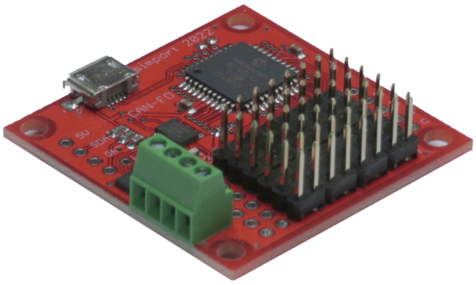
CAN in Simulation

Keyboard Module



Rev. 1.2

Keyboard Module

The Keyboard module is equipped with 24 digital inputs designed to accommodate various types of switches, particularly pushbuttons. When these switches are closed, they establish a connection to ground. Consequently, a ‚key down‘ message is generated. Conversely, when the switch subsequently opens, a ‚key up‘ message is generated, indicating the release of the key. While the ‚key down‘ message transmits a predefined keycode for every input along with a modifier byte, the ‚key up‘ message contains a keycode value of zero

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
720h	node	0Bh	item	num	mod	key	0	0

CAN Message

The item in the service code byte is a key ID that associates a keystroke with a specific key or button.

The keycodes employed in data byte 1 (*key*) are in accordance with the *HID Usage Table 0x07* designated for USB Keyboards and Keypads, as defined and published by the USB Implementer‘s Forum (Link). For a comprehensive keycode table, please refer to the appendix or download it from the following location (Link).

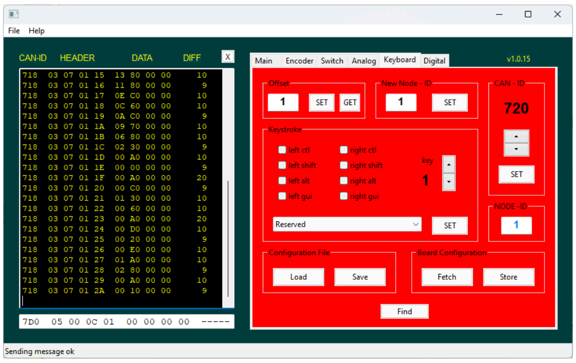
The modifier byte (*mod*) operates on a bitwise level and indicates which modifier key(s) (*Ctrl, Shift, Alt, Gui*) are being pressed simultaneously with the key.

Its structure is as follows:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
left Ctrl	left Shift	left Alt	left GUI	right Ctrl	right Shift	right Alt	right GUI

Modifier Byte

This is how the Configuration Tool views a Digital Module (more on page 7):



```
CAN-ID   HEADER        DATA          DIFF
718   03 07 01 15   13 80 00 00     10
718   03 07 01 16   11 80 00 00      9
718   03 07 01 17   0E C0 00 00     10
718   03 07 01 18   0C 60 00 00     10
718   03 07 01 19   0A C0 00 00      9
718   03 07 01 1A   09 70 00 00     10
718   03 07 01 1B   06 80 00 00     10
718   03 07 01 1C   02 30 00 00      9
718   03 07 01 1D   00 A0 00 00     10
718   03 07 01 1E   00 00 00 00      9
718   03 07 01 1F   00 A0 00 00     20
718   03 07 01 20   00 C0 00 00      9
718   03 07 01 21   01 30 00 00     10
718   03 07 01 22   00 60 00 00     10
718   03 07 01 23   00 A0 00 00     20
718   03 07 01 24   00 D0 00 00     10
718   03 07 01 25   00 20 00 00      9
718   03 07 01 26   00 E0 00 00     10
718   03 07 01 27   01 A0 00 00     10
718   03 07 01 28   02 80 00 00      9
718   03 07 01 29   00 A0 00 00     10
718   03 07 01 2A   00 10 00 00      9

7D0   05 00 0C 01   00 00 00 00   -----
```

Offset: 1

New Node - ID: 1

CAN - ID: 720

NODE -ID: 1

Keystroke: ☐ left ctl ☐ right ctl ☐ left shift ☐ right shift ☐ left alt ☐ right alt ☐ left gui ☐ right gui; key 1; Reserved

Sending message ok

The CAN-ID is displayed as 720 (hexadecimal), and the node ID is identified as 1.

