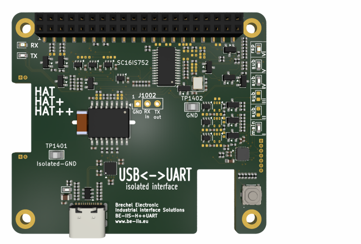
Isolated USB to Dual UART Industrial HAT++ for Raspberry Pi

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



Document ID: BE-IIS-HPP-UART_RevB_Datasheet
Revision: Rev. B
Date: May 20, 2026
Document Status: Released
Product Status: Final Testing

The UART available on the RSP 40-pin header enables direct serial communication and provides a reliable debug interface if other communication interfaces fail.

This HAT provides a galvanically isolated FTDI-based UART-to-USB interface for robust communication and protection against ground potential differences and noise coupling. As part of the BE-IIS HAT++ ecosystem, it is stackable with other HATs from the portfolio for seamless system expansion and simplified system integration.

Key Features

- FT234X USB-to-UART bridge
- Integrated ESD protection
- Galvanic isolation
- TX/RX activity LEDs
- Pi UART support (Instance VI)
- I^2C-UART support (Instances I, IV, V)
- Stackable HAT++ design
- RSP HAT+ compliant (2024)
- Stackable HAT (BE-IIS-HAT++)
- Configurable addressing and IRQ routing
- RoHS compliant

Product Description

The UART Industrial HAT is a Raspberry Pi HAT+ compliant board providing reliable UART access for debugging, service, or additional USB-to-UART interfaces.
It integrates an SC16IS752 dual UART controller and an isolated UART interface with galvanic isolation between the host system and the external interface. In fallback scenarios, the native Raspberry Pi UART can also be used.

Applications

- UART debugging and service interfaces
- Sevelopment and prototyping
- Education and training environments
- Protocol gateway and converter design
- Industrial communication evaluation and testing

Contents

1 Introduction

The BE-IIS HAT++ UART Industrial HAT is a Raspberry Pi HAT+ compliant interface board providing reliable UART connectivity for debugging, service, and system integration in industrial and laboratory environments.

The board integrates an SC16IS752 [1] dual UART controller and an FT234X [2] USB-to-UART bridge, enabling flexible expansion of serial interfaces while maintaining compatibility with standard operating system drivers.

Galvanic isolation separates the host system from the external interface, ensuring robust operation and protection against ground potential differences.

UART communication provides a simple and widely supported interface for low-level system access, debugging, and service operations. Its direct and deterministic nature makes it particularly suitable for diagnostics, bring-up, and fallback communication paths in complex systems.

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**, **CAN-FD**, and additional communication and measurement interfaces.

The HAT can be used to extend a Raspberry Pi system with additional UART interfaces or as a dedicated fallback interface when primary communication channels are unavailable. In such scenarios, the native Raspberry Pi UART remains accessible as an alternative debug interface.

The HAT is compatible with a wide range of Raspberry Pi platforms, including **Raspberry Pi 2, 3, 4, and 5**, as well as **Raspberry Pi Zero and Zero 2 W**.

All required software components, including drivers and system configuration, are provided through the **BE-IIS installer**, enabling a simple and reproducible setup process.

In line with the BE-IIS design philosophy of transparency, **schematics**, **PCB layout data**, and **3D models** are available, allowing full insight into the design and enabling users to build upon it.

The HAT can be used for evaluation, prototyping, system integration, 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.

2 Design Resources

All design files and software resources are publicly available.

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

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

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

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

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

GIT **GitHub Repository**
https://github.com/be-iis/be-iis-installer

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

3 Hardware Configuration

The hardware architecture is designed for robustness, flexibility, and seamless integration into industrial and embedded environments.

3.1 Main Features

The BE-IIS HAT++ UART Industrial HAT is a Raspberry Pi HAT+ compliant interface board that extends Raspberry Pi platforms with additional UART interfaces via USB, enabling flexible serial communication for debugging, service, and system integration.

It enables standard Raspberry Pi systems (e.g. Raspberry Pi Zero or Raspberry Pi 3/4/5, excluding Compute Module variants) to operate with multiple independent UART channels.

Communication between the Raspberry Pi and the UART interfaces is realized via an SC16IS752 I^2C-connected UART bridge [1] in combination with an FT234X USB-to-UART interface [2], ensuring compatibility with standard operating system drivers and seamless integration.

