# Hardware Guide for Revision 3.0

# Contents

# 1. Overview

This document describes the hardware installed in the Kontron Portable (P100) server as well as the wiring and interaction between the installed components.
This is especially useful if you intend to use the P100 without the supplied OS.

The P100 is split into several subsystems which interact to each other, see the hardware block diagram in the next chapter for an overview. The subsystems are referred to these acronyms in this document:

- **CPU**: x86_64 based computing unit
- **MCU**: Microcontroller and Sensor unit
- **PU**: Power distribution and management subsystem
- **WAU**: Wireless AccessPoint unit

Each unit will be described in detail, as well as (possible) interactions between the units to allow you to operate the P100.

## 1.1. Operating your P100

This section contains vital information on how to operate your P100 safely. Please pay close attention to the following chapters:

### 1.1.1. Environmental and Installation Information

The following environmental conditions apply to the P100 which must not be exceeded:

| Type | Air temperature | Humidity (non-condensing) |
|---|---|---|
| Storage | 0°C to 60°C | 20% to 80% |
| Operating | 10°C to 30°C<br>35°C for 3 minutes | 20% to 80% |

For operation, the P100 must be secured in a way so it can only be accessed by authorized personnel and is locked into position. The device is intended for indoor use only, it can be mounted and operated either horizontally or vertically in a way that the front panel remains accessible.

### 1.1.2. Battery Warnings

- Note that the device contains two Lithium-Ion batteries. Do not disassemble, deform, or heat these batteries. Do not throw batteries into fire. Never apply force to insert or remove a battery from the P100.
- Keep batteries away from direct sunlight, high temperature and high humidity environments.
- Never insert a different battery type than RRC2054.
- In addition to the user accessible Li-Ion batteries, an additional CR2032 based lithium button cell CMOS / RTC backup battery is installed in the P100.

### 1.1.3. Power Supply

Only use the AC power supply unit provided with the P100 or a suitable DC power supply as described in chapter 4.2.1. To remove power from the P100, unplug the AC adapter from the wall plug or the DC supply from the P100 power socket and remove the RRC2054 batteries as described in chapter 4.1.3. Make sure that the wall plug is always accessible.

**ATTENTION:** Power outages must not be less than 200ms.

### 1.1.4. Regulatory Information

#### 1.1.4.1. FCC

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

This device contains FCC ID:

- Z9W-JLP
- PD99260NG
- N7NEM75L

#### 1.1.4.2. IC

The equipment complies with CAN ICES-3 (A)/NMB-3(A)

This device contains IC:

- 11468A-JLP
- 1000M-9260NG
- 2417C-EM75L

# 2. Hardware Changes

## 2.1. Hardware changes from Revision 1.0 to 1.1

The following changes have been implemented in this hardware revision, refer to the according chapter in this document for details:

- **Upgraded ADS-B Receiver hardware**
  - New ADS-B receiver hard- and firmware which supports signal strength indication
- **Added software-switchable power supplies for various MCU components:**
  - ADS-B receiver
  - Graphic display
  - 9-DOF and environmental sensors
- **Changed graphic display data connection to I2C Bus #1**
- Added feature to disable LTE modem via MCU GPIO
- Added feature to detect if CPU is powered on via MCU GPIO
- Added multiple GPIO connections between MCU and WAU
- Default location for WAU antennas has been swapped (WAU antennas installed in bottom antenna slot instead of top)
- Added discrete unit option (available as ordering option)
- Added DC input voltage measurement feature for MCU
- Added wiring for PROGRAM and RESET pins of the MCUs microcontroller

## 2.2. Hardware changes from Revision 1.1 to 2.0

The following changes have been implemented in this revision:

- **Changed graphic display data connection from I2C to SPI**
- **Changed ADS-B data connection from CPU to MCU**

## 2.3. Hardware changes from Revision 2.0 to 3.0

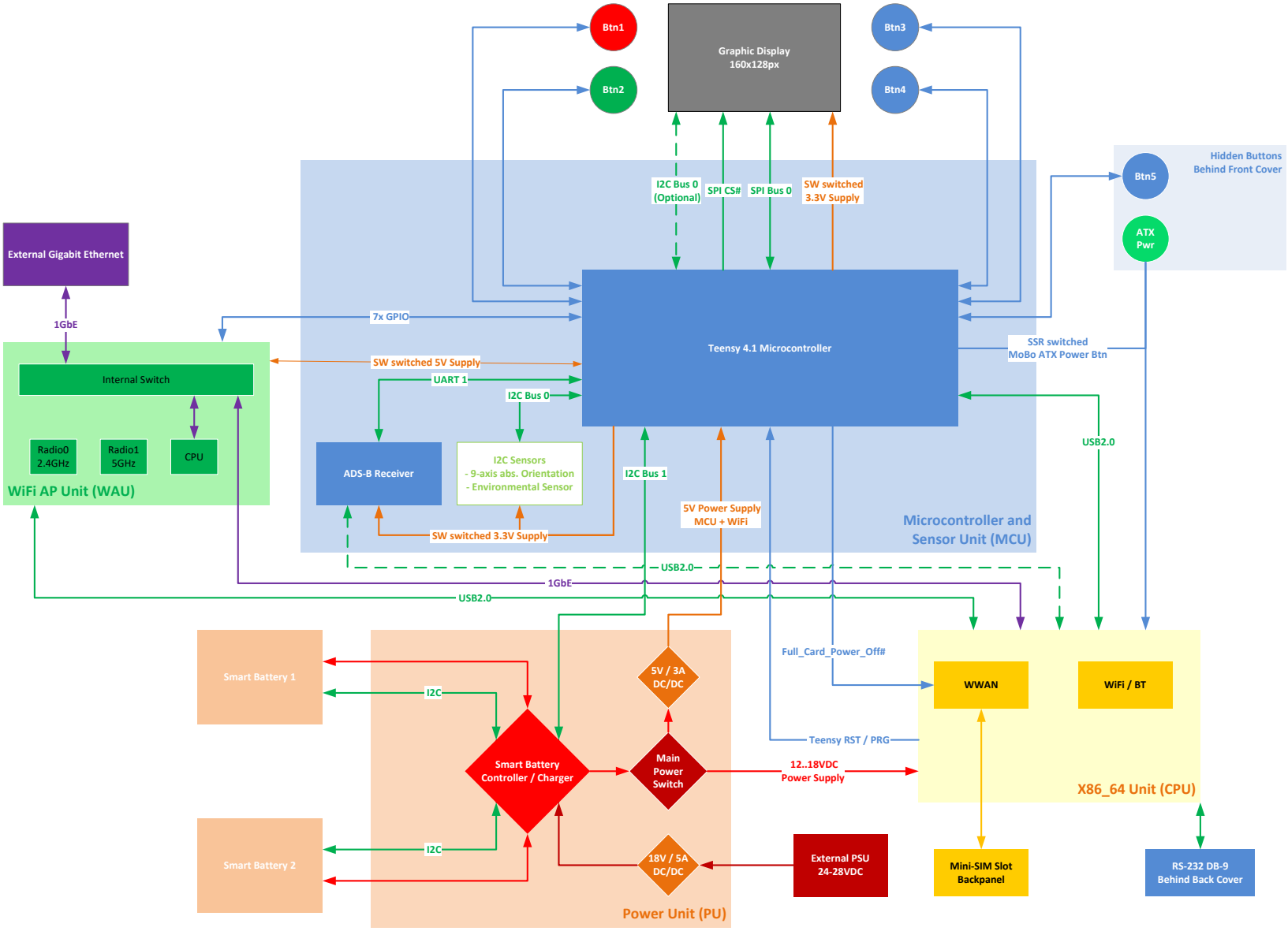
The following changes have been implemented in this revision:

- **Changed CPU board manufacturer**
  - **Changed programmatic method to access CPU GPIO pins**
  - **Changed PCIe bus numbers for peripherals (NVMe disks)**
  - **Required Power-On button-press duration if Super-S5-ECO-Mode is enabled**
- **Upgraded MCU microcontroller from Teensy 3.6 to Teensy 4.1**
  - **Changed multiple GPIO connections between MCU and WAU**
  - **Changed graphic display SPI CS pin**
  - **Changed I2C Bus #1 pins**
  - **Changed Analog DC-Input voltage measurement**
  - **Changed optional Display I2C connection to I2C Bus #0**
- Upgraded PU to support higher DC input voltage
- Changed external AC adapter for new PU
- Changed barrel type power connector to circular connector type KPJX-PM4

- Removed optional Discrete Unit
- Removed optional 10GbE interface
- Removed unused Fan connector from MCU board

**ATTENTION:** Changes marked **bold** might require software changes (i.e. MCU firmware) compared to the previous hardware revision, verification recommended.

