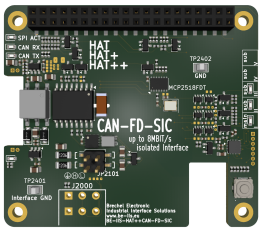
Isolated CAN FD (SIC) Industrial HAT++ for Raspberry Pi

Brechel Electronic
Industrial Interface Solutions
Designed by Philipp Brechel (Germany)
www.be-iis.eu | www.github.com/be-iis



Document ID: BE-IIS-HPP-CAN_RevB_Datasheet
Revision: Rev. B
Date: June 8, 2026
Document Status: Released
Product Status: In Production

CAN-FD with SIC supports data rates up to 8 Mbit/s over a differential two-wire bus, allowing reliable communication between multiple nodes within typical automotive and industrial network topologies.

This HAT enables CAN-FD multidrop communication with Signal Improvement Capability (SIC) and, as part of the BE-IIS HAT++ ecosystem, is stackable with other HATs from the portfolio for seamless system expansion and simplified system integration.

Key Features

- CAN-FD with SIC, up to 8 Mbit/s
- MCP2518FD CAN-FD controller
- TCAN1472 CAN-FD SIC transceiver
- Galvanically isolated transceiver
- Configurable 120 Ω termination
- RSP HAT+ compliant (2024)
- Stackable HAT (BE-IIS-HAT++)
- Configurable CS and IRQ routing
- Two connector options
- RoHS compliant
- Quality component suppliers

Product Description

The CAN-FD Industrial HAT is a Raspberry Pi HAT+ compliant interface board supporting CAN-FD with SIC according to ISO 11898.
It integrates an MCP2518FD controller and a TCAN1472 SIC transceiver with galvanic isolation between logic and bus interface.

Applications

- CAN and CAN-FD network evaluation
- Technology evaluation
- Prototyping
- Education and laboratory use

Contents

1 Introduction

1.1 General

The BE-IIS HAT++ CAN-FD-SIC Industrial HAT is a Raspberry Pi HAT+ compliant interface board for CAN and CAN-FD communication in industrial and laboratory environments.

The board integrates an MCP2518FD [1] CAN-FD controller and a TCAN1472 [3] SIC transceiver. Galvanic isolation separates the logic domain from the CAN bus domain.

CAN and CAN FD enable robust, deterministic multi-node communication and are widely used as proven standards in industrial and automotive applications. The arbitration-based bus access ensures reliable and predictable behavior even in systems with multiple active nodes and in electrically noisy environments.

The design is fully aligned with the **HAT++ ecosystem**, enabling **stackable operation** of multiple interface boards on a single Raspberry Pi. The HAT++ concept ensures **conflict-free resource allocation** and allows flexible combinations of different communication interfaces within one system.

The HAT++ ecosystem includes a growing portfolio of industrial interface modules such as **10BASE-T1S**, **10BASE-T1L**, **Ethernet (SPI-based)**, **Modbus/RS-485**, and additional communication and measurement interfaces.

The HAT supports multi-node CAN bus operation and can be used for evaluation, prototyping, test setups, and educational purposes.

If you intend to use the HAT in a commercial product, please contact Brechel Electronic to adapt and optimize the design according to your specific requirements. “‘tex

1.2 Raspberry Pi Compatibility

The board follows the Raspberry Pi HAT+ mechanical and electrical interface concept. Compatibility depends on the selected Raspberry Pi model, operating system architecture, enabled interfaces, and the installed BE-IIS software configuration.

Model	32-bit	64-bit
Raspberry Pi 2	✓	
Raspberry Pi 3	✓	✓
Raspberry Pi 4	✓	✓
Raspberry Pi 5	✓	✓
Raspberry Pi Zero	✓	
Raspberry Pi Zero 2 W	✓	✓

Table 1: Raspberry Pi compatibility

Note: Compatibility depends on the installed operating system, kernel version and enabled interfaces.“

“‘tex

1.3 HAT Compatibility Level

The board supports the Raspberry Pi HAT concept and extends it with the BE-IIS HAT++ stackability concept.

Level	Description	Support	Configuration
HAT	The board acts like a standard Raspberry Pi HAT.	✓	3 configurable hardware configurations
HAT+	The board provides the Raspberry Pi HAT+ auto-detection mechanism.	✓	Instance ID 1 configurable
HAT++	The board offers full detection and stackability according to the BE-IIS HAT++ specification.	✓	3 configurable hardware configurations

Table 2: HAT compatibility level

“‘

1.4 Introduction: Optional HAT++ Stacking

HAT++ is an extended function of the Raspberry Pi HAT+ concept. It enables stacking of up to five different HATs in one system. Depending on the supported instance modes, the same HAT type can be used up to three times in one stack.

Figure 1 gives a short overview of the required hardware configuration and the supported instance modes.