Galvanic separation between the host system and the external interface improves robustness against ground potential differences and enhances overall system reliability.

3.2 Main Features

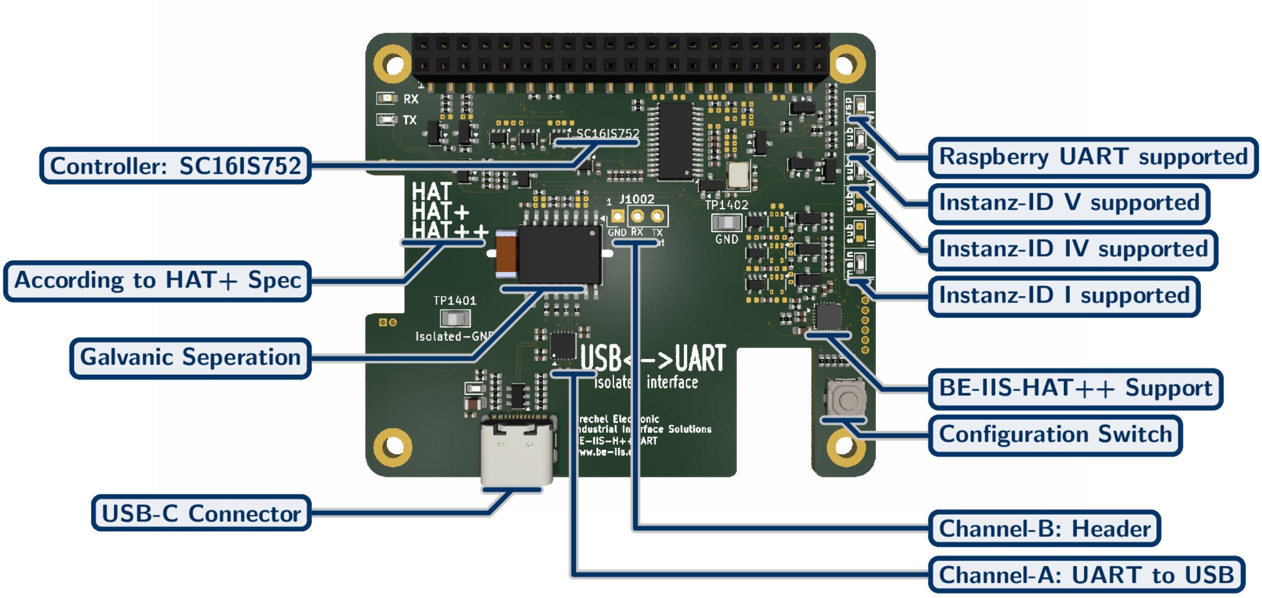


Figure 1: BE-IIS-H++UART top view with annotations

Device

- Supplier: FTDI
- Part: FT234XD
- Oscillator: 3.072MHz

UART Interface

- Pi UART Interface (14/15)
- Additional USB-to-UART interface (ttyUSB)

Separation

- Galvanically separated Interface
- Digital isolator (ISO734x, TI)

Signal Levels

- UART (TTL-level)
- USB-powered interface
- No user configuration required

Signal Levels

- Channel A: UART to USB
- Channel B: Routed to Header, 3V3 level

LEDs

- RX LED
- TX LED

3.3 Block Diagram

The block diagram shown in Figure 2 is simplified. It illustrates the power domains, separation barriers, main functional blocks, and principal signal paths.

The interrupt signal routing is not shown.