You can change these values using the spin buttons for the CAN-ID or by editing the node ID. Clicking the SET button afterward will overwrite the corresponding value in the module.

The Find button searches for any attached digital modules, which is useful when the modules are changed.

There are 1 + 24 parameters available that can be altered upon request, which are described in the next paragraph.

Available Parameters

Offset

The Digital module is capable of handling up to 24 key or buttons, each of which is assigned a unique ID.

Starting with the *offset* value, the 24 keys are given consecutive ID values, which will be included in the CAN message sent by the board.

Since the ID values are 1 byte wide, up to 256 different output lines can be distinguished under a given Node-ID.

Keystroke

The keyboard module transmits *keystroke* values along with a keycode (representing the specific key pressed) and a modifier that signifies a combination involving the *Shift*, *Ctrl*, *Alt*, and *GUI* key(s). Each input has its own associated keystroke, which can be configured individually.

Parameter Setting

To modify the parameters of a module, the Module Configuration Service (MCS) is utilized. The MCS is assigned a unique CAN-ID of `7D0h` (equivalent to decimal value `2000`):

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
`7D0h`	`node`	`type`	`0Dh`	`pid`	`par1`	`par2`	`par3`	`0`

node ID: CAN node ID (*`node`*)

data type: UCHAR (*`0Ah, 10d`*) for offset parameter
UCHAR3 (*`1Bh, 27d`*) for keystroke parameter

service code: MCS (`0D`)

message code: *`Pid=1`* for offset parameter
`Pid=5` for keystroke parameter

message data: Offset or Modifier *(`par1`)*
keycode *(`par2`)*
Key ID *(par3)*

The parameter ID (pid) is used to identify which specific parameter needs to be modified. Data byte 0 contains the value of the parameter and data byte 1 specifies the affected output port.

Parameters IDs

index	parameter	value(s)
1	offset	1 … 255
5	keystroke	Modifier / keycode pair

Upon completion of the parameter modification request, the response message will have a message code of **`0`** (zero) if the operation was successful. However, if the requested parameter is out of the valid range or the parameter ID is invalid, the response message will contain a message code of **`-6`**.

CAN-ID Setting

The CAN-ID range for Keyboard Module messages is **720h..72Fh** (decimal **1832..1839**).

To change the CAN-ID of the Keyboard Module, the CAN Identifier Setting Service (CSS) can be used. The message code (parameter ID) should be set to 0.

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
7D0h	node	0Ch	0Eh	0	0	0	xh	xl

node ID:	CAN node ID (*node*)
data type:	SHORT (*0Ch, 12d*)
service code:	CSS (*0Eh, 14d*)
message code:	0
message data:	New CAN ID high byte (xh, data byte 2) New CAN ID low byte (xl, data byte 3)

Upon completion of the CAN Identifier Setting request, the response message will have a message code of **0** (zero) if the operation was successful, or **-6** if the ID is out of the valid range.

Node-ID Setting

To change the Node-ID of the Keyboard Module, the Node ID Setting Service (NIS) can be used. Node-ID values are in the range of 1 to 255.

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
7D0h	node	0	0Bh	x	0	0	0	0

node ID:	CAN node ID (*node*)
data type:	NODATA (*00h, 0d*)
service code:	NIS (*0Bh, 11d*)
message code:	New node ID (*1 ≤ X ≤ 255*)
message data:	0

Upon completion of the Node Identifier Setting request, the response message will have a message code of **0** (zero) if the operation was successful.

Module Information

The CAN bus bus messages are designed to be triggered only when there is a change in the state of any key pressed or released. The Module Information Service (MIS) enables the interrogation of a key's modifier and code. A total of 24 messages must be generated to provide an up-to-date reflection of the current presets for all key inputs on the board.

