGB LED can be changed using CAN messages.



Transmit	Identifier	Shortcut	RTR	Ext.	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
Transmit	00012002	None	☐	☑	8	00	00	00	00	00	00	00	00
Transmit	00012002	None	☐	☑	8	01	00	00	00	00	00	00	00
Transmit	00012002	None	☐	☑	8	04	00	00	00	00	00	00	00
Transmit	00012002	None	☐	☑	8	02	00	00	00	00	00	00	00
Transmit	00012002	None	☐	☑	8	05	00	00	00	00	00	00	00

By selecting the Transmit button, the corresponding message is sent to the keyboard. The button then changes the color.

4 Example parameterization keyboard and CAN I/O module 1G

In the following section the parameterization and the interaction of two miunske CAN devices in a network will be clarified.

In this example, the miunske CAN keyboard 2G6 and the miunske CAN I/O module 1G4 are used. A new project is created in the existing project folder.

The creation of a new project takes place in the same way as described in the "Create a new project in the solution" section. The newly created project is the basis for the following description.

4.1 Selection of required components

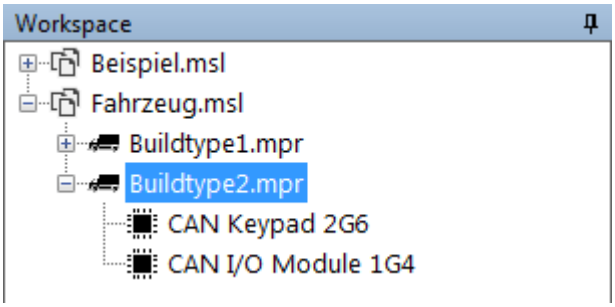
For this example the project is added a miunske CAN keyboard 2G6 and a miunske CAN I/O module 1G4.

For this example the project is added a miunske CAN keyboard 2G6 and a miunske CAN I/O module 1G4.

The newly created project is selected. Afterwards, the desired device can be selected and added to the project via the menu "FILE -> New -> Device".

In the example, the devices: "CAN keyboard 2G6" and the "CAN I/O module 1G4 / Nano" are added to the project.

(For a detailed explanation of the individual device parameters, please refer “HELP -> Instructions”).
Once the devices have been assigned to the project, they can be parameterized accordingly.



4.2 Parameterization

The device settings are displayed by selecting the corresponding device.