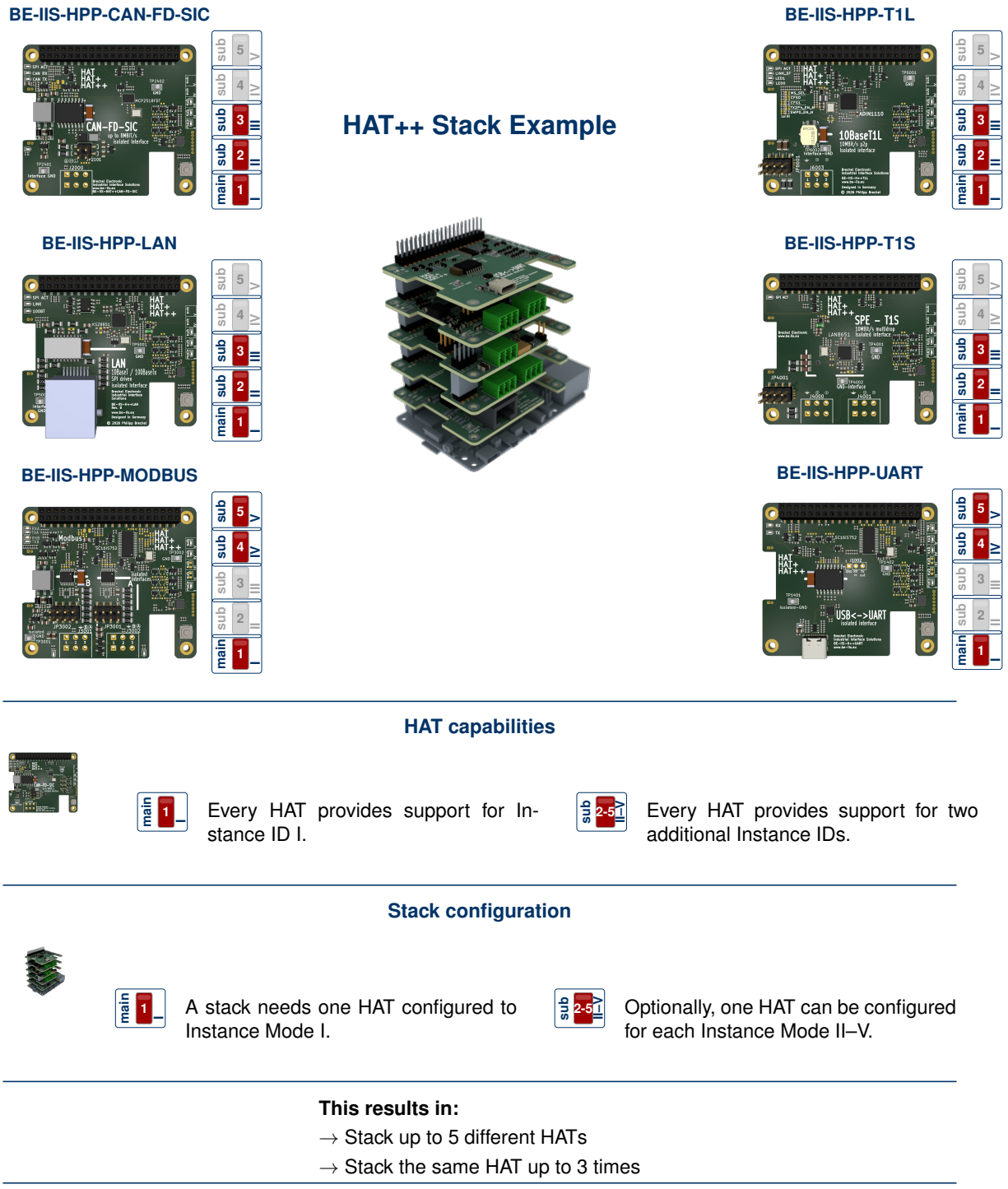


Figure 1: HAT++ stacking overview and instance mode configuration.

1.5 Design Resources

All design files and software resources are publicly available.

HTML **Product Page**
https://www.be-iis.eu/products/BE-IIS-HPP-CAN_B

PDF **Datasheet (PDF)**
https://www.be-iis.eu/products/BE-IIS-HPP-CAN_B/datasheet.pdf

PDF **Schematic (PDF)**
https://www.be-iis.eu/products/BE-IIS-HPP-CAN_B/schematic.pdf

HTML **Layout & BOM (Interactive)**
https://www.be-iis.eu/products/BE-IIS-HPP-CAN_B/ibom.html

STL **3D Model (STEP/STL)**
https://www.be-iis.eu/products/BE-IIS-HPP-CAN_B/model.zip

GIT **GitHub Repository**
https://github.com/be-iis/be-iis-installer/tree/main/products/BE-IIS-HPP-CAN-FD-SIC-REVB

GIT **Installer Script**
https://github.com/be-iis/be-iis-installer/tree/main/scripts/install/install-all.sh

DigiKey

Available from DigiKey
DigiKey product page

2 Hardware Configuration

2.1 Main Features

The BE-IIS-CAN-FD-SIC HAT enables CAN-FD communication with Signal Improvement Capability (SIC) on Raspberry Pi platforms. It allows any standard Raspberry Pi (e.g. Raspberry Pi Zero or Raspberry Pi 3/4/5, excluding Compute Module variants) to act as a CAN-FD-SIC node. Communication between the Raspberry Pi and the CAN controller is implemented via the SPI interface with an interrupt line for efficient event handling.

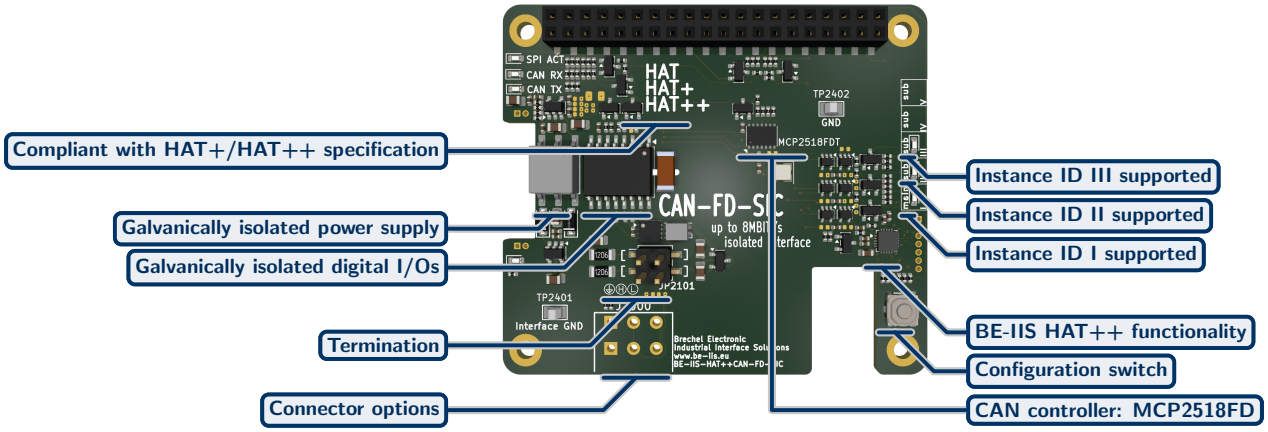


Figure 2: BE-IIS CAN-FD-SIC top view with annotations

CAN Controller

- Supplier: Microchip
- MCP2518FD CAN-FD controller
- MCP2518FD Crystal-Configuration: 40MHz

