Quick Start Guide

Required Parts

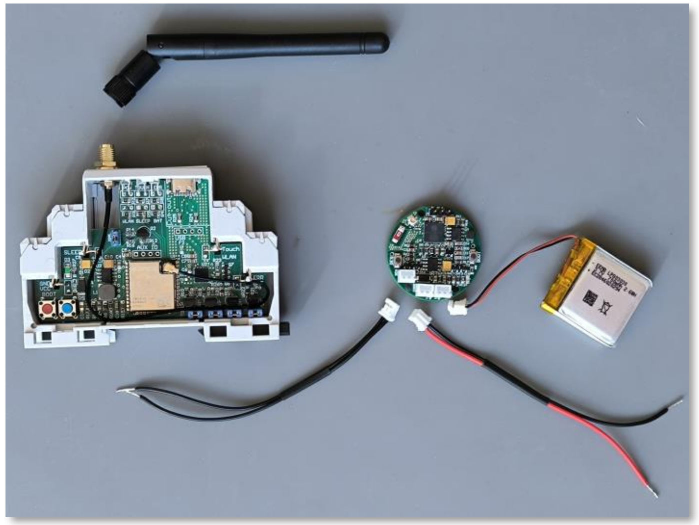
- Assembled **Basestation/Server PCB version 3.0**
- External **WLAN Antenna**
- Assembled **Client PCB version 2.0** (setup for on board antenna)
- **LiPo battery** with proper connector (JST PH2P BU). ATTENTION: the polarity on the connector is not normed, double check the polarity before you connect your own LiPo.
- **Two wires with a proper connector** (JST PH2P BU) to serve as sensor switch replacement

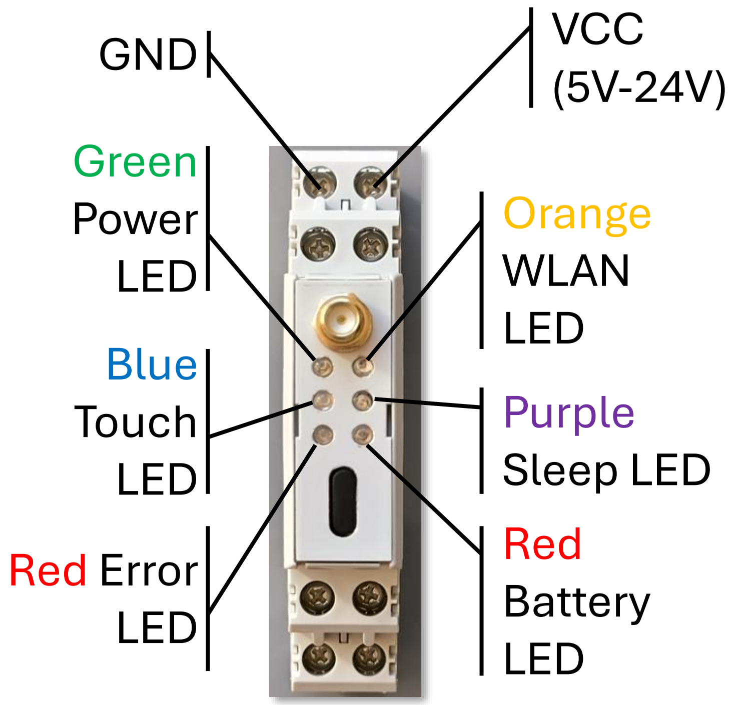
Basestation/Server Setup

- Mount the antenna to the Basestation/Server
- Connect a Power Supply to VCC (5-24V) and GND
- The green power LED should be on
- The orange WLAN LED is flashing, indicating that the server is looking for a client

Sensor/Client Setup

- Connect the Sensor cable to the "SEN" socket on the round Sensor PCB and make a short circuit between the to wires of the sensor cable (the sensor is normally closed, NC).
- Connect the battery to the "BAT" socket on the round Sensor PCB.
- Now the RGB LED on the back of the Sensor PCB should start fading from 0% to 100% in white (if not, open the sensor wires for a short moment, indicating a wake up).
- After reaching 100%, the white LED also flashes, indicating the search for a Basestation.



