



International Partnership on Innovation
SAMS - Smart Apiculture Management Services

Deliverable N°3.5

**Manual on
HIVE Construction and
Operation**

Work package N° 3 – HIVE System

Horizon 2020 (H2020-ICT-39-2017)

Project N°780755














This project has received funding from the European Union's Horizon 2020 research and innovation programme under **grant agreement N° 780755**. The sole responsibility for the content of this document lies with the authors. It does not necessarily reflect the opinion of the EU.

Project information		
Lead partner for the deliverable	University of Kassel, Faculty of Organic Agricultural Sciences, Agricultural and Biosystems Engineering. Head of Department and Project lead: Prof. Dr. Oliver Hensel	
Document type	Deliverable	
Dissemination level	Public	
Date and status	31.08.2020	submitted 31.08.2020
Author(s)	Sascha Fiedler, Dr. Sascha Kirchner	
Reviewer(s)	GIZ	

This document is issued by the consortium formed for the implementation of the SAMS project under Grant Agreement N° 780755.

SAMS consortium partners

Logo	Partner name	Short	Country
 Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ) GmbH	Deutsche Gesellschaft für internationale Zusammen-arbeit (GIZ) GmbH (Coordinator)	GIZ	Germany
 UNIKASSEL VERSITÄT	University of Kassel	UNIKAS	Germany
 KARI-FRANZENS-UNIVERSITÄT GRAZ UNIVERSITY OF GRAZ 	University of Graz (Institute for Biology)	UNIGRA	Austria
 Latvia University of Life Sciences and Technologies	Latvia University of Life Sciences and Technologies	UNILV	Latvia
 iceaddis	ICEADDIS – IT-Consultancy PLC	ICEADDIS	Ethiopia
  IQQO Oromia Agricultural Research Institute	Oromia Agricultural Research Institute, Holeta Bee Research Center	HOLETA	Ethiopia
 Universitas Padjadjaran	University Padjadjaran	UNPAD	Indonesia
  PRIMARY TRAINING & CONSULTING	Commanditaire Vennootschap (CV.) Primary Indonesia	CV.PI	Indonesia

List of Abbreviations

A/D	Analog/ digital
AC	Alternating current
ADC	Analog Digital Converter
ARM	Advanced RISC Machine
DAT	Data
DC	Direct current
DSS	Decision Support System
DW	Data Warehouse
ET	Ethiopia
FFT	Fast Fourier Transformed
GND	Ground
GPIO	General-purpose input/output
GSM	Global System for Mobile Communication
HI-FI	High fidelity
ICT	Information and Communication Technology
IDN	Indonesia
IP	Internet Protocol
LAN	Local area network
LED	Light-emitting diode
MU	Monitoring unit
PCB	Printed Circuit Board
PWM	Pulse Width Modulation
RAM	Random-access memory
SD	Secure Digital
SMS	Short message service
SDG	Sustainable Development Goals
SPL	Sound pressure level
VCC	Common Collector Voltage
VGA	Video Graphics Array

Summary of the project

SAMS is a service offer for beekeepers that allows active monitoring and remote sensing of bee colonies by an appropriate and adapted ICT solution. This system supports the beekeeper in ensuring bee health and bee productivity, since bees play a key role in the preservation of our ecosystem, the global fight against hunger and in ensuring our existence. The high potentials to foster sustainable development in different sectors of the partner regions are they are often used inefficient.

Three continents - three scenarios

(1) In Europe, consumption and trading of honey products are increasing whereas the production is stagnating. Beside honey production, pollination services are less developed. Nevertheless, within the EU 35% of human food consumption depend directly or indirectly on pollination activities.

(2) In Ethiopia, beekeepers have a limited access to modern beehive equipment and bee management systems. Due to these constraints, the apicultural sector is far behind his potential.

(3) The apiculture sector in Indonesia is developing slowly and beekeeping is not a priority in the governmental program. These aspects lead to a low beekeeper rate, a low rate of professional processing of bee products, support and marketing and a lack of professional interconnection with bee products processing companies.