CAN Tranceiver

- Supplier: Texas Instruments
- TCAN1472 CAN-FD SIC transceiver

SPI Interface

- SPI0.0, SPI0.1, and SPI0.2 supported
- Persistent selection via push button

Isolation

- Galvanically separated interface
- Isolator rated up to 3 kV

Protocol Support

- Compatible with Classical CAN
- Compatible with CAN-FD
- CAN-FD with SIC support

Bus Topology

- Multi-node CAN bus architecture
- Node count depending on system design

Node Configuration

- Configuration via `ip` and SocketCAN
- Standard CAN arbitration supported

2.2 Block Diagram

The block diagram shown in Figure 3 is simplified. It illustrates the main functional blocks, power domains, isolation barriers, and the principal data paths of the system.

The interrupt signal routing is not shown. It is configured using the same scheme as the chip-select (CS) routing. Reset, and pull-up control signals is omitted for clarity as well.

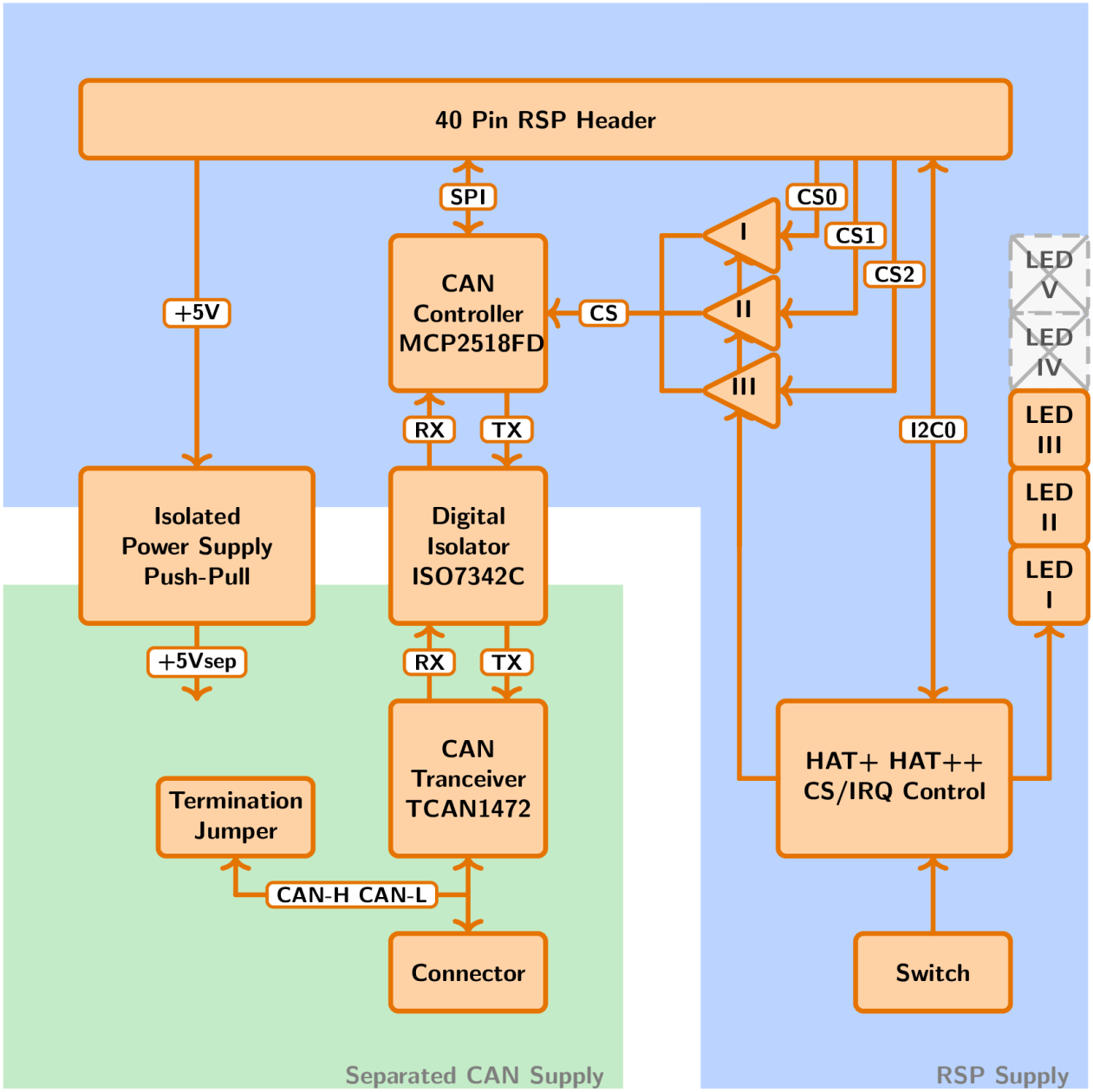


Figure 3: Simplified block diagram

2.3 Hardware Components

The selected hardware components and their interconnection are optimized to support **high-speed CAN FD communication** up to **8 Mbit/s**, including **galvanic isolation** of the bus interface. The design is fully aligned with the **HAT++ ecosystem**, enabling modular integration and conflict-free operation in stacked configurations.

2.3.1 MCP2518FD CAN FD Controller

The **MCP2518FD** by *Microchip Technology* is a stand-alone **CAN FD controller** with an SPI interface, designed to add CAN and CAN FD connectivity to systems without native CAN support.

It supports both **Classical CAN (CAN 2.0B)** and **CAN FD** according to ISO 11898-1, with data rates of up to **8 Mbit/s** in the data phase. Communication with the host is performed via a high-speed **SPI interface**, while an external CAN transceiver is required to interface with the physical bus.