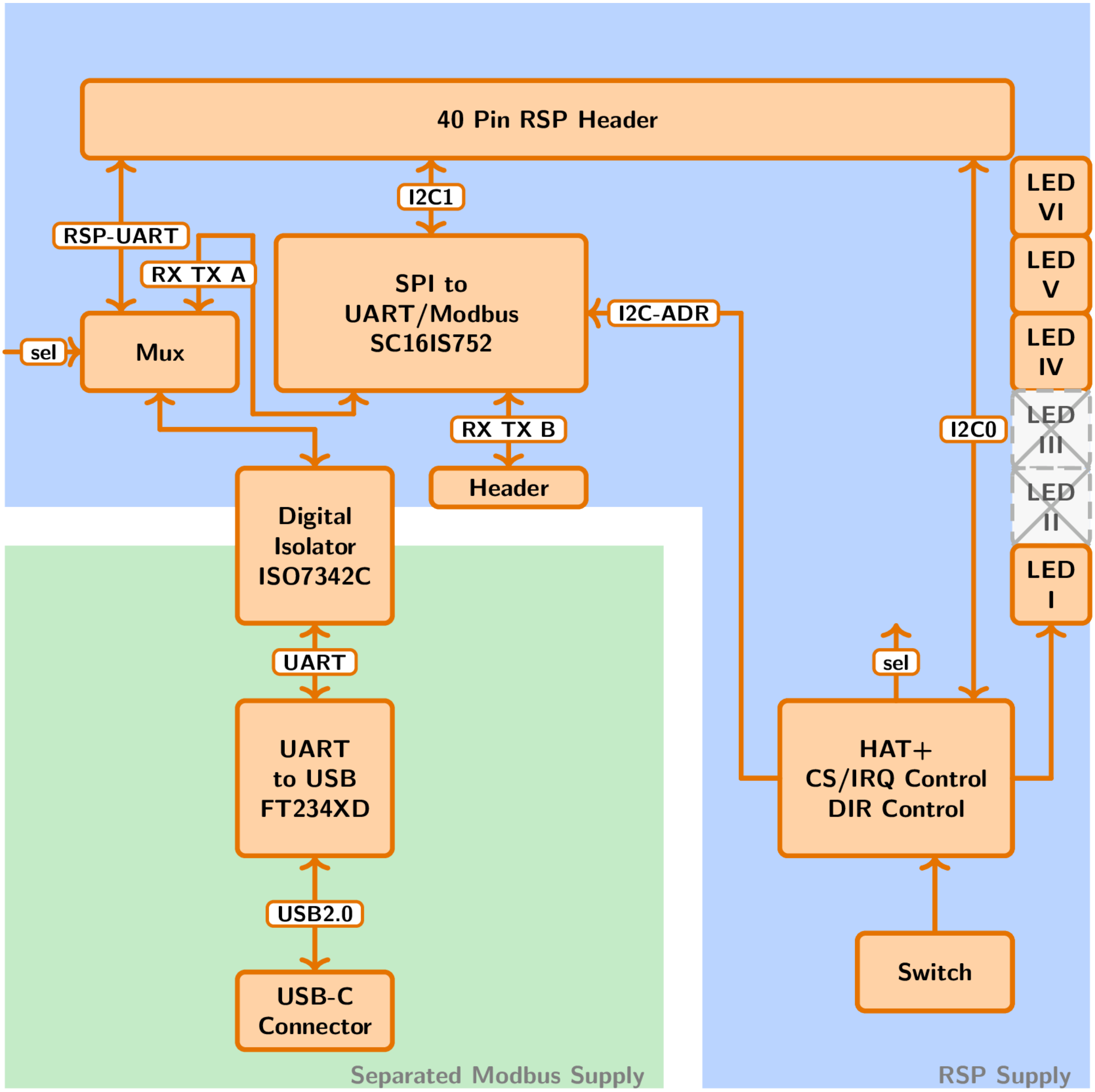


Figure 2: Simplified block diagram

3.4 Hardware Components

The selected hardware components and their interconnection are optimized to provide **reliable UART communication**, including **galvanic isolation** of the external interface. The design is fully aligned with the **HAT++ ecosystem**, enabling modular integration and conflict-free operation in stacked configurations.

Additionally, the design provides the option to use the Raspberry Pi UART in Instance ID VI, which operates outside the BE-IIS HAT++ ecosystem. In this mode, potential conflicts must be managed by the user.

3.4.1 SC16IS752 Serial-to-UART Converter

The **SC16IS752** is a dual-channel UART bridge providing two independent UART interfaces via an I²C interface.

Key characteristics:

- Dual UART channels
- I²C interface to Raspberry Pi
- Integrated FIFOs for reliable data buffering

Oscillator:

- External oscillator defines baud rate generation
- Frequency: 22.1184 MHz

The HAT can be configured in I²C-to-UART mode in Instance IDs I, IV, and V. Additionally, it is possible to connect the Raspberry Pi UART to the UART-to-USB circuit in Instance ID VI. In this mode, Channel A of the SC16IS752 is not used (left unconnected).

Channel B is always connected to a 3-pin header.