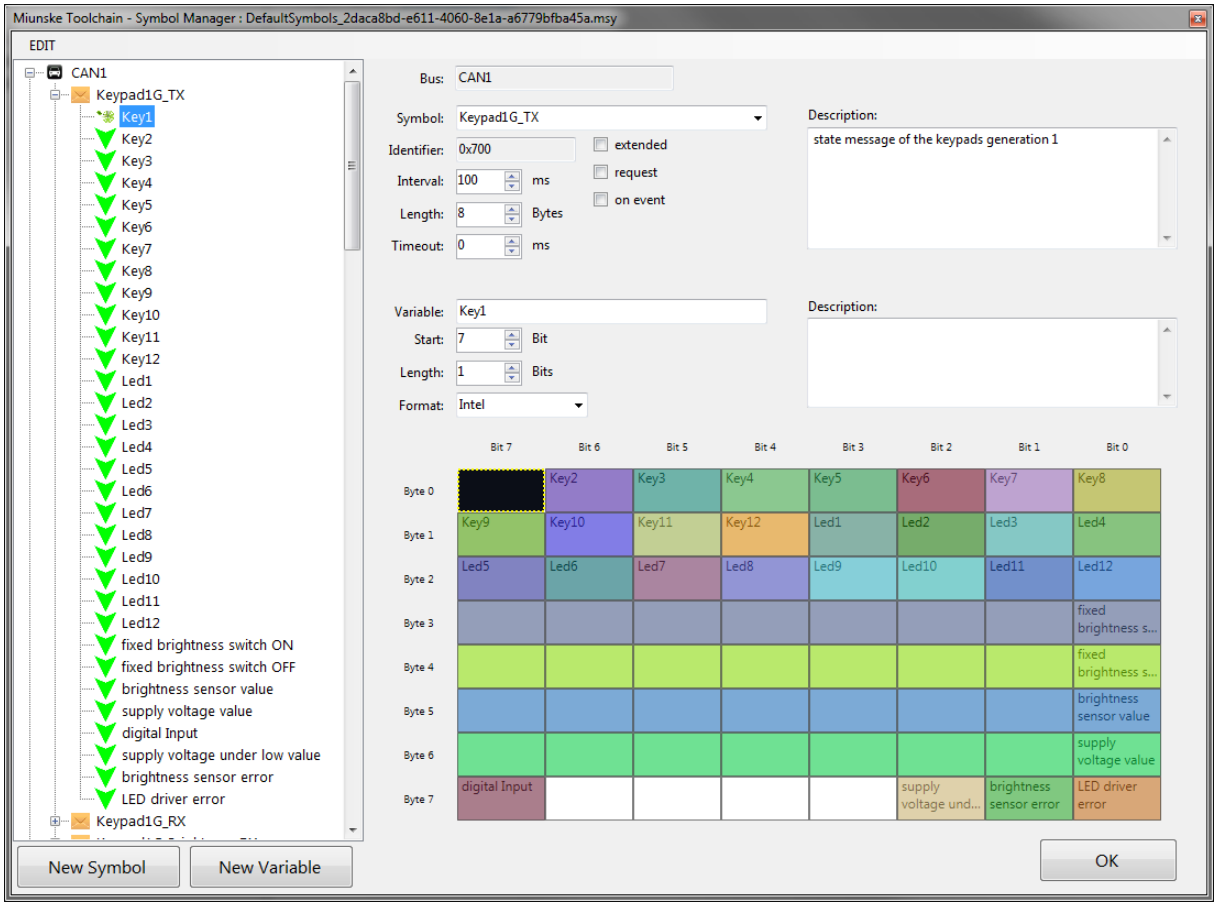
4.2.1 CAN keyboard 2G6

The settings for the individual switching or display elements are made in the "Fields" tab. By selecting the field to be parameterized on the keyboard layout, its settings can be changed.

Proceed:

Button 1 is selected in the keyboard layout. The "Switch - High active" function is selected in the "Keypad" area. The CAN message is then assigned to the field, with which the change of the actuation (ON or OFF) is sent in the CAN bus.

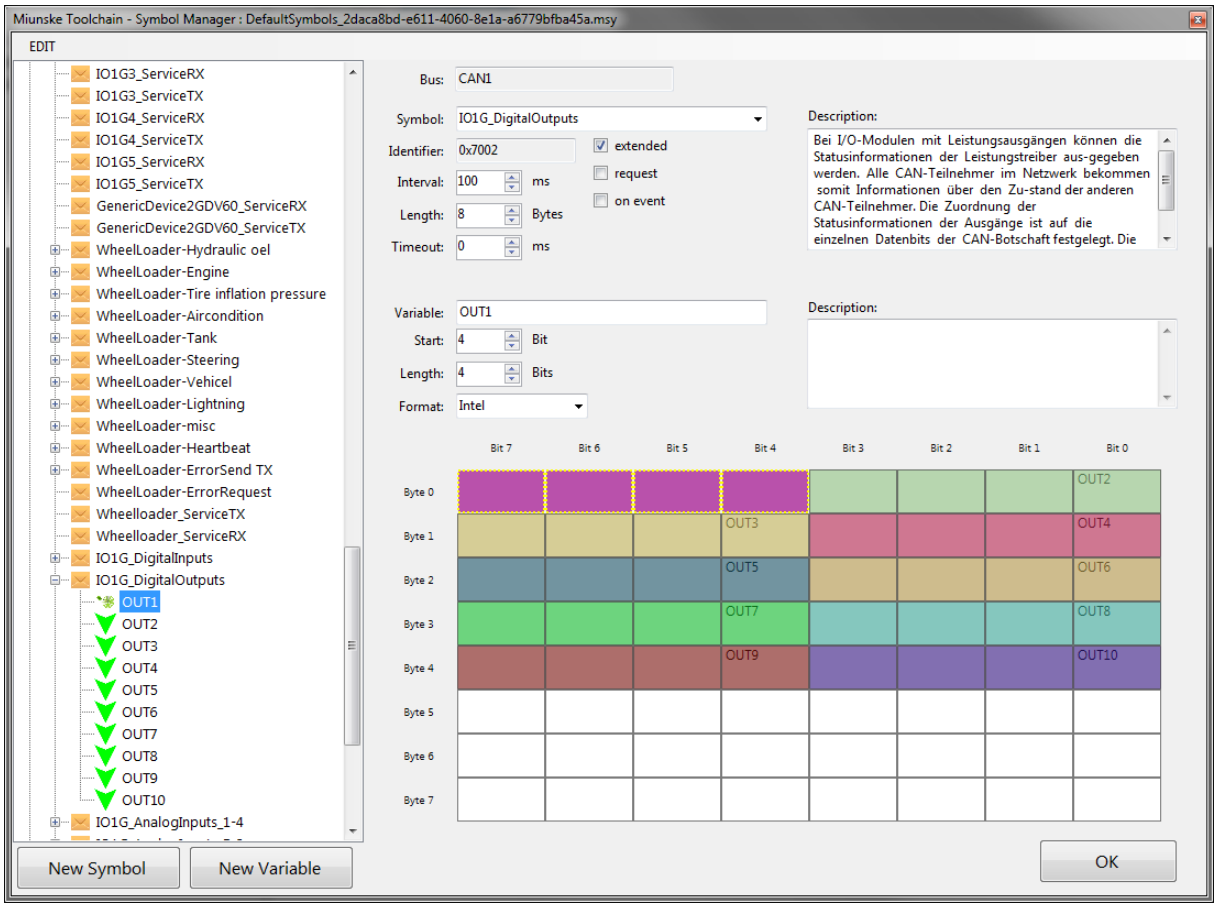
When selecting or creating the CAN message, make sure that the variable length is only **1 bit**. The desired CAN message for the function is assigned by selecting a symbol in the keyboard field. To do this, select "..." and the Symbol Manager is opened.