The device integrates a flexible message handling system with multiple **FIFOs**, **filters**, and **timestamping** capabilities. This allows efficient handling of CAN traffic while offloading real-time constraints from the host CPU.

In Linux-based systems, the MCP2518FD is supported by the `mcp251xfd` driver and integrates seamlessly into the **SocketCAN** framework, appearing as a standard CAN network interface (e.g. `can0`). For further details, refer to the supplier's product documentation (see 1).

2.3.2 ISO7342C Digital Isolator

The **ISO7342C** by *Texas Instruments* is a quad-channel **digital isolator** designed to provide galvanic isolation for high-speed digital signals in industrial and embedded systems.

It features **capacitive isolation technology**, enabling robust data transmission across isolation barriers while maintaining high noise immunity. The device supports data rates up to **25 Mbps** and provides a typical isolation rating of up to **3 kVrms**, depending on the package variant.

The ISO7342C integrates four unidirectional channels with a defined channel direction configuration, making it suitable for isolating interfaces such as SPI, UART, or control signals. Its low propagation delay and tight channel-to-channel matching ensure reliable communication in time-critical applications.

Typical use cases include **industrial communication interfaces**, **isolated fieldbus systems**, and **mixed-voltage designs**, where protection against ground loops, voltage transients, and electromagnetic interference is required. For details on system-level isolation, see section 2.4. For further details, refer to the supplier's product documentation (see 2).

2.3.3 TCAN1472 CAN FD Transceiver

The **TCAN1472** by *Texas Instruments* is a robust **CAN FD transceiver** designed for high-speed and industrial-grade communication systems. It provides the physical layer interface between a CAN controller (e.g. MCP2518FD) and the CAN bus.

The device supports both **Classical CAN** and **CAN FD** with data rates up to **8 Mbit/s**. It features excellent **electromagnetic compatibility (EMC)** and high **electrostatic discharge (ESD)** robustness, making it suitable for harsh environments.

The TCAN1472 includes advanced protection features such as **bus fault protection**, **dominant timeout**, and **thermal shutdown**. It ensures reliable operation even under fault conditions like short circuits or voltage transients on the bus lines.

Designed for seamless integration, the transceiver operates with a wide supply voltage range and provides direct interfacing to modern CAN FD controllers. Typical applications include **industrial automation**, **automotive systems**, and **robust embedded communication interfaces**. For details on system-level isolation, see section 2.4. For further details, refer to the supplier's product documentation (see 3).

2.3.4 Isolated Power Supply

The board integrates an isolated power supply based on a push-pull converter topology.

Characteristics:

- Input: 5 V (Raspberry Pi side)
- Output: 5 V (isolated field side)
- Galvanic isolation between logic and field domain

For details on system-level isolation, see section 2.4.

2.3.5 HAT++ Control Logic

The control logic manages HAT+, HAT++, and application-specific functionality. A **Texas Instruments microcontroller** is used to implement the control logic. On the **I²C0 bus**, it behaves like an **AT28-compatible EEPROM**, thereby enabling the **HAT++ functionality** while maintaining compatibility with the standard HAT+ detection mechanism.

Tri-state buffers are used to switch the control paths and to selectively connect the required control electronics depending on the active operating mode.

Functions:

- HAT+ detection (EEPROM interface)
- HAT++ Instance ID detection
- LED control for status and mode indication
- Instance mode selection via push button
- Board resource management based on the selected Instance ID

Board Resource Management:

- Control of I²C addresses for I²C-based HATs
- Selection of SPI chip select (CS) for SPI-based HATs
- IRQ routing and handling
- Enable I²C pull-ups in Instance ID I
- HAT-specific control functions

For details on the HAT++ system, see section 3.

2.4 Isolation

Galvanic isolation is implemented between the field-side interface and the Raspberry Pi domain.

The interface and the Raspberry Pi are galvanically isolated. The isolation barrier provides a minimum clearance distance of $\geq 5\,\mathrm{mm}$. In areas where this spacing cannot be maintained, isolation slots are implemented to ensure a creepage distance of at least **5 mm**.

All components bridging the isolation barrier are specifically designed and specified for isolation applications. A detailed list of these components is provided in 3.

The isolation implemented on the board provides **functional galvanic isolation**. The boards are delivered without labeling or certified testing and must therefore be considered as providing **functional insulation only**.

Higher isolation ratings can be achieved by using an alternative BOM, application-specific validation, testing, labeling, and certification. This can be provided upon request (see 6.3).

RefDes	Supplier	MPN	Description
TR2101	Bourns	PAD002-T764113S	Transformer, see 2.3.4
C2012	PSK	FK21X102K502EGG	X1/Y2 Capacitor
U2003	TI	ISO7342C	Digital Isolator, see 2.3.2

Table 3: Isolation Components

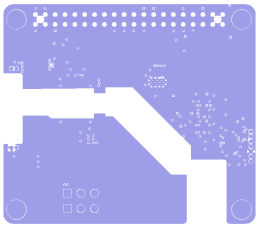
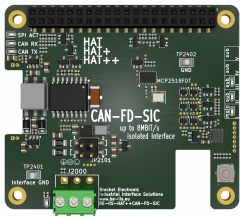
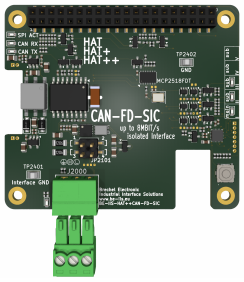


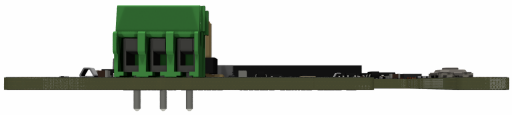
Figure 4: Isolation Barrier

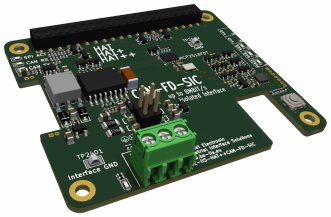
2.5 Connectors

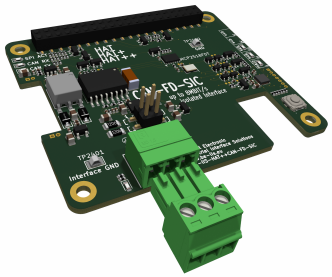
Connectors J2000 provides the physical CAN-FD bus interface. Two commonly used connector options can be populated.