Based on the User Centered Design the core activities of SAMS include the development of marketable SAMS Business Services, the adaption of a hive monitoring system for local needs and usability as well as the adaption of a Decision Support System (DSS) based on an open source system. As a key factor of success SAMS uses a multi stakeholder approach on an international and national level to foster the involvement and active participation of beekeepers and all relevant stakeholders along the whole value chain of bees.

The aim of SAMS is to:

- enhance international cooperation of ICT and sustainable agriculture between EU and developing countries in pursuit of the EU commitment to the UN Sustainable Development Goal (SDG N°2) “End hunger, achieve food security and improved nutrition and promote sustainable agriculture”
- increases production of bee products
- creates jobs (particularly youths/ women)
- triggers investments and establishes knowledge exchange through networks..

Project objectives

The overall objective of SAMS is to strengthen international cooperation of the EU with developing countries in ICT, concentrating on the field of sustainable agriculture as a vehicle for rural areas. The SAMS Project aims to develop and refine an open source remote sensing technology and user interaction interface to support small-hold beekeepers in managing and

monitoring the health and productivity in their own bee colonies. Highlighted will be especially the production of bee products and the strengthening of resilience to environmental factors.

- Specific objectives to achieve the aim:
- Addressing requirements of communities and stakeholder
- Adapted monitoring and support technology
- Bee related partnership and cooperation
- International and interregional knowledge and technology transfer
- Training and behavioral response
- Implementation SAMS Business cooperation

Contents

SAMS consortium partners.....	2
Summary of the project	4
Project objectives	4
Contents.....	6
1. Introduction	10
1.1 Function principles	10
1.2 Photovoltaic System.....	11
2. Hardware.....	12
2.1 Components.....	12
2.2 Circuit plan and Printed Circuit Board.....	13
2.3 Case and Cables.....	15
2.4 Sensor frame and placement	17
2.5 Scale unit	18
2.6 WiFi Router adaption.....	20
2.7 Witty Pi power management.....	21
3. Software.....	22
3.1 Image installation	22
3.2 WiFi Setup	23
3.3 Raspberry Pi Setup	23
3.4 Witty Pi Setup.....	25
3.5 SAMS Software Calibration	27
3.6 Features.....	28
3.7 Process.....	28
3.8 Automatic error control	31
4. Data Warehouse	32
4.1 Setup	32
4.2 Online configuration	32
4.3 Report	33
5. Protocol	34
Annexes	35

List of tables / figures

Figure 1: Top view of a ready to print circuit layout for the SAMS HIVE Monitoring System.....	12
Figure 2: PCB female header	12
Figure 3: Circuit plan and wiring	13
Figure 4: Screw terminal blocks (blue)	13
Figure 5: RGB LED Cathode Pinout (www.circuitbread.com).....	13
Figure 6: CAD Sketch of SAMS HIVE Case.....	14
Figure 7: 3D printed SAMS HIVE Case.....	14
Figure 8: Opened 3D printed SAMS HIVE Case with PCB and components	14
Figure 9: 14-conduct flat cable with IDC connection	15
Figure 10: D-Sub 9-Pin Plug for Sensorframe connection	15
Figure 11: 9-Pin D-Sub pinout (backside)	15
Figure 12: DC power plug.....	15
Figure 13: SAMS HIVE Monitoring System with sensor	15
Figure 14: Placement of SAMS HIVE System in common beehive.	16
Figure 15: CAD sketch of 3D printable SAMS Sensor case	16
Figure 16: Sensor case installed in regular brood frame	17
Figure 17: Sensor case installed in regular brood frame connected with flat cable.....	17
Figure 18: Measurements of metal plate and spacer for scale setup	17
Figure 19: Measurements and Setup of Load cell (bosche.eu)	18
Figure 20: Scale unit with load cells H30A	18
Figure 21: WiFi Router access	19
Figure 22: WiFi Router adaption for auto-switch-ON function (left); Access data (top right)	19
Figure 23: Witty Pi Power Connection (red and black cable connected to DC/DC converter)	20
Figure 24: Optional Witty Pi Power Connection for detailed power monitoring	20
Figure 25: SAMS Image installation on microSD-Card with software w32 DiskImager	21
Figure 26: BerryLAN	22
Figure 27: SSH Software PuTTY	22
Figure 28: SSH Software PuTTY	23
Figure 29: Set Timezone	23
Figure 30: Expand Filesystem	23
Figure 31: Witty Pi Schedule Script Generator (http://www.uugear.com/app/wittypi-scriptgen/)	24
Figure 32: Data Warehouse HW Config.....	25
Figure 33: Create Witty Pi Schedule Script via SSH Terminal	25
Figure 34: Start Witty Pi Software (bash script)	25
Figure 35: Sign up in Data Warehouse to get your HIVE system credentials.....	26
Figure 36: SAMS system software interface	26
Figure 37: Display of messages in the data warehouse during the Debug mode	27
Figure 38: Software flow chart.....	29
Figure 39: Hardware Config selection	31