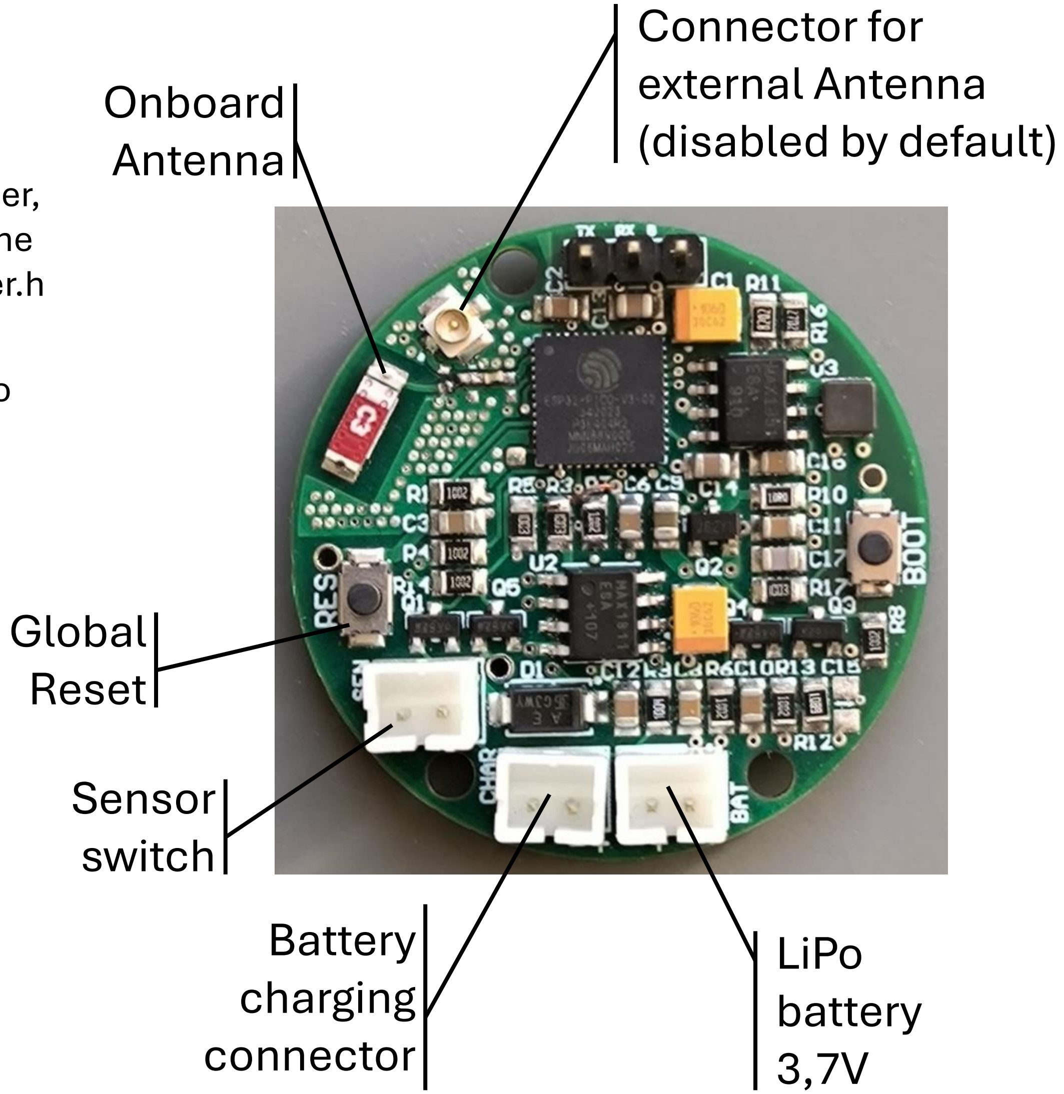


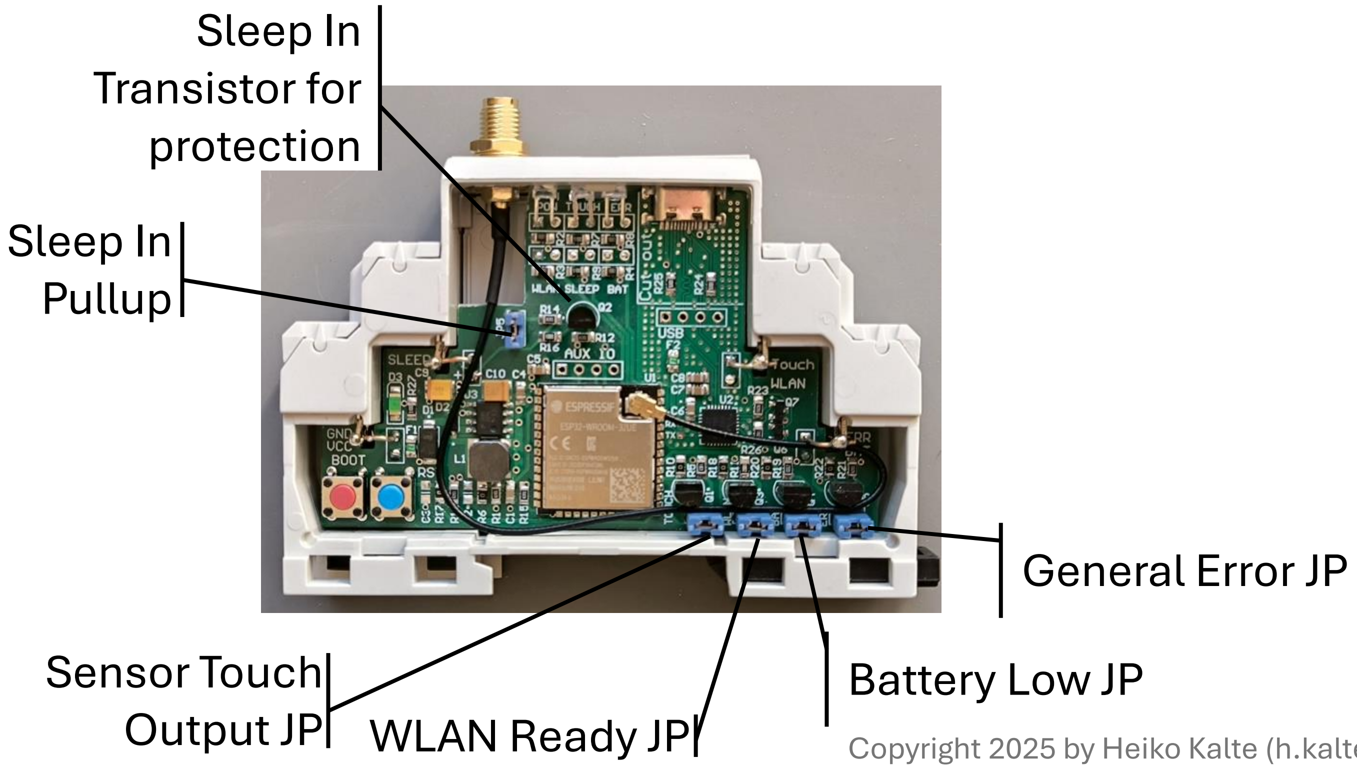
Copyright 2025 by Heiko Kalte (h.kalte@gmx.de)

Quick Start Guide (continued)

Operation

- After a short period of time the Basestation and the Sensor should find each other, indicated by a steady orange LED on the server and a steady white RGB LED on the Sensor (if the sensor LED is not steady white, look up the colors at the 3D-Header.h for more information)
- Now, you can simulate a sensor touch by opening the Sensor cable connected to the "SEN" socket. When the sensor is open, the RGB LED is blue as well as the Touch LED at the Basestation.


Onboard Antenna
Connector for external Antenna (disabled by default)
Global Reset
Sensor switch
Battery charging connector
LiPo battery 3,7V



Copyright 2025 by Heiko Kalte (h.kalte@gmx.de)