Description:
3-pin 3.5 mm rising cage screw terminal block
Optimized for One-HAT-Operation

Option1:
Phoenix Contact PT 1,5/3-3,5-H
Order Code: 1984620 [5]

Option2:
Wuerth Elektronik WR-TBL 300VAC 10A 3P
Order Code: 691214110003 [6]

Note: or equivalent component with compatible mechanical and electrical specifications
Pinning: 1: SHIELD (optional, cable shield) 2: CAN-H 3: CAN-L

Description:
3-pin 3.5m mm PCB header and connector
Optimized for Stacked-HAT-Operation

Option1:
PCB Header: Phoenix Contact - MC 1,5/ 3-G-3,5 [7]
PCB Connector: Phoenix Contact - MC 1,5/ 3-ST-3,5 [8]
Order Code: 1844223(Header), 1840379(Connector)

Option2:
PCB Header: Wurth Electronic - WR-TBL 300VAC 12A 3P [9]
PCB Connector: Wurth Elecztronic - WR-TBL 300VAC 10.5Am Vertical 26-16AWG [10]
Order Code: 691305140003(Header), 691361100003 (Connector)

Pinning: 1: SHIELD (optional, cable shield) 2: CAN-H 3: CAN-L
Note: This connector is included in the delivery

2.6 Jumper and Configuration

Jumper JP2101 configures the CAN bus termination.
For proper CAN and CAN-FD operation, termination must only be enabled at the two physical end nodes of the bus segment.
End nodes: Jumper installed (120 Ω termination enabled)
Intermediate nodes: Jumper not installed (termination disabled)
In **point-to-point operation**, both connected devices are considered end nodes. Therefore, termination must be enabled on both nodes (jumper installed on both sides).
Figure 5 shows the jumper configuration. The installed jumpers are highlighted in yellow.

Pin	Signal	Pin	Signal
1	Termination	2	CAN-Low
3	Termination	4	CAN-High

Table 4: JP2101 pin assignment

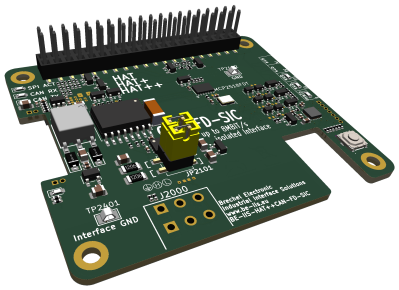


Figure 5: JP2101 termination jumper configuration for CAN/CAN-FD operation. Highlighted jumpers indicate enabled 120 Ω termination at the bus end nodes.

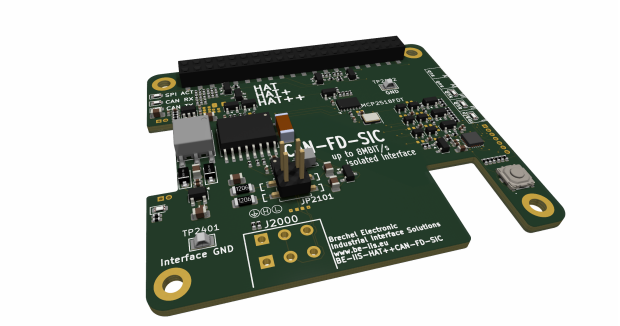


Figure 6: JP2101 jumper configuration for CAN/CAN-FD operation with termination disabled. Open jumpers indicate intermediate node configuration (no 120 Ω termination).

2.7 Indicators (LEDs)

In addition to the LED status bar on the side, which indicates the selected instance mode, there are additional LEDs located on the opposite side of the HAT:

- **SPI ACT**: Indicates active SPI communication
- **CAN RX**: Indicates CAN receive activity
- **CAN TX**: Indicates CAN transmit activity

An additional LED is located on the isolated CAN supply domain. This LED indicates proper operation of the isolated power supply.

2.8 Signal Polarity and Wiring Orientation

The CAN bus interface requires correct signal polarity. CAN_H and CAN_L must be connected to their respective terminals and must not be swapped.

The PCB silkscreen marking indicates the signal assignment:

- Ⓗ: CAN-High

- Ⓛ: CAN-Low
- ⏚: CANGND reference (separated from Raspberry Pi GND)

This applies to normal data communication as well as to bus wiring in industrial setups.

3 HAT++

HAT++ is designed to enable conflict-free stacking of multiple HATs while keeping hardware and software integration simple.

The concept is based on transparency rather than a black-box design. All hardware resources, configurations, and software components are openly accessible following the BE-IIS transparency principles.

HAT++ is not limited to multi-board systems. The same concept can also be used with a single HAT, providing a consistent and scalable approach from simple setups to complex systems.

3.1 HAT++ Compatibility Concept

The board is designed according to the BE-IIS HAT++ design principles.

HAT++ is fully backward compatible with:

- Raspberry Pi HAT+
- Standard Raspberry Pi HAT

The HAT can be operated in three different modes:

- **HAT (manual)** – full manual configuration
- **HAT+ (autodetect)** – automatic detection via EEPROM
- **HAT++ (autodetect + stackable)** – extended functionality, support for stacking multiple HATs

3.2 Instance ID

Each BE-IIS HAT++ provides multiple **Instance ID**.
An **Instance ID** defines how the HAT is connected to the Raspberry Pi in terms of hardware resources, including:

- Chip-Select (SPI)
- Interfaces
- Interrupt signals
- I^2C target address

Each **Instance ID** represents a unique hardware configuration, allowing multiple HATs to operate in parallel without resource conflicts.