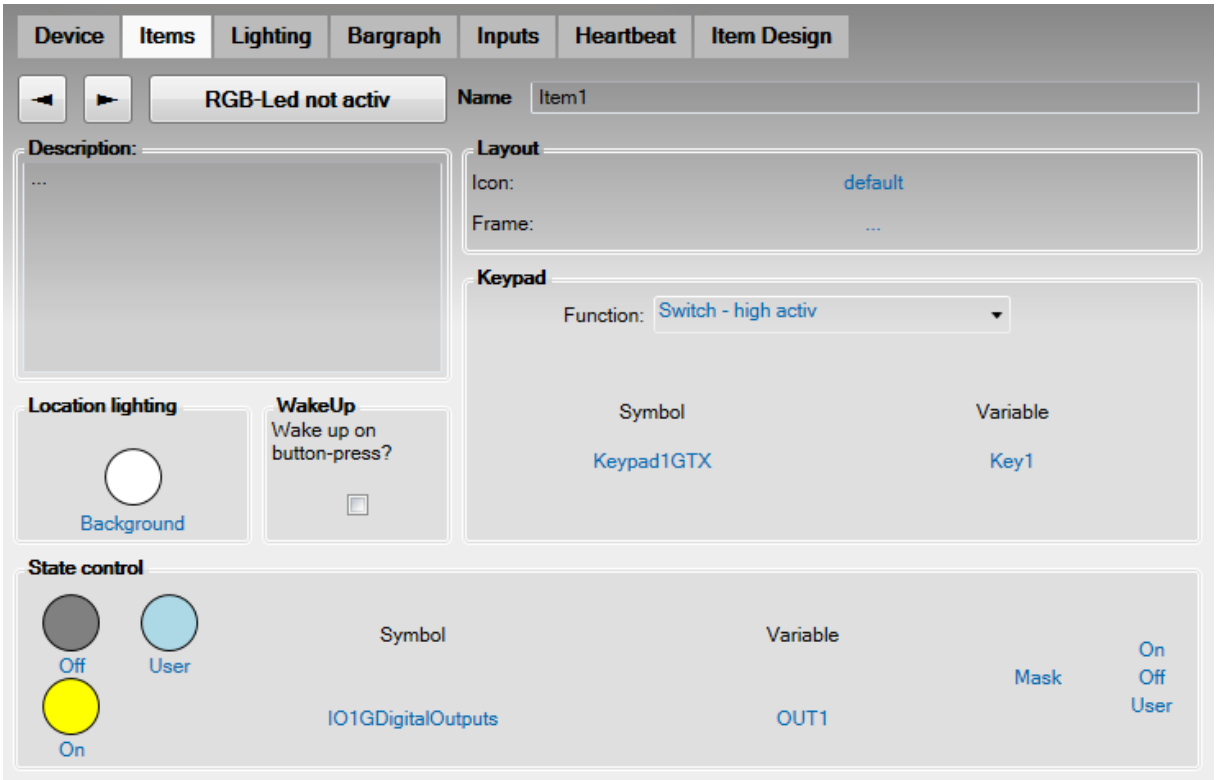
If the CAN message was accepted successfully, the symbol and the variable are displayed with corresponding names. In the example, the message with ID 0x700 and bit position 7 in byte 0 is selected.

The status change should be triggered by the CAN I/O module 1G4 and transmitted via the CAN message to the keyboard. In order for the LED to change and display the status, the LED is parameterized to the corresponding CAN message.

This is done analogously to the example "Application as pure display field" and is set via "State control".



In this example, the message ID 0x7002 and bit position 0 (0 to 3) are selected in byte 0 and confirmed with "OK". The symbol and variable name are displayed in the overview.



Note:

If an output at I/O module switched, a message is generated and sent. In addition to the status, the status information of the output is also transmitted in this message. For this information, 4 bits are required.