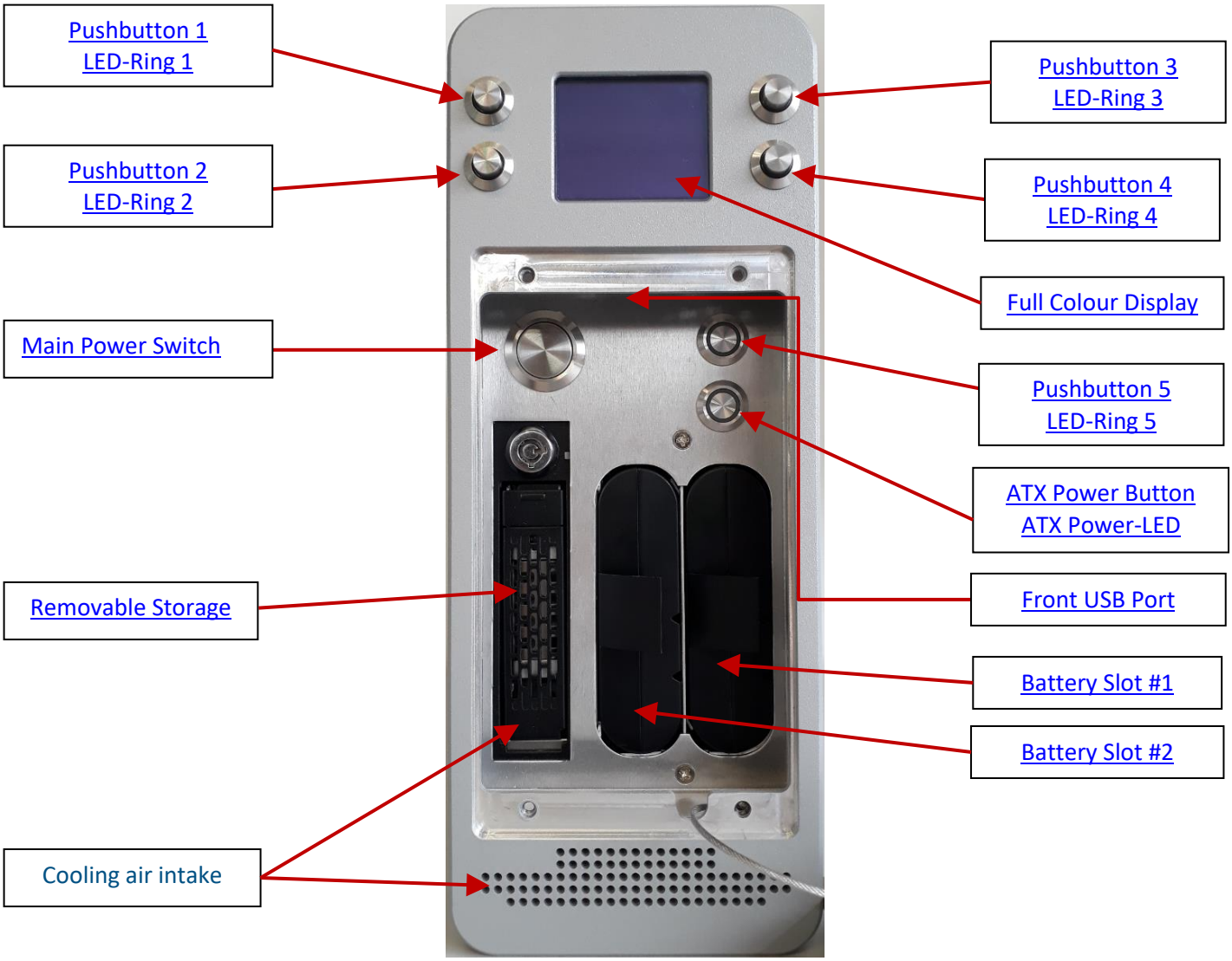
## 3. Hardware Block Diagram



# 4. Components and Connectors

## 4.1. P100 Front Panel

The P100 has the following connectors and components on the front:




To power up the CPU, first press the main power switch, then **press and hold the ATX power button until the LED ring will light up green.**
**Note**: Depending on the ordered hardware setup, powering up the CPU might require a long press (up to 8 seconds) on the ATX power button. For details see chapter ATX Power button and Super-S5-ECO-Mode.
To shut down the OS running on the CPU, shortly press the ATX power button to initiate shutdown (depends on ACPI OS configuration). The green LED ring will turn off when the CPU has finished shutdown.

**ATTENTION:** Make sure that the cooling air intakes marked above are not covered to avoid overheating the P100. Unused battery slots must not remain open to allow correct airflow through the U.2 cage. The P100 will be shipped with appropriate battery slot cover sheets according to your order (single or no battery setup).

### 4.1.1. Main Power Switch

The latching main power switch allows you to mechanically power on and off the P100. When the P100 main power switch is enabled, the MCU will boot the firmware and the P100 UI on the front panel will become available (depending on the installed MCU firmware).

### 4.1.2. Removable Storage

The removable storage is an ordering option, which allows you to equip the P100 with a removable NVMe based storage. The installed disk has an SFF-8639 plug equal to an U.2 NVMe disk.
Depending on the equipped CPU module, the external slot provides between 1 and 4 PCIe 3.0 lanes to the installed disk.

### 4.1.3. Smart Batteries

The P100 is equipped with one or two Li-Ion based Smart batteries. In addition to this, the P100 may be operated without Li-Ion batteries installed. If running without Li-Ion batteries, also pay attention to chapter 4.2.1.
These batteries can be either charged when installed in the P100 and external power is provided, or in an optionally available external charging bay.

The batteries are hot pluggable. To remove a battery, open the P100 front panel cover by unscrewing the knurled thumbscrews. Then gently pull out the desired battery with the plastic strap.
To re-insert a battery, make sure the plastic strap is on the left side of the battery and gently push the battery into the slot until it is fully inserted.

**ATTENTION:** Never apply force when inserting a battery into the P100!

After connecting the power supply, the batteries will be charged automatically. It is not necessary to power up the CPU using the ATX power button or to turn on the main power switch.
Charging two fully depleted batteries installed in the P100 will take about 3 hours.

**ATTENTION:** When storing batteries consider these specifications:

- Batteries need to be set into shipping mode.
- The recommended storage temperature is +25°C.
- Extended storage at > 40°C could degrade battery performance and lifetime.
- To avoid an under voltage cell situation, store the battery pack at 30% state of charge (SOC) and recharge it every 12 months.
- Do not store depleted batteries; recharge to at least 30% SOC.

For details on how to place the batteries into shipping mode, see chapter Hardware-only MCU Firmware. Batteries placed into shipping mode will disable the power outputs and disable the internal electronics, allowing you to store the batteries (also when installed in the P100).

The battery has 4 LEDs to show the current state of charge. Each LED segment represents 25 percent of the full charge capacity. The LED pattern definition is given in the table below. The LED's illuminate for 4 seconds following the pushbutton activation on the battery. If the battery voltage is too low, there will be no LED indication. Immediately recharge the battery in this case.

| Capacity (SOC) | LED Indicators | | | | Note |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | |
| < 10% | ■ | | | | Blinks |
| 10% - 25% | ■ | | | | Lit for 4 seconds |
| 26% - 50% | ■ | ■ | | | Lit for 4 seconds |
| 51% - 75% | ■ | ■ | ■ | | Lit for 4 seconds |
| 76% - 100% | ■ | ■ | ■ | ■ | Lit for 4 seconds |