Selection:

- The active **Instance ID** is selected using the on-board push button
- The selected mode is indicated by LEDs on the right side of the PCB
- The selected mode is stored permanently 2 s after being set
- The selected mode becomes active after a power cycle

Purpose:

- Enables conflict-free stacking of multiple HATs
- Allows flexible system configuration
- Provides deterministic hardware resource mapping

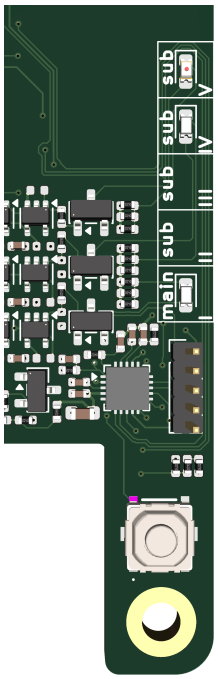


Figure 7: Instance ID indication on the PCB

Instance ID HW resources:

This board uses the following interfaces, either exclusively or shared with other HATs in the system:

Instance Mode	Signal	Pinname
I	CSN0	GP8
I	IRQ0	GP6
II	CSN1	GP7
II	IRQ1	GP5
III	CSN2	GP16
III	IRQ2	GP12

Table 5: Exclusive HW resources

Instance Mode	Signal	Pin
I & II & III	SCLK	GP11
I & II & III	MISO	GP9
I & II & III	MOSI	GP10
I & II & III & IV & V	RESET	GP13
I & II & III & IV & V	SCL0	GP1
I & II & III & IV & V	SDA0	GP0

Table 6: Shared HW resources

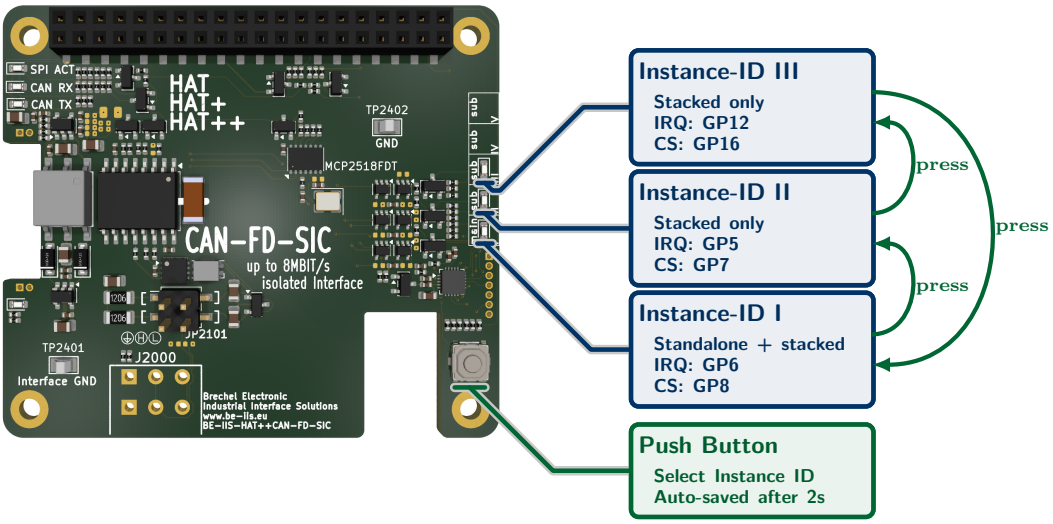


Figure 8: HAT Instance ID HW resources

3.3 Standalone and Stacked Operation

The board is designed to operate as part of the **HAT++ ecosystem**, enabling seamless combination with other HAT++ boards in a stacked configuration.
However, it can also be used as a standalone board, fully compatible with standard **Raspberry Pi HAT** / **HAT+** usage.
The following section provides a comparison of both operation modes.

Manual	Auto-detect (HAT++)
Configuration: • At least one HAT must be configured to **Instance ID I** to provide the required **I²C pull-up resistors** in accordance with the **Raspberry Pi HAT specification** • The BE-IIS installer is not used, or • BE-IIS overlays are removed from `/boot/firmware/BE-IIS*` if the installer has been used previously • If the **HAT+ auto-detection mechanism** is used, the overlay name must match `BE-IIS-HPP-CAN-SIC-I` and the Instance ID must be I • Otherwise, a custom overlay must be provided and applied manually (e.g., via `config.txt` or systemd)	**Configuration:** • At least one HAT must be configured to **Instance ID I** to provide the required **I²C pull-up resistors** according to the **Raspberry Pi HAT specification** and to enable the **auto-detection mechanism** • Each HAT must use a unique **Instance ID**
Software: • Device Tree overlays must be created and applied manually • Kernel modules must be built and installed manually • Practical examples and references can be found at [4]	**Software:** • Run the BE-IIS installer from GitHub [4] • Automatic overlay deployment and driver setup • All devices are enumerated and available after boot
Stacking: • Up to 5 HAT++ boards can be stacked • Each HAT must use a unique **Instance ID** • Alternatively, a custom **resource management** scheme can be implemented	**Stacking:** • Up to 5 HAT++ boards can be stacked • Each HAT must use a different **Instance ID** • **Instance ID I** must be used at least once
Result: • Full system control • Maximum flexibility • Highest integration effort	**Result:** • Automatic system integration • Scalable from single to multi-board setups • Minimal integration effort

4 Software and System Configuration

The BE-IIS-HAT++ system provides a unified platform for fast system integration.

- Predefined drivers and kernel modules
- Support for prebuilt modules and custom kernel builds
- Ready-to-use build and configuration scripts
- Centralized software repository [TODO]
- Typical setup time below a few minutes

After installation, the system can be used without further software modification.

4.1 System Support

The hardware is designed for use with Raspberry Pi platforms running a standard Raspberry Pi OS. All listed configurations have been validated or are expected to operate reliably with mainline Linux drivers.