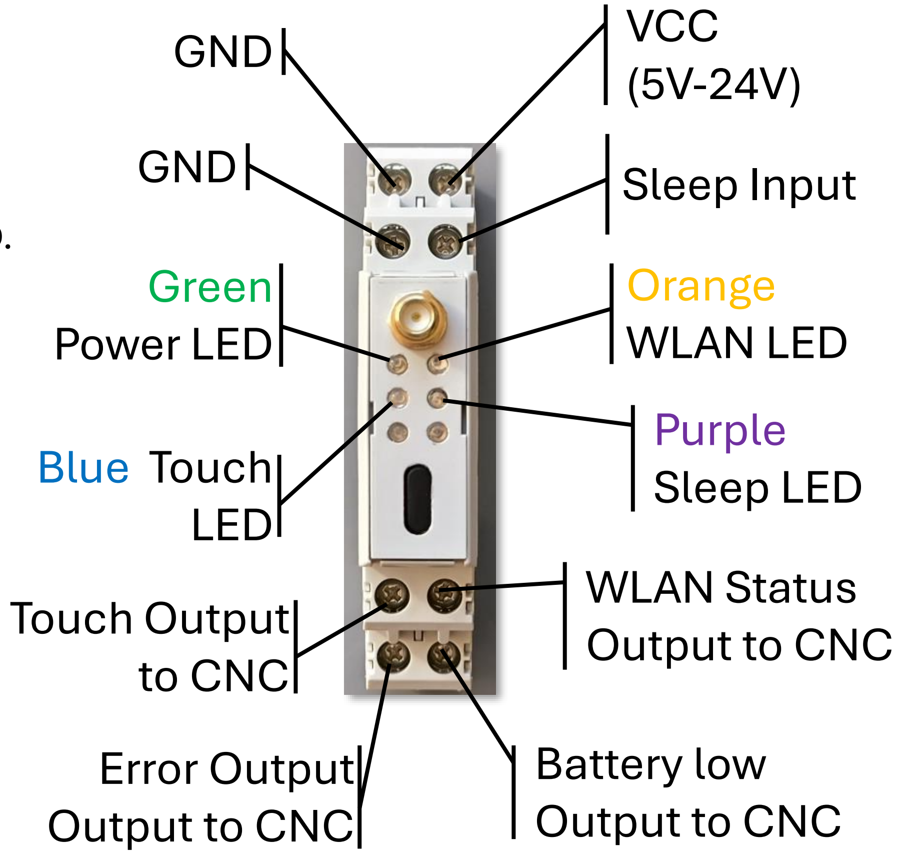
Possible Next Steps

Sending the Sensor asleep and waking up

- Precondition: the sensor and Basestation are up and running (steady orange LED at Basestation) and steady white LED on the sensor. If the sensor LED is blue, make a short circuit at the “SEN” wires to indicate that currently no workpiece is touched.
- For sending the Sensor asleep connect the Sleep input of the Basestation to GND and watch the purple LED on the Basestation, while the white RBG LED of the Sensor is fading from 100% down to 0%. Sensor is now asleep and Power consumption is at a minimum.
- The Basestation will go back the searching for a Sensor, indicated by a flashing orange LED.
- Now you can wake up the Sensor by opening the Sensor wire for a short moment and the startup will take place again.

Watch the Electrical Outputs of the Basestation

- Before you connect the Basestation to your CNC controller watch the electrical Output the Basestation.
- Make a short circuit by connecting/bridging all 4 Jumpers in the Basestation at the bottom right corner (if not already the case). That sets all 4 outputs to “normal” mode instead of “open collector” mode.
- Connect a multimeter to the “TOUCH” output of the Basestation (use GND as reference).
- Start up the system and bring it into normal operation (Basestation LEDs green and steady orange and Sensor is steady white).
- Now open the “SEN” wires and watch the blue LEDs while the Voltage of the multimeter should change from 0 to VCC (supply voltage you connected to Basestation) or the other way around, depending on the polarity.



Copyright 2025 by Heiko Kalte (h.kalte@gmx.de)

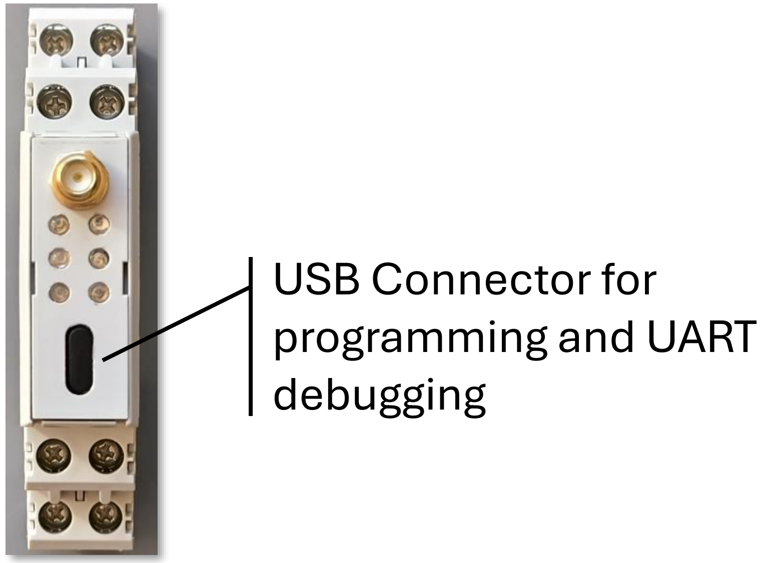
Possible Next Steps (continued)