UART Channel A:

- Instance IDs I, IV, and V: Connected to UART-to-USB circuit
- Instance ID VI: Open (not connected)

UART Channel B:

- Connected to a 3-pin header
- Pin 1: GND
- Pin 2: RX (input)
- Pin 3: TX (output)
- I/O voltage: 3.3 V
- Isolation: None
- Pitch: 2.54 mm
- Reference: J1002

For further details, refer to the supplier's product documentation (see 1).

3.4.2 UART to USB FT234XD

The **FT234XD** by *FTDI* is a compact **USB-to-UART bridge** designed to provide a simple and reliable interface between USB hosts and serial UART devices.

It enables direct connection of UART interfaces to USB systems such as the Raspberry Pi, supporting standard communication speeds and low-latency data transfer.

The device integrates the complete USB protocol handling, allowing seamless operation without the need for external USB controllers. Communication is presented to the operating system as a virtual COM port, ensuring broad compatibility across platforms.

In Linux-based systems, the FT234XD is supported by the `ftdi_sio` driver and integrates into the standard **TTY subsystem**, allowing straightforward access to the UART interface.

The FT234XD is optimized for embedded and industrial applications requiring robust, easy-to-use USB-to-serial connectivity for debugging, service, and system integration tasks. For further details, refer to the supplier's product documentation (see 2).

3.4.3 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.

The digital isolator handles only Channel A of the SC16IS752 in Instance IDs I, IV, and V, or the Raspberry Pi UART in Instance ID IV. Channel B of the SC16IS752 is not covered by this isolation circuit.

For details on system-level isolation, see section 3.5. For further details, refer to the supplier's product documentation (see 3).

3.4.4 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 **I^{2}C0 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 I^{2}C addresses for I^{2}C-based HATs
- Selection of SPI chip select (CS) for SPI-based HATs
- IRQ routing and handling
- Enable I^{2}C pull-ups in Instance ID I
- HAT-specific control functions

Besides the BE-IIS HAT++ mechanism, Instance ID VI is supported by this HAT. This mode is outside the BE-IIS HAT++ ecosystem. It can only be used once, as it utilizes the native Raspberry Pi UART, which becomes occupied when enabled.

Please ensure that this interface is not already in use by internal Raspberry Pi functions, such as Bluetooth.

This feature was implemented because access to the native UART is a common and useful debug option when working with the Raspberry Pi.

For details on the HAT++ system, see section 4.

3.5 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 1.

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 7.3).

RefDes	Supplier	MPN	Description
C1006	PSK	FK21X102K502EGG	X1/Y2 Capacitor
U1003	TI	ISO7342C	Digital Isolator, see 3.4.3

Table 1: Isolation Components

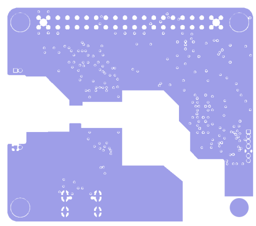


Figure 3: Isolation Barrier

The digital isolator handles only Channel A of the SC16IS752 in Instance IDs I, IV, and V, or the Raspberry Pi UART in Instance ID IV. Channel B of the SC16IS752 is not covered by this isolation circuit.

3.6 Connectors

The board features a USB Type-C connector for power supply and data communication.

The USB-C interface provides the connection to the host system and is used for the USB-to-UART communication channel. It ensures reliable mechanical stability and reversible plug orientation [5].

Although a USB Type-C connector is used, the electrical interface operates as a USB 2.0 device [4]. No USB 3.x functionality is supported.

The connector does not require external configuration for standard operation.

3.7 Signal Polarity and Wiring Orientation

The USB Type-C connector is fully reversible and can be inserted in either orientation without affecting functionality [5].

4 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.

4.1 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

4.2 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^{2}C 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

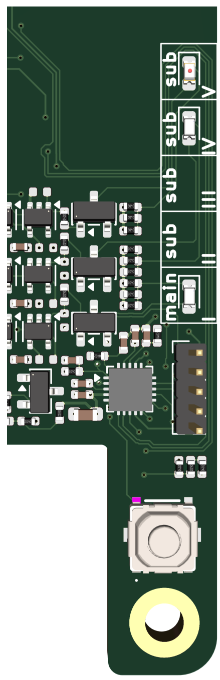


Figure 4: Instance ID indication on the PCB

Instance ID HW resources:

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

Instance Mode	Target Address / Signal	Pin
I	0x9A	GP6
IV	0x98	GP14
V	0x92	GP25
VI	UART TX	GP14
VI	UART RX	GP15