The state of the output (ON or OFF) is shown in bit 0. Bit 1 and 2 provide information on the status of the output (e.g.: overtemperature or short circuit). Bit 3 serves as a spare bit.

After all the buttons and indicators have been set accordingly, the parameterization is transferred to the device. With the "Write Parameters" button in the Communication window, the configuration is transferred to the device.

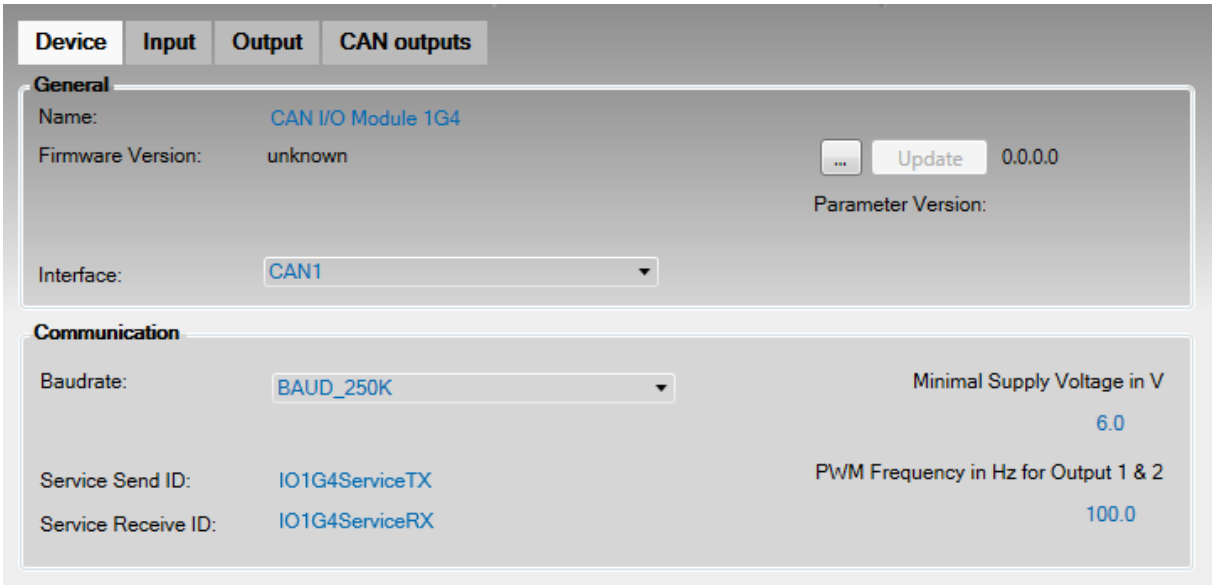
(Default baud rate of all miunske CAN keyboards is 250 kBit/s)

If the data has been successfully transferred to the keyboard, a corresponding message is displayed. This is also signaled if the transmission is faulty or aborted. The progress of a transfer can be tracked at the bottom left of the toolchain window. In addition, information about the current activities of the toolchain and its communication is printed in text form in the Output window.

4.2.2 CAN I/O 4 module

All miunske CAN I/O modules 1G have configured a build rate of 500 kBit/s at the factory. For communication, with the configured keyboard, the baud rate of the CAN I/O module must be adapted accordingly. (Default baud rate of the miunske CAN keyboard is 250 kBit/s)

The setting is made in the "Device" tab in the Communication field. The baud rate is set to the value "BAUD_250K" in the "Baud rate" selection list.



The inputs of the module are configured in the "Physical inputs" tab. A distinction is made between digital and analog inputs.

The outputs of the module are configured in the "Physical outputs" tab. In general, an output can always be controlled via CAN message or via physical input. Simultaneous activation is not provided in the standard firmware.

In order for an output of the CAN I/O module 1G4 to react to the keyboard, the corresponding CAN messages must be assigned to an output.

The assignment is made in the tab "Physical outputs". All outputs of the CAN I/O module 1G4 can be configured differently.

The following functions are available:

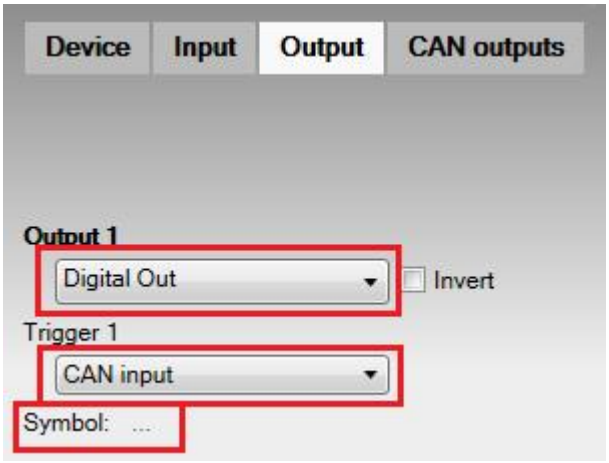
- Off
- Digital Out
- Impuls Delay
- Switch On Delay
- Switch Off Delay
- Blinking
- Threshold
- PWM

Each function includes parameters that define the behavior within it. A detailed overview can be found in the help.