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
7D0h	node	0	0Ch	05h	0	0	key	0

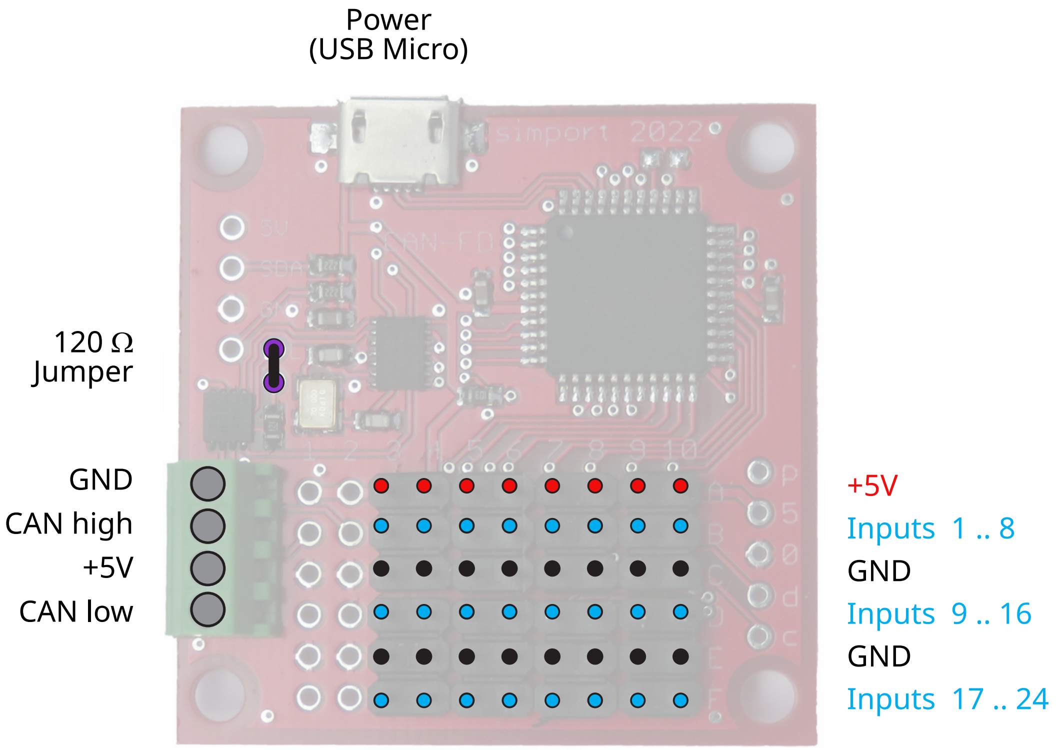
node ID:	CAN node ID (*node*)
data type:	NODATA (*00h, 0d*)
service code:	MIS (*0Ch*, 12*d*)
message code:	05h (keystroke parameter)
message data:	Key ID (data byte 2)

The response message appears as follows:

CAN ID	node ID	data type	service code	message code	data byte 0	data byte 1	data byte 2	data byte 3
7D1h	node	1Bh	0	0	par1	par2	par3	0

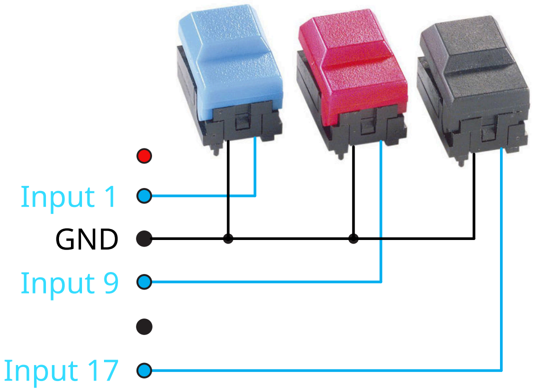
node ID:	CAN node ID (*node*)
data type:	NODATA (*1Bh, 27d*)
service code:	0
message code:	0
message data:	modifier (par1) Key code (par2) Key ID (par3)