Figure 40: Device Config adaption	32
Figure 41: DW Report on Weight (11.03. 6:00 - 13.03. 07:00)	32
Figure 42: DW Report on Temperature (11.03. 6:00 - 13.03. 07:00).....	33
Figure 43: Events Log	33
Figure 44: Device Log	33
Figure 45: Error Codes	33
Table 1: Power Consumption per day	9
Table 2: Global horizontal irradiation per Country/ EU and year in average (globalsolaratlas.info)	10
Table 3: Components of the hi-fi HIVE Monitoring System.....	11

1. Introduction

1.1 Function principles

Up to 10 monitoring systems are supplied with power by one photovoltaic system. A DC/DC converter regulates the input voltage of the photovoltaic battery to 5V. One monitoring system (SAMS HIVE system) is based on a single-board computer Raspberry Pi Zero W with WiFi. The computer can be equipped with an attachable Witty Pi 3 Mini extension to obtain very differentiated switch-on and switch-off cycles. The Witty Pi mini allows the Raspberry Pi to be completely disconnected from the power supply between measurements, thus saving energy. Various sensors can be connected to the RaspberryPi. The software is prepared to connect temperature and humidity sensor (DHT22), temperature sensor (DS18B20), microphone for recording the frequency spectrum (SP0645) as well as a load cell (H30A). The components are soldered, plugged and connected to a PCB made for this purpose. The software must be calibrated before the first operation. For this purpose, the user needs to connect to the system using a smartphone via WiFi and perform the calibration through a web browser. A standard mobile GSM Wifi router is used to connect to the Internet. The router can link up to 10 systems and is connected to one of the Raspberry Pis for its power supply.

The measuring intervals and parameters can be changed online in the Data Warehouse (DW). The online configuration is checked by the system before each measurement and changes are updated if necessary. The software is Open Source and located on a githubserver. Software updates are automatically requested after each measurement and installed if necessary. Afterwards the system is shut down automatically, disconnected from the power by Witty Pi and waits for the next predefined start by Witty Pi. The Witty Pi Schedules can also be adjusted online. They are also updated with the online configuration. If no energy saving as well as differentiated measuring intervals are required, the Witty Pi can be omitted and the software starts the next measurement automatically after a certain time. This time setting can also be changed online. All collected data is sent to the DW and stored there. Reports can be created and data can be graphically visualized. Log files are also sent to the DW. This enables a differentiated error diagnosis. The software also includes automatic error handling for a wide range of error scenarios and is designed to display the status via RGB LED for fast on-site error diagnosis.