In the example, an output is used as a digital output. This should respond to a CAN message on the keyboard.

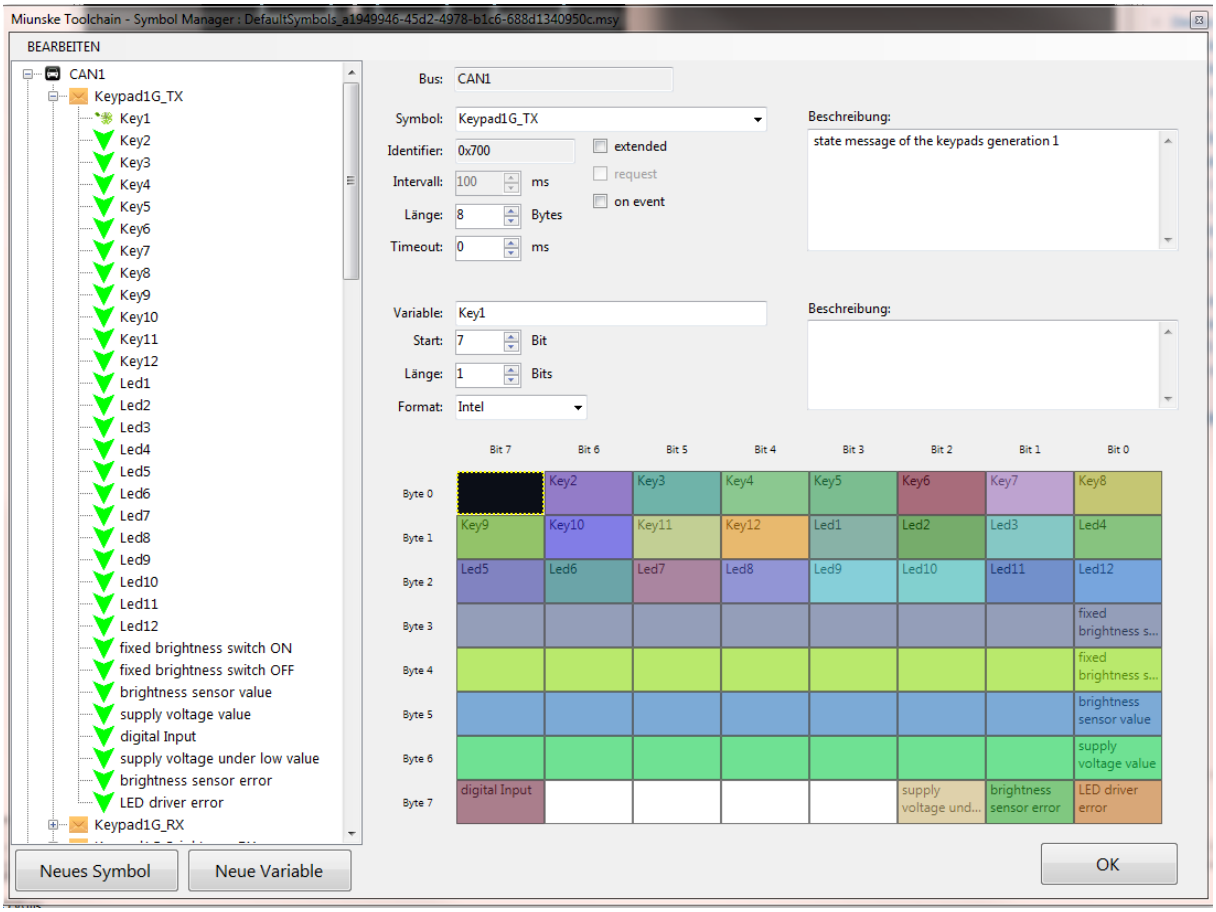
For output 1, select "Digital output" in the selection list. The "Trigger" selection list determines which signal (physical inputs on the CAN I/O module 1G4 or CAN input) responds to.

If "CAN input" is selected in the selection list, the "Symbol" field is displayed.

