AMBER PI REFERENCE MANUAL

2609017281001

VERSION 2.1

SEPTEMBER 20, 2019

Revision history

Manual version	Driver version	HW version	Notes	Date
1.5	1.0.0 - 1.2.0	2.1	• Initial version	January 2018
1.6	1.6.3, 2.0.0	2.1	• New corporate design and structure • Description of the driver version 2.0.0	November 2018
1.7	1.6.3, 2.0.0	3.0	• Description of the new hardware version 3.0 • Added chapter `Hardware history`	February 2019
1.8	1.6.3, 2.0.0	3.0	• Added chapter `Regulatory compliance information`	February 2019
1.9	1.6.3, 2.0.0	3.0	• Package content image updated.	March 2019
2.0	3.0.0	3.0	• Renamed product names in driver • Moved other drivers to Wireless Connectivity SDK • Moved description of other drivers on Wireless Connectivity SDK manual	April 2019
2.1	3.1.0	3.0	• Added the requirements for operation with Raspberry Pi 4 • Updated wiringPi install instructions	September 2019

⋆ For hardware history see chapter `Hardware history`

Abbreviations and abstract

Abbreviation	Name	Description
ACK	Acknowledgement	Acknowledgement pattern confirming the reception of the transmitted data packet.
CS	Checksum	
DC	Duty cycle	Transmission time in relation of one hour. 1% means, channel is occupied for 36 seconds per hour.
FSE	Field Sales Engineer	Support and sales contact person responsible for limited sales area
0xhh [HEX]	Hexadecimal	All numbers beginning with 0x are stated as hexadecimal numbers. All other numbers are decimal.
HIGH	High signal level	
LOW	Low signal level	
LSB	Least significant bit	
MSB	Most significant bit	
PL	Payload	The real, non-redundant information in a frame/packet.
RF	Radio frequency	Describes everything relating to the wireless transmission.
UART	Universal Asynchronous Receiver Transmitter	Universal Asynchronous Receiver Transmitter allows communicating with the module of a specific interface.
US	UserSettings	Any relation to a specific entry in the UserSettings is marked in a special font and can be found in the respective chapter.
VCC	Supply voltage	

Contents

1 Summary & introduction

The AMBER PI is an expansion board for the Raspberry Pi that equips the Raspberry Pi with the sub GHz RF interface provided by Würth Elektronik eiSos. The included sensor boards as well as the delivered RF-stick, that run as remote station for radio transmissions, allow to develop various creative applications on top of the Raspberry Pi.

Besides the hardware of the AMBER PI, Würth Elektronik eiSos provides a driver in C-code that includes the functions for the integrated RF-module as well as the connected sensors to enable an easy and quick custom application development.

2 Package content

Würth Elektronik eiSos delivers a box that contains the AMBER PI, the sensor boards and an empty SPI prototype board. The included radio-stick can be used as remote station to communicate with the AMBER PI via radio.

To complete this development kit, the user has to add its Raspberry Pi and the corresponding power supply to the placeholders that are provided in the box.

With this and the corresponding AMBER PI driver, that can be downloaded from the AMBER PI web page [1], the kit is ready to bring numerous remarkable custom ideas to life.

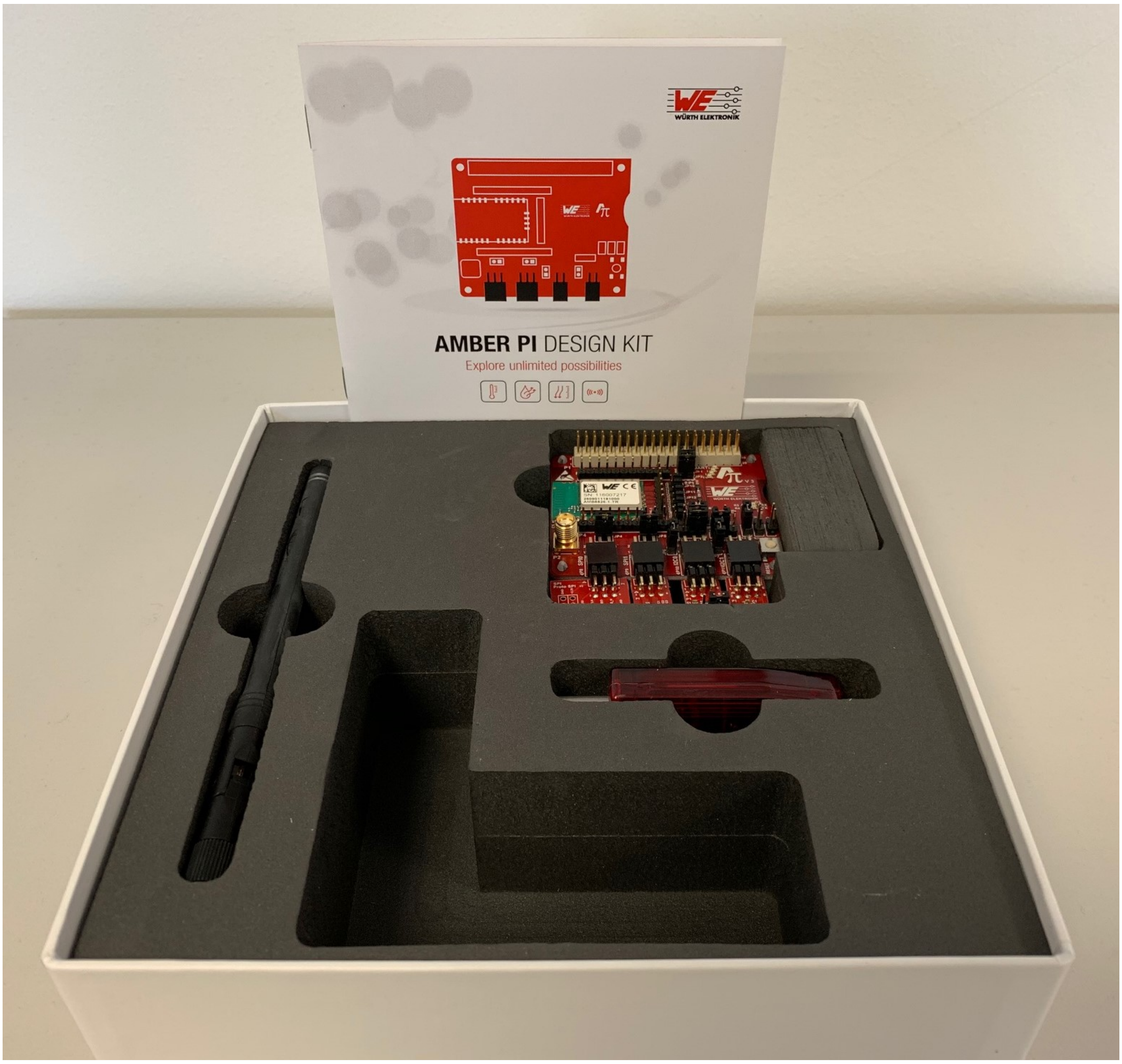


Figure 1: Package content

3 Raspberry Pi compatibility

The AMBER PI uses a 40 pin connector to be mounted on top of the Raspberry Pi. The AMBER PI as well as this manual have been created using the Raspberry Pi 3B as base providing sufficient computing power for a convenient application development. But nevertheless, also other types of Raspberry Pi may be compatible.

Compatible Raspberry Pi versions:

- Raspberry Pi 4B (1, 2 and 4GB)
  - Requires: Raspbian Buster or newer
  - Requires: Wiring Pi version 2.52 or newer (manual steps are required till this version is in the official repositorys. See instructions at: *wiringpi.com*).
- **Raspberry Pi 3B (recommended)**
- Raspberry Pi Zero W
  - When using Raspbian with GUI most of the computing power is used for the GUI
- Others
  - To be tested

4 Quickstart guide: Taking into operation

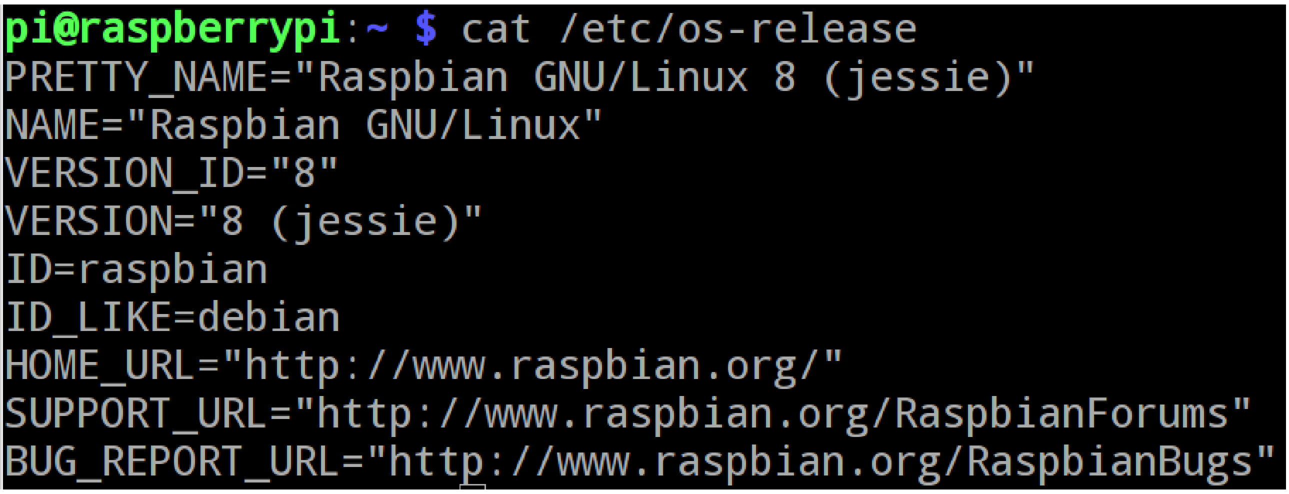
To take the AMBER PI into operation the AMBER PI driver has to be installed. To do so the following chapter contains the complete description how to setup the Raspberry Pi as well as the AMBER PI driver. This description has been created using a Raspberry Pi 3B.

Please perform the following steps:

4.1 Install the Raspbian OS on the Raspberry Pi

1. First of all the Raspberry Pi has to be installed and configured.

   a) Download the latest Raspbian with GUI from *https://www.raspberrypi.org/downloads/raspbian/*



```
pi@raspberrypi:~ $ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 8 (jessie)"
NAME="Raspbian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
```

Figure 2: Tested with Raspbian Jessie with PIXEL and Raspbian Buster

   b) Install the Raspbian OS by writing its image on your SD-card. In Windows the Win32DiskImager tool can be used, as described here *www.raspberrypi.org/documentation/installation/installing-images/windows.md*

2. After installing the image on the SD card, insert it into the Raspberry Pi's SD card slot, connect your monitor, mouse and keyboard. Now the Raspberry Pi is ready to boot up. Please start it by powering it up.

3. After booting the Raspberry, switch off the Bluetooth interface by clicking on the Bluetooth button on the right upper corner of the screen (see figure 3).

4. Then turn on the WiFi for connecting to the internet by clicking on the WiFi button on the right upper corner of the screen and selecting the WiFi of your choice.

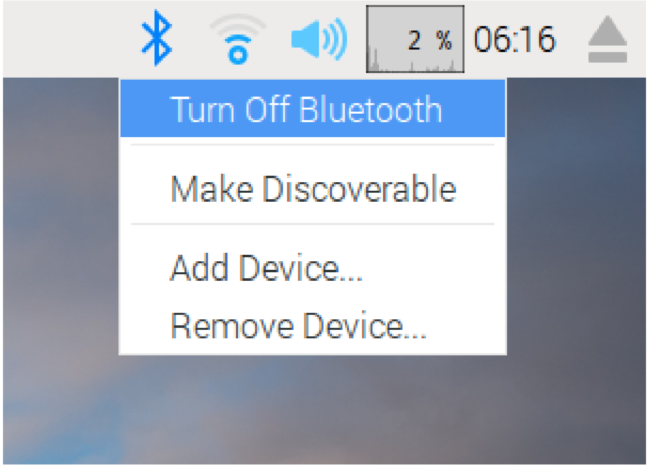


Figure 3: Switch off the Bluetooth and connect to internet via WiFi

5. After connecting to the internet make sure your Raspberry Pi is up to date with the latest versions of Raspbian OS. To update the system open a terminal by clicking on the terminal symbol in the left upper corner (see figure 4).

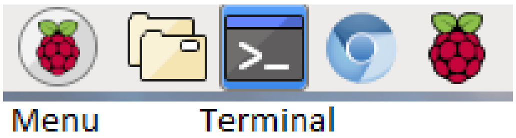


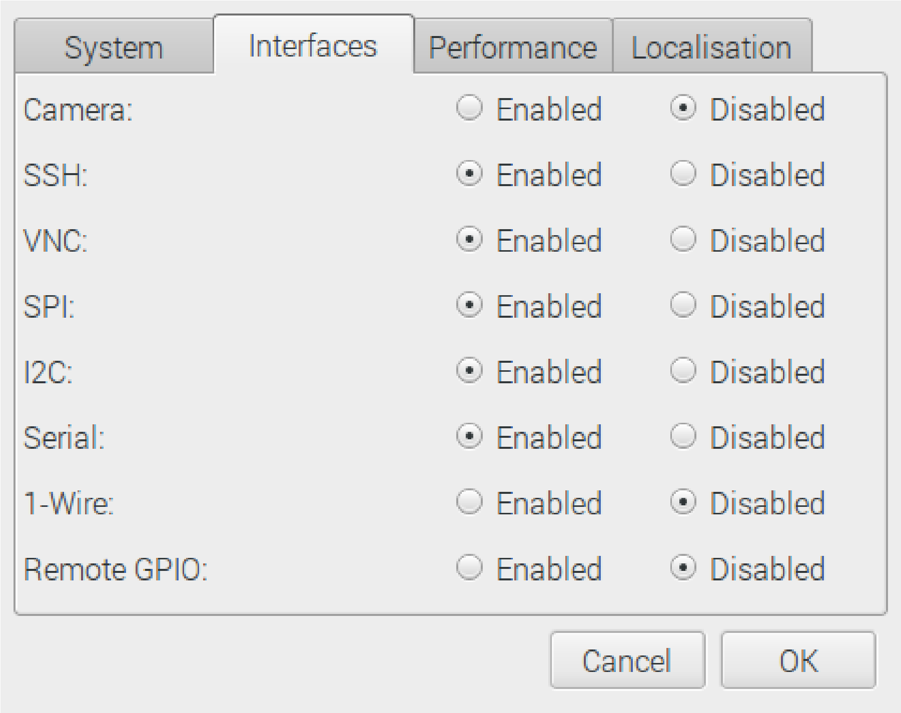
Figure 4: Terminal button

6. Then upgrade the Raspbian OS by typing in terminal:

```
sudo apt-get update
sudo apt-get upgrade
```

4.1.1 Configuring the peripherals

1. Next, the peripherals have to be enabled. To do so open the menu by clicking on the Raspberry symbol on the left upper corner of the screen and open the **Preferences** → **Raspberry Pi Configuration** window (see figure 5). Enable the SPI, I2C and SERIAL interface. The SPI and I2C are used for the sensors, the SERIAL interface (UART) drives the integrated RF-module.



Camera:	○ Enabled	● Disabled
SSH:	● Enabled	○ Disabled
VNC:	● Enabled	○ Disabled
SPI:	● Enabled	○ Disabled
I2C:	● Enabled	○ Disabled
Serial:	● Enabled	○ Disabled
1-Wire:	○ Enabled	● Disabled
Remote GPIO:	○ Enabled	● Disabled

Figure 5: Raspberry Pi interface configuration

2. After enabling the interfaces a dialog should appear asking for a reboot to apply the changes. If no dialog appears reboot by clicking on the Raspberry symbol on the left upper corner of the screen and select **Shutdown**.

The Raspbian OS claims the serial interface for console output. We need to disable this feature to use the serial interface for the communication with our integrated RF-module.

3. Now, after enabling the serial interface, the Raspbian OS claims it for console output. To disable this feature, please remove the string "console = serial0,115200" from the file /**boot/cmdline.txt** and save it. Root privilige is needed to change the file. To open the file accordingly type in terminal:

```
sudo leafpad /boot/cmdline.txt
```

4. Please check whether the serial interface is still enabled by opening the file /**boot/config.txt** and check whether the string "enable_uart=1" is still included. If not, please add it and save the file. Root privilige is needed to change the file. To open the file accordingly type in terminal:

```
sudo leafpad /boot/config.txt
```

5. Please reboot as before and check whether the files /**boot/cmdline.txt** and /**boot/config.txt** are still as described in the previous two points. If not, adapt the two files again as described before and reboot. Otherwise the UART-interface to the module is not active and thus the module communication fails.