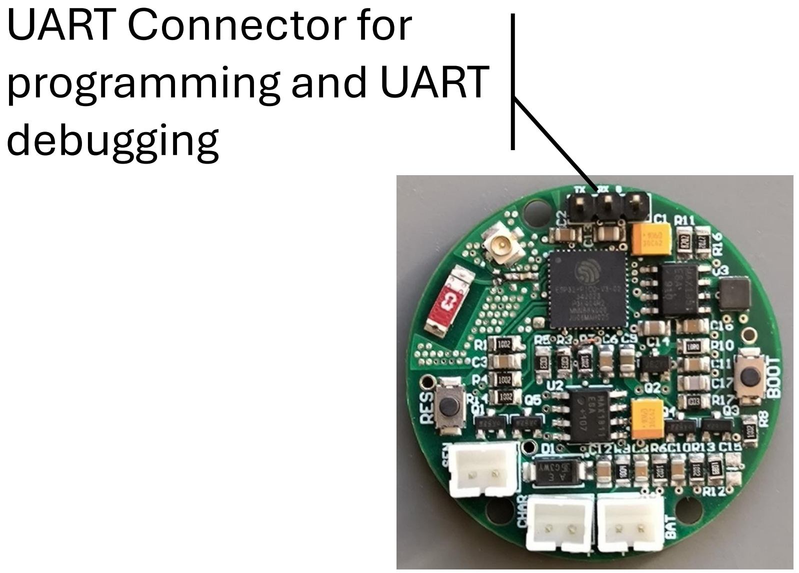
Watch the Electrical Outputs of the Basestation (continued)

- The output polarity of all 4 outputs depends on the settings in 3D-Header.h (SERVER_WLAN_OUT_POLARITY, SERVER_TOUCH_OUT_POLARITY, SERVER_ERROR_OUT_POLARITY, SERVER_BAT_ALM_OUT_POLARITY)

Watching the Debug Outputs

- Assuming that the initial software on the Basestation and on the Sensor is compiled with "#define DEBUG" in the 3D-Header.h you will be able to watch all stuff going on, on the Basestation and on the Sensor by a UART viewer.
- On the Basestation, just connect a USB cable from the Basestation to a computer and setup a UART viewer to the correct USB port with setting up the BAUD rate to 9600 (can be changed by BAUD_RATE in 3D-Header.h). Possible UART viewers are e.g. the Arduino IDE or simply use PuTTY.
- For the Sensor/Client it is a bit more complicated, because due to space limitations there is no USB-UART converter on board. You need an external USB to UART converter. I use this one, but there are hundreds around. Connect the GND and TD and RX pins between the Sensor PCB and the external USB-UART converter. When connecting, do not forget to cross the pins, RX of the Sensor must be connected to TX of the converter etc.
- Neither the Basestation nor the Sensor/Client hardware can be powered by the UART or USB, that means you have to connect the normal power supply (e.g. Lipo for the Sensor).
- Now watch one or both serial/UART outputs and make yourself familiar with what is going on. It can be helpful to reset both hardware to see the startup.





Copyright 2025 by Heiko Kalte (h.kalte@gmx.de)