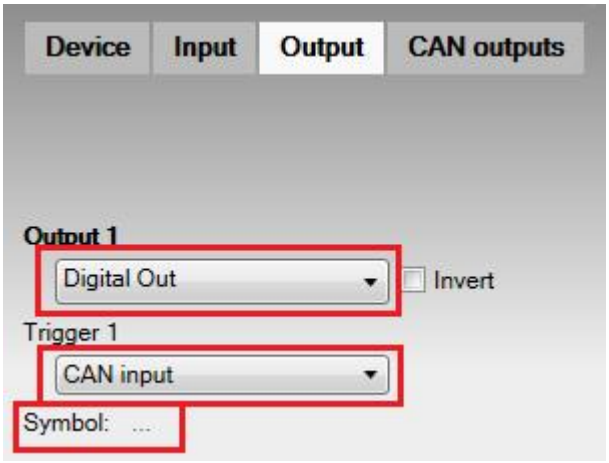


The assignment of the desired CAN message for the output is made by selecting a symbol. To do this, select "..." and the Symbol Manager is opened.

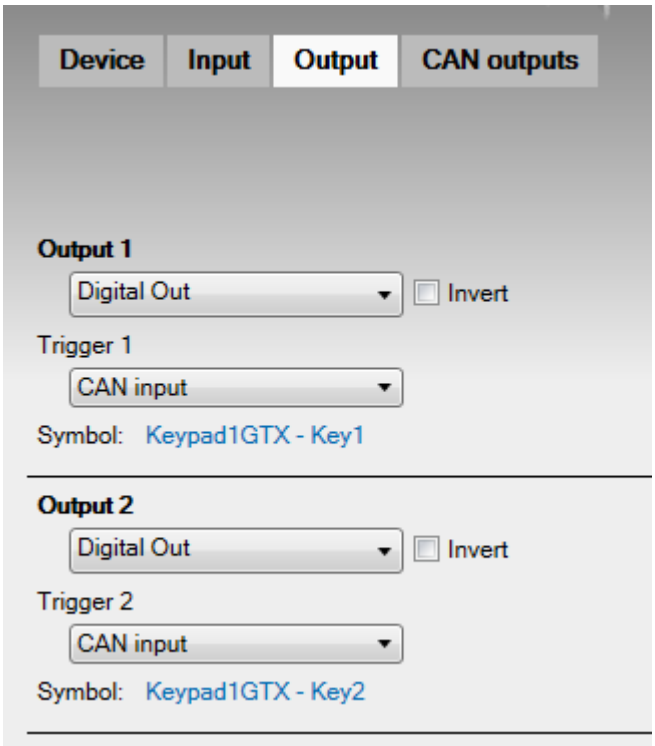
Key 1 of the keyboard has been parameterized as "Switch - High active". The assignment of a CAN message (ID 0x700, Byte 0, Bit 7), which represents the status change (ON or OFF) on the CAN bus, was also carried out. This message must be assigned to the output of the I/O module. In the Symbol Manager, select the corresponding symbol and the appropriate variable and confirm with OK.
It is also possible to have several outputs respond to a CAN message.


Miunske Toolchain - Symbol Manager : DefaultSymbols_a1949946-45d2-4978-b1c6-688d1340950c.msy
BEARBEITEN
CAN1
Keypad1G_TX
Key1
Key2
Key3
Key4
Key5
Key6
Key7
Key8
Key9
Key10
Key11
Key12
Led1
Led2
Led3
Led4
Led5
Led6
Led7
Led8
Led9
Led10
Led11
Led12
fixed brightness switch ON
fixed brightness switch OFF
brightness sensor value
supply voltage value
digital Input
supply voltage under low value
brightness sensor error
LED driver error
Keypad1G_RX
Neues Symbol
Neue Variable
Bus: CAN1
Symbol: Keypad1G_TX
Identifier: 0x700
Intervall: 100 ms
Länge: 8 Bytes
Timeout: 0 ms
extended
request
on event
Beschreibung:
state message of the keypads generation 1
Variable: Key1
Start: 7 Bit
Länge: 1 Bits
Format: Intel
Beschreibung:
Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0
Byte 0 Key2 Key3 Key4 Key5 Key6 Key7 Key8
Byte 1 Key9 Key10 Key11 Key12 Led1 Led2 Led3 Led4
Byte 2 Led5 Led6 Led7 Led8 Led9 Led10 Led11 Led12
Byte 3 fixed brightness s...
Byte 4 fixed brightness s...
Byte 5 brightness sensor value
Byte 6 supply voltage value
Byte 7 digital Input supply voltage und... brightness sensor error LED driver error
OK

If the message has been accepted successfully, this message is displayed at the output.