In case there is no Internet, the data is stored on the microSD card and can be read out locally through SSH access via WiFi or directly from the SD card with a computer. In case of internet breakdowns, the data will be cached and sent to the DW with the next internet access. All data that has been sent successfully will be automatically deleted from the device to reduce storage space. If the storage space becomes insufficient, the system deletes the data of every second measurement until storage can be released again. To reduce measurement uncertainties, the results of several short series of measurements are taken as an average. If there are strong outlier values between two measurements, the measurement is repeated. The respective parameters can also be set in the online configuration. The sensors are built into sensor frames provided for this purpose and can thus be placed in the bee colony. The connection is made via a cable. The scale is located in a frame under the beehive. The computer with PCB is installed in a waterproof case in the scale framework. A quick start guide can be found in the appendix X. To build a HIVE system based on ESP8266 see appendix XI.

1.2 Photovoltaic System

The photovoltaic system is only used for power supply. Any other power source adapted to the system can be used. DC power sources between 5V and 23V as well as standard AC power sources with micro-USB power supply are suitable. The dimensioning of the power supply is based on the standard measuring interval of the SAMS HIVE system (Chapter 3.4). It can be extended as needed and is independent of the HIVE system. Table 1 shows the calculation of the total energy demand. Table 2 shows the values of the Global horizontal irradiation for the partner countries Indonesia and Ethiopia as well as EU. For the project and as a recommendation the following configuration is used (also see Report 3.3):

A monocrystalline 12 Volt Phaesun Sun Plus 50 S solar module with 50 Wattpeak output was selected as solar module. A 10 Ampere PWM Phocos eco 10 charge controller was installed between the battery and the solar module. A standard car battery (100 Ah) is needed as accumulator. Charge controller and battery needs to be placed in a water-protected box and need to be connected with standard 12 Volt cables to the solar module and the SAMS HIVE system.

Table 1: Power Consumption per day

Consumer	Amount of Measurements per Day	Amount of MU	Length of Measurement [min]	Total Consumption per Day [h]	Power Consumption per Device [mA]	Total Consumption [Ah]
Monitoring Unit (MU)	56	9	5	4,67	70-200	8,40
Mobile GSM Router	56	1	10	9,33	400	3,73
					Total	12,13

Table 2: Global horizontal irradiation per Country/ EU and year in average (globalsolaratlas.info)









Country	Min	Max	Average	Monthly	Daily	Unit
ET	1753	2483	2118	177	6	kWh/m ²
IDN	1314	2191	1753	146	5	kWh/m ²
EU	803	2118	1461	122	4	kWh/m ²
GLOBAL	803	2702	1753	146	5	kWh/m ²
GER	951	1257	1104	92	3	kWh/m ²

2. Hardware

2.1 Components

Table 3 lists the main components of the SAMS HIVE system.

Table 3: Components of the SAMS HIVE system

Component	Function	Specification	Photo
Raspberry Pi zero w	Single board computer	ARM v7 1GHz processor, 512MB RAM, On-board Wireless LAN 2.4Ghz, On-board Bluetooth and micro-SD Card reader	
UUGear Witty Pi 3 mini incl. DS3231SN	Energy manager + Real time clock	Exact time for up to 17 hours	
Dallas DS18B20	Temperature inside hive	Operating range -10°C to +85°C +/- 0.5°C	
Aosong DHT22	Outdoor temperature and humidity	Operating range humidity 0-100% +/-2% temperature -40 to +80°C +/- 0.5°C	
Microphon Adafruit I2S MEMS SP0645	Records acoustics inside hive	Operating range 50Hz - 15KHz	
BOSCHE H30A load cell	Weight measurement	Accuracy class C3 and nominal load of 200kg. IP65, as outside hive	
AVIA Semiconductor HX711	A/D converter	24-bit analog-to-digital converter	
Step-down converter mini-360	Connects the device to 12V photovoltaic system	Output voltage is adjustable between 1.0 - 23V	

2.2 Circuit plan and Printed Circuit Board (PCB)

The Gerber files for manufacturing the Printed Circuit Board (PCB) are located on the [SAMS Github page](#). The design of the PCB (Figure 2) is based on the circuit plan shown in Figure 3 and the related wiring. The components are soldered to the designated slots. The slots are