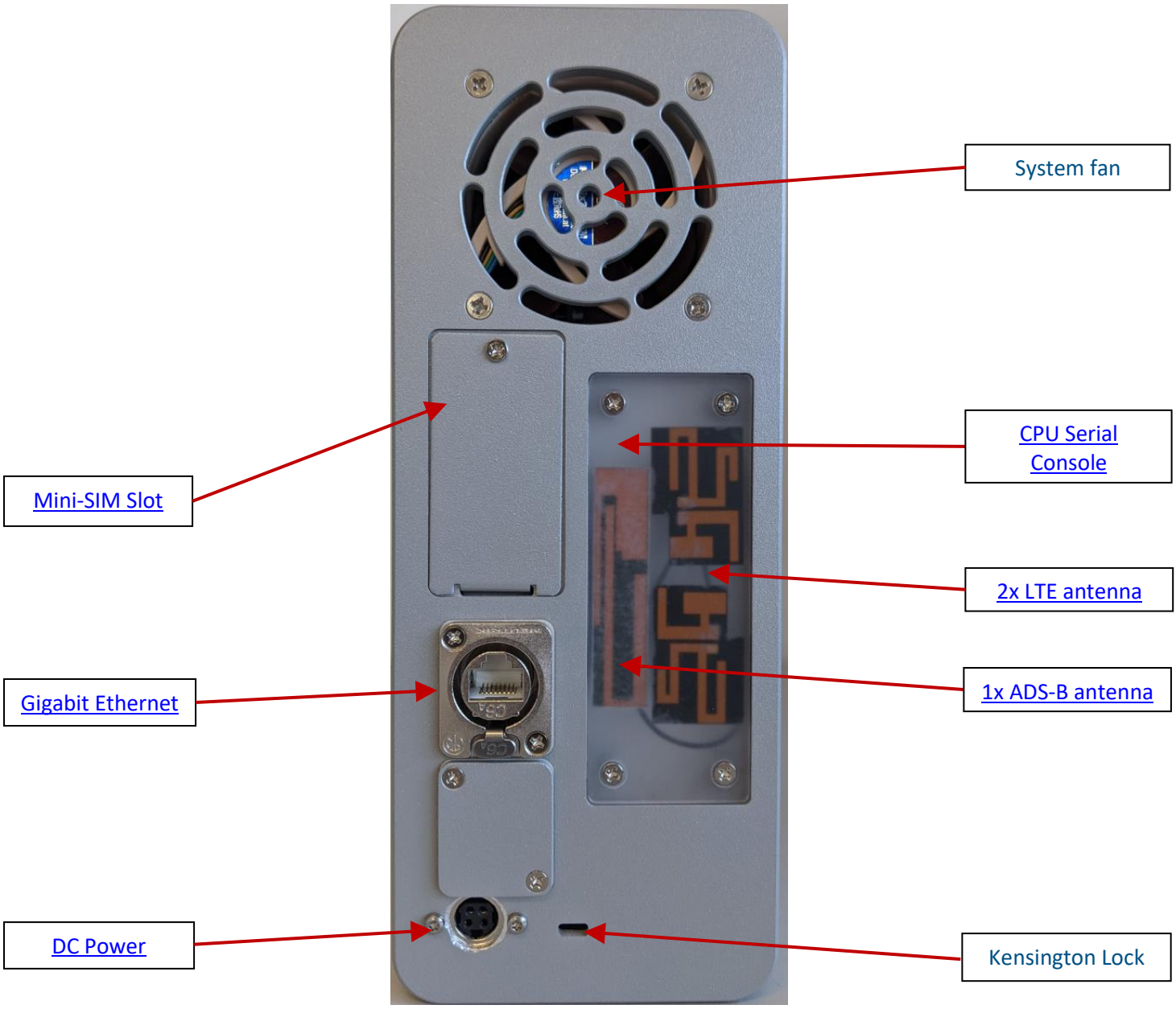
**ATTENTION:** Running the P100 on batteries with battery voltage less than 12V will lead to system (CPU) instability. For fresh batteries, this voltage will be underrun at a SOC of around 5% but could raise as the battery wears out during its lifecycle. Using the possibilities provided by the MCU by integrating an appropriate power management is strongly recommended.

### 4.1.4. Front USB3.0 Port

The P100 provides an USB3.0 (5Gbit/s) port on the top of the hidden front panel. To access the port, open the P100 front panel cover by unscrewing the knurled thumbscrews.

## 4.2. P100 Rear Panel

The P100 has the following connectors and components on the rear panel:




CPU serial console port as well as 1x ADS-B and 2x LTE antennas are hidden behind the acrylic cover.

**ATTENTION:** System fan exhausts on the rear, **do not cover the fan outlet**.

### 4.2.1 DC Power Socket

**ATTENTION:** Only use the 24V AC/DC power supply unit provided with the P100 or DC supply which fulfills the following requirements as an external power source:

| | Minimum | Nominal | Maximum |
|---|---|---|---|
| Nominal DC Supply Voltage [V] | | 24 | 28 |
| Normal Operation DC Supply Voltage (max. ratings) [V] | 21 | | 53 |
| No-Charge DC Supply Voltage range [V] (*) | 15 | | 21 |
| Maximum DC power requirement [W] | | | 90 |

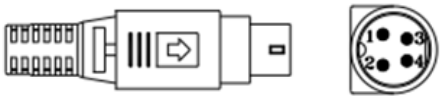
(*) Depending on the batteries state of charge, system might be operated by DC-In Voltage, but batteries will not be charged. If input voltage decreases below the minimum No-Charge Input Voltage, the system will be powered by battery.

In addition to the electrical specification above, 3rd party DC power supplies must fulfill PS2 classification according to EN 62368-1 or LPS classification according to EN 60950-1. The short circuit current of the power supply must not exceed 8A.

The following safety features are included in the PU:

- ESD protection diode
- 5A fuse (slow)
- Reverse polarity protection
- Inrush current limiter

The P100 is shipped with an external power supply unit providing 90W power at 24V direct current. It requires an input of 90 to 240V AC at 50 to 60Hz.

| Connector | Pin | Output |
|---|---|---|
| Kycon KPPC-4P  | 1 | +V |
| | 2 | +V |
| | 3 | -V |
| | 4 | -V |

**ATTENTION:** Depending on the connected power supply, the P100 might take up to 90 seconds to start up if powered on while no or fully depleted Li-Ion batteries are inserted. This behavior is by design.

## 4.3. Microcontroller and Sensor Unit (MCU)

The P100 is equipped with a microcontroller board that can perform the following tasks:

- Provide power to or powering on/off several system components
- Verify states of system components
- Measure environmental data and absolute orientation
- Receive ADS-B data
- Handle external UI (buttons, display)
- Communicate with PU

The MCU was designed to be able to aid you to maximize battery lifetime of the P100 by powering on components only when required (i.e. WAU and CPU). Sensors as well as ADS-B data will support you in finding the current flight state.

### 4.3.1. Microcontroller GPIO overview

This section will give you a detailed view of available in- and outputs of the MCU.

As a reference concerning pinout of the installed Teensy 4.1 microcontroller you might use the following sheet which shows the available I/Os as well as their possible default and alternate functionality:
https://www.pjrc.com/teensy/card11a_rev4_web.pdf
https://www.pjrc.com/teensy/card11b_rev4_web.pdf

All supported functions will be described in detail in the subsequent chapters.

These I/Os are mapped on the MCU as shown in this table:

| Pin Function | Mode | Function | to Device | Note |
|---|---|---|---|---|
| 0 | RX1 | UART1 for ADS-B | ADS-B | |
| 1 | TX1 | UART1 for ADS-B | ADS-B | |
| 2 | GPIO | Digital I/O from/to WAU | WAU (GPIO3) | |
| 3 | GPIO | Digital I/O from/to WAU | WAU (GPIO4) | |
| 4 | GPIO | Digital I/O from/to WAU | WAU (GPIO5) | |
| 5 | D-OUT | WAU Power | WAU | Setting to HIGH enables power to WAU |
| 6 | D-OUT | CPU ATX Power Button | CPU | Setting to HIGH shorts ATX power button from CPU |
| 7 | D-OUT | CPU LTE Modem Full_Card_Power_Off# | CPU (Modem CONF) | Setting to HIGH disables modem by pulling Full_Card_Power_Off# Pin on the Modem to GND |
| 8 | D-IN | Pushbutton 1 | Pushbutton | Requires Software INPUT_PULLUP |
| 9 | GPIO | Digital I/O from/to WAU | WAU (GPIO0) | |
| 10 | GPIO | Digital I/O from/to WAU | WAU (GPIO1) | |
| 11 | MOSI0 | SPI-MOSI – Bus #0 | Display | |
| 12 | MISO0 | SPI-MISO – Bus #0 | Display | |
| 13 | SCK0 | SPI-SCK – Bus #0 | Display | |
| 14 | D-OUT | SW-switchable 5V Power-Supply | For optional I2C Display | Only used for optional I2C data connection to display (J21) |
| 15 | Analog-IN | DC-IN Voltage readout | PU DC-Input | |
| 16 | SCL1 | I2C-SCL – Bus #1 | Battery-Backplane | I2C clock max. 100kHz |
| 17 | SDA1 | I2C-SDA – Bus #1 | Battery-Backplane | I2C clock max. 100kHz |
| 18 | SDA0 | I2C-SDA – Bus #0 | BNO055 / BME280 | |
| 19 | SCL0 | I2C-SCL – Bus #0 | BNO055 / BME280 | |
| 20 | CTS1 | UART1 for ADS-B | ADS-B | |
| 21 | RTS1 | UART1 for ADS-B | ADS-B | |
| 22 | D-OUT | SW-switchable 3V3 Power-Supply | BNO055 / BME280 ADS-B | |
| 23 | D-OUT | SW-switchable 3V3 Power-Supply | Display | |
| 24 | | Reserved | | |
| 25 | D-IN | Pushbutton 2 | Pushbutton | Requires Software INPUT_PULLUP |
| 26 | D-IN | Pushbutton 3 | Pushbutton | Requires Software INPUT_PULLUP |
| 27 | D-IN | Pushbutton 4 | Pushbutton | Requires Software INPUT_PULLUP |
| 28 | D-IN | Pushbutton 5 | Pushbutton | Requires Software INPUT_PULLUP |
| 29 | D-OUT | LED-Ring 1 | LED-Ring | |
| 30 | D-OUT | LED-Ring 2 | LED-Ring | |
| 31 | D-OUT | LED-Ring 3 | LED-Ring | |
| 32 | D-OUT | LED-Ring 4 | LED-Ring | |
| 33 | | Reserved | | |
| 34 | D-IN | 3V3 sense | CPU GPIO 3V3 | Detection if CPU powered on (HIGH if powered on) |
| 35 | D-OUT | SPI-CS0 | Display | SPI chip select (active low) |
| 36 | D-IN | I2C-INTB# – Bus #1 | PU | Optional SMBus interrupt line (active low) |
| 37 | GPIO | Digital I/O from/to WAU | WAU (GPIO58) | |
| 38 | GPIO | Digital I/O from/to WAU | WAU (GPIO63) | |
| 39 | D-OUT | LED-Ring 5 | LED-Ring | |
| 40 | | Reserved | | |
| 41 | | Reserved | | |
| PROG# | D-IN | Program | CPU GPIO#1 | Open Collector Inputs to GND<br>Requires additional CONF GPIO#0 from CPU |
| ON/OFF# | D-IN | ON/OFF | CPU GPIO#2 | Open Collector Input to GND<br>Requires additional CONF GPIO#0 from CPU |

### 4.3.2. UI Components

All external UI components are wired to the MCU, their functionality is freely programmable:

- 4x visible Pushbuttons on the front panel with LED-Rings
- 1x hidden pushbutton with LED-ring behind the front panel cover
- 160x128px graphic display

**ATTENTION:** the 6th Pushbutton ("ATX Power Button") is hard wired to the CPU's ATX-button and its LED-Ring is hard wired to the ATX Power-LED. This button is not software controllable.

#### 4.3.2.1. Pushbuttons and LED-Rings

The non-latching pushbuttons are of type "normally opened". To be able to use them, you will need to enable the microcontroller's internal pullup resistors on the corresponding GPIO pins.
Example code (Arduino IDE) to enable GPIO as Input with internal pullup:

```
pinMode(<PUSHBUTTON-GPIO>, INPUT_PULLUP);
```

Pushing a button will drive the corresponding pin LOW while the button is being pressed.
LED-Rings are high active: configure their corresponding microcontroller GPIO as output and set to HIGH to enable the LED.

#### 4.3.2.2. Graphic Display

The installed 1.8" graphic display is connected to the MCU using **SPI Bus #0**. It provides a full color display with 160x128 pixels and 2MB of flash memory which can be used to store a bootup logo for example. Display uses MCU **GPIO#35** as SPI CS# (active low). Optionally, the display can be wired to **I2C Bus #0**. This hardware change must be communicated at ordering time.

**ATTENTION:** The graphic display integrates an **OLED panel**, which causes the risk of wearing out over time. It is highly recommended to implement countermeasures in the MCU firmware to avoid this.

**ATTENTION:** The display has a software-switchable power supply which allows you to turn off the display for power saving. Before accessing the display, make sure to enable the supply voltage by setting MCU **GPIO#23** to HIGH. If the display uses the optional I2C data connection, you must set **GPIO#14** to HIGH instead.

You can find detailed information on how to operate the display here:
https://www.digole.com/images/file/Tech_Data/Digole_Serial_Display_Adapter-Manual.pdf

### 4.3.3. Sensors

The MCU board is equipped with two sensor chips:

- Bosch BNO055 9-axis absolute orientation sensor
- Bosch BME280 Air pressure, temperature and humidity sensor

Both Sensors are connected via **I2C Bus #0**, the I2C slave addresses of the sensors are 0x76h for BME280 and 0x28h for BNO055.

For details how to operate the sensors, please consult the sensors data sheets:
https://www.bosch-sensortec.com/bst/products/all_products/bme280
https://www.bosch-sensortec.com/bst/products/all_products/bno055

**ATTENTION:** Both sensors are supplied via a 3.3V software-switched power supply. Before accessing the sensors, make sure to enable their common supply voltage by setting MCU **GPIO#22** to HIGH.

### 4.3.4. ADS-B Receiver

A GNS5892R ADS-B receiver module is mounted on the MCU. The module is connected via port **UART1** directly to the microcontroller of the MCU or can be optionally wired to the CPU using an FTDI USB-to-Serial converter installed on the MCU. Note that this optional hardware wiring must be communicated at ordering time.
In both cases, use a baud rate of 921.600bps (8N1 with hardware flow control) to communicate with the receiver.

An external antenna is connected to the ADS-B receiver, which is located behind the acrylic cover on the rear panel of the P100.

**ATTENTION:** The receiver is connected via a 3.3V software-switched power supply. Before accessing the receiver, make sure to enable its supply voltage by setting MCU **GPIO#22** to HIGH.
The optional data connection to the CPU via the FTDI USB-to-Serial converter uses a dual-port chip. Connect to the first (lower) of the 2 available ports on the CPU to communicate with the ADS-B receiver.

**ATTENTION:** When using the USB based connection to the CPU to communicate with the ADS-B receiver, it is crucial to NOT initialize the UART connection on the Teensy microcontroller to avoid clashes on the data connection to the ADS-B chip. This means, that Teensy pins 0, 1, 20 and 21 must not be used in that case (must remain in the default DISABLED mode).

#### 4.3.4.1. ADS-B Receiver Command Line Reference

*Commands*
Commands always consist of at least two ASCII characters which compose 1 binary command or parameter value. Command and parameter values as well as consecutive parameter values are separated by a delimiter character (“-“). Every command starts with a hash (“#”) and is finished by Carriage Return (0x0D).

Available commands and their function:

| *Command* | *Value* | *Parameter* | *Example* | *Description* |
|---|---|---|---|---|
| Reset Receiver | FF | None | #FF<CR> | Performs HW-Reset |
| ADS-B Mode 0 | 49 | 00 | #49-00<CR> | Set mode 0 (stop data output) |
| ADS-B mode 2 | 49 | 02 | #49-02<CR> | Set mode 2 (output ALL DF – Data) |
| ADS-B mode 3 | 49 | 03 | #49-03<CR> | Set mode 3 (output DF17 / DF18 / DF19 – Data only) |
| ADS-B mode 2+ | 49 | 82 | #49-82<CR> | Set mode 2 (output ALL DF – Data) with additional Signal Strength Indicator |
| ADS-B mode 3+ | 49 | 83 | #49-83<CR> | Set mode 3 (output DF17 / DF18 / DF19 – Data only) with additional Signal Strength Indicator |

*Command Replies*
Replies consist always of 16 binary values, each value formed of two ASCII characters. Consecutive values are separated by a delimiter character (“-“). Every reply starts with a hash (“#”) and ends with LF (0x0A). First and following reply values represent the echo of the previous command. Unused values are padded with “00”.
In case of command error, the first value is “FF”.

*GNS 5892R command reply examples:*

```
#49-02<CR>
#49-02-00-00-00-00-00-00-00-00-00-00-00-00-00-00-<LF>
```

➔ Command accepted

```
#55<CR>
#FF-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-<LF>
```

➔ Unknown command

*ADS-B data reception example (mode 2)*

In case of ADS-B mode 2 (reception of all DF-Data) the 56bit or 112bit raw data frames are output as 14 bytes ASCII frames, or 7 bytes ASCII frames, MSB first.
Every ASCII frame begins with an asterisk ("*") and is finished by ';' + <LF>.

```
*8D4B1621994420C18804887668F9;
*02E1991058EF31;
*A0000CB0958000316C00004D38CA;
```

*ADS-B data reception example (mode 2+)*

In case of ADS-B mode 2+ (reception of all DF-Data with additional Signal Strength Indicator) the 56bit or 112bit raw data frames are output as 15 bytes ASCII frames, or 8 bytes ASCII frames, MSB first.
Every ASCII frame begins with a "+" and is finished by ";" + <LF>.
The first byte following the "+" is the signal strength indicator of the current ADS-B frame.