Board Layout



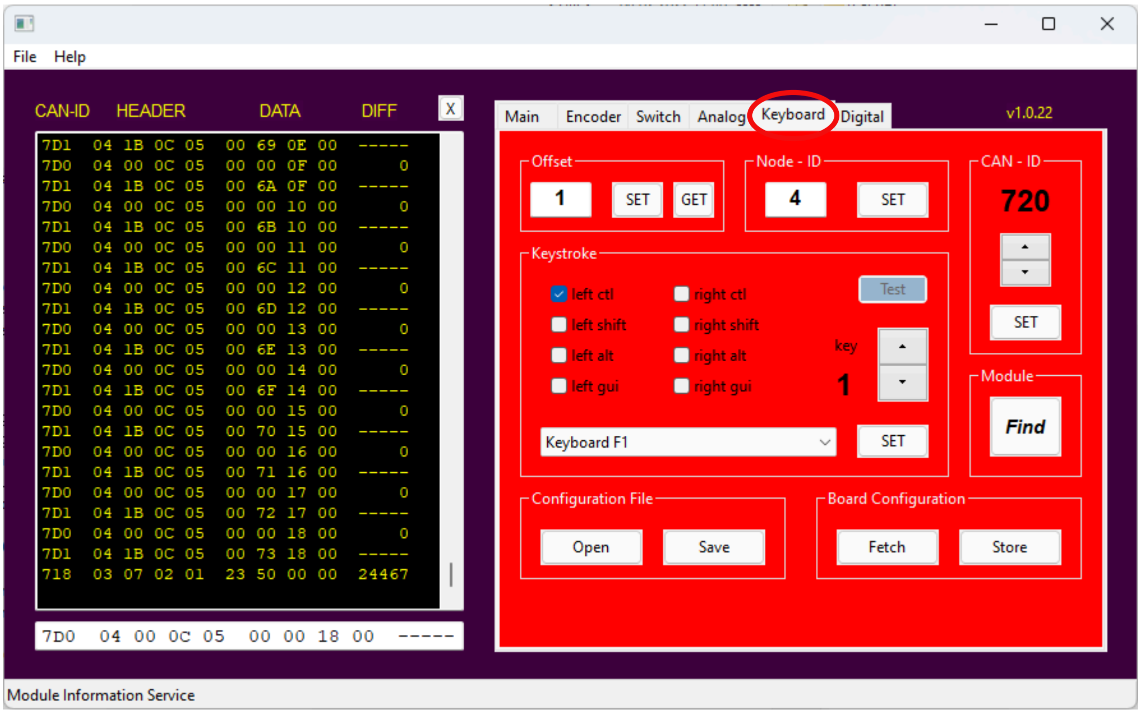
The 120 Ω jumper places a termination resistor between the CAN high and CAN low line.

Wiring Example



The Configuration Tool

The configuration tool features a "Keyboard" panel on the right side, which appears as follows:



```
CAN-ID  HEADER       DATA          DIFF
7D1   04 1B 0C 05   00 69 0E 00   -----
7D0   04 00 0C 05   00 00 0F 00       0
7D1   04 1B 0C 05   00 6A 0F 00   -----
7D0   04 00 0C 05   00 00 10 00       0
7D1   04 1B 0C 05   00 6B 10 00   -----
7D0   04 00 0C 05   00 00 11 00       0
7D1   04 1B 0C 05   00 6C 11 00   -----
7D0   04 00 0C 05   00 00 12 00       0
7D1   04 1B 0C 05   00 6D 12 00   -----
7D0   04 00 0C 05   00 00 13 00       0
7D1   04 1B 0C 05   00 6E 13 00   -----
7D0   04 00 0C 05   00 00 14 00       0
7D1   04 1B 0C 05   00 6F 14 00   -----
7D0   04 00 0C 05   00 00 15 00       0
7D1   04 1B 0C 05   00 70 15 00   -----
7D0   04 00 0C 05   00 00 16 00       0
7D1   04 1B 0C 05   00 71 16 00   -----
7D0   04 00 0C 05   00 00 17 00       0
7D1   04 1B 0C 05   00 72 17 00   -----
7D0   04 00 0C 05   00 00 18 00       0
7D1   04 1B 0C 05   00 73 18 00   -----
718   03 07 02 01   23 50 00 00   24467

7D0   04 00 0C 05   00 00 18 00   -----
```