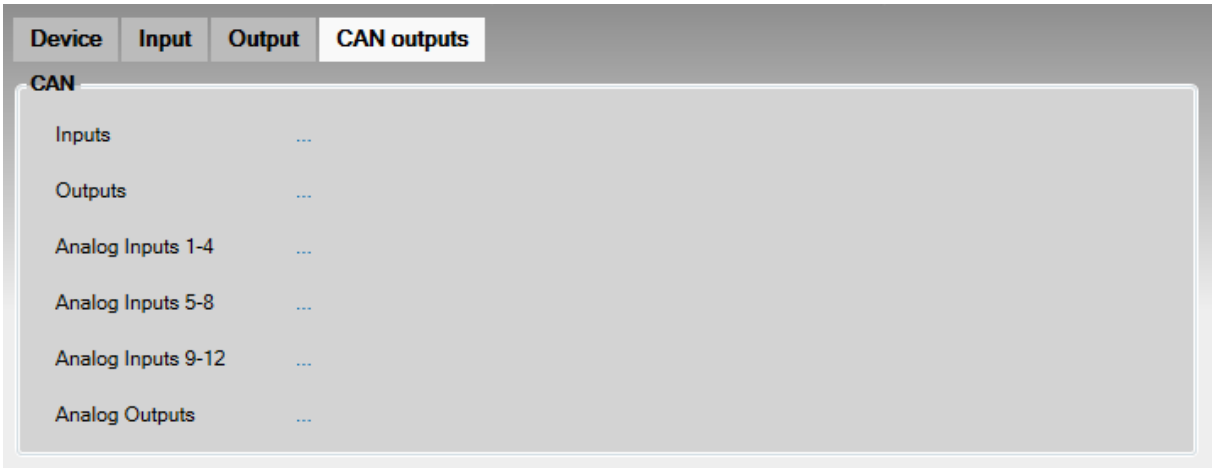


Output 2 is to respond to button 2 and is parameterized accordingly to "digital output" with the trigger CAN input. The assignment of the desired CAN message for the output is made by selecting a symbol. To do this, select "..." and the Symbol Manager is opened. The assignment of a CAN message (ID 0x700, byte 0, bit 6) with which the status change of key 2 is sent on the CAN bus has been assigned.



Analogously to this procedure, the remaining outputs are assigned to the remaining keys.

In order for the status display of the keyboard to change accordingly, the I/O module must announce the status of the individual outputs and inputs to the CAN network. This is done via the tab "CAN outputs".



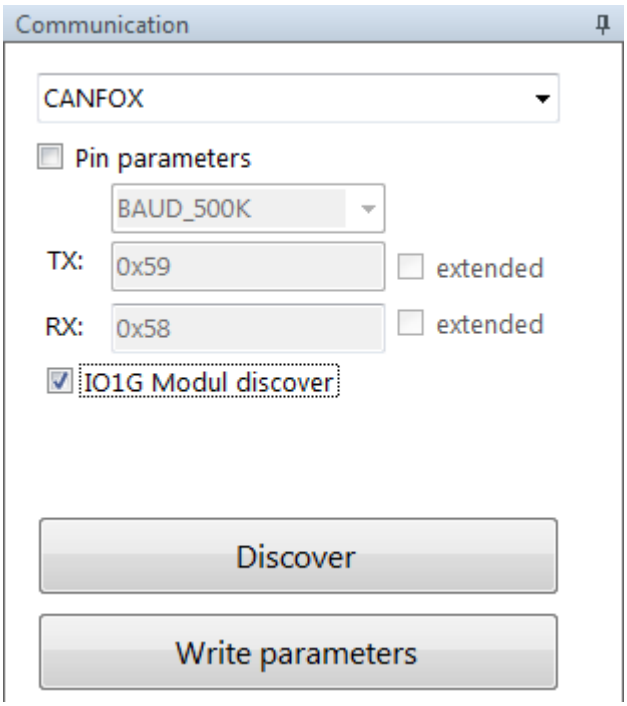
The CAN I/O module 1G4 can announce the status of its digital and analogue inputs and outputs as separate messages in the CAN network.

Possible messages are:

- Digital inputs (status of the physical inputs)
- Digital outputs (status of the physical outputs)
- Analog inputs (status and value of the analog inputs)
- Analog outputs (status and value of the analog outputs)

The desired CAN message is assigned by selecting a symbol. To do this, select "..." and the Symbol Manager is opened. However, the assignment of a CAN message was made without specifying a variable because the structure of the outgoing CAN messages of 1Gx modules is fixed. An adaptation is possible via the API interface via C code.

The message with the ID 0x7001 is selected for the digital inputs and accepted with OK. The message with the ID 0x7002 is selected for the digital outputs and accepted with OK. The remaining messages (analog inputs and outputs) are not required for this example, but can be assigned in the same way as this procedure.



After all inputs and outputs have been set appropriately, the parameterization is transferred to the device. With the "Write Parameters" button in the Communication window, the configuration is transferred to the device. Follow the corresponding instructions of the Toolchain. The transmission must take place individually, which means only the CAN I/O module 1G4 may be connected / active.
(Default baud rate of the miunske CAN I/O module 1G is 500 kBit/s.)