The following bytes represent the received ADS-B data:

```
+1B5D3DDDD1707EA1;
+1AA800023AC4600030A80000FB50EB;
+1A80E1969158B51233C9445A63D2C7;
+195DC03ABCB09F1C;
```

**Note:** The Signal Strength Indicator preceded by the data is only an indicator of the field strength of the received frame and not a measured value from which the distance to the transmitter can be derived. The range of the value is approximately between 0x18h (far distance signals) and 0x2dh (near distance signals).

### 4.3.5. Input Voltage measurement

It is possible to measure the internal DC supply voltage using MCU **GPIO #15**. The voltage is measured at the input of the battery backplane, so it is either ~0V, or ~18VDC when the system is running on external power.

**Note:** When measuring DC-input voltage, keep in mind that you will not measure the actual voltage applied to the DC-In connector of the P100. Due to the technical implementation, the output voltage of the initial DC step-down converter is measured, which generates 18VDC regardless of the DC voltage supplied to the P100.

Use the following call to read the measured voltage:

```
externalVoltage = (((float)analogRead(15)) / 1024.0f) * 30.0f;
```

### 4.3.6. Programming the MCU

There are several possible development environments available for programming the microcontroller installed on the MCU, the most common are:

- Arduino IDE with Teensyduino

https://www.arduino.cc/en/Main/Software

https://www.pjrc.com/teensy/teensyduino.html

- Microsoft Visual Studio Code with PlatformIO Addon

https://code.visualstudio.com/download
https://platformio.org/install/ide?install=vscode

Both solutions are available as freeware.

#### 4.3.6.1. PROGRAM and ON/OFF pins

The Microcontroller ROGRAM# pin is required to flash new microcontroller firmware using the USB-connection to the CPU.
As improper usage of these pins can lead to a total loss of the firmware on the microcontroller, a protective logic circuit has been added to prevent unintentional usage of these pins by using an additional "configuration input", which needs to be set HIGH to allow usage of ON/OFF# and PROGRAM# pins.
Please note that both pins on the microcontroller are open collector (OC) inputs, thus they need to be grounded for activation. This is however inverted by the protective logic circuit, please see the truth table below:

| *Function* | *CONF_IN* | *PROG_IN* | *ON/OFF_IN* | *µC PROGRAM#* | *µC ON/OFF#* |
|---|---|---|---|---|---|
| **CPU GPIO** | **P1** | **P2** | **P3** | | |
| No function | LOW | X | X | Z | Z |
| PROGRAM set | HIGH | HIGH | LOW | GND | Z |
| RESET set | HIGH | LOW | HIGH | Z | GND |
| PROG + RESET set | HIGH | HIGH | HIGH | GND | GND |

*Explanation:*
X.. logic state of input does not matter
Z.. output is inactive (high impedance)
GND.. output is active (grounded)

**Note:** The 3 CONF GPIOs are hardwired to the documented CPU GPIO pins. These pins are set to INPUT mode by default, a pulldown resistor on the MCU board leads to a LOW level for P1 to P3 in this state.

**ATTENTION:** Be aware that the ON/OFF# pin for Teensy 4.1 allows you to set the µC into a low-power OFF state, in contrary to the previous µC (Teensy 3.6), where it has been acting as a reset pin. Thus, this pin is also not required any longer for programming the Teensy 4.1 (see next chapter).

Quote from PJRC concerning the ON/OFF# pin:
*A special low power state which turns off the 3.3V power can be controlled by the On/Off# pin. A pushbutton is meant to be connected between On/Off and GND. While running, holding the button for 4 seconds turns off power. Pressing for 0.5 seconds while power is off turns the 3.3V power back on and reboots the processor.*

#### 4.3.6.2. Programming sequence

**Note:** Using PROGRAM# to upgrade the microcontroller firmware might not be necessary, however usage is recommended for predictive results.

Follow this procedure to upgrade the microcontroller firmware:

- Configure CPU **GPIO P1 and P2** as outputs and set them to LOW
- Set **P1 and P2** to HIGH to switch Teensy into programming mode
- Delay for 2 seconds
- Set **P2** to LOW
- Delay for 1 second

- Use teensy_loader_cli to update the firmware:
  - `teensy_loader_cli -mmcu=TEENSY41 <Firmware.hex>`
- Delay for 2 seconds and set **P1** to LOW

The source for the teensy_loader_cli program can be found here:
https://github.com/PaulStoffregen/teensy_loader_cli

### 4.3.7. Solid State Relays

The MCU is equipped with a galvanically isolated solid-state relay (SSR). The installed IC provides two "normally opened", potential free electronic switches that can be closed through the MCU **GPIO#6 and #7**.

#### 4.3.7.1. ATX Power Button and Super-S5-ECO-Mode

**GPIO#6** is used to control the CPUs power button, which can also be operated with the physical button available underneath the front panel cover. The LED-Ring of this pushbutton will also represent the current state of the CPU. If it is lit, the CPU is powered on.
As an alternative to the physical pushbutton, the SSR which can be controlled by the microcontroller through **GPIO#6** is wired to the physical switch in parallel. This allows you to toggle the power state of the CPU through the MCU.

Note that the power-on functionality of the ATX power switch depends if your P100 has the **"Super-S5-ECO-Mode"** enabled or not. This mode allows (nearly) zero power consumption of the CPU while powered down. This is especially recommended for environments, where the P100 is running on battery power.
The desired operational state for Super-S5-ECO-Mode must be defined when ordering the P100 (enabled by default).

- *Super-S5-ECO-Mode enabled*

Press the ATX power button for ~ 5-8 seconds to boot the CPU.
Note that no other wake-up sources are possible in this mode. The CPU needs to be powered on using the power button, automatic boot after power-loss (for example) is not supported.

**Note:** When using the MCU to power up the CPU with enabled Super-S5-ECO-Mode we strongly recommend the following approach:

- "Press" the CPU ATX-Power-Button by setting MCU **GPIO#6** to HIGH
- Monitor MCU CPU power-on detection pin **GPIO#34** until it is HIGH
- then immediately "release" CPU ATX-Power-Button by setting MCU **GPIO#6** to LOW

- *Super-S5-ECO-Mode disabled*

In that case, the CPU is working like a normal PC in S5 state. Use a short (<500msec) press on the power button to boot the CPU. Note that this results in higher power drain (5VSB line) when the unit is powered down.
In this mode, the CPU can also be automatically woken up, i.e. by configuring automatic boot after power-loss in BIOS.

Any other functions of the ATX power button are as on any other ATX / ACPI compliant hardware. Depending on the OS (ACPI aware), you may initiate a shutdown (or any other configurable action) with a short (<500msec) press on the power button. When the CPU is powered on, pressing the button for longer than 5 seconds will force-off the system (without shutting down the OS).
Any of these actions can be performed through either the physical switch or MCU **GPIO#6**.

#### 4.3.7.2. LTE Modem Power Control

Setting MCU **GPIO#7** to HIGH level allows you to shut down the LTE modem which is installed on the CPU. This is done by pulling the `Full_Card_Power_Off#` control signal for the modem to GND level (OC input).

### 4.3.8. CPU power-on detection

In order to verify if the CPU is powered on, a microcontroller GPIO is wired to the 3.3V line of the CPU. To check the state of the CPU, configure the **GPIO#34** as INPUT and read its state. While the state is LOW, the CPU is powered off, when in HIGH state, it is powered on.

## 4.4. Power Unit (PU)

The P100 is equipped with a SmartBattery compliant power subsystem. It can handle 2 Li-Ion batteries, a controller chip automatically handles power paths of DC sources and load.

Technical data for the power unit:

| Parameter | Minimum | Typical | Maximum |
|---|---|---|---|
| DC Input Voltage [V] (*) | 15 | 18 | 53 |
| DC Input current [A] | | | 4.5 |
| DC output current [A] | | | 9 |
| Single Battery output current [A] | | | 4.5 |
| DC Output Voltage [V] | 11 | 15 | 18 (**) |
| Battery charging current [A] | 0 | 1 | 2 |

(*) Includes wide DC-In input voltage range, see chapter DC Power Socket.
(**) maximum PU output voltage limited to 18V by initial DC-In Step-Down converter.