Offset: 1

Node - ID: 4

CAN - ID: 720

Keystroke: ☑ left ctl ☐ right ctl ☐ left shift ☐ right shift ☐ left alt ☐ right alt ☐ left gui ☐ right gui

key 1

Keyboard F1

v1.0.22

Module Information Service

When you open this panel for the first time, it automatically searches for a Keyboard module on the CAN bus. If a board is found, its *Node-ID*, *CAN-ID*, and *Offset* parameter are displayed. Subsequently, clicking the **Find** button initiates a new search.

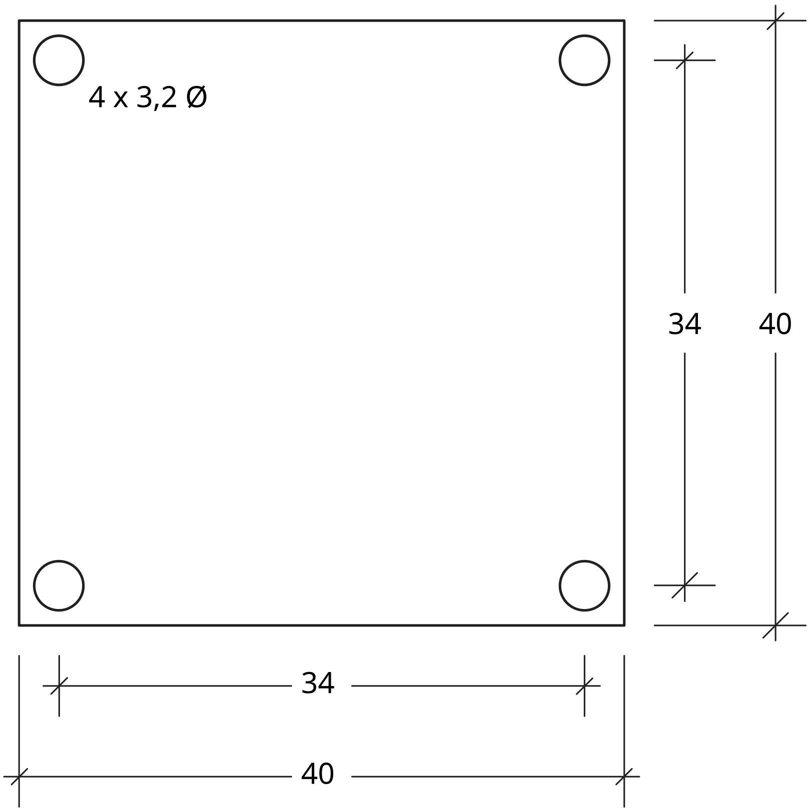
The *Node-ID* and *Offset* parameters can be adjusted by editing the numbers in their respective fields. Pressing the **SET** button will immediately update these parameters in the module. The *CAN-ID* can be adjusted using the spin buttons.

In the 'Keystroke' field, you can select one of the 24 keys using the spin buttons, with the key number displayed to the left. The dropdown menu of the combo box lets you choose a specific key code, such as 'F1' in the example. The checkboxes above allow you to add modifier bits for the selected key code. In the example, the 'left ctl' modifier is checked. Finally, the **SET** button updates this configuration in an internal list. Once all adjustments are made, the list can be saved to a text file (in JSON format) using the **Save** button, or it can be stored in the board's internal memory with the **Store** button.

The **Fetch** button retrieves the values from the board's memory and updates the internal list accordingly. Alternatively, you can **Open** a previously saved configuration file from disk to update the internal list.

You can monitor all CAN bus activity in the left window.

Board Dimensions [mm]


4 x 3,2 Ø
34
40
34
40