4.2 Install the wiringPi library

The wiringPi library is used to easily access the peripherals of the Raspberry Pi.

1. First check if wiringPi is already installed. In a terminal type:

   ```
   gpio −v
   ```

   If you get a version number, then you have it installed already. Make sure that in case of Raspberry Pi 4 the version of wiringPi is at least 2.52 (instuctions follow in the next step).
   In this case continue with the next chapter 4.3.

2. If it is not installed, install wiringPi by

   ```
   sudo apt−get install wiringpi
   ```

   Note: For Raspberry Pi 4B, currently a manual update is required as shown in this instruction: *wiringpi.com 2.52 manual install*).

3. The result of

   ```
   gpio −v
   ```

   should now be the version number of the installed wiringPi version.

4. Double check that

   ```
   cd /usr/lib/
   ls −l *wiring*
   ```

   is showing that *libwiringPi.so* is found at this location.

4.3 Install the AMBER PI driver

The AMBER PI driver was developed in the codeblocks development environment.

1. Thus first download and install the software codeblocks. Therefore open a terminal and type:

   ```
   sudo apt−get install codeblocks
   ```

2. Now download the AMBER PI driver as zip file from (*www.we-online.com/amber-pi*) to the location ~/**Downloads**

3. The file is going to be extracted to the folder ~/**Projects**. If the folder does not exist create it by typing in terminal:

   ```
   mkdir ~/Projects
   ```

4. Now extract the AMBER PI project to ~/**Projects** by typing in terminal:

   ```
   unzip ~/Downloads/AMBER_PI.zip −d ~/Projects
   ```

5. Then start the project via codeblocks by typing in terminal

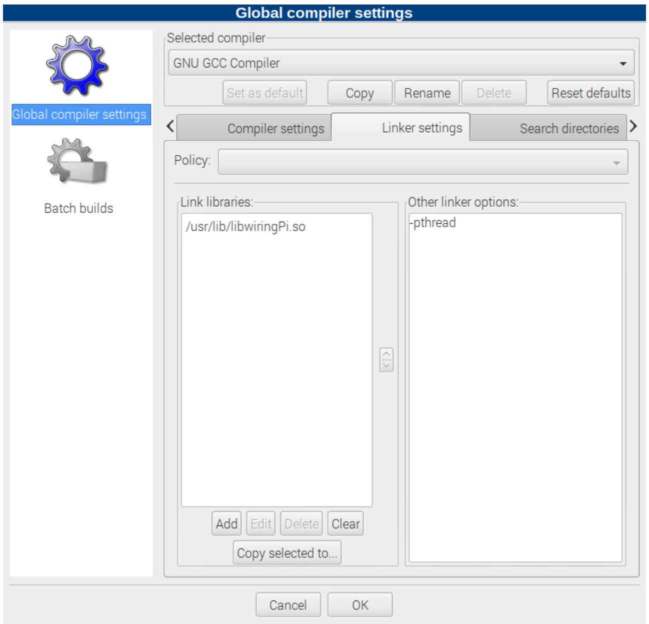
   ```
   sudo codeblocks ~/Projects/AMBER_PI/AMBER_PI.workspace &
   ```

It's important to start the project with root permissions (sud...). Otherwise the access to the peripherals is blocked and thus the Raspbian OS freezes.

6. Now include the wiringPi libraries into codeblocks by opening the global linker settings in **Settings** → **Compiler** → **Linker Settings** and adding the library **/usr/lib/libwiringPi.so** to the **Link libraries** field (see figure 6).

7. Additionally add **-pthread** in the **Other linker options** field. Close the linker settings again.

All necessary libraries are also linked in the projects linker settings to not run into trouble in case they have not been linked in the global linker settings.



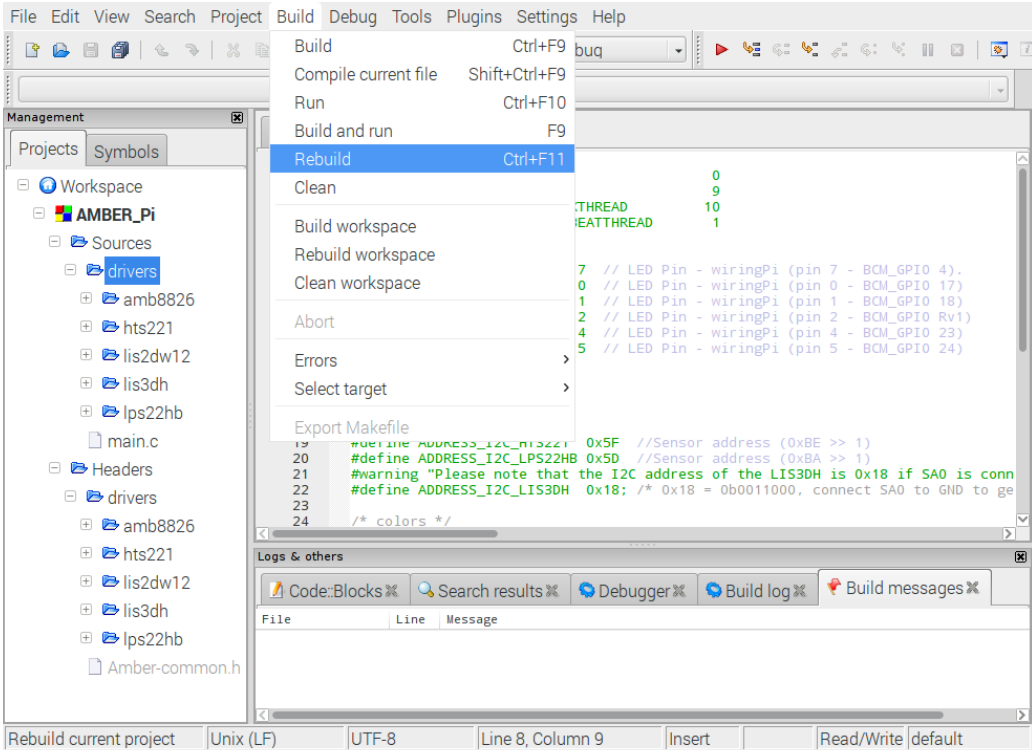
Selected compiler: GNU GCC Compiler

Link libraries: /usr/lib/libwiringPi.so

Other linker options: -pthread

Figure 6: Codeblocks linker settings

8. Then press **Build** → **Rebuild** to build the project (see figure 7).



```
                                    0
                                    9
THREAD                             10
EATTHREAD                           1

7  // LED Pin - wiringPi (pin 7 - BCM_GPIO 4).
0  // LED Pin - wiringPi (pin 0 - BCM_GPIO 17)
1  // LED Pin - wiringPi (pin 1 - BCM_GPIO 18)
2  // LED Pin - wiringPi (pin 2 - BCM_GPIO Rv1)
4  // LED Pin - wiringPi (pin 4 - BCM_GPIO 23)
5  // LED Pin - wiringPi (pin 5 - BCM_GPIO 24)

#define ADDRESS_I2C_HTS221  0x5F  //Sensor address (0xBE >> 1)
#define ADDRESS_I2C_LPS22HB 0x5D  //Sensor address (0xBA >> 1)
#warning "Please note that the I2C address of the LIS3DH is 0x18 if SA0 is conn
#define ADDRESS_I2C_LIS3DH  0x18; /* 0x18 = 0b0011000, connect SA0 to GND to ge

/* colors */
```

Figure 7: Rebuild the application

9. If it builds without errors the Raspberry Pi, WiringPi and AMBER PI driver setup succeeded. Warnings can be ignored as they just provide further information.
   In case of compilation errors, see also chapter `FAQ - Frequently asked questions`.

4.4 Setup the AMBER PI hardware

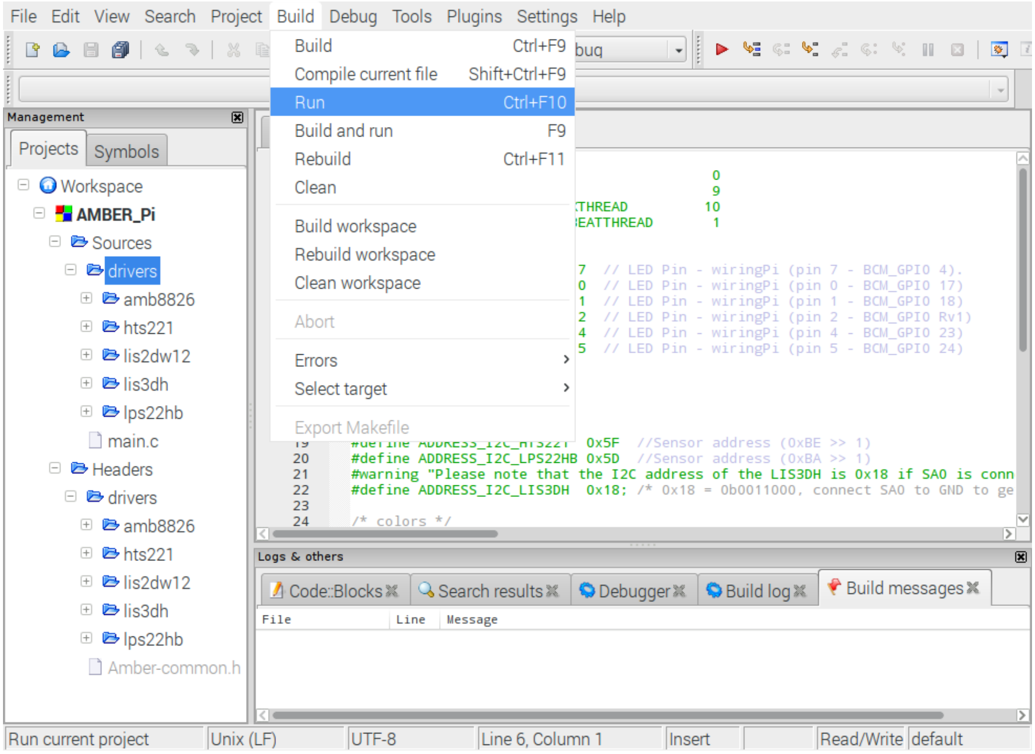
Set the jumpers according to the default configuration (see Table 1).

1. Now attach the sensors to the predefined connectors.
   a) The motion sensor LIS2DW12 to the SPI1 connector.
   b) The pressure sensor LPS22HB to one of the I2C connectors.
   c) The humidity and temperature sensor HTS221 to the remaining I2C connector.
2. Now the AMBER PI is ready for operation. Turn the Raspberry Pi off, connect the AMBER PI and start the Raspberry Pi again.
3. After reboot, open the project again via codeblocks by

```
sudo codeblocks ~/Projects/AMBER_PI/AMBER_PI.workspace &
```

4.5 Run the AMBER PI

1. Please press **Build** → **Run** to run the application on the Raspberry Pi (see figure 8).



```
                      0
                      9
THREAD               10
EATTHREAD             1

7  // LED Pin - wiringPi (pin 7 - BCM_GPIO 4).
0  // LED Pin - wiringPi (pin 0 - BCM_GPIO 17)
1  // LED Pin - wiringPi (pin 1 - BCM_GPIO 18)
2  // LED Pin - wiringPi (pin 2 - BCM_GPIO Rv1)
4  // LED Pin - wiringPi (pin 4 - BCM_GPIO 23)
5  // LED Pin - wiringPi (pin 5 - BCM_GPIO 24)

#define ADDRESS_I2C_HTS221  0x5F  //Sensor address (0xBE >> 1)
#define ADDRESS_I2C_LPS22HB 0x5D  //Sensor address (0xBA >> 1)
#warning "Please note that the I2C address of the LIS3DH is 0x18 if SA0 is conn
#define ADDRESS_I2C_LIS3DH  0x18; /* 0x18 = 0b0011000, connect SA0 to GND to ge

/* colors */
```

Figure 8: Run the default application

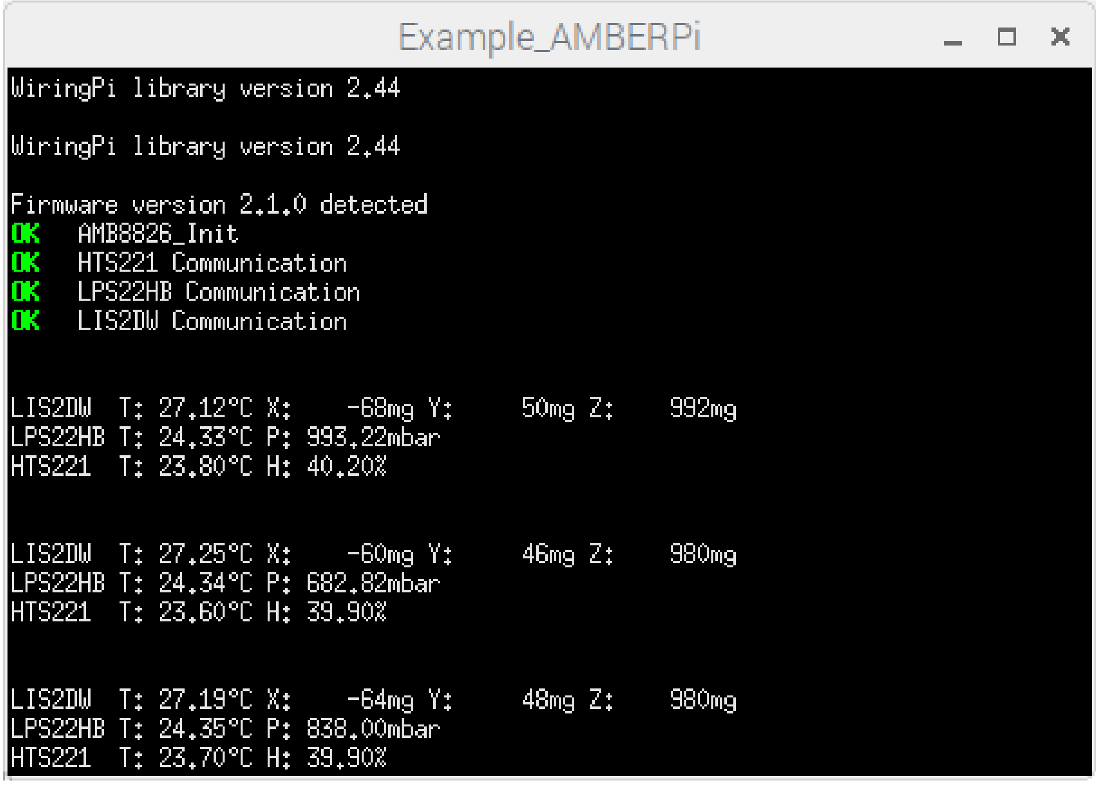
STOP

If the module communication fails, most probably the Raspberry Pi claims the serial interface for its console output. Please repeat the instructions in chapter 4.1.1 to fix this issue.

2. The default application starts and configures the Tarvos-III module as well as the connected sensors, reads their measurement values (temperature, humidity, motion, pressure) once per second and transmits the data via the Tarvos-III.
   Each time the Tarvos-III transmits data the red TX LED on the AMBER PI is flashing.

3. To receive the transmitted data the delivered RF-dongle can be used. Thus please connect the RF stick to the USB port of a Microsoft Windows machine and open the resulting COM port using a terminal program of your choice (e.g. *hterm*) with its default UART settings (115200Baud, Data 8, Stop 1, Parity None, see figure 10). For better readability of the received data the "new line after ... ms" can be set to 100ms.

The Tarvos-III Plug is compatible to the Tarvos-III.