The installed Analog Devices LTC1760 controller chip will select the power source automatically, batteries will be charged if external DC power is available and drained simultaneously while on battery power. The controller provides read and write access via a SMBus compliant interface.
The controller is wired to the **I2C Bus #1** on the MCU board and can be accessed using address 0x0a for the controller and 0x0b for the batteries.

**ATTENTION:** Both batteries share the same I2C address! You will need to select the battery which you want to access by writing the according bits to the SmartBattery controllers status register. See data sheets for details:
https://www.analog.com/media/en/technical-documentation/data-sheets/1760fc.pdf
http://sbs-forum.org/specs/sbdat110.pdf

## 4.5. WiFi Access Point Unit (WAU)

The WiFi access point unit supports DBDC (Dual Band Dual Concurrency) 802.11a/b/g/n/ac in MU-MIMO 2x2 and provides 2 x 1000 Base-T interfaces. The first gigabit interface (eth0) is wired internally to the CPU and the second interface (eth1) is wired to the external RJ-45 jack on the P100 back panel.
The WAU runs OpenWRT 18.04 OS or later, all wired and wireless interfaces are bridged on one software bridge in the default configuration. Wireless interfaces are disabled by default, there is **no password** set on the OpenWRT system.
The WAU can be configured by connecting to the OpenWRT SSH shell using the default IP address 192.168.1.1.
Two WiFi antennas are connected to the WAU, they are located beneath the bottom antenna cover.

**ATTENTION:** the WAU is powered by the MCU. This allows you to power off the WAU in times where no WiFi communication is required or allowed. To enable power for the WAU, you will need to set MCU **GPIO#5** to HIGH.

### 4.5.1. GPIO Connections to MCU

Several GPIO connections from the WAU are wired to the MCU board as shown in the table:

| *WAU GPIO* | *MCU GPIO* |
|---|---|
| 0 | 9 |
| 1 | 10 |
| 3 | 2 |
| 4 | 3 |
| 5 | 4 |
| 58 | 37 |
| 63 | 38 |

GPIO pins in OpenWRT need to be set up using sysfs, see: https://www.kernel.org/doc/Documentation/gpio/sysfs.txt

Example Setup for WAU **GPIO#0** on the WAU console:

```
#Initialize GPIO#0
echo 0 > /sys/class/gpio/export
# Define as OUTPUT
echo out > /sys/class/gpio/gpio0/direction
# Set to HIGH
echo 1 > /sys/class/gpio/gpio0/value
```

**ATTENTION:** The GPIOs are hard wired according to the table above. Avoid setting both GPIO (i.e. WAU **GPIO#0** and MCU **GPIO#9**) to OUTPUT state. Default (unconfigured) state of all GPIOs on both WAU and MCU is INPUT.

### 4.5.2. Accessing the WAU Serial Console

A **serial console** is connected to the CPU through a USB connection. The corresponding device node can be identified i.e. by examining the kernel log under Linux OS:

```
[root@localhost ~]# dmesg | grep FTDI
[    5.888242] usb 1-3.1: Manufacturer: FTDI
[   12.029507] usbserial: USB Serial support registered for FTDI USB Serial Device
[   12.038373] ftdi_sio 1-3.1:1.0: FTDI USB Serial Device converter detected
[   12.070003] usb 1-3.1: FTDI USB Serial Device converter now attached to ttyUSB1
[root@localhost ~]#
```

The resulting character device as shown in the example is `/dev/ttyUSB1`. You will be able to connect to the WAU console with a terminal program of your choice using 115200 baud with 8N1 and hardware flow control disabled.

### 4.5.3. VLAN Configuration

The WAU supports VLAN interfaces (ingress / egress filtering), which needs to be configured as described in this chapter.

Be aware of the following hard- and software limitations concerning VLAN usage on the WAU:

- VLAN IDs from 0 to 127 are supported
- Traffic to WAU processor (Port 0) must always be tagged
- It is recommended not to use VLAN ID 1 and 2, as these might be used internally by OpenWRT

First, the internal WAU ethernet switch needs to be set to a VLAN aware configuration.
To do so, create the file `/etc/sysctl.d/99-edma_vlan.conf` with the following content:

```
net.edma.default_group1_vlan_tag = 0
net.edma.default_group2_vlan_tag = 0
net.edma.default_lan_tag = 0
net.edma.default_wan_tag = 0
```

Now add a VLAN configuration in the `/etc/config/network` file. The sample shown below sets up a sample configuration for:

- eth0 interface will be configured with untagged VLAN 5 (friendly name “INT”) and tagged VLAN 60 (friendly name “MGMT”)
- eth1 interface will be tagged with MGMT VLAN and untagged VLAN 70 (friendly name “EXT”), an DHCP address will be requested on eth1 VLAN EXT
- An internal software bridge (friendly name “VL5BR”) with static IP-address 10.1.1.1/24 will be added to the INT VLAN and eth0 (a WiFi will be bound to this in the next step)
- MGMT VLAN will be bridged using an IP-less bridge across eth0 and eth1 (friendly name “VL60BR”)

```
#### Network Device Configuration ####
## eth0 VLAN configuration
# Virtual device for VLAN 5
config device
        option name 'eth0-vl5'
        option type '8021q'
        option ifname 'eth0'
        option vid '5'

# Virtual device for VLAN 60
config device
        option name 'eth0-vl60'
        option type '8021q'
        option ifname 'eth0'
        option vid '60'

## eth1 VLAN configuration
# Virtual device for VLAN 60
config device
      option name 'eth1-vl60'
      option type '8021q'
      option ifname 'eth1'
      option vid '60'

# Virtual device for VLAN 70
config device
      option name 'eth1-vl70'
      option type '8021q'
      option ifname 'eth1'
      option vid '70'

#### Interface configuration ####
## Virtual Interface for eth1 / EXT VLAN
config interface
      option ifname 'eth1-vl70'
      option proto 'dhcp'
```

```
#### Software Bridge configuration ####
## Bridge for VLAN INT
config interface 'VL5BR'
      option type 'bridge'
      option ifname 'eth0-vl5'
      option proto 'static'
      option ipaddr '10.1.1.1'
      option netmask '255.255.255.0'

## Bridge for VLAN MGMT
config interface 'VL60BR'
      option type 'bridge'
      option ifname 'eth0-vl60 eth1-vl60'
      option proto 'none'


#### Internal SWITCH Configuration ####
## Reset config and enable VLAN support
config switch 'switch0'
      option reset '1'
      option enable_vlan '1'

## Configure VLAN 5 (INT); Tagged on processor, untagged on eth0
config switch_vlan 'INT'
      option device 'switch0'
      option vlan '5'
      option vid '5'
      option ports '0t 4'

## Configure VLAN 60 (MGMT); Tagged on processor, eth0 and eth1
config switch_vlan 'MGMT'
      option device 'switch0'
      option vlan '60'
      option vid '60'
      option ports '0t 4t 5t'

## Configure VLAN 70 (EXT); Tagged on processor and untagged on eth1
config switch_vlan 'EXT'
      option device 'switch0'
      option vlan '70'
      option vid '70'
      option ports '0t 5'
```

This sets up the desired VLAN configuration internally and for the wired interfaces.

We now add a 2.4GHz wireless interface (radio0) to our default VLAN (ESSID “Internal”), and an additional 5GHz WiFi (radio1) to our MGMT VLAN (ESSID “Management”).
This needs to be set up via the `/etc/config/wireless` file:

```
config wifi-iface 'pub_radio0'
      option device 'radio0'
      option network 'VL5BR'
      option mode 'ap'
      option ssid 'Internal'
      option encryption 'psk2'
      option key 'SecurePassword'

config wifi-iface 'mgmt_radio1'
      option device 'radio1'
      option network 'VL60BR'
      option mode 'ap'
      option ssid 'Management'
      option encryption 'psk2'
      option key 'SecurePassword2'
```

**NOTE:** Physical interface settings for radio0 and radio1 are not described, you may use the default settings provided in the `wireless` config file.

We recommend the following OpenWRT guides as additional documentation source for configuring networking on the WAU:
https://openwrt.org/docs/guide-user/network/wifi/basic
https://openwrt.org/docs/guide-user/base-system/basic-networking
https://openwrt.org/docs/guide-user/network/vlan/switch_configuration
https://openwrt.org/docs/guide-user/network/vlan/switch

## 4.6. x86_64 Unit (CPU)

The P100 is equipped with an Intel Denverton 64bit x86 based computer.