- Supported Raspberry Pi platforms:
  - Raspberry Pi 2
  - Raspberry Pi 3B, 3B+, 3A+
  - Raspberry Pi 4B
  - Raspberry Pi 5
  - Raspberry Pi Zero, Zero W, Zero 2 W
- Supported Linux kernel versions:
  - $\geq$ **6.12**
  - Older kernel versions may work but are not officially supported
- Supported operating systems:
  - Raspberry Pi OS (32-bit and 64-bit)
  - Raspberry Pi OS Full and Lite
- Other Linux distributions:
  - Debian, Ubuntu, and other Linux distributions may work
  - Not tested or officially supported at this time

4.2 Driver & Integration

A standard Raspberry Pi OS installation is used as the base system. The provided installer configures all required components automatically, including kernel modules, Device Tree overlays, and systemd services.

- Prepare a Raspberry Pi hardware platform and operating system from the System Support list
- Run the provided Git-based installer
- Reboot the system to apply all configurations
- System supports the full BE-IIS-HAT++ portfolio

```
# install git
$ sudo apt install -y git

# clone installer
$ git clone https://github.com/be-iis/be-iis-installer.git

# enter directory
$ cd be-iis-installer

# run installer
$ ./scripts/install/install-all.sh
```

Drop-in CMD

```
sudo apt install -y git && cd  /Downloads && git clone https://github.com/be-iis/be-iis-installer.git && cd be-iis-installer && ./scripts/install/install-all.sh
```

After running the installer, a summary is printed to indicate the installation status and applied system changes.

Example output (shortened):

```
[INFO] Installation complete.
[INFO] Total scripts : 6
[INFO] Successful    : 6
[INFO] Failed        : 0

[INFO] Changes active after reboot:
[INFO]   - systemd service
[INFO]   - udev rules
[INFO]   - module autoload / runtime setup
Press ENTER to reboot now or CTRL+C to cancel...
```

4.3 Hardware–Software Interaction

The Instance ID can be changed during normal operation using the on-board control interface. The selected Instance ID is stored persistently after a short delay.

- Instance ID can be changed during runtime
- The selected Instance ID is stored persistently after approximately 2 seconds
- After changing the Instance ID, the interface becomes temporarily unavailable
- A system reboot restores full functionality with the updated configuration
- Instance ID **0** must be present at least once in the system
- In stacked configurations, each board must use a unique Instance ID

4.4 System Inspection

The system status and integration process can be inspected using standard Linux tools. All BE-IIS related services provide detailed runtime information via the system journal.

- View system integration logs:

```
# show BE-IIS system integration log
$ journalctl -b | grep BE-IIS
```

Example output (shortened):

```
BE-IIS Instance I   (0-0050): HAT detected -> BE-IIS-HPP-T1S-I
BE-IIS Instance II  (0-0060): HAT detected -> BE-IIS-HPP-CAN-SIC-II
BE-IIS Instance III (0-0070): HAT detected -> BE-IIS-HPP-LAN-III
BE-IIS Instance IV  (0-0074): HAT detected -> BE-IIS-HPP-UART-II
BE-IIS Instance V   (0-0076): HAT detected -> BE-IIS-HPP-MODBUS-III
BE-IIS HAT++ system integration complete.
```

Drop-in CMD

```
journalctl -b | grep BE-IIS
```

4.5 Interface Naming

All interfaces are assigned deterministic and persistent names using udev rules. This ensures stable device identification across reboots and different hardware configurations.

- Network interfaces are named based on function and instance index
- UART interfaces are exposed via symbolic links
- Naming is independent of kernel enumeration order
- udev rules location: `/etc/udev/rules.d/70-beiis-names.rules`

Example:

```
beiis-t1s0   # 10BASE-T1S interface
beiis-t1l0   # 10BASE-T1L interface
beiis-lan0   # Ethernet interface
beiis-can0   # CAN interface
beiis-uart0a # UART and MODBUS channel A
beiis-uart0b # UART and MODBUS channel B
```

```
$ cat /etc/udev/rules.d/70-beiis-names.rules
```

Drop-in CMD

```
cat /etc/udev/rules.d/70-beiis-names.rules
```

5 Electrical Specifications

5.1 Supply Voltage

Parameter	Min	Typ	Max
3.3 V Input [V]	2.7	3.3	3.6
5 V Input [V]	4.5	5	5.5

Table 7: Voltage Supply

5.2 Power Consumption

Parameter	Typ	Unit
Current @ 3.3 V	26	mA
Current @ 5 V	150	mA

Table 8: Current Consumtion

6 Environmental Conditions

6.1 Conditions

Condition	Min	Max
Operating Temperature [°C]	-40	+85
StorageTemperature [°C]	-20	+105
Relative humidity [%]	5	95

Table 9: Operation Conditions

6.2 Usage

Condition	parameter
Usage	indoor
Pullution degree	2
Operating altitude	up to 2000m

Table 10: Operation usage

6.3 EMC and Environmental Compliance (Preliminary)

The standard version of the board is provided without formal EMC or safety certification.
The hardware design is developed with consideration of commonly applied IEC standards, including:

- **ESD immunity:** IEC 61000-4-2
- **Electrical fast transient (EFT/Burst):** IEC 61000-4-4
- **Surge immunity:** IEC 61000-4-5
- **Conducted RF immunity:** IEC 61000-4-6
- **Radiated RF immunity:** IEC 61000-4-3
- **EMC immunity (industrial):** IEC 61000-6-2
- **EMC emission (industrial):** IEC 61000-6-4
- **Safety / isolation reference:** IEC 62368-1

These standards are not verified for the standard product variant.
Compliance with specific standards, test levels, or safety requirements is not guaranteed unless explicitly specified.
If defined EMC or isolation requirements are provided, application-specific validation, testing, and certification can be supported. Upon request, product variants with validated performance, including labeling, certification, and test reports (e.g. Hi-Pot testing), can be delivered.

7 Delivery Condition and Assembly

The product is delivered as a partially assembled kit intended for final user assembly. Mechanical accessories and connector components required for standard evaluation and stacked operation are included.