Table 2: Exclusive HW resources

Instance Mode	Signal	Pin
I & IV & V & SCL1	GP3	
I & IV & V & SDA1	GP2	
I & II & III & IV & V	RESET	GP13
I & II & III & IV & V	SCL0	GP1
I & II & III & IV & V	SDA0	GP0

Table 3: Shared HW resources

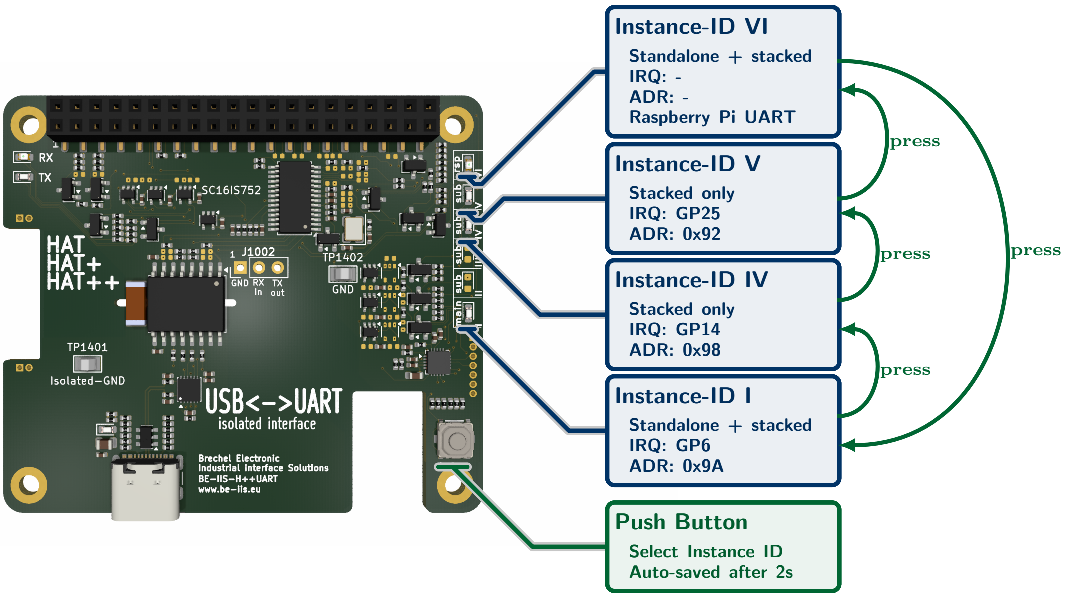


Figure 5: HAT Instance ID HW resources

4.3 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 **I^2C 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-UART-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 **I^2C 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 [8]	**Software:** • Run the BE-IIS installer from GitHub [8] • 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

5 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.

5.1 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

5.2 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...
```

5.3 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

5.4 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
```

5.5 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`

6 Electrical Characteristics

6.1 Supply Voltage

Parameter	Min	Typ	Max
3.3 V Input [V]	3.0	3.3	3.6
5 V Input (Pi) [V]	–	5.0	–
5 V Input (USB-C) [V]	4.75	5.0	5.25

Table 4: Voltage supply

Note: The board does not draw power from the Raspberry Pi 5 V rail. The logic domain is supplied by the 3.3 V rail of the Raspberry Pi header, while the USB/PHY domain is powered independently via the USB-C [5] input.

6.2 Current Consumption

Parameter	Typ	Unit
Current @ 3.3 V (Raspberry Pi)	15	mA
Current @ 5 V (Raspberry Pi)	0	mA
Current @ 5 V (USB-C)	14	mA

Table 5: Current consumption

Note: The board does not draw power from the Raspberry Pi 5 V rail. The logic domain is supplied by the 3.3 V rail of the Raspberry Pi header, while the USB/PHY domain is powered independently via the USB-C [5] input.

7 Environmental Conditions

7.1 Conditions

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

Table 6: Operating conditions

7.2 Usage

Condition	Parameter
Usage	indoor
Pollution degree	2
Operating altitude	up to 2000 m

Table 7: Operating usage

7.3 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.