The CPU provides:

- x86_64 architecture CPU with 2-16 cores
- Up to 32GB DDR4
- Up to 3x PCIe based internal NVMe disks
- Optional 1x removable PCIe based NVMe disk
- 1x LTE modem w/ Mini-SIM-slot and config pins
- 1x WiFi / Bluetooth module
- 8x GPIO
- 1x RS-232
- 2x USB2.0 and up to 2x USB3.0
- 1x Gigabit Ethernet
- System and processor Fan connectors

**Note:** Low-End systems based on Denverton C3308 only provide one USB3.0 port, which is connected to the Front USB port.
Additional limitations in the number of available PCIe lanes for NVMe disks need to be considered for low-end systems (down to a single PCIe 3.0 lane for NVMe disks on C3308).

### 4.6.1. Serial Console Port

As the P100 is a headless system, the provided DE-9 RS-232 serial console port is of special importance, as this is the only possibility to gain console access to the CPU.
To access the serial console port, remove the 4 screws in the corners of the acrylic cover on the back panel of the P100, then gently pull the cover outwards (take care of antennas and cables attached to the acrylic cover!). When re-installing the acrylic cover, do not harm the acrylic cover by applying too much force to the screws.
To gain access to the console you will probably need a null modem adapter. Use 115200 baud with 8N1 and a terminal program of your choice to access the CPU.

**Note:** the serial console port is wired to COM1 (ttyS0 in Linux) of the CPU. BIOS serial console redirection is by default enabled on this port.

### 4.6.2. LTE Modem

A Sierra Wireless EM7565 LTE modem is installed on the CPU. It can operate worldwide and provides one Mini-SIM slot on the back panel of the P100.
To install a SIM card for the LTE modem, open the SIM-card slot by unscrewing the Philips screw on the top. Gently pull the hatch outwards to access the SIM card holder.
**Note:** SIM cards are not hot-swappable. Install the SIM card when the CPU is powered off and make sure that the SIM card holder is fully locked before closing the hatch.

The factory default configuration for the modem is MBIM operation mode, which works out of the box with the ModemManager package of many up to date Linux distributions.

In depth documentation for the modem can be obtained through the manufacturer's website (registration required):
https://source.sierrawireless.com/devices/em-series/em7565/

**ATTENTION:** When using `Full_Card_Power_Off#` configuration signal to power down the modem, keep in mind that the modem will do a proper shutdown after drawing the signal to GND level (i.e. by setting MCU **GPIO#7** to HIGH). After a shutdown phase of ~20 seconds, the modem will no longer show up in the CPU's OS. After enabling the modem it will also take ~30 seconds for the modem to boot.
It is vital not to change the state of the `Full_Card_Power_Off#` signal during modem boot or shutdown to avoid damage to the modem, see modem manual for detailed power-on and power-off sequences.

### 4.6.3. WiFi and Bluetooth module

In addition to the WAU, the CPU also has an Intel 9260 WiFi module installed. This adapter supports Bluetooth 5 and 2x2 802.11ac Wi-Fi including wave 2 features such as 160MHz channels.
Two combined WiFi and Bluetooth antennas are installed beneath the top cover.

### 4.6.4. CPU GPIOs

The CPU is equipped with 8 GPIO pins which can be configured individually as in- or outputs.
By default, all GPIO pins are configured as INPUTS. Configured pin states remain until the CPU is rebooted or reset.

#### 4.6.4.1. Sample program usage

**Note:** The tool described in this section can be obtained from your distributor.

`S283AGPIO -v`
Print overview of all GPIO pins current state.

`S283AGPIO -m`
Print overview of all GPIO pins in machine readable CSV format.

`S283AGPIO -i <Pin>`
Configure GPIO pin to input mode and print the current status.

`S283AGPIO -opp <Pin> <Data>`
Configure GPIO pin to push-pull output mode and set to high/low.

`S283AGPIO -ood <Pin> <Data>`
Configure GPIO pin to open-drain output mode and set to high/low.

**ATTENTION:** The GPIO pins support two different output modes: open-drain and push-pull. Latter is required to communicate with the MCU on P1-P3. Setting open-drain output mode to P1 to P3 will cause unpredictable behavior on the MCU Teensy microcontroller.

`S283AGPIO -r <Pin>`
Print single GPIO pin state, independent of configured in- or output state
`<Pin>` refers to the pin of the CPUs GPIO connector (valid values: p1 to p8, case insensitive):

| Connector Pin Number | GPIO Pin Function | Function |
|---|---|---|
| 1 | P1 | MCU CONF_IN |
| 2 | P2 | MCU PROG_IN |
| 3 | P3 | MCU ON/OFF_IN |
| 4 | P4 | Unused |
| 5 | P5 | Unused |
| 6 | P6 | Unused |
| 7 | P7 | Unused |
| 8 | P8 | Unused |
| 9 | GND | |
| 10 | 3V3 | MCU à CPU ON detection |

`<Data>` is either H for logical high level (3.3V) or L for logical low level (GND), value is case insensitive.

The program supports return codes of either 0 (success) or 1 (failure).

**Note:** P1 to P3 of the CPU GPIOs are hardwired to the MCU for microcontroller programming, please see previous chapter for details.

#### 4.6.4.2. Usage Examples

Read current state of P3:

```
[root@sgphost ~]# ./S283AGPIO -r p3
LOW
[root@sgphost ~]#
```

Set P5 to push-pull output mode in LOW state:

```
[root@sgphost ~]# ./S283AGPIO -opp p5 L
[root@sgphost ~]#
```

Show overview of all GPIO pins:

```
[root@sgphost ~]# ./S283AGPIO -v
MS-S283A GPIO Pin header

================================================
 GPIO     Mode          PP/OD     Current status
================================================
  P1      Input          N/A            LOW
  P2      Input          N/A            LOW
  P3      Input          N/A            LOW
  P4      Input          N/A            HIGH
  P5      Output     Push-Pull          LOW
  P6      Input          N/A            HIGH
  P7      Input          N/A            HIGH
  P8      Input          N/A            HIGH

 [root@sgphost ~]#
```

#### 4.6.4.3. Hardware Information – GPIO and fan control

The GPIO pins are provided by the Nuvoton NCT6106D Super-IO chip. Additional information on how to access this chip can be found in the data sheet:
https://www.nuvoton.com/resource-files/NCT6102D_NCT6106D_Datasheet_V1_0.pdf

The chips I/O pins are wired to the GPIO connector as described below:

| GPIO Pin Function | NCT6106D Pin |
|---|---|
| P1 | 109 (GPIO00) |
| P2 | 117 (GPIO10) |
| P3 | 124 (GPIO17) |
| P4 | 123 (GPIO16) |
| P5 | 1 (GPIO24) |
| P6 | 121 (GPIO14) |
| P7 | 115 (GPIO06) |
| P8 | 116 (GPIO07) |
| GND | |
| 3V3_S0 | Wired to MCU |

Each GPIO is wired to 3V3 voltage through a 10kOhms pullup resistor. The Super-IO chip's SIO I/O address configured to 4Eh/4Fh.
The Super-IO chip is also responsible for handling fan control for both CPU and system (chassis) fan. Note that both fans have temperature regulated speed control enabled by default and it is not recommended to change the fan control behavior.
Depending on the running OS, it might be possible to manually control the fan settings through sysfs:

Fan control files:
`/sys/class/hwmon/hwmon3/pwm2` → CPU fan
`/sys/class/hwmon/hwmon3/pwm1` → Chassis fan

In example, to run a fan at a user defined speed, you first need to enable manual control for the chassis fan:
```
echo "1" > /sys/class/hwmon/hwmon3/pwm1_enable
```

Afterwards, set the desired fan speed (0-255 where 255 is max speed):
```
echo "255" > /sys/class/hwmon/hwmon3/pwm1
```

To reset the fan to automatic temperature-based speed control:
```
echo "5" > /sys/class/hwmon/hwmon3/pwm1_enable
```

Note that the described configuration is not persistent and needs to be reconfigured at every boot (if desired). System will always come up with temperature based fan control for both fans.

The OS is required to provide an appropriate driver for the NCT6106D Super-IO chip, which is usually handled by the nct6775 kernel module (information provided as-is without support).

**ATTENTION:** Misconfiguration of the Super-IO chip can lead to hardware malfunctions that might not be covered by warranty!

### 4.6.5. CPU BIOS Settings

This chapter describes some BIOS settings which might be important for operating the P100. To enter the BIOS configuration utility, press the DEL key during POST. By default, no BIOS password is set on the P100.