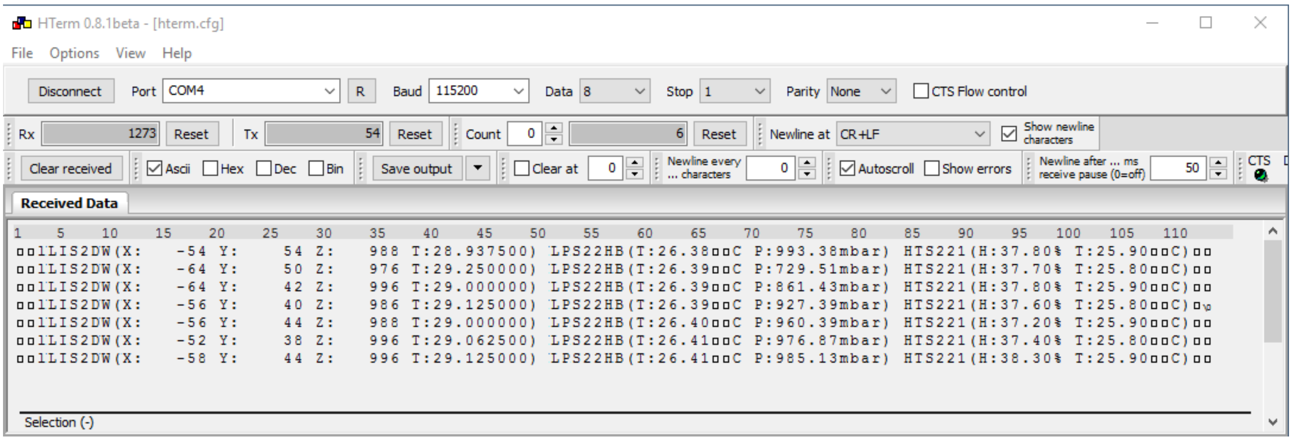


```
WiringPi library version 2.44

WiringPi library version 2.44

Firmware version 2.1.0 detected
OK   AMB8826_Init
OK   HTS221 Communication
OK   LPS22HB Communication
OK   LIS2DW Communication


LIS2DW  T: 27.12°C X:    -68mg Y:     50mg Z:    992mg
LPS22HB T: 24.33°C P: 993.22mbar
HTS221  T: 23.80°C H: 40.20%


LIS2DW  T: 27.25°C X:    -60mg Y:     46mg Z:    980mg
LPS22HB T: 24.34°C P: 682.82mbar
HTS221  T: 23.60°C H: 39.90%


LIS2DW  T: 27.19°C X:    -64mg Y:     48mg Z:    980mg
LPS22HB T: 24.35°C P: 838.00mbar
HTS221  T: 23.70°C H: 39.90%
```

Figure 9: Default application

If your computer does not recognize the connected RF-stick as COM port, please install the latest driver from *http://www.ftdichip.com/Drivers/VCP.htm* first.

The Tarvos-III Plug uses the so called command mode. Thus when receiving data via radio a `CMD_DATA_IND` message including the received radio data is transmitted on the UART.



Port COM4 Baud 115200 Data 8 Stop 1 Parity None □ CTS Flow control

Rx 1273 Tx 54 Count 0 6 Newline at CR+LF ☑ Show newline characters

☑ Ascii □ Hex □ Dec □ Bin □ Clear at 0 Newline every ... characters 0 ☑ Autoscroll □ Show errors Newline after ... ms receive pause (0=off) 50

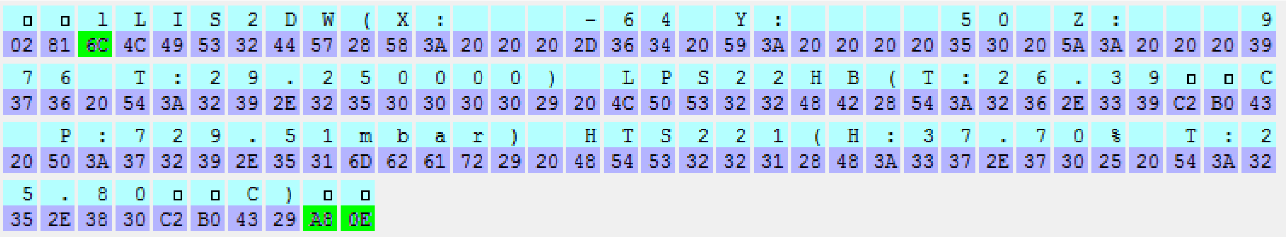
```
□□lLIS2DW(X:   -54 Y:    54 Z:   988 T:28.937500) LPS22HB(T:26.38□□C P:993.38mbar) HTS221(H:37.80% T:25.90□□C)□□
□□lLIS2DW(X:   -64 Y:    50 Z:   976 T:29.250000) LPS22HB(T:26.39□□C P:729.51mbar) HTS221(H:37.70% T:25.80□□C)□□
□□lLIS2DW(X:   -64 Y:    42 Z:   996 T:29.000000) LPS22HB(T:26.39□□C P:861.43mbar) HTS221(H:37.80% T:25.90□□C)□□
□□lLIS2DW(X:   -56 Y:    40 Z:   986 T:29.125000) LPS22HB(T:26.39□□C P:927.39mbar) HTS221(H:37.60% T:25.80□□C)□□
□□lLIS2DW(X:   -56 Y:    44 Z:   988 T:29.000000) LPS22HB(T:26.40□□C P:960.39mbar) HTS221(H:37.20% T:25.90□□C)□□
□□lLIS2DW(X:   -52 Y:    38 Z:   996 T:29.062500) LPS22HB(T:26.41□□C P:976.87mbar) HTS221(H:37.40% T:25.80□□C)□□
□□lLIS2DW(X:   -58 Y:    44 Z:   996 T:29.125000) LPS22HB(T:26.41□□C P:985.13mbar) HTS221(H:38.30% T:25.90□□C)□□
```

Selection (-)

Figure 10: Receiving the transmitted data using the Tarvos-III Plug radio stick and the terminal program HTerm

The `CMD_DATA_IND` frame is of the following form, and contains besides the received payload, also the RSSI value.

Start signal	Command	Length	Payload	RSSI	CS
0x02	0x81	1 Byte	x Byte	1 Byte	1 Byte



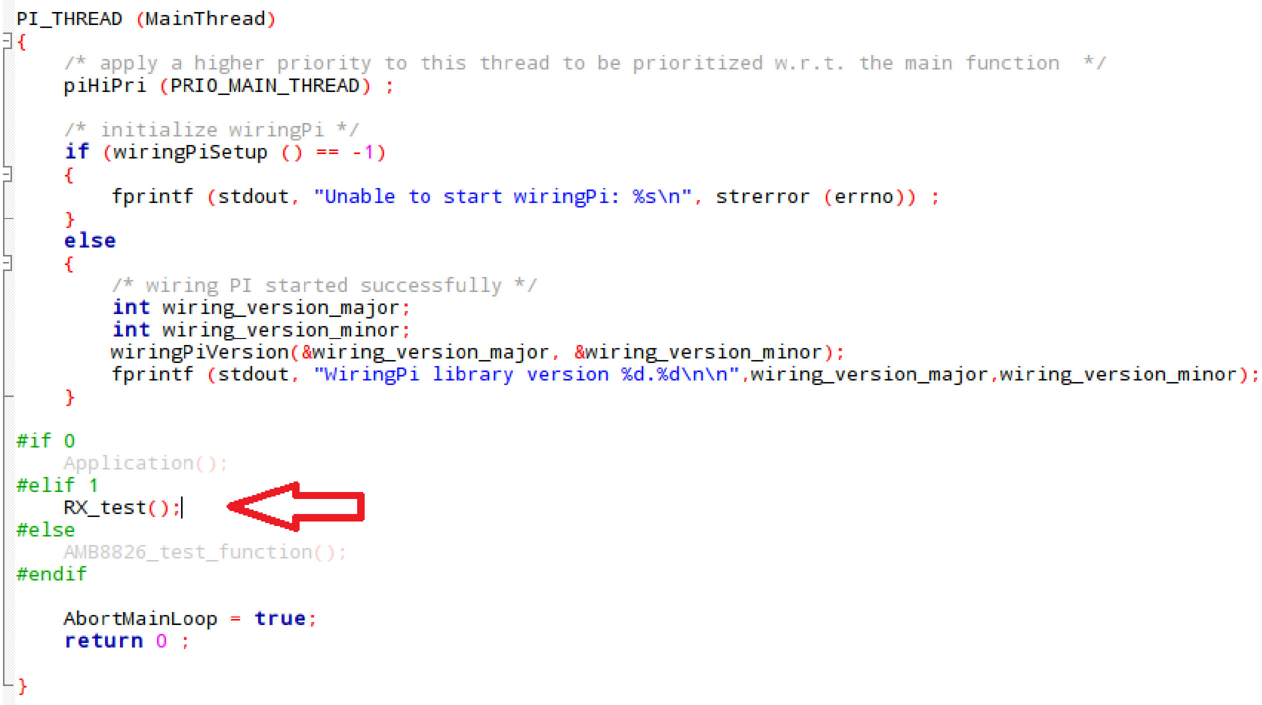
```
 □  □  l  L  I  S  2  D  W  (  X  :           -  6  4     Y  :              5  0     Z  :           9
02 81 6C 4C 49 53 32 44 57 28 58 3A 20 20 20 2D 36 34 20 59 3A 20 20 20 20 35 30 20 5A 3A 20 20 20 39
 7  6     T  :  2  9  .  2  5  0  0  0  0  )     L  P  S  2  2  H  B  (  T  :  2  6  .  3  9  □  □  C
37 36 20 54 3A 32 39 2E 32 35 30 30 30 30 29 20 4C 50 53 32 32 48 42 28 54 3A 32 36 2E 33 39 C2 B0 43
    P  :  7  2  9  .  5  1  m  b  a  r  )     H  T  S  2  2  1  (  H  :  3  7  .  7  0  %     T  :  2
20 50 3A 37 32 39 2E 35 31 6D 62 61 72 29 20 48 54 53 32 32 31 28 48 3A 33 37 2E 37 30 25 20 54 3A 32
 5  .  8  0  □  □  C  )  □  □
35 2E 38 30 C2 B0 43 29 A8 0E
```

Figure 11: Received `CMD_DATA_IND` in hex with length field of 0x6C, RSSI of 0xA8 and CS of 0x0E.

4.6 Advanced: Bidirectional transmission

To demonstrate also the receiving capabilities of the Tarvos-III the test function **RX_test()**, that simply starts the Tarvos-III driver and stays in a while-loop, can be used.

1. Please enable this function in the MainThread of the driver (see figure 12), build and run the program again.



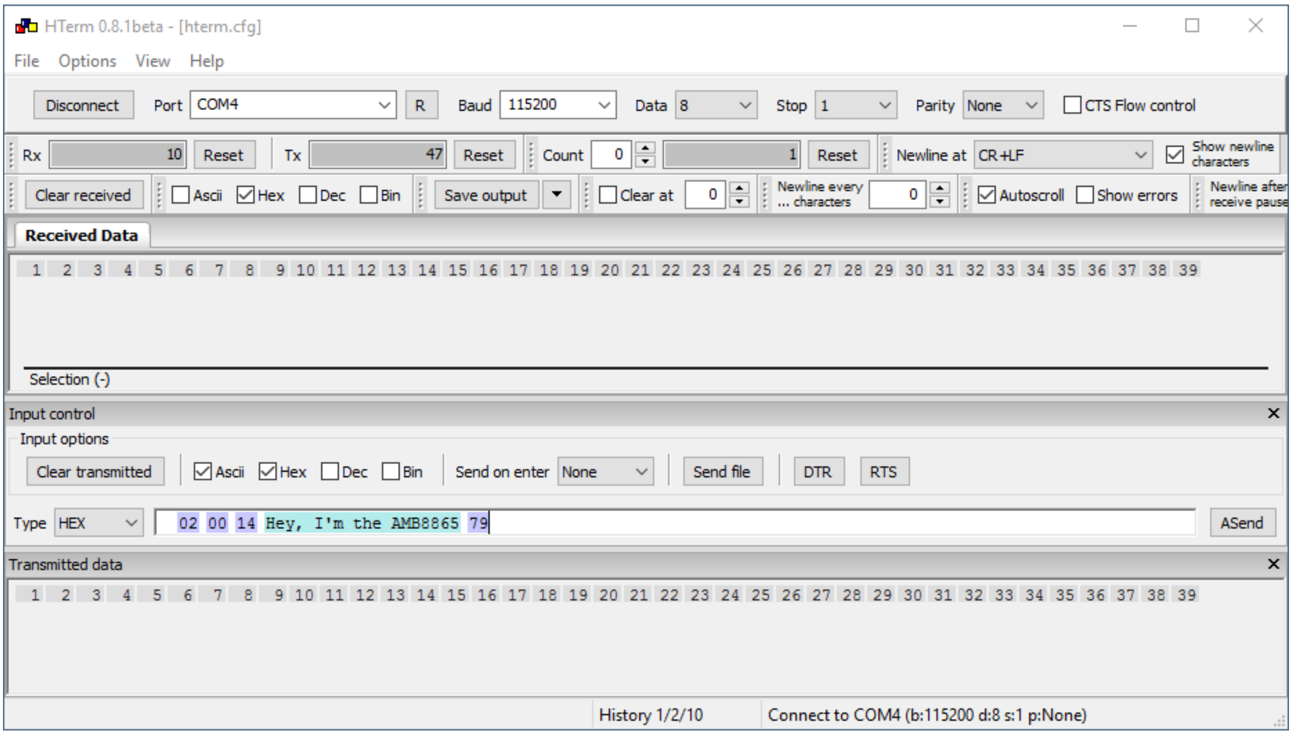
```
PI_THREAD (MainThread)
{
    /* apply a higher priority to this thread to be prioritized w.r.t. the main function  */
    piHiPri (PRIO_MAIN_THREAD) ;

    /* initialize wiringPi */
    if (wiringPiSetup () == -1)
    {
        fprintf (stdout, "Unable to start wiringPi: %s\n", strerror (errno)) ;
    }
    else
    {
        /* wiring PI started successfully */
        int wiring_version_major;
        int wiring_version_minor;
        wiringPiVersion(&wiring_version_major, &wiring_version_minor);
        fprintf (stdout, "WiringPi library version %d.%d\n\n",wiring_version_major,wiring_version_minor);
    }

#if 0
    Application();
#elif 1
    RX_test();
#else
    AMB8826_test_function();
#endif

    AbortMainLoop = true;
    return 0 ;

}
```

Figure 12: Enabling the RX_test() function in the driver application

2. Then, on your windows machine, you have to use a `CMD_DATA_REQ` command to transmit data with the Tarvos-III Plug. The format of the `CMD_DATA_REQ` is as follows:

Start signal	Command	Length	Payload	CS
0x02	0x00	1 Byte	Length Byte	1 Byte

Thus, to transmit a string like "Hey, I'm the AMB8865", the command looks like shown in figure 13. In this example the length field is 0x14 (20) and the checksum is 0x79.


HTerm 0.8.1beta - [hterm.cfg]

Port COM4, Baud 115200, Data 8, Stop 1, Parity None, ☐ CTS Flow control

Rx 10, Tx 47, Count 0, 1, Newline at CR+LF, ☑ Show newline characters

☐ Ascii ☑ Hex ☐ Dec ☐ Bin, ☐ Clear at 0, Newline every ... characters 0, ☑ Autoscroll ☐ Show errors

Input options: ☑ Ascii ☑ Hex ☐ Dec ☐ Bin, Send on enter None

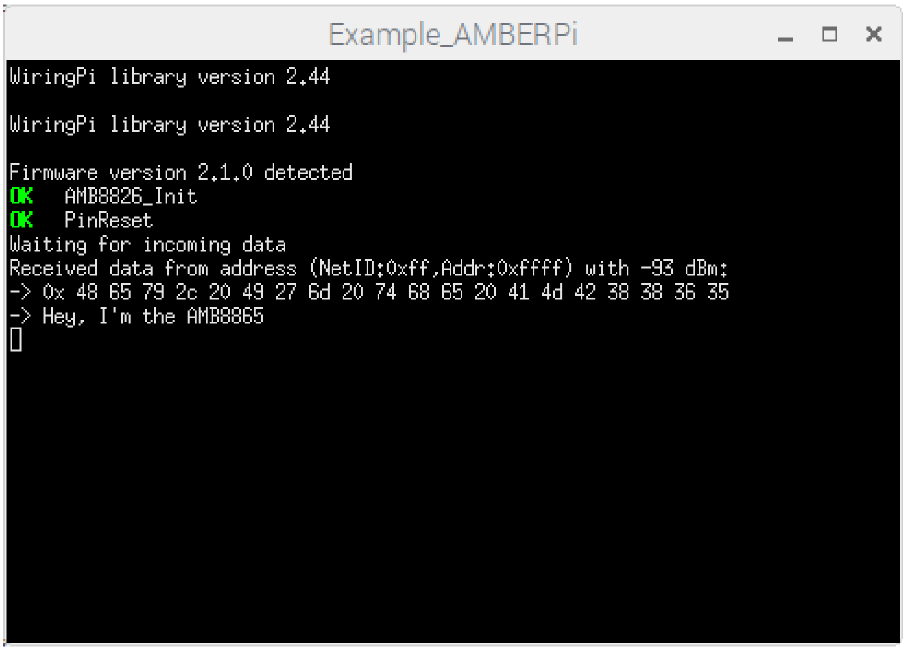
Type HEX: `02 00 14 Hey, I'm the AMB8865 79`

History 1/2/10

Connect to COM4 (b:115200 d:8 s:1 p:None)

Figure 13: Transmitting an arbitrary string using the Tarvos-III Plug radio stick

3. The Tarvos-III on the AMBER PI will receive the data transmitted and gives it to the driver (see figure 14).

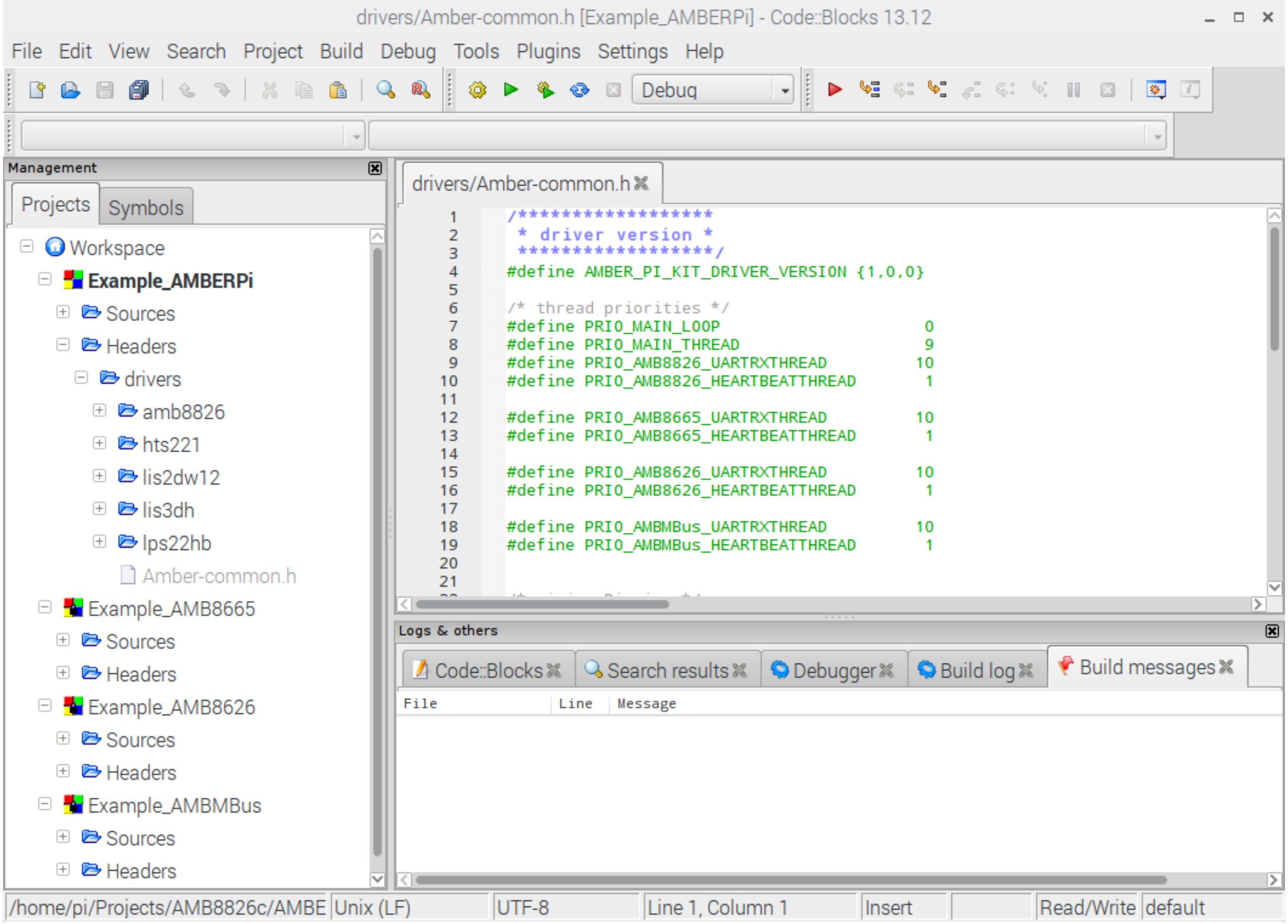


```
WiringPi library version 2.44

WiringPi library version 2.44

Firmware version 2.1.0 detected
OK   AMB8826_Init
OK   PinReset
Waiting for incoming data
Received data from address (NetID:0xff,Addr:0xffff) with -93 dBm:
-> 0x 48 65 79 2c 20 49 27 6d 20 74 68 65 20 41 4d 42 38 38 36 35
-> Hey, I'm the AMB8865
```

Figure 14: Receiving the transmtted data

5 The AMBER PI driver

The AMBER PI driver contains drivers of the delivered sensors, of the integrated Tarvos-III RF-module as well as of additional Würth Elektronik eiSos products.



```
/******************
 * driver version *
 ******************/
#define AMBER_PI_KIT_DRIVER_VERSION {1,0,0}

/* thread priorities */
#define PRIO_MAIN_LOOP                    0
#define PRIO_MAIN_THREAD                  9
#define PRIO_AMB8826_UARTRXTHREAD        10
#define PRIO_AMB8826_HEARTBEATTHREAD      1

#define PRIO_AMB8665_UARTRXTHREAD        10
#define PRIO_AMB8665_HEARTBEATTHREAD      1

#define PRIO_AMB8626_UARTRXTHREAD        10
#define PRIO_AMB8626_HEARTBEATTHREAD      1

#define PRIO_AMBMBus_UARTRXTHREAD        10
#define PRIO_AMBMBus_HEARTBEATTHREAD      1
```

Figure 15: AMBER PI driver

5.1 Tarvos-III - Würth Elektronik eiSos 868MHz radio module

The Tarvos-III is one of the compact and low-cost radio data transmission module for wireless half-duplex communication provided by Würth Elektronik eiSos. For full information we refer to the Tarvos-III manual [2] and Tarvos-III datasheet [3].

The Tarvos-III acts as a slave and can be fully controlled by the Raspberry Pi. The configuration as well as the operation of the module can be managed by predefined commands that are sent as telegrams over the UART interface. The AMBER PI driver implements this command interface to provide a simple API to the user. A short description of the command interface can be found in chapter `5.1.1`.

Besides the routines to set the module into low power mode, reset it or transmit and receive data, several functions are provided to readout and adapt the module's configuration parameters.

To run the Tarvos-III driver the function **TarvosIII_Init()** has to be called first. Its input arguments are:

- the connected pin numbers (reset, wake up, boot) of the Raspberry Pi
- the data baudrate to setup the serial connection to the integrated RF-module
- the callback function that returns the data that has been received by the module via RF
- the address mode that defines number of available addresses and thus the structure of the `CMD_DATAEX_CMD` command

> ! Please note that when adapting the baudrate or address mode of the module, the driver has to be restarted (deinit and init again). The default UART baudrate is 115200 Baud and the default address mode is mode 0. Please check the Tarvos-III manual [2] for all default values.

> ! Please be cautious when frequently updating the non-volatile user settings. These settings are stored in the module's flash and thus can be updated only a limited number of times before hardware failure.

5.1.1 The Tarvos-III command interface

The commands of the command interface can be divided into 3 groups:

- Requests: The host requests the module to trigger any action, e.g. in case of the request `CMD_RESET_REQ` the host asks the Tarvos-III to perform a reset.
- Confirmations: On each request the module answers with a confirm message to give a feedback on the requested operation status. In case of a `CMD_RESET_REQ`, the module answers with a `CMD_RESET_CNF` to tell the host whether the reset will be performed or not.
- Indications and Responses: The module indicates spontaneously when a special event occurred. The `CMD_DATAEX_IND` indicates for example that data was received via RF.

Thus for example, when calling the function **TarvosIII_Reset()**, the driver sends the `CMD_RESET_REQ` message and waits for the corresponding `CMD_RESET_CNF` message that is send back from the module to confirm that the reset request was received and will be executed.

Second example: When the driver spontaneously receives a `CMD_DATAEX_IND` message that contains data received by the module via RF, the driver gives the received RF packet data to its registered callback function to inform the user about the received RF data.

The commands itself have to following format:

Start signal	Command	Length	Payload	CS
0x02	1 Byte	1 Byte	Length Byte	1 Byte

Example 1: CMD_RESET_REQ
This command triggers a software reset of the module. The reset is performed after the acknowledgement is transmitted. All volatile settings are initialized with their defaults.

The `CMD_RESET_REQ` has the command number 0x05 with length field 0x00. Thus the payload field is empty.

Format:

Start signal	Command	Length	CS
0x02	0x05	0x00	0x07

Example 2: CMD_DATA_REQ
This command serves the simple data transfer in the command mode. Transmission takes place on the configured channel to the previously parameterised destination address. The `CMD_DATA_REQ` has the command number 0x00. The length field is the number of bytes we'd like to transmit via RF.

Format:

Start signal	Command	Length	Payload	CS
0x02	0x00	1 Byte	Length Byte	1 Byte

Sending "Hello World!" means

Start signal	Command	Length	Payload	CS
0x02	0x00	0x0C	0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21	0x0F

where we send 12 bytes (0x0C), which are "Hello World!" (0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21). The resulting checksum is 0x0F.

5.2 HTS221 - Digital sensor for relative humidity and temperature

The HTS221 is an ultra-compact sensor for relative humidity and temperature. It includes a sensing element and a mixed signal ASIC to provide the measurement information through digital serial interfaces.

The AMBER PI driver contains the main functions to configure and run the HTS221. For the description of the driver functions we refer to the documentation that is provided in the source code. For the description of this sensor, we refer to its datasheet [4].

When connecting the HTS221 sensor board please align it following the white "triangle marking" located on top.

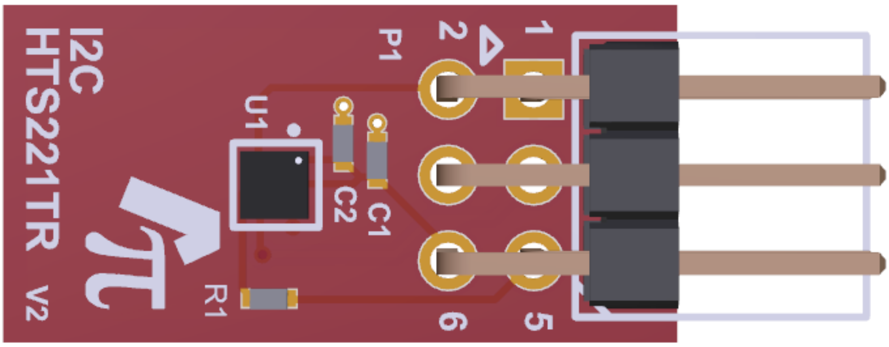


Figure 16: HTS221 sensor board

5.3 LPS22HB - MEMS nano pressure sensor

The LPS22HB is an ultra-compact piezoresistive absolute pressure sensor which functions as a digital output barometer. The device comprises a sensing element and an IC interface which communicates through I2C or SPI from the sensing element to the application. The AMBER PI driver contains the main functions to configure and run the LPS22HB. For the description of the driver functions we refer to the documentation that is provided in the source code. For the description of this sensor, we refer to its datasheet [5].

When connecting the LPS22HB sensor board please align it following the white "triangle marking" located on top.

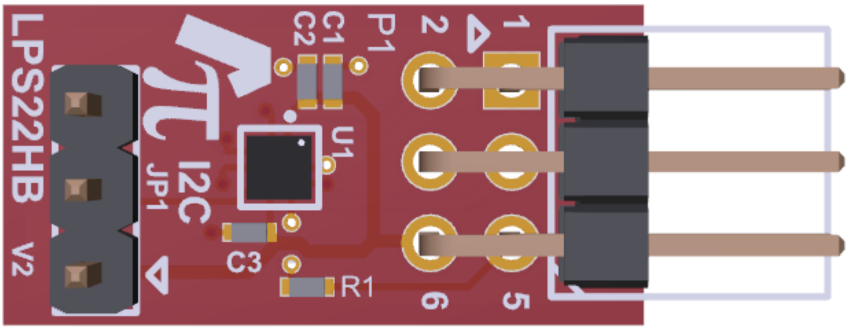


Figure 17: LPS22HB sensor board

5.4 LIS2DW12 - MEMS digital output motion sensor

The LIS2DW12 is an ultra-low-power high-performance three-axis linear accelerometer belonging to the "femto" family which leverages on the robust and mature manufacturing processes already used for the production of micromachined accelerometers.

The LIS2DW12 has user-selectable full scales of ±2g/±4g/±8g/±16g and is capable of measuring accelerations with output data rates from 1.6 Hz to 1600 Hz.

The AMBER PI driver contains the main functions to configure and run the LIS2DW12. For the description of the driver functions we refer to the documentation that is provided in the source code. For the description of this sensor, we refer to its datasheet [6].

Please connect the LIS2DW12 sensor board on P9 (SPI1) connector.

When connecting the LIS2DW12 sensor board please align it following the white "triangle marking" located on top.

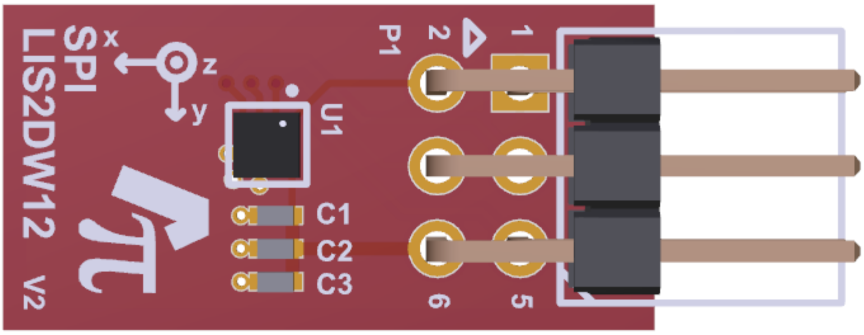


Figure 18: LIS2DW12 sensor board

5.5 PROTO SPI - Mini breadboard for SPI customized connections

The PROTO SPI is a mini breadboard with onboard all the typical 4-wires SPI signals plus the power signals +3.3V and GND.

Please refer to the schematics chapter for detailed signal connections.

When connecting the PROTO SPI board please align it following the white "triangle marking" located on top.

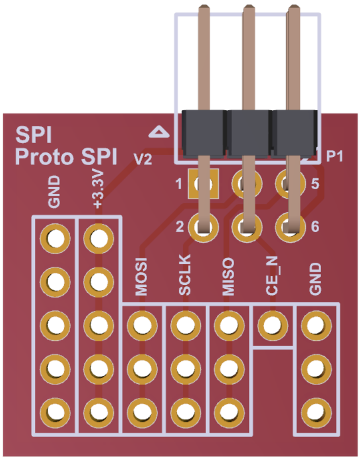


Figure 19: PROTO SPI board

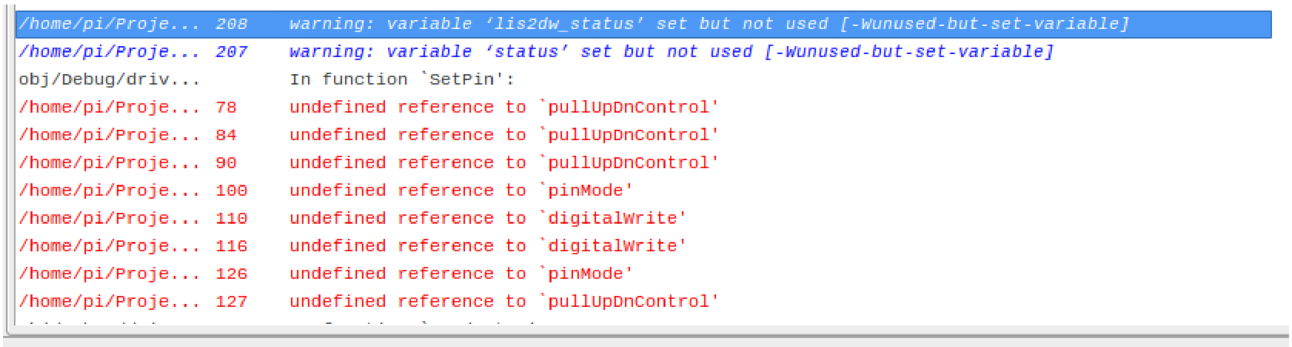
6 FAQ - Frequently asked questions

6.1 Is my Raspberry Pi compatible to the AMBER PI?

Please refer to the chapter 3 to check your compatibility.

6.2 I get the following compilation errors, what can I do?

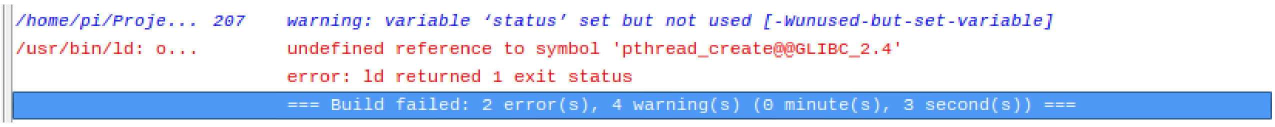
- Undefined reference to pullUpDnControl, pinMode, digitalwrite,...



```
/home/pi/Proje...  208    warning: variable ‘lis2dw_status’ set but not used [-Wunused-but-set-variable]
/home/pi/Proje...  207    warning: variable ‘status’ set but not used [-Wunused-but-set-variable]
obj/Debug/driv...         In function `SetPin':
/home/pi/Proje...  78     undefined reference to `pullUpDnControl'
/home/pi/Proje...  84     undefined reference to `pullUpDnControl'
/home/pi/Proje...  90     undefined reference to `pullUpDnControl'
/home/pi/Proje...  100    undefined reference to `pinMode'
/home/pi/Proje...  110    undefined reference to `digitalWrite'
/home/pi/Proje...  116    undefined reference to `digitalWrite'
/home/pi/Proje...  126    undefined reference to `pinMode'
/home/pi/Proje...  127    undefined reference to `pullUpDnControl'
```

Solution: Probably the wiringPi was not linked correctly. Please refer to figure 6.

- Undefined reference to symbol pthread_create,...



```
/home/pi/Proje...  207    warning: variable ‘status’ set but not used [-Wunused-but-set-variable]
/usr/bin/ld: o...         undefined reference to symbol 'pthread_create@@GLIBC_2.4'
                          error: ld returned 1 exit status
                          === Build failed: 2 error(s), 4 warning(s) (0 minute(s), 3 second(s)) ===
```

Solution: Probably the pthread library was not linked correctly. Please refer to figure 6.

6.3 After installation of the AMBER PI driver, I get a "NOK" when calling the TarvosIII_Init-function. What can I do?

Probably the configuration of the serial interface (UART) did not work properly. To use the AMBER PI, the serial interface has to be enabled. By default, the Raspberry Pi uses the enabled serial interface for console output. To make the serial interface available for the AMBER PI communication, the console output has to be disabled.
Please follow the instructions in chapter 4.1.1 to solve this issue.

6.4 When starting codeblocks, there are many warnings in the console. Is this a problem?

Warning: Mismatch between the program and library build versions detected. The library used 3.0 (wchar_t,compiler with C++ ABI 1010,wx containers,compatible with 2.8), and your program used 3.0 (wchar_t,compiler with C++ ABI 1009,wx containers,compatible with 2.8)

No it is not, since a compatible version of codeblocks has been installed. If it is still a problem the newest version of codeblocks has to be compiled and installed as described in: *http://wiki.codeblocks.org/index.php/Installing_Code::Blocks_from_source_on_Linux*

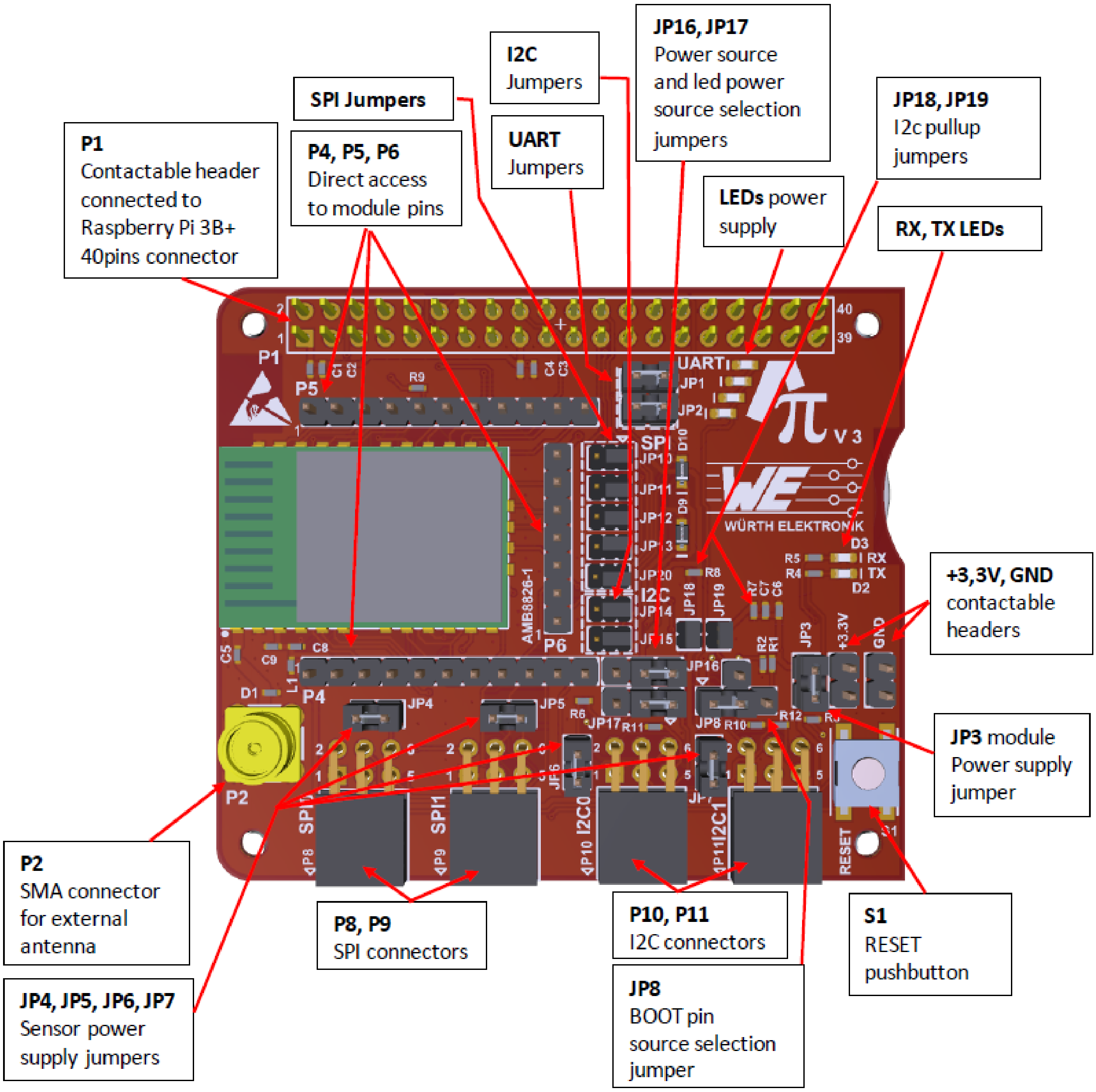
6.5 How can I get the AMBER PI driver?

Please check the download section of the AMBER PI website
www.we-online.com/amber-pi

6.6 Is there also a driver for other radio modules of Würth Elektronik eiSos?

Yes, in version 2.1.0 of the driver the support of other Würth Elektronik eiSos products has been moved to the Wireless Connectivitiy SDK. It is available on our website
www.we-online.com

7 AMBER-PI development board



P1 Contactable header connected to Raspberry Pi 3B+ 40pins connector

SPI Jumpers

P4, P5, P6 Direct access to module pins

I2C Jumpers

UART Jumpers

JP16, JP17 Power source and led power source selection jumpers

LEDs power supply

JP18, JP19 I2c pullup jumpers

RX, TX LEDs

+3,3V, GND contactable headers

JP3 module Power supply jumper

P2 SMA connector for external antenna

JP4, JP5, JP6, JP7 Sensor power supply jumpers

P8, P9 SPI connectors

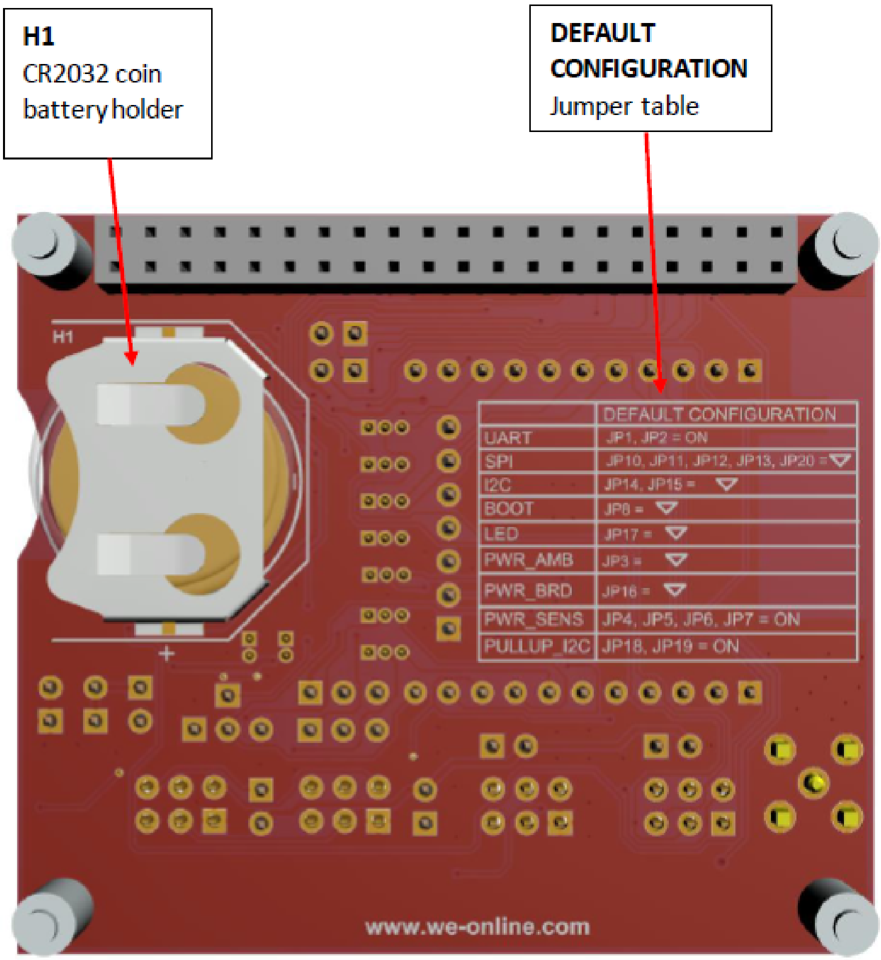
P10, P11 I2C connectors

JP8 BOOT pin source selection jumper

S1 RESET pushbutton

Figure 20: Top view (mounted on Raspberry Pi 3B)

Please connect the sensor boards as indicated by the "triangle markings" on the top side for the correct alignment.



	DEFAULT CONFIGURATION
UART	JP1, JP2 = ON
SPI	JP10, JP11, JP12, JP13, JP20 = ▽
I2C	JP14, JP15 = ▽
BOOT	JP8 = ▽
LED	JP17 = ▽
PWR_AMB	JP3 = ▽
PWR_BRD	JP16 = ▽
PWR_SENS	JP4, JP5, JP6, JP7 = ON
PULLUP_I2C	JP18, JP19 = ON

Figure 21: Bottom view

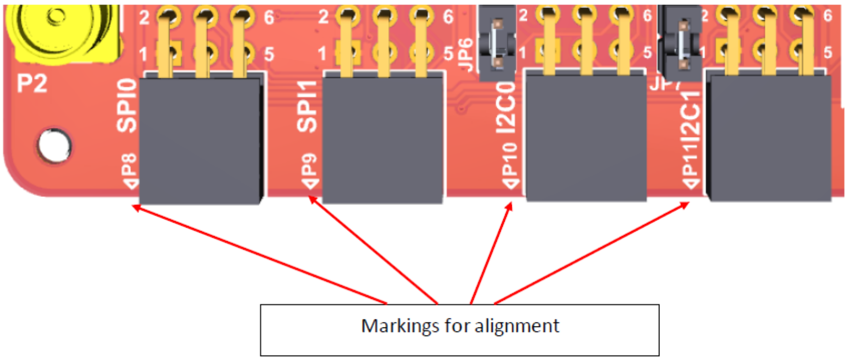


Figure 22: Sensor boards alignment

7.1 Tarvos-III long range radio module

For detailed information of the Tarvos-III radio module, please refer to the Tarvos-III manual [2].

Please note that we do not guarantee the proper operation of the radio profile 3 (0.625 kbps long range mode) of the Tarvos-III over the full temperature range. This only holds for Tarvos-III of hardware version 2.2 or older and serial number 116.002000 or smaller.

7.2 Jumper settings

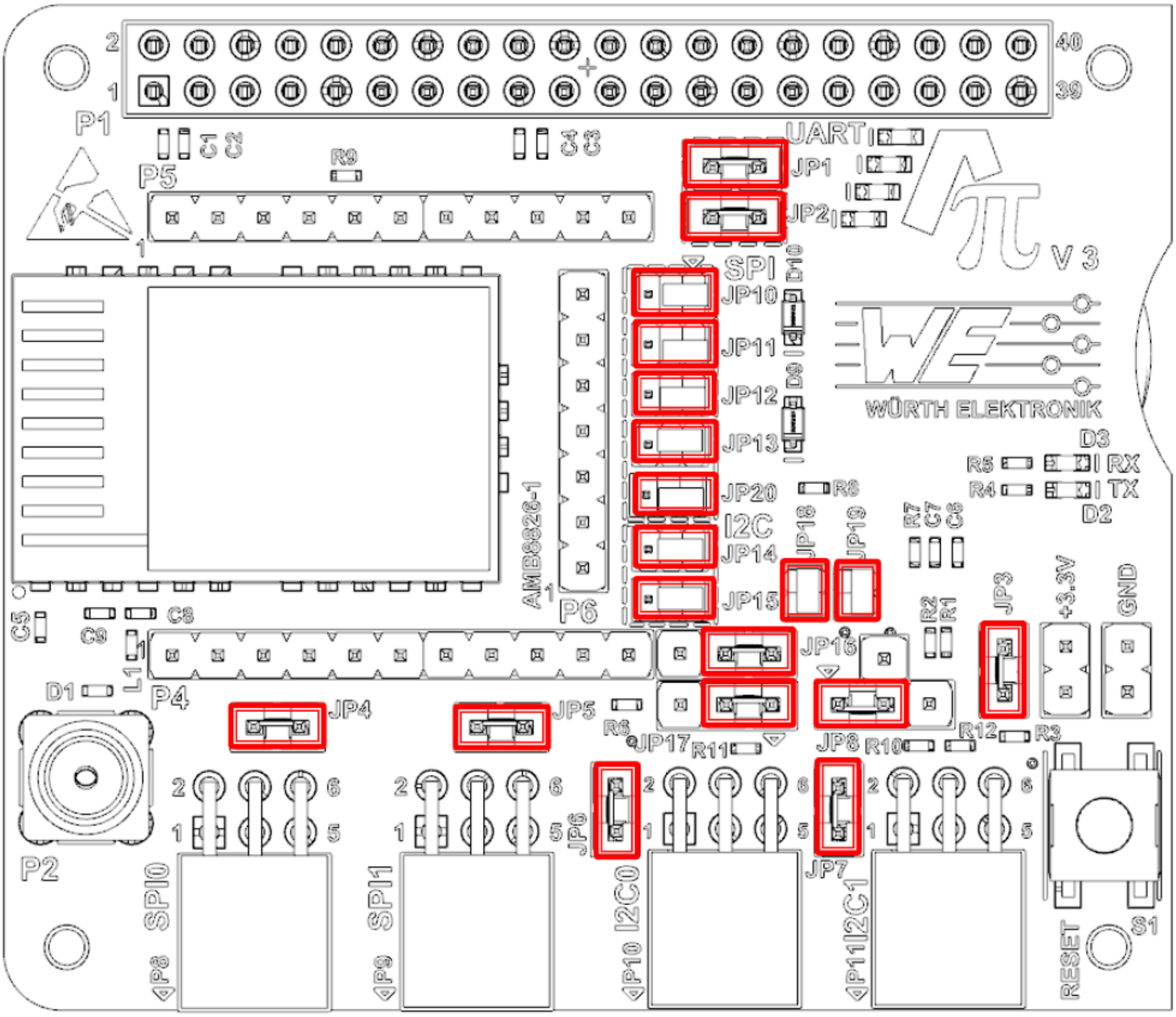


Figure 23: Jumper settings

▭ Jumper is set

Jumper designator		Function	Default location
JP1-2	UART	Connect the Raspberry Pi UART pins to the Tarvos-III UART pins. Set = Connected, Not set = Not connected	Set
JP3	Module power supply	Supply module with power. Set = module is powered, Not set = module is not powered	Set
JP4-7	Sensor power supply	Supply sensors with power. Set = Sensor is powered, Not set = Sensor is not powered	Set
JP8	Boot	Connect the Tarvos-III Boot pin. Left = Connected to GND (Normal operation), Middle = Boot pin is handled by the Raspberry Pi, Right = Connected to high (Bootloader starts after reset)	Left
JP10-13, JP20	SPI	Connect Raspberry Pi SPI pins or Tarvos-III SPI pins to the SPI sensors. Right = Raspberry Pi SPI pins are connected to the SPI sensors, Left = Tarvos-III SPI pins are connected to the SPI sensors	Right
JP14-15	I2C	Connect Raspberry Pi I2C pins or Tarvos-III I2C pins to the I2C sensors. Right = Raspberry Pi I2C pins are connected to the I2C sensors, Left = Tarvos-III I2C pins are connected to the I2C sensors	Right
JP16	Power source	Power source for the AMBER PI. Left = Battery is used as power source, Right = Raspberry Pi is used as power source	Right
JP17	Power LED source	Supply the power. Right = PWM of Raspberry Pi is used, Left = Constant 3.3V signal is used	Right
JP18-19	I2C Pull up	Set: Pull-up resistors are connected, Not set = Pull-up resistors are not connected	Set
LPS22HB-JP1	LPS22HB SA0/SD0	Switch the address of this sensor	Right

Table 1: Jumper documentation

7.3 Power supply

There are two possibilities to supply the evaluation board and the module with voltage.

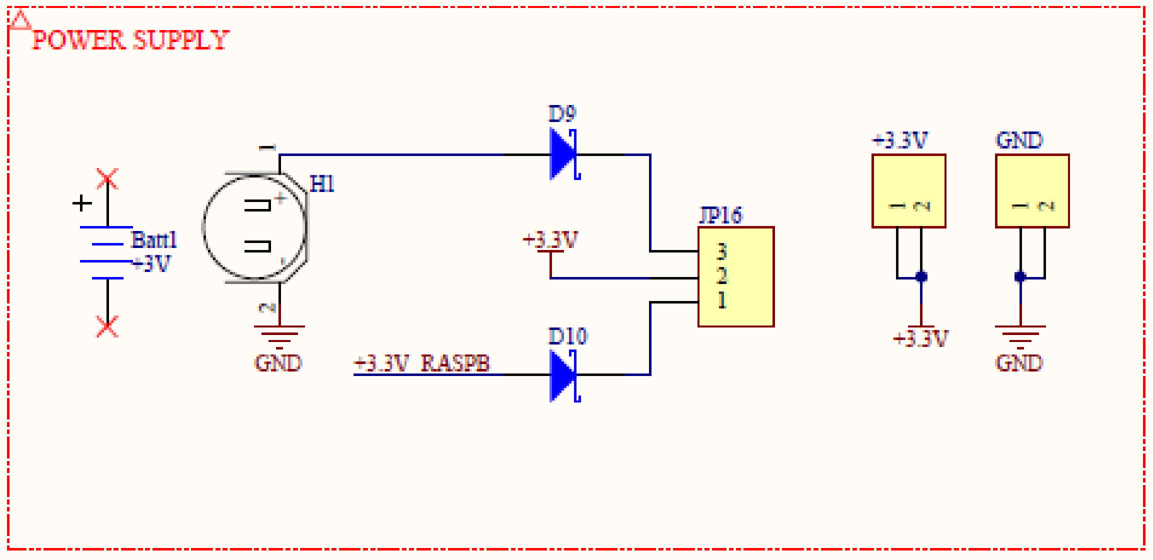


Figure 24: Power supply schematic

The different power sources are connected with diodes (approx. dropout 0.2V) for protection reasons. The power source is simply chosen mounting the JP16 jumper in the wanted position.
For convenience two 2x1 contactable headers provide +3.3V and GND potentials.

7.3.1 Battery source

The development board has the possibility to be powered on by a standard 3V CR2032 coin battery (e.g. standalone application).
Battery has to be inserted in the battery holder named H1 located in the bottom side of the pcb.

STOP

Please respect the polarity when inserting the battery as indicated on the H1 battery holder serigraphy.

7.3.2 Raspberry Pi source

In this case the development board is powered on using the +3.3V lines coming from the Raspberry Pi through the P1 connector.

7.4 Current measurement

The development board provides the opportunity to measure the current consumption of each stage, especially useful when designing a dedicated application in which current budget is a key parameter.

Please consider the overall current consumption when designing a final application.
When using battery as power source, please refer to CR2032 coin battery datasheet.
When using Raspberry Pi as power source, please refer to its datasheet.

7.4.1 Tarvos-III current measurement

For Tarvos-III current measurement please follow these steps:

1. Remove the jumper on JP3
2. Connect a current meter in series on JP3 (refer to the schematics pages for reference)

For normal operation please set again the jumper on JP3 in default position.

7.4.2 Sensor current measurement

For sensor current measurement please follow these steps:

1. Remove the relative jumpers related to the sensors under analysis (JP4, JP5, JP6 or JP7)
2. Connect a current meter in series on relative JP position (refer to the schematics pages for reference)

For normal operation please set again the jumpers on default positions.

7.5 Interfaces

7.5.1 Extended connector P1

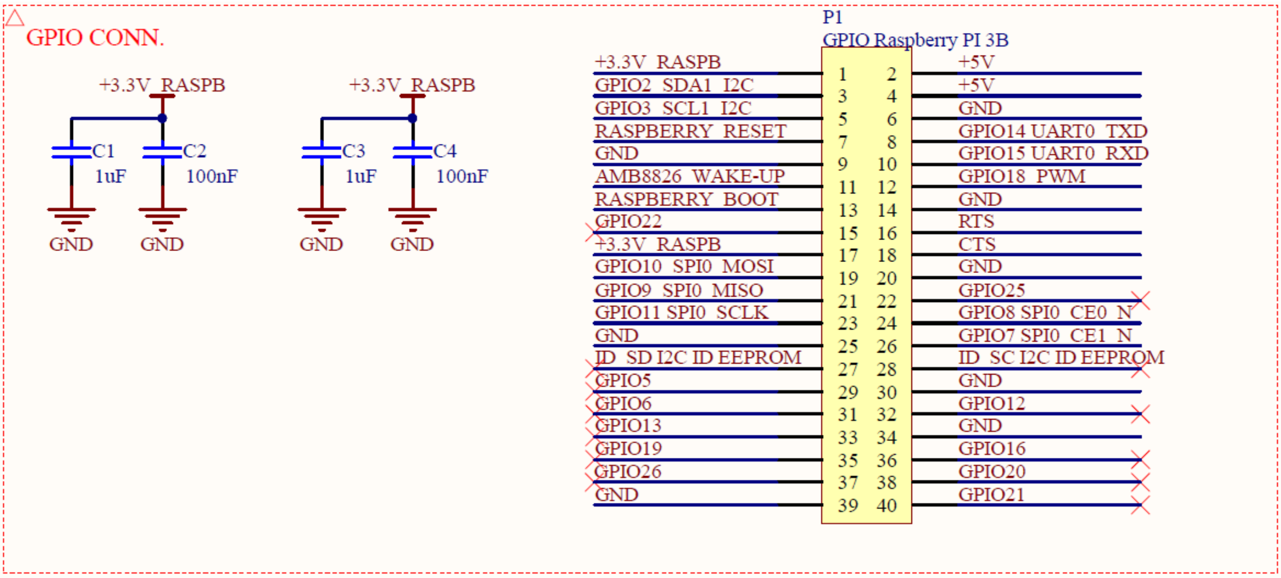


Figure 25: Extended connector P1 from Raspberry Pi board

The connector P1 extends the power and GPIO signals coming from the Raspberry Pi board on the AMBER PI board.
This allows the user the possibility to contact the shared pins on it once the two boards are mounted together.

7.5.2 SMA connector P2

The development board gives the user the possibility to use an external antenna (when using a module with RF pad) through the P2 SMA connector.

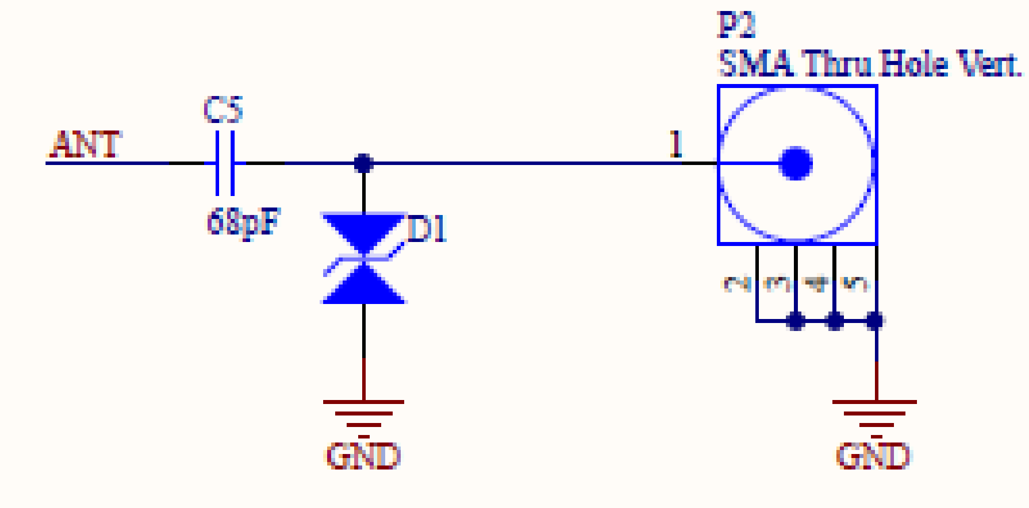


Figure 26: SMA connector P2 for external antenna connection

7.5.3 Headers P4, P5, P6

All pins of the RF module, except for the RF-pin, are available on 2.54 mm pitch headers P4, P5, P6 (not mounted).

7.5.4 SPI connectors P8, P9

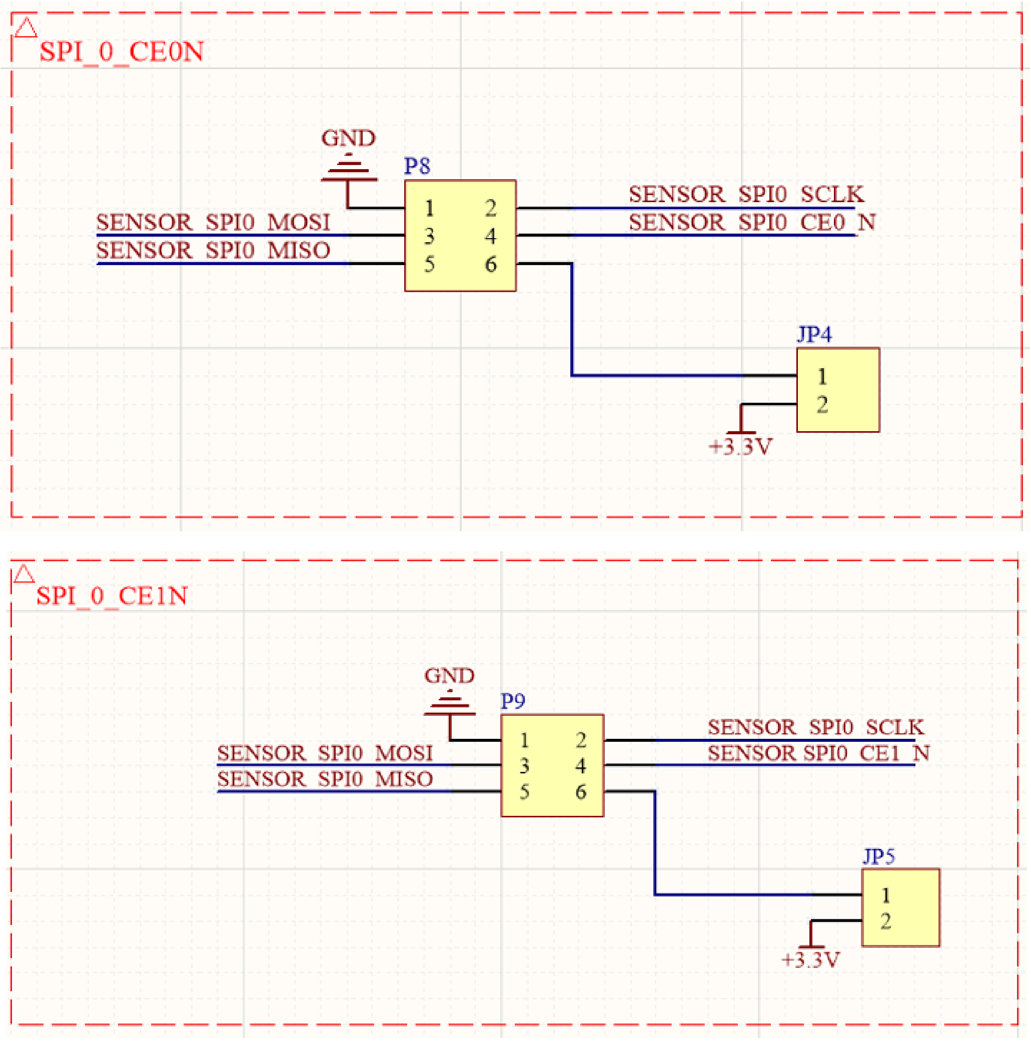


Figure 27: SPI connectors P8 and P9

On the development board the connectors P8 and P9 share the SPI signals plus the +3.3 and GND power signals.
Depending on how the SPI jumpers are set (please refer to chapter 7.1), there is the possibility to drive the P8 connector directly with the SPI signals coming from RF module (e.g. dedicated programmed firmware on it).
In default SPI jumper settings, both P8 and P9 are driven directly from Raspberry Pi SPI signals as shown in the schematics.

Please be sure that the jumpers are set on JP4 and JP5 in order to provide power to P8 and P9 connectors.

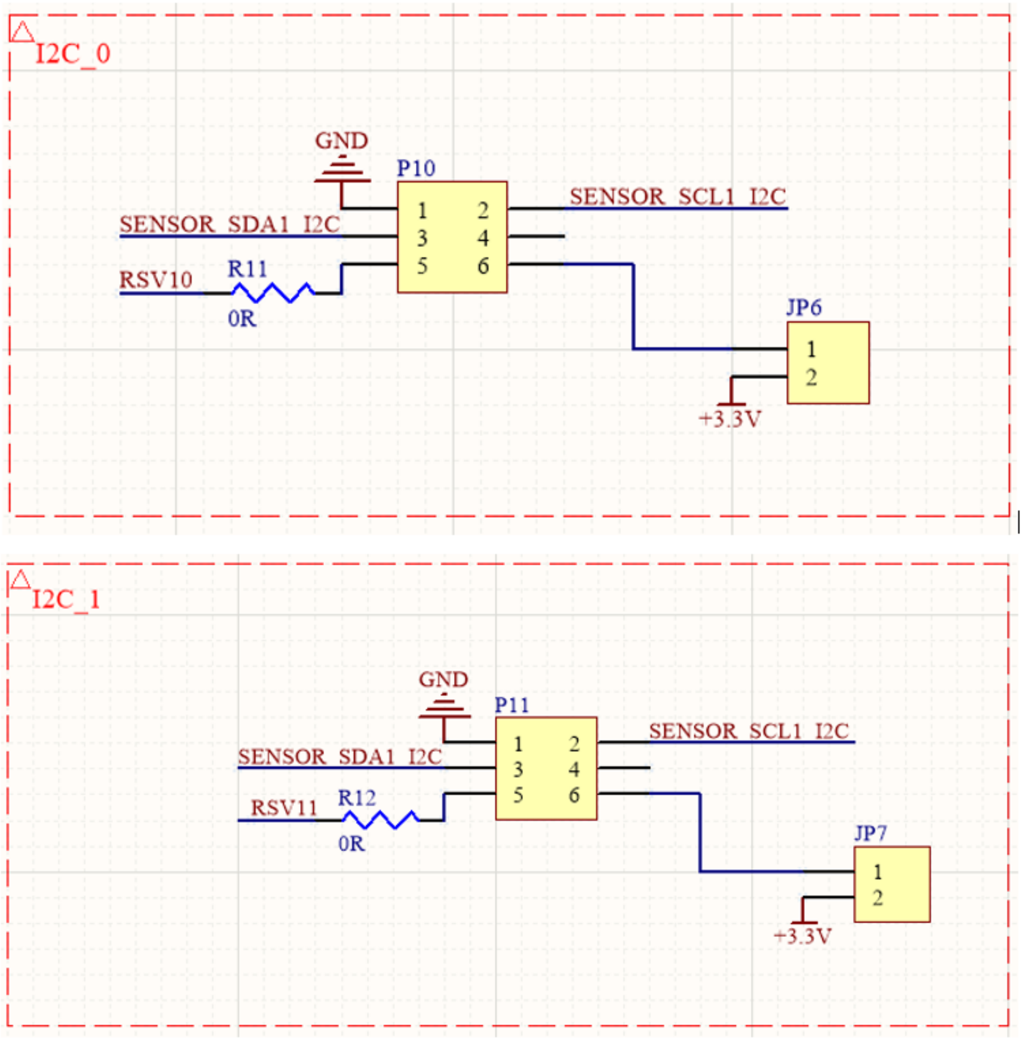


Figure 28: I2C connectors P10 and P11

7.5.5 I2C connectors P10, P11

On the development board the connectors P10 and P11 share the I2C signals plus the +3.3 and GND power signals.
Depending on how the I2C jumpers are set (please refer to chapter 7.1), there is the possibility to drive the P10 and P11 connectors directly with the I2C signals coming from RF module (e.g. dedicated programmed firmware on it).
In default I2C jumper settings, both P10 and P11 are driven directly from Raspberry Pi I2C signals as shown in the schematics.

> Please be sure that the jumpers are set on JP6 and JP7 in order to provide power to P10 and P11 connectors.

7.5.6 BOOT

Based on the jumper position on JP8 (please refer to chapter 7.1), the *BOOT* pin of the module can be set in three different status:

1. Fixed to 1 logic level through a pull-up resistor
2. Fixed to 0 logic level through a pull-down resistor
3. Connected and driven by Raspberry Pi

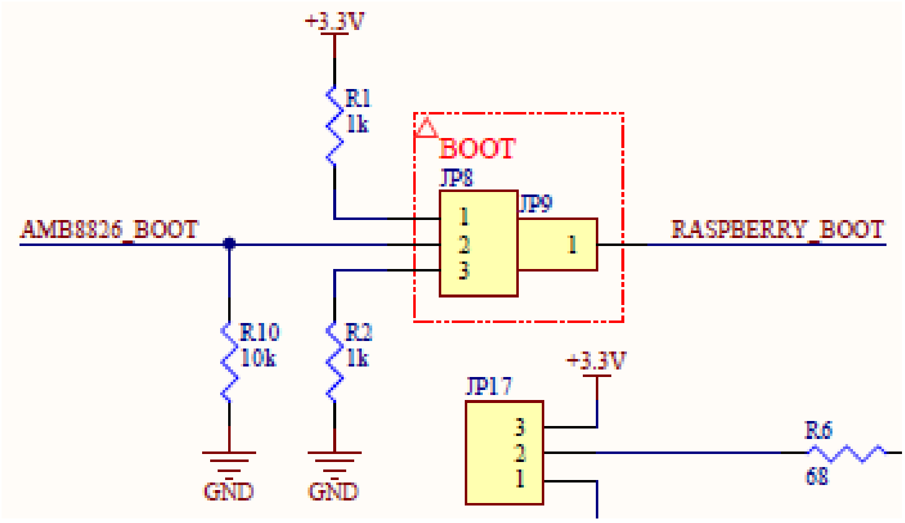


Figure 29: BOOT pin connections

7.5.7 RESET

The RESET signal can be provided to the module in two ways:

1. Driven directly from Raspberry Pi board by the driver
2. Using the S1 pushbutton

7.5.8 LEDs

Three different kind of leds are located on the development board in order to show the respective status:

- RX led D3 (GREEN) flashes when the module receives packets
- TX led D2 (RED) flashes when the module transmits packets

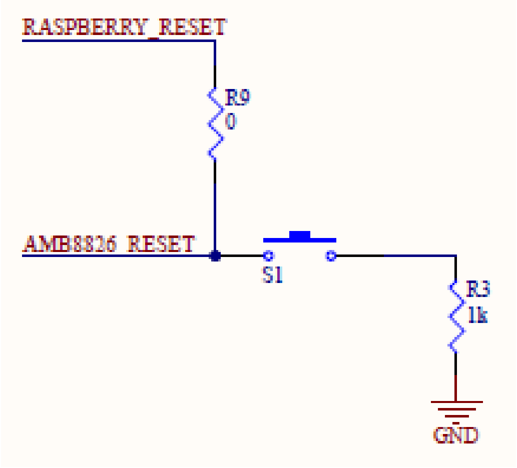


Figure 30: RESET pin connections

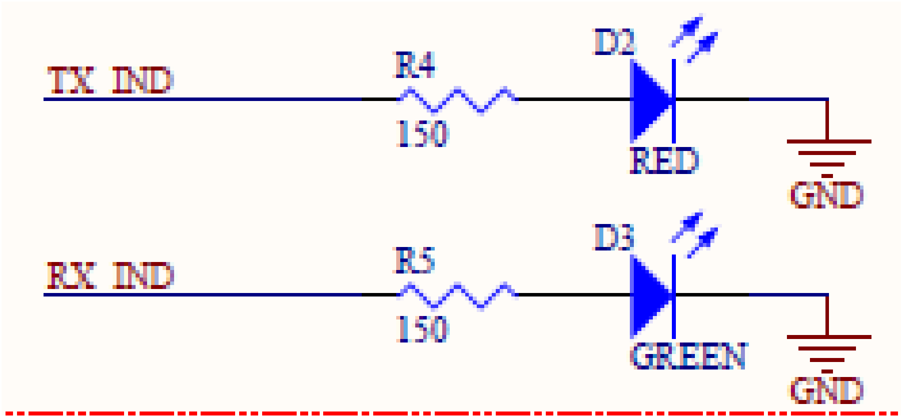


Figure 31: TX and RX leds

- POWER SUPPLY leds D4-D8 (BLUE) flashing when the development board is powered on. Based on the jumper position on JP17, these leds can be constantly turned on or driven by the PWM of the Raspberry Pi board (please refer to chapter 7.1).

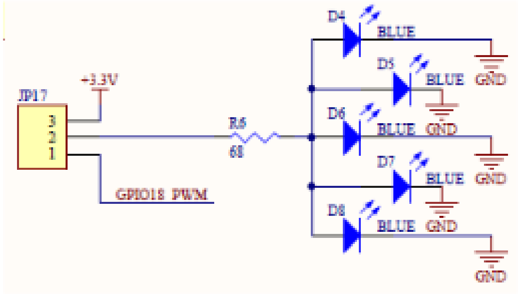


Figure 32: POWER SUPPLY leds

7.6 Schematics

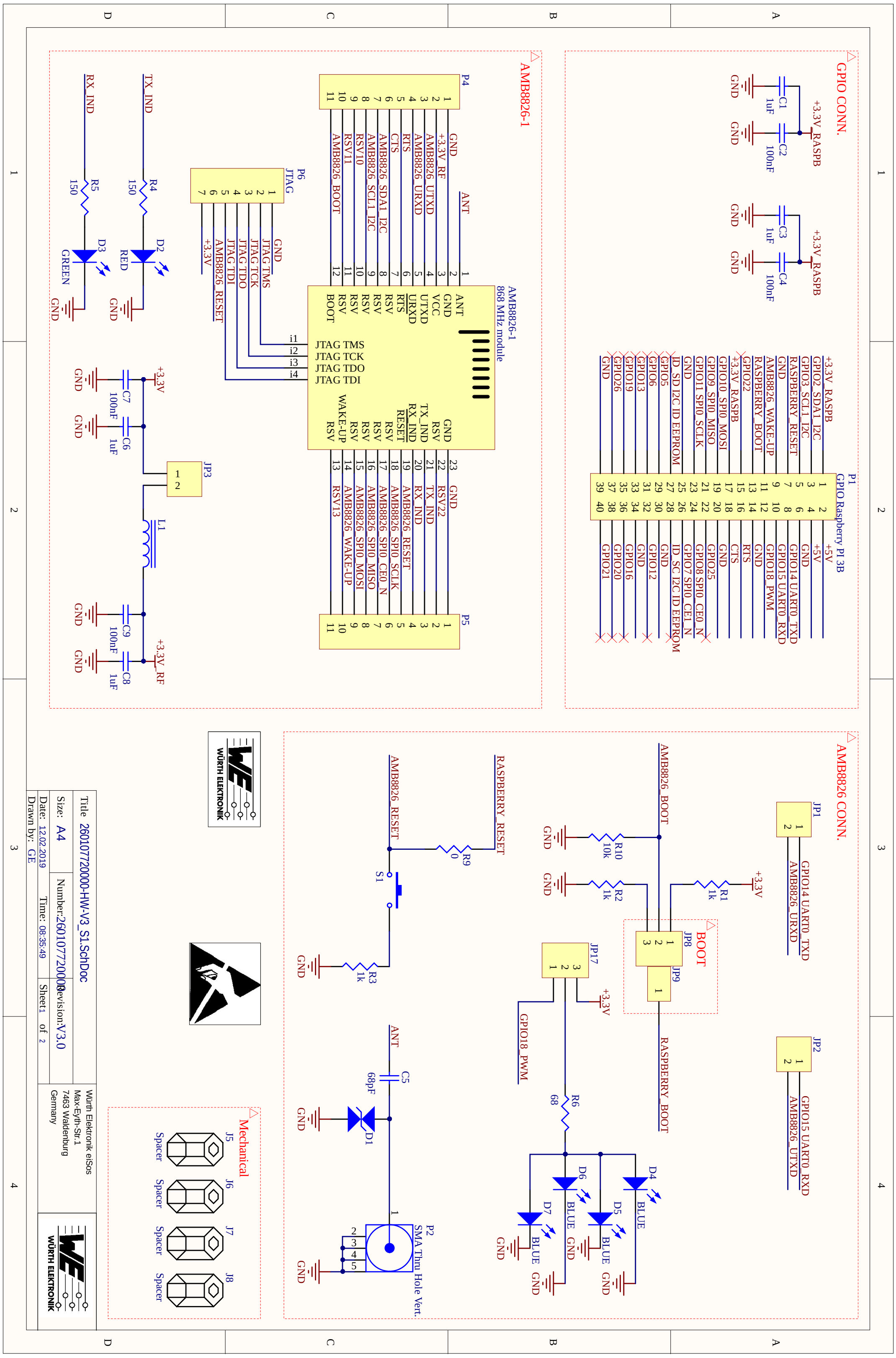


Figure 33: Schematic page 1 of 2

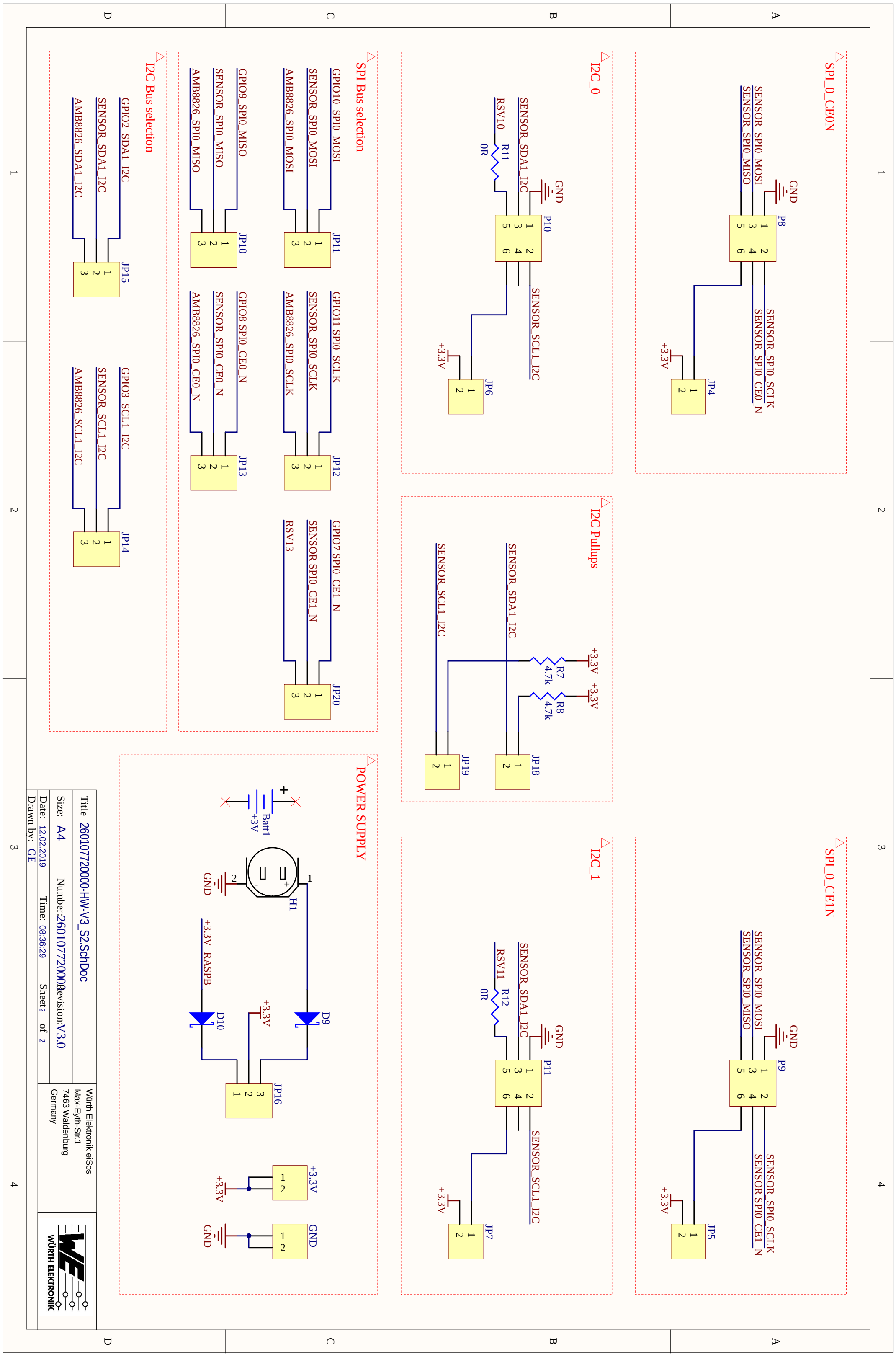


Figure 34: Schematic page 2 of 2

Designator	Value	Voltage	Tolerance	Description	Manufacturer	Manufacturer part number	Quantity
+3.3V, GND, JP3, JP4, JP5, JP6, JP7				Header Male Vertical 2X1	Wurth Electronics Inc.	61300211121	7
AMB8826-1		2.2-3.8V		Tarvos-III 868 MHz module with RF on pad	Wurth Electronics Inc.	Tarvos-III-1	1
C1, C3, C6, C8	1uF	6.3V	±20%	Capacitor SMD	Wurth Electronics Inc.	885012105006	4
C2, C4, C7, C9	100nF	10V	±10%	Capacitor SMD	Wurth Electronics Inc.	885012205018	4
C5	68pF	50V	±5%	Capacitor SMD	Wurth Electronics Inc.	885012005060	1
D1				TVS Diode	Murata Electronics North America	LXES15AAA1-133	1
D2				Diode LED	Wurth Electronics Inc.	150060SS75000	1
D3				Diode LED	Wurth Electronics Inc.	150060VS75000	1
D4, D5, D6, D7, D8				Diode LED	Kingbright	APT1608LVBC/D	5
D9, D10				Schottky Diode	Toshiba Semiconductor and Storage	CUS10S30,H3F	2
H1				Coin Battery Holder	Keystone Electronics	3034	1
J5, J6, J7, J8				Board spacer	Wurth Electronics Inc.	709670110	4
JP1, JP2, JP18, JP19				Header Male Vertical 2X1	Sullins Connector Solutions	GRPB021VWVN-RC	4
JP8				Header Male Vertical 3X1	Wurth Electronics Inc.	61300311121	1
JP9				Header Male Vertical 1X1	Wurth Electronics Inc.	61300111121	1
JP10, JP11, JP12, JP13, JP14, JP15, JP16, JP17				Header Male Vertical 3X1	Sullins Connector Solutions	GRPB031VWVN-RC	8
L1				Ferrite bead SMD	Wurth Electronics Inc.	7427927311	1
P1				Header, Female, Dual Row, 40-Pin, 2.54mm pitch, Pressfit, Dual row, Pin length 12.2mm	EPT	962-60202-12	1
P2				SMA RF Coaxial PCB Connector, Thru-Hole, Vertical Mount Plug, 50 Ohm Impedance	Samtec Inc.	SMA-J-P-H-ST-TH1	1
P4, P5				Header Male Vertical 11X1, 2.54mm pitch	Sullins Connector Solutions	PRPC011SAAN-RC	2
P6				Header Male Vertical 7X1, 2.54mm pitch	Wurth Electronics Inc.	61300711121	1
P8, P9				3x2 pins, 2.54mm Right Angle, Dual Socket Header WR-PHD	Wurth Electronics Inc.	613006243121	2
P10, P11				2x2 pins, 2.54mm Right Angle, Dual Socket Header WR-PHD	Wurth Electronics Inc.	613004243121	2
R1, R2, R3	1k		±5%	Resistor SMD	TE Connectivity Passive Product	CRG0402J1K0	3
R4, R5	150		±5%	Resistor SMD	Yageo	RC0402JR-07150RL	2
R6	68		±5%	Resistor SMD	Yageo	RC0402JR-0768RL	1
R7, R8	4.7k		±5%	Resistor SMD	Samsung Electro-Mechanics America, Inc.	RC1005J472CS	2
R9	0		Jumper	Resistor SMD	Yageo	RC0402JR-070RL	1
R10	10k		±5%	Resistor SMD	TE Connectivity Passive Product	CRG0402J10K/10	1
S1				SWITCH TACTILE SPST-NO	Wurth Electronics Inc.	430471025826	1

Table 2: Bill of material

8 Hardware history

Version 2.1 "Production"

- First production release.
- Amber PI packet contains the radio stick AMB8665 (Tarvos-II Plug).

Version 3.0 "Production"

- Sensor connectors have been updated. 4-pin I2C sensor interface was replaced by a 6-pin sensor interface.
- PCB color changed to red.
- In delivery state, the radio stick AMB8665 (Tarvos-II Plug) was replaced by the AMB8865 (Tarvos-III Plug).
  - The Tarvos-III Plug supports all radio profiles of the Tarvos-III radio module, which is mounted on the Amber PI. Thus radio profile 3 (long range mode) can be used in this package version.
  - Unlike Tarvos-II Plug, the Tarvos-III Plug uses a command interface on the UART. Thus the communication protocol on the COM port of the radio dongle has been modified. Applications using the Tarvos-II Plug from Amber PI version 2.1 need to be updated to the new communication interface of the Tarvos-III Plug. To do so, the Tarvos-III Plug driver, which is part of the Wireless Connectivity SDK has be used as a basis.

9 References

[1] "AMBER PI," 2017. [Online]. Available: *https://www.we-online.com/wireless-connectivity*.

[2] "Tarvos-III Manual," 01.04.2017. [Online]. Available: *https://www.we-online.com/wireless-connectivity*.

[3] "Tarvos-III Datasheet," 01.04.2017. [Online]. Available: *https://www.we-online.com/wireless-connectivity*.

[4] "HTS221," Datasheet Capacitive digital sensor for relative humidity and temperature, 01.04.2017. [Online]. Available: *www.st.com/en/mems-and-sensors/hts221.html*.

[5] "LPS22HB," Datasheet MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer, 01.04.2017. [Online]. Available: *www.st.com/en/mems-and-sensors/lps22hb.html*.

[6] "LIS2DW12," Datasheet MEMS digital output motion sensor:high-performance ultra-low-power 3-axis "femto" accelerometer, 01.04.2017. [Online]. Available: *www.st.com/en/mems-and-sensors/lis2dw12.html*.

[7] "LIS3DH," Datasheet MEMS digital output motion sensor ultra low-power high performance 3-axes "nano" accelerometer, 01.04.2017. [Online]. Available: *www.st.com/en/mems-and-sensors/lis3dh.html*.

[8] "Tarvos-II Manual," 01.05.2017. [Online]. Available: *https://www.we-online.com/wireless-connectivity*.

10 Regulatory compliance information

Pursuant to Article 1 (2.) of the EU directive 2014/53/EU, Article 1 (2.) the directive does not apply to equipment listed in Annex I (4.): custom-built evaluation kits destined for professionals to be used solely at research and development facilities for such purposes.

10.1 Exemption clause

Relevant regulation requirements are subject to change. Würth Elektronik eiSos does not guarantee the accuracy of the before mentioned information. Directives, technical standards, procedural descriptions and the like may be interpreted differently by the national authorities. Equally, the national laws and restrictions may vary with the country. In case of doubt or uncertainty, we recommend that you consult with the authorities or official certification organizations of the relevant countries. Würth Elektronik eiSos is exempt from any responsibilities or liabilities related to regulatory compliance.

Notwithstanding the above, Würth Elektronik eiSos makes no representations and warranties of any kind related to their accuracy, correctness, completeness and/or usability for customer applications. No responsibility is assumed for inaccuracies or incompleteness.

11 Important notes

The following conditions apply to all goods within the wireless connectivity product range of Würth Elektronik eiSos GmbH & Co. KG:

11.1 General customer responsibility

Some goods within the product range of Würth Elektronik eiSos GmbH & Co. KG contain statements regarding general suitability for certain application areas. These statements about suitability are based on our knowledge and experience of typical requirements concerning the areas, serve as general guidance and cannot be estimated as binding statements about the suitability for a customer application. The responsibility for the applicability and use in a particular customer design is always solely within the authority of the customer. Due to this fact, it is up to the customer to evaluate, where appropriate to investigate and to decide whether the device with the specific product characteristics described in the product specification is valid and suitable for the respective customer application or not. Accordingly, the customer is cautioned to verify that the documentation is current before placing orders.

11.2 Customer responsibility related to specific, in particular safety-relevant applications

It has to be clearly pointed out that the possibility of a malfunction of electronic components or failure before the end of the usual lifetime cannot be completely eliminated in the current state of the art, even if the products are operated within the range of the specifications. The same statement is valid for all software sourcecode and firmware parts contained in or used with or for products in the wireless connectivity and sensor product range of Würth Elektronik eiSos GmbH & Co. KG. In certain customer applications requiring a high level of safety and especially in customer applications in which the malfunction or failure of an electronic component could endanger human life or health, it must be ensured by most advanced technological aid of suitable design of the customer application that no injury or damage is caused to third parties in the event of malfunction or failure of an electronic component.

11.3 Best care and attention

Any product-specific data sheets, manuals, application notes, PCN's, warnings and cautions must be strictly observed in the most recent versions and matching to the products firmware revisions. This documents can be downloaded from the product specific sections on the wireless connectivity homepage.

11.4 Customer support for product specifications

Some products within the product range may contain substances, which are subject to restrictions in certain jurisdictions in order to serve specific technical requirements. Necessary information is available on request. In this case, the field sales engineer or the internal sales person in charge should be contacted who will be happy to support in this matter.

11.5 Product improvements

Due to constant product improvement, product specifications may change from time to time. As a standard reporting procedure of the Product Change Notification (PCN) according to the JEDEC-Standard, we inform about major changes. In case of further queries regarding the PCN, the field sales engineer, the internal sales person or the technical support team in charge should be contacted. The basic responsibility of the customer as per section 11.1 and 11.2 remains unaffected. All wireless connectivity module driver software ¨wireless connectivity SDK¨ and it's source codes as well as all PC software tools are not subject to the Product Change Notification information process.

11.6 Product life cycle

Due to technical progress and economical evaluation we also reserve the right to discontinue production and delivery of products. As a standard reporting procedure of the Product Termination Notification (PTN) according to the JEDEC-Standard we will inform at an early stage about inevitable product discontinuance. According to this, we cannot ensure that all products within our product range will always be available. Therefore, it needs to be verified with the field sales engineer or the internal sales person in charge about the current product availability expectancy before or when the product for application design-in disposal is considered. The approach named above does not apply in the case of individual agreements deviating from the foregoing for customer-specific products.

11.7 Property rights

All the rights for contractual products produced by Würth Elektronik eiSos GmbH & Co. KG on the basis of ideas, development contracts as well as models or templates that are subject to copyright, patent or commercial protection supplied to the customer will remain with Würth Elektronik eiSos GmbH & Co. KG. Würth Elektronik eiSos GmbH & Co. KG does not warrant or represent that any license, either expressed or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, application, or process in which Würth Elektronik eiSos GmbH & Co. KG components or services are used.

11.8 General terms and conditions

Unless otherwise agreed in individual contracts, all orders are subject to the current version of the "General Terms and Conditions of Würth Elektronik eiSos Group", last version available at *www.we-online.com*.

12 Legal notice

12.1 Exclusion of liability

Würth Elektronik eiSos GmbH & Co. KG considers the information in this document to be correct at the time of publication. However, Würth Elektronik eiSos GmbH & Co. KG reserves the right to modify the information such as technical specifications or functions of its products or discontinue the production of these products or the support of one of these products without any written announcement or notification to customers. The customer must make sure that the information used corresponds to the latest published information. Würth Elektronik eiSos GmbH & Co. KG does not assume any liability for the use of its products. Würth Elektronik eiSos GmbH & Co. KG does not grant licenses for its patent rights or for any other of its intellectual property rights or third-party rights.

Notwithstanding anything above, Würth Elektronik eiSos GmbH & Co. KG makes no representations and/or warranties of any kind for the provided information related to their accuracy, correctness, completeness, usage of the products and/or usability for customer applications. Information published by Würth Elektronik eiSos GmbH & Co. KG regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof.

12.2 Suitability in customer applications

The customer bears the responsibility for compliance of systems or units, in which Würth Elektronik eiSos GmbH & Co. KG products are integrated, with applicable legal regulations. Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of Würth Elektronik eiSos GmbH & Co. KG components in its applications, notwithstanding any applications-related in-formation or support that may be provided by Würth Elektronik eiSos GmbH & Co. KG. Customer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences lessen the likelihood of failures that might cause harm and take appropriate remedial actions. The customer will fully indemnify Würth Elektronik eiSos GmbH & Co. KGand its representatives against any damages arising out of the use of any Würth Elektronik eiSos GmbH & Co. KG components in safety-critical applications.

12.3 Trademarks

AMBER wireless is a registered trademark of Würth Elektronik eiSos GmbH & Co. KG. All other trademarks, registered trademarks, and product names are the exclusive property of the respective owners.

12.4 Usage restriction

Würth Elektronik eiSos GmbH & Co. KG products have been designed and developed for usage in general electronic equipment only. This product is not authorized for use in equipment where a higher safety standard and reliability standard is especially required or where a failure of the product is reasonably expected to cause severe personal injury or death, unless the parties have executed an agreement specifically governing such use. Moreover, Würth Elektronik eiSos GmbH & Co. KG products are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation (automotive control, train control, ship control), transportation signal, disaster prevention, medical, public information network etc. Würth Elektronik eiSos GmbH & Co. KG must be informed about the intent of such usage before the design-in stage. In addition, sufficient reliability evaluation checks for safety must be performed on every electronic component, which is used in electrical circuits that require high safety and reliability function or performance. By using Würth Elektronik eiSos GmbH & Co. KG products, the customer agrees to these terms and conditions.

13 License terms

This License Terms will take effect upon the purchase and usage of the Würth Elektronik eiSos GmbH & Co. KG wireless connectivity products. You hereby agree that this license terms is applicable to the product and the incorporated software, firmware and source codes (collectively, "Software") made available by Würth Elektronik eiSos in any form, including but not limited to binary, executable or source code form.
The software included in any Würth Elektronik eiSos wireless connectivity product is purchased to you on the condition that you accept the terms and conditions of this license terms. You agree to comply with all provisions under this license terms.

13.1 Limited license

Würth Elektronik eiSos hereby grants you a limited, non-exclusive, non-transferable and royalty-free license to use the software and under the conditions that will be set forth in this license terms. You are free to use the provided Software only in connection with one of the products from Würth Elektronik eiSos to the extent described in this license terms. You are entitled to change or alter the source code for the sole purpose of creating an application embedding the Würth Elektronik eiSos wireless connectivity product. The transfer of the source code to third parties is allowed to the sole extent that the source code is used by such third parties in connection with our product or another hardware provided by Würth Elektronik eiSos under strict adherence of this license terms. Würth Elektronik eiSos will not assume any liability for the usage of the incorporated software and the source code. You are not entitled to transfer the source code in any form to third parties without prior written consent of Würth Elektronik eiSos.
You are not allowed to reproduce, translate, reverse engineer, decompile, disassemble or create derivative works of the incorporated Software and the source code in whole or in part. No more extensive rights to use and exploit the products are granted to you.

13.2 Usage and obligations

The responsibility for the applicability and use of the Würth Elektronik eiSos wireless connectivity product with the incorporated Firmware in a particular customer design is always solely within the authority of the customer. Due to this fact, it is up to you to evaluate and investigate, where appropriate, and to decide whether the device with the specific product characteristics described in the product specification is valid and suitable for your respective application or not.
You are responsible for using the Würth Elektronik eiSos wireless connectivity product with the incorporated Firmware in compliance with all applicable product liability and product safety laws. You acknowledge to minimize the risk of loss and harm to individuals and bear the risk for failure leading to personal injury or death due to your usage of the product.
Würth Elektronik eiSos' products with the incorporated Firmware are not authorized for use in safety-critical applications, or where a failure of the product is reasonably expected to cause severe personal injury or death. Moreover, Würth Elektronik eiSos' products with the incorporated Firmware are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation (automotive control, train control, ship control), transportation signal, disaster prevention, medical, public information network etc. You shall inform Würth Elektronik eiSos about the intent of such usage before

design-in stage. In certain customer applications requiring a very high level of safety and in which the malfunction or failure of an electronic component could endanger human life or health, you must ensure to have all necessary expertise in the safety and regulatory ramifications of your applications. You acknowledge and agree that you are solely responsible for all legal, regulatory and safety-related requirements concerning your products and any use of Würth Elektronik eiSos' products with the incorporated Firmware in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by Würth Elektronik eiSos. YOU SHALL INDEMNIFY WÜRTH ELEKTRONIK EISOS AGAINST ANY DAMAGES ARISING OUT OF THE USE OF WÜRTH ELEKTRONIK EISOS' PRODUCTS WITH THE INCORPORATED FIRMWARE IN SUCH SAFETY-CRITICAL APPLICATIONS.

13.3 Ownership

The incorporated Firmware created by Würth Elektronik eiSos is and will remain the exclusive property of Würth Elektronik eiSos.

13.4 Firmware update(s)

You have the opportunity to request the current and actual Firmware for a bought wireless connectivity Product within the time of warranty. However, Würth Elektronik eiSos has no obligation to update a modules firmware in their production facilities, but can offer this as a service on request. The upload of firmware updates falls within your responsibility, e.g. via ACC or another software for firmware updates. Firmware updates will not be communicated automatically. It is within your responsibility to check the current version of a firmware in the latest version of the product manual on our website. The revision table in the product manual provides all necessary information about firmware updates. There is no right to be provided with binary files, so called "Firmware images", those could be flashed through JTAG, SWD, Spi-Bi-Wire, SPI or similar interfaces.

13.5 Disclaimer of warranty

THE FIRMWARE IS PROVIDED "AS IS". YOU ACKNOWLEDGE THAT WÜRTH ELEKTRONIK EISOS MAKES NO REPRESENTATIONS AND WARRANTIES OF ANY KIND RELATED TO, BUT NOT LIMITED TO THE NON-INFRINGEMENT OF THIRD PARTIES' INTELLECTUAL PROPERTY RIGHTS OR THE MERCHANTABILITY OR FITNESS FOR YOUR INTENDED PURPOSE OR USAGE. WÜRTH ELEKTRONIK EISOS DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT RELATING TO ANY COMBINATION, MACHINE, OR PROCESS IN WHICH THE WÜRTH ELEKTRONIK EISOS' PRODUCT WITH THE INCORPORATED FIRMWARE IS USED. INFORMATION PUBLISHED BY WÜRTH ELEKTRONIK EISOS REGARDING THIRD-PARTY PRODUCTS OR SERVICES DOES NOT CONSTITUTE A LICENSE FROM WÜRTH ELEKTRONIK EISOS TO USE SUCH PRODUCTS OR SERVICES OR A WARRANTY OR ENDORSEMENT THEREOF.

13.6 Limitation of liability

Any liability not expressly provided by Würth Elektronik eiSos shall be disclaimed.
You agree to hold us harmless from any third-party claims related to your usage of the Würth Elektronik eiSos' products with the incorporated Firmware, software and source code. Würth Elektronik eiSos disclaims any liability for any alteration, development created by you or your customers as well as for any combination with other products.

13.7 Applicable law and jurisdiction

Applicable law to this license terms shall be the laws of the Federal Republic of Germany. Any dispute, claim or controversy arising out of or relating to this license terms shall be resolved and finally settled by the court competent for the location of Würth Elektronik eiSos' registered office.

13.8 Severability clause

If a provision of this license terms is or becomes invalid, unenforceable or null and void, this shall not affect the remaining provisions of the terms. The parties shall replace any such provisions with new valid provisions that most closely approximate the purpose of the terms.

13.9 Miscellaneous

Würth Elektronik eiSos reserves the right at any time to change this terms at its own discretion. It is your responsibility to check at Würth Elektronik eiSos homepage for any updates. Your continued usage of the products will be deemed as the acceptance of the change.
We recommend you to be updated about the status of new firmware and software, which is available on our website or in our data sheet and manual, and to implement new software in your device where appropriate.
By ordering a wireless connectivity product, you accept this license terms in all terms.

List of Figures

List of Tables

WÜRTH ELEKTRONIK

more than you expect

Internet of Things

Monitoring & Control

Automated Meter Reading

Contact:
Würth Elektronik eiSos GmbH & Co. KG
Division Wireless Connectivity & Sensors

Rudi-Schillings-Str. 31
54296 Trier
Germany

Tel.: +49 651 99355-0
Fax.: +49 651 99355-69
www.we-online.com/wireless-connectivity