8 Delivery

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

Table 8: Delivery condition and included parts

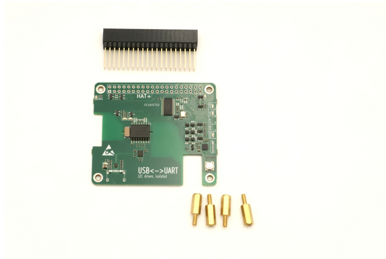


Figure 6: Delivery condition

9 Mechanical

9.1 Board Format

- Form factor: Raspberry Pi HAT+
- Mechanical dimensions: Raspberry Pi HAT compatible [7]
- Mounting hole pattern: Raspberry Pi HAT compatible [7]
- Stackszize: 16mm

9.2 3D Data

- Available on the BE-IIS product page [9]

10 Assembly

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

10.1 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.

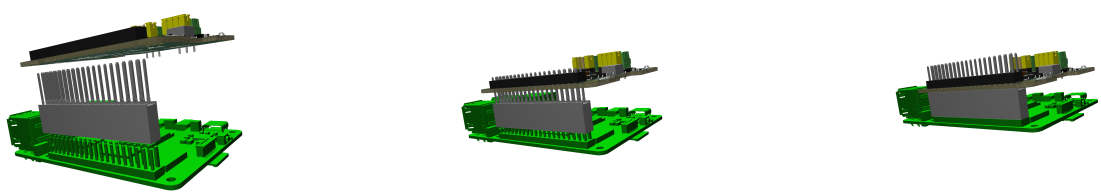


Figure 7: Assembly of the 2x20-pin stackable header between Raspberry Pi and HAT

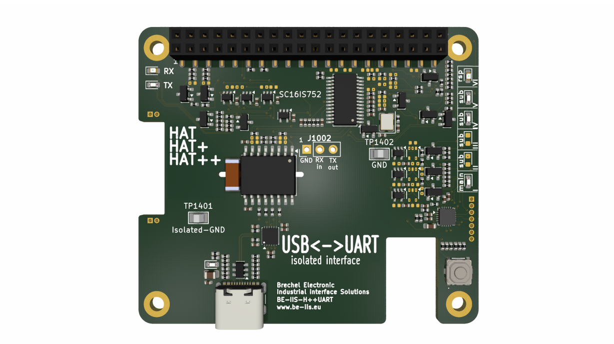
10.2 Assemble Spacer

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

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

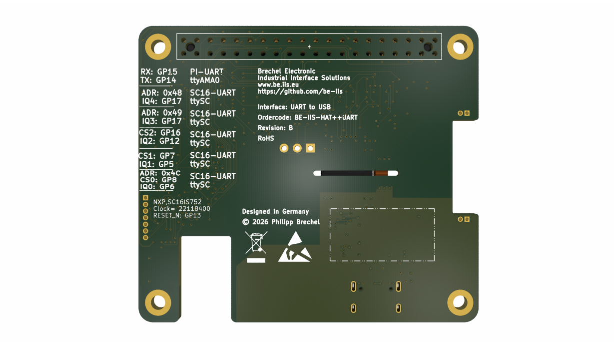
The spacers define the stacking distance and provide mechanical fixation of the HAT.

10.3 Board Overview

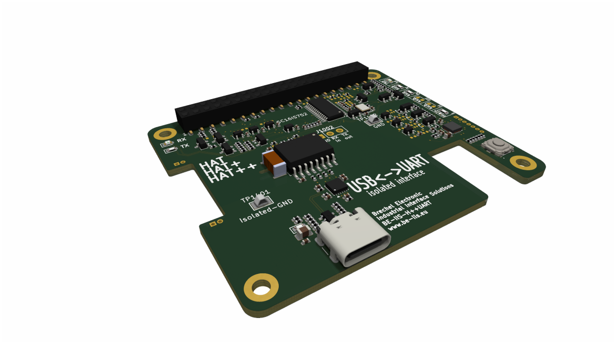


Top view

Front view



Bottom view



Side view

Figure 8: BE-IIS-HPP-UART – mechanical overview

11 References

1. NXP SC16IS752 Datasheet
2. FTDI FT234XD Datasheet
3. Texas Instruments ISO734x Datasheet
4. USB 2.0 Specification
5. USB Type-C Cable and Connector Specification
6. NXP I²C-bus Specification and User Manual
7. Raspberry Pi HAT+ Specification
8. BE-IIS Installer (Software and Setup Tools)
9. Schematic, PCB-Viewer, BOM, 3D-Model

12 Revision History

Revision	Date	Description
A.00	2026-02-12	Initial draft
B.00	2026-05-05	First version of B

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/philipp-brechel`
Languages	`German and English`

Brechel Electronic – Industrial Interface Systems

Designed by Philipp Brechel (Germany)