#### 4.6.5.1. CSM – Legacy BIOS Boot

By default, the P100 is set up for EFI boot only, which is the recommended mode of operation. To support legacy boot, the CSM (compatibility service module) needs to be enabled in the BIOS configuration menu:

- Enter BIOS Setup Utility
  - ***Advanced*** Tab
    - CSM Configuration
      - CSM Support à [Enabled]
  - ***Save & Exit*** Tab
    - Save Changes and Reset

**Note:** Re-enter BIOS Setup Utility afterwards, change the Boot order as required and save the changes.

#### 4.6.5.2. Power State after power loss

The CPU supports to automatically power on after a power loss. If enabled (disabled by default), the CPU will startup automatically under the following conditions:

- Main Power Switch is being switched ON
- Power is restored after a full power source loss (i.e. both batteries drained, no DC-Input)

To enable the feature, follow this procedure:

- Enter BIOS Setup Utility
  - ***IntelRC Setup*** Tab
    - State after G3 à [Power On]
  - ***Save & Exit*** Tab
    - Save Changes and Reset

**ATTENTION:** This feature requires that Super-S5-ECO-Mode is disabled!

#### 4.6.6. TPM 2.0

The CPU is equipped with a TCG version 2.0 compliant trusted platform module (TPM) which is enabled by default.
You might find the following links useful for implementing the TPM in Linux:
https://github.com/tpm2-software/tpm2-tools/wiki/How-to-use-tpm2-tools
https://github.com/tpm2-software

# 5. Software Description

Depending on your order (Appliance or hardware-only), a plain and unconfigured setup of the OS might be preinstalled on the P100. The OS can be accessed using the serial console port to log on to the system with user **root** and password **ready2go**.

**Note:** In the factory default state, the external NVMe disk installed in the removable storage slot is not used (unformatted and not mounted). No IP settings are configured on the CPUs gigabit interface.

# 6. Hardware-only Guidelines

This chapter contains additionally hardware related information in case you do not use the SGOS software for your P100.

## 6.1. Installing Operating Systems other than SGOS

If you are planning to install other operating systems than SGOS on the P100, please consider these hints:

- This is a headless system without any graphic adapter available
- BIOS is only accessible externally via serial console redirection
- Depending on the OS you are trying to install, you might be able to use an USB graphic adapter along with a suitable USB-Hub to get a video signal during the installation. Note that an USB-Graphic adapter will not work until the OS has booted – you will still need the serial console to boot your OS installer

## 6.2. Hardware-only MCU Firmware

If you are working on a P100 that has been shipped without OS (hardware-only), your MCU will be pre-programmed with a Hardware-only firmware. The functionality of this firmware will be described in this chapter.

**Note:** This firmware is provided as-is without any software support. The firmware is only stored in the MCU microcontroller flash of the P100, neither source code nor a binary image is available in addition to this.

### 6.2.1. User Interface and Functionality

The display of the Hardware-only MCU firmware will remain blank after powering on the P100, also the LED-rings will remain off.
The hardware-only MCU firmware will set both LTE-modem and WAU to ON state after startup.

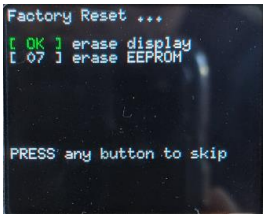
#### 6.2.1.1. Factory Reset Function

The Hardware-only MCU firmware is equipped with a factory reset function. This function will allow you to clear any static memory of the MCU as well as the OLED-display, except the firmware code itself.
The factory reset function will also allow you to **set the installed batteries into shipping mode**, which is required in the following cases:

- You are preparing the P100 for delivery.
- You are planning to store the batteries (either installed in the P100 or separately).

Enabling shipping mode will charge (or discharge) the installed batteries to a SOC of 30%. After this, the batteries will be set into shipping mode through special commands that will fully disable the power output of the batteries. You can also place the batteries into shipping mode only, without resetting the MCUs static memories, see next chapter.

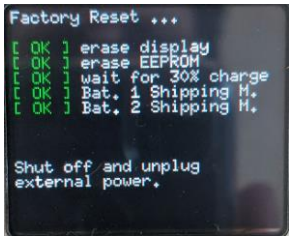
To initiate the factory reset function, press Pushbutton 5 **for 15 seconds** and follow the instructions on the OLED screen.




**Note:** if you do not skip the “erase display” and “erase EEPROM” steps by pressing any pushbutton (1-5), the MCU will automatically re-flash the display on the next boot to restore the default background image (blank screen). This does not have any impact the functionality of the MCU firmware.

**ATTENTION:** After clearing the static memories, the factory reset function will display the charge state of both batteries and will wait until both batteries have reached 30% SOC. If the SOC is above this threshold, you need to unplug the external power to drain the batteries, if they are below 30%, plug in an external supply to charge them.


Alternatively, you can skip the charge/discharge process by pressing any pushbutton (1-5). Pay attention that this will immediately set all installed batteries into shipping mode, forcefully powering off the whole P100 if no external power supply is connected.




The factory reset process has finished now. External power is the only power source remaining at this stage.

**ATTENTION:** Batteries that have been set to shipping mode will automatically leave this mode as soon as DC input power is re-applied to the P100, or if the batteries are placed into a charging station.
If you intend to keep batteries in shipping mode, do not re-apply external power to the P100 after finishing the factory reset process.

#### 6.2.1.2. Placing batteries into shipping mode only

In alternative to the factory reset function, it is also possible to only place the installed batteries into shipping mode. This can be done at 2 different SOC stages:

- Press and hold Pushbutton 5 then in addition also press and hold Pushbutton 3 for a total of 10 seconds together.

➔ Shipping Mode with **30% charge**

- Press and hold Pushbutton 5 then in addition also press and hold Pushbutton 4 for a total of 10 seconds together.

➔ Shipping Mode with **60% charge**

## 6.3. Additional Information

### 6.3.1. Placing batteries into shipping mode using I2C communication

See FAQ below for a description on how to implement shipping mode into your custom built MCU firmware using the hardware I2C/SMBus interface:

https://www.rrc-ps.com/faq

*"How can I set a battery into shipping mode?"* will provide the required commands and timings.

## 7. Documentation History

| Date | Version | Author | Chapters affected | Remarks |
|---|---|---|---|---|
| 10/02/2021 | 1.0 | jpichlbauer | All | - Initial Release for HW Rev. 3.0 |
| 09/03/2021 | 1.1 | jpichlbauer | 1.1.<br>2.3.<br>4.1.3.<br>4.3.7.1.<br>6.1. | - More precise specification of allowed operating conditions and electrical handling.<br>- Added missing information in hardware changes list<br>- Added information about operating mode w/o Li-Ion batteries<br>- Corrected required power-on button press time when Super-S5-ECO-Mode is enabled<br>- Added description for hardware-only MCU firmware |
| 16/06/2021 | 1.2 | jpichlbauer | 4.2.1.<br>4.3.1. | - Added safety information for 3rd party power supplies; added inrush current limiting feature<br>- Added missing pins 40+41 in Teensy I/O mapping; removed connector names from GPIO list |
| 24/08/2021 | 1.3 | jpichlbauer | 6.1. | - Updated hardware-only MCU Firmware chapter |
| 19/01/2022 | 1.4 | jpichlbauer | 4.1.<br>4.1.3.<br>4.3.1.<br>4.3.6.1.<br>4.3.7.1.<br>4.5.<br>6.1.1.2. | - Corrected CPU Power-up description<br>- Added reference to SB shipping mode<br>- Reworked and updated MCU Teensy Pinout<br>- Corrected functionality of Teensy ON/OFF# pin (prev. RESET#)<br>- Detailed information on how to power-up CPU with Super-S5-ECO-Mode<br>- Corrected Typo in WAU FW Version<br>- Added chapter for placing batteries into shipping mode |
| 08/06/2022 | 1.5 | jpichlbauer | 4.1.<br>4.1.3.<br>6.* | - Specified requirement of correct battery cover sheet to ensure airflow through U.2 cage.<br>- Updated information on requirement of shipping mode when storing/shipping batteries<br>- Restructured HW-only chapter, added information on how to programmatically place batteries into shipping mode |
| 15/06/2023 | 1.6 | jpichlbauer | 4.1.3.<br>4.3.2.2.<br>4.6.2.<br>4.6.4.3. | - Added more specific information about CPU instability when running on drained batteries<br>- Added information about OLED display burn-in<br>- Added information on modem power on/off restrictions<br>- Added information on how to manually control fan speeds |
| 09/01/2024 | 1.7 | P. Smeeth | All | Updated format and logos |