If the data has been successfully transmitted to the CAN I/O module, a corresponding message is displayed and the CAN I/O module 1G4 must be restarted. This is also signaled if the transmission is faulty or aborted. The progress of a transfer can be tracked at the bottom left of the toolchain window. In addition, information about the current activities of the toolchain and its communication is output in text form in the Output window.

4.3 Test of the parameterization

If the parameters have been successfully transferred to the CAN I/O module and the keyboard, a first test can be done. For this purpose, the keyboard and the I/O module are connected to one another using a CAN network.

Press the 1 key on the keyboard. A corresponding message is sent in the CAN network. The CAN I/O module 1G4 receives and processes the message accordingly. Output 1 is switched. This is announced by a message on the CAN bus. The keyboard processes the I/O Module 1G4 feedback. The LED of button 1 changes the state.

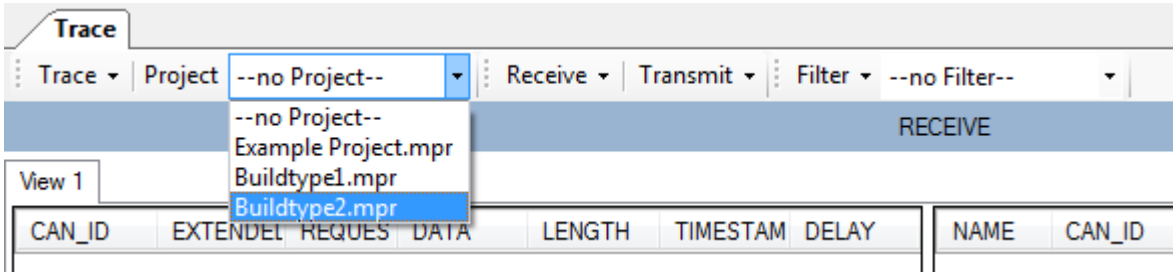
The communication between keyboard and CAN I/O module 1G4 can be monitored and tracked using the toolchain trace function. Alternatively, this can be done using tools (e.g.: PCAN View, CAN Explorer, etc.).

4.3.1 Presentation of the communication with the toolchain trace function

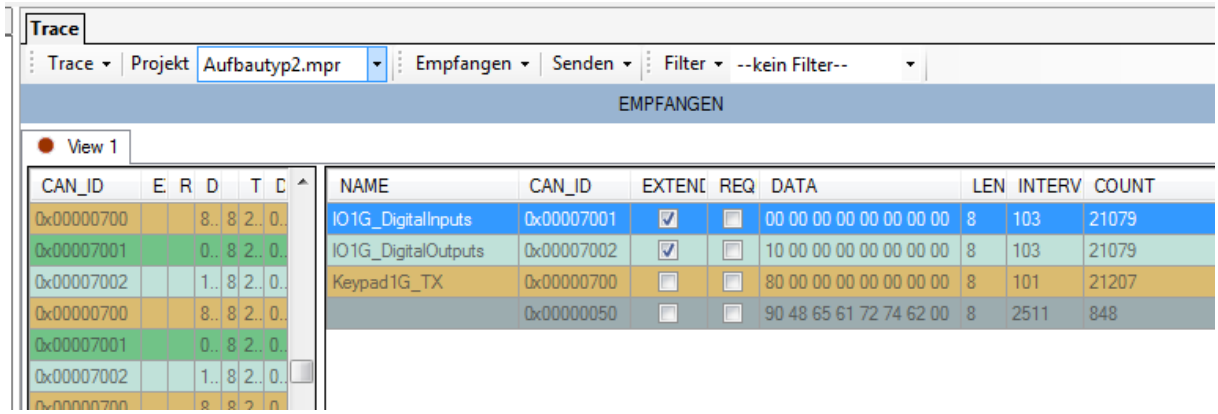
The following section shows the communication, between the CAN keyboard 2G6 and the CAN I/O module 1G4, using the toolchain trace function.

The trace function of the Toolchain makes it possible to send and receive messages in a CAN network without any further tools.
The trace function uses the data stored in the communication window for the interface and baud rate and is started via the "Connect Trace" button.

The sample project "Build Type2" is assigned to the trace. This assignment allows the CAN messages that are created or defined in the symbol manager to be used.



In order to receive and send messages, the trace must be started in the receive view by pressing the "Record" button. All messages of the CAN bus are displayed. Since the trace function has been linked to the example project, the corresponding symbol names are displayed in addition to the CAN ID's in the receive overview.



The key press of the key 1 on the miunske Keypad 2G6 is displayed as a message (Keypad1G_TX) in the receive view. The I/O module 1G4 switches the corresponding output and announces this again on the CAN bus as a CAN message (IO1G_DigitalOutputs).