Order Code	BE-IIS-HPP-CAN
Condition	Assembly kit
Status	Partially assembled
Included Items	1× BE-IIS-HPP-CAN-PCBA-FT 4× M2.5x16 mm spacers 1× 2×20 pin stackable header 2x Jumper 1x PCB Header (see 2.5) 1x PCB Connector (see 2.5)
REACH & RoHS	Compliant with EU Directive 2011/65/EU and REACH Regulation (EC) No 1907/2006

Figure 9: Delivery condition

Available from DigiKey

DigiKey product page

8 Mechanical

8.1 Board Format

- Form factor: Raspberry Pi HAT+
- Mechanical dimensions: Raspberry Pi HAT compatible [11]
- Mounting hole pattern: Raspberry Pi HAT compatible [11]
- Stackszize: 16mm

8.2 3D Data

- Available on the BE-IIS product page [12]

9 Assembly

This product is delivered as a kit and requires basic soldering and mechanical assembly.

9.1 Assemble 2x20-Pin Main Connector

The 2x20-pin connector provides the interface to the Raspberry Pi. For proper HAT functionality, the connector must be assembled carefully.
A stackable 2x20-pin header is included in the delivery and is recommended for most applications, especially when using the BE-IIS HAT++ stacking system.
The header must be inserted between the Raspberry Pi and the HAT: the header is first mounted onto the Raspberry Pi, and the HAT is then plugged onto the header.

9.2 Assemble Spacer

To ensure mechanical stability and correct stacking height, spacers must be installed.

- Recommended spacer height: see Section 8.1
- Fix the PCB using appropriate screws and spacers
- Ensure stable mechanical mounting to avoid stress on the connector

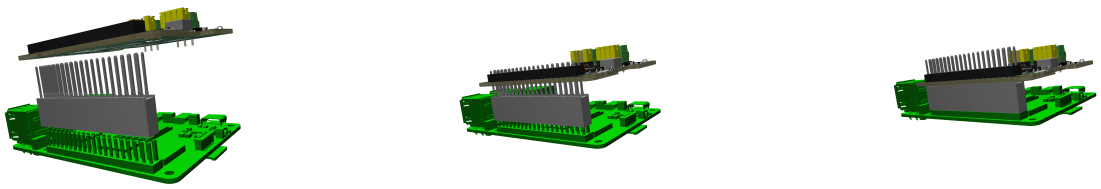


Figure 10: Assembly of the 2x20-pin stackable header between Raspberry Pi and HAT

The spacers define the stacking distance and provide mechanical fixation of the HAT.

9.3 3.5 mm Terminal Block Connector

A suitable screw terminal block is typically included in the delivery. Alternatively, a compatible PCB header (plug or socket variant) may be used, depending on the application. Refer to the corresponding product section for supported connector types.
Assembly instructions:

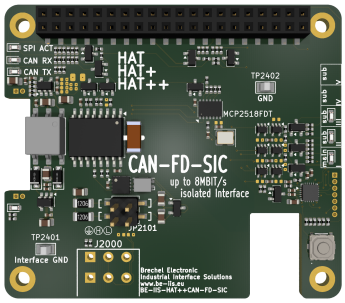
- Ensure correct orientation before soldering: the cable entry openings must face outwards from the PCB edge
- Insert the connector fully into the PCB to ensure proper mechanical alignment
- Solder all pins carefully with sufficient wetting

Important notes:

- Avoid direct contact between the soldering iron and the plastic housing of the connector, as this may cause visible damage or deformation
- Ensure clean solder joints without excessive solder to maintain proper mechanical fit

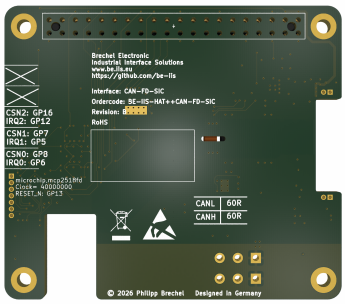
Correct assembly ensures reliable electrical contact and proper usability of the terminal interface.

9.4 Board Overview

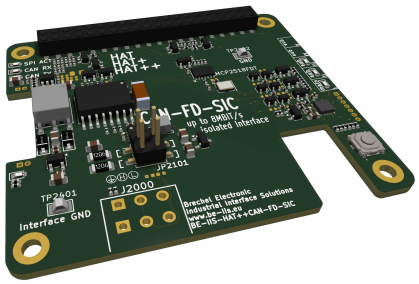


Top view

Front view



Bottom view



Side view

Figure 11: BE-IIS-HPP-CAN – mechanical overview

10 References

1. Microchip MCP2518FD product page
2. Texas Instruments ISO7342C Product Page
3. Texas Instruments TCAN1472-Q1 product page
4. BE-IIS Installer (Software and Setup Tools)
5. PhoenixContact PT 1,5/ 3-3,5-H - PCB terminal block
6. Wurth Electronic - WR-TBL Series 2141 - 3.50 mm Horiz. Entry Modular
7. PhoenixContact - MC 1,5/ 3-G-3,5 - PCB header
8. PhoenixContact - MC 1,5/ 3-ST-3,5 - PCB connector
9. Wurth Electronic - WR-TBL 300VAC 12A 3P - PCB header
10. Wurth Electronic - WR-TBL 300VAC 10.5Am Vertical 26-16AWG
11. Raspberry Pi HAT+ Specification
12. Schematic, PCB-Viewer, BOM, 3D-Model

11 Revision History

Revision	Date	Description
A.00	2026-02-12	Initial draft
B.00	2026-05-05	First version of B
B.01	2026-06-08	Added Raspberry Pi compatibility and HAT compatibility level sections

Company Information

Manufacturer

BE-IIS welcomes technical feedback, suggestions and improvement ideas. Business inquiries, cooperation requests and distribution opportunities are welcome.

Name	Philipp Brechel
Company	Brechel Electronic
Business	Industrial Interface Systems
Address	Hindenburgstraße 100/1
	73207 Plochingen, Germany

Contact

Email	`contact@be-iis.eu`
Web	`www.be-iis.eu`
GitHub	`github.com/be-iis`
LinkedIn	`linkedin.com/in/brechel`
Digikey	`https://www.digikey.de/de/supplier-centers/industrial-interface-solutions`
Languages	`German and English`