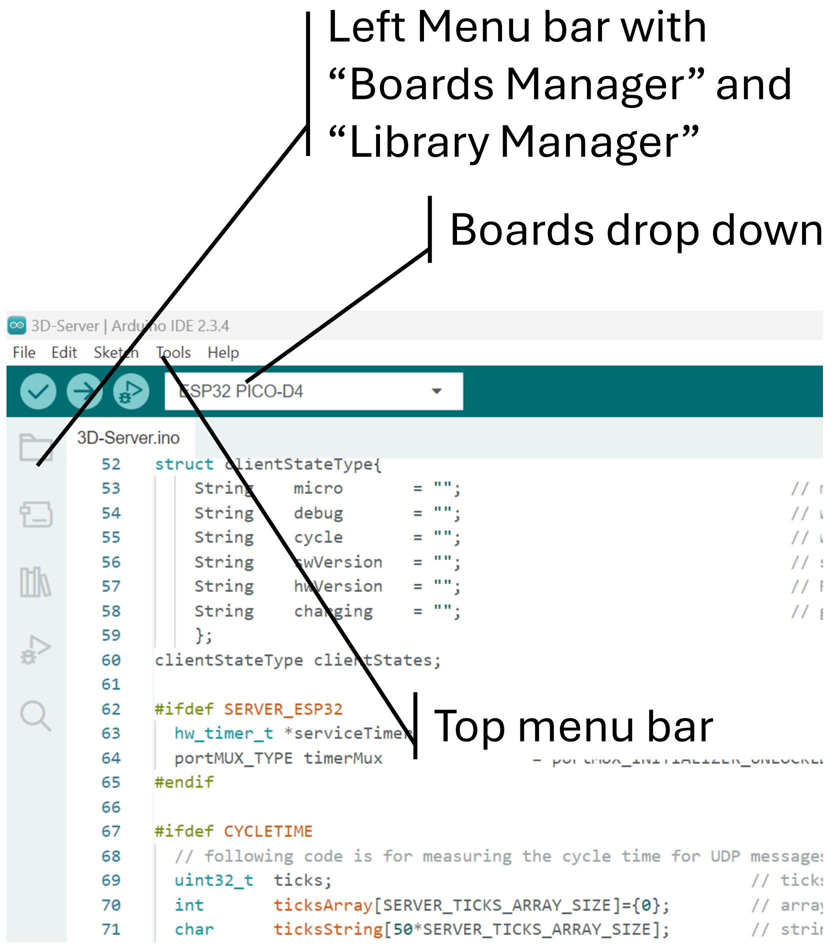
Programming the ESP32s

Source Code and Github

- Clone the github repository: https://github.com/HeikoKa/Wireless-3D-Touch-Probe-Sensor or just grab the three files /src/3D-Header.h, /src/3D-Client/3D-Client.ino and /src/3D-Server/ 3D-Server.ino
- Set the correct (absolute) path to the 3D-Header.h in the two *.ino files (to my knowledge the IDE can only use absolute paths)
- Have a look into the 3D-Header.h file, e.g. to globally enable/disable Debug Mode (sending all sorts of Text via UART). All parameters are stored in this file, do not change the I/O pin numbers when using the "official" PCBs
- It is recommended to not change anything during the first compile and download run. Only gradually start changing things.

Arduino Development Environment

- Download the Arduino IDE from: https://docs.arduino.cc/software/ide/
- In the left menu bar select the "Boards Manager", search for "esp32" by Espressif Systems and install that package
- Go to Libraray Manager in the left menu bar and search for "Adafruit NeoPixel" by Adafruit and install the library (it is for the RGB LED of the Sensor/Client.
- Open one or both of the *.ino files.
- In the boards dropdown (see picture) select "ESP32 PICO-D4" for the Client/Sensor PCB and "ESP32-WROOM-DA Module" for the Basestation/Server. DO NOT MIX THAT UP, ALWAYS DOUBLE CHECK. When wrong, in the best case it shows weird behavior in the worst case the PCB is damaged.


Left Menu bar with "Boards Manager" and "Library Manager"
Boards drop down
Top menu bar
3D-Server | Arduino IDE 2.3.4
File Edit Sketch Tools Help
ESP32 PICO-D4
3D-Server.ino

```
struct clientStateType{
    String    micro       = "";                                      // r
    String    debug       = "";                                      // w
    String    cycle       = "";                                      // w
    String    swVersion   = "";                                      // s
    String    hwVersion   = "";                                      // F
    String    charging    = "";                                      // g
    };
clientStateType clientStates;

#ifdef SERVER_ESP32
  hw_timer_t *serviceTimer
  portMUX_TYPE timerMux
#endif

#ifdef CYCLETIME
  // following code is for measuring the cycle time for UDP message
  uint32_t  ticks;                                               // ticks
  int       ticksArray[SERVER_TICKS_ARRAY_SIZE]={0};             // array
  char      ticksString[50*SERVER_TICKS_ARRAY_SIZE];             // strin
```

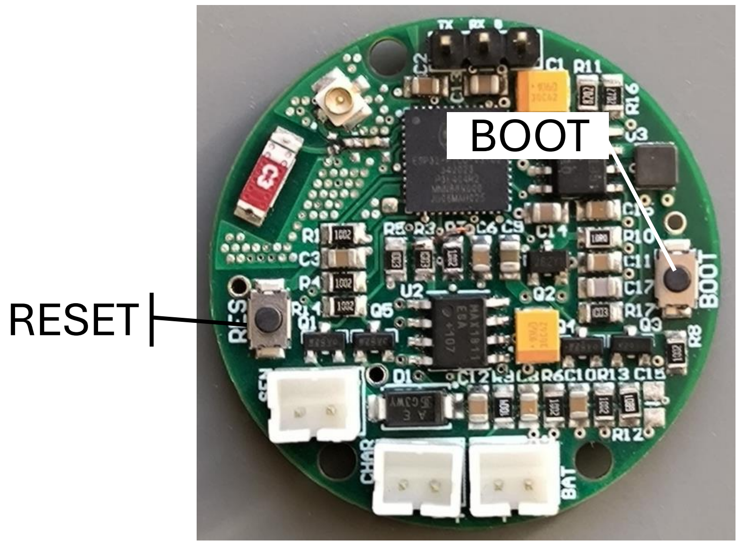
Programming the ESP32s (continued)

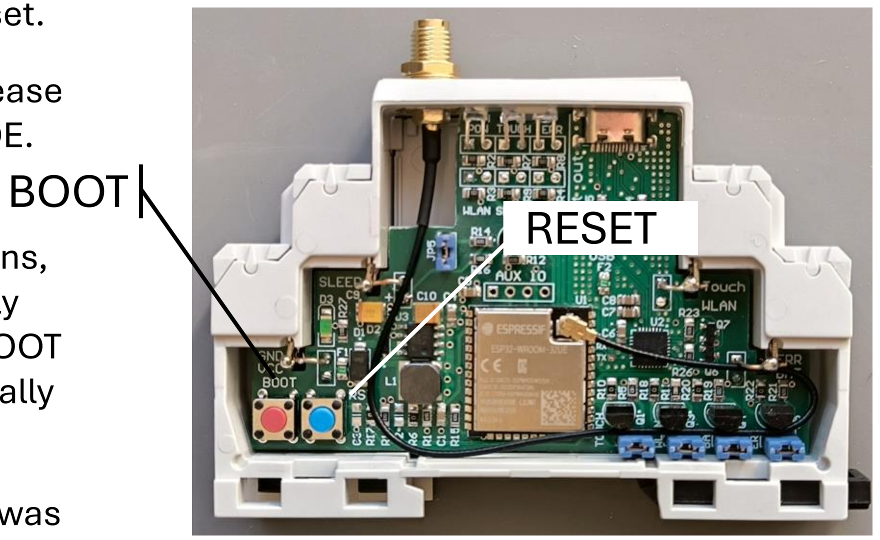
Arduino Development Environment (continued)

- Connect the PCBs to the computer by a proper USB cable. Be aware that some USB cables are just for Charging purposes and do not have any data wires.
- In the Arduino top menu select "Tools" → Port and select the right USB port. If not clear, just remove the cable and re-plug it again and see, which port is dis-appearing and re-appearing Do that for both PCBs if you like to program both.
- Connect the USP cable to the PCBs as described in the "Watching the Debug Outputs" section.

PCBs

- ESP32 microcontrollers must be put in "programming mode" before the IDE downloads the compiled code. This is done by a high voltage level on the IO0 during startup or after reset.
- For the Client pcb this means: press BOOT button on the PCB, press RESET Button, release the RESET Button and Release the BOOT button. Now press download in the Arduino IDE. Repeat this procedure every time you want to program the PCB.
- For the Server/Basestation pcb you do not need to care about the Reset and Boot buttons, because a circuit handles that. However, sometimes the timing seems not to fit properly and the PCB does not get into programming mode. If so, just help out by pressing the BOOT button during the Arduino compile and download process (the reset is down automatically before the downoad).#
- In the top menu switch to Tools → Serial Monitor to watch the debug outputs if DEBUG was enabled.





Copyright 2025 by Heiko Kalte (h.kalte@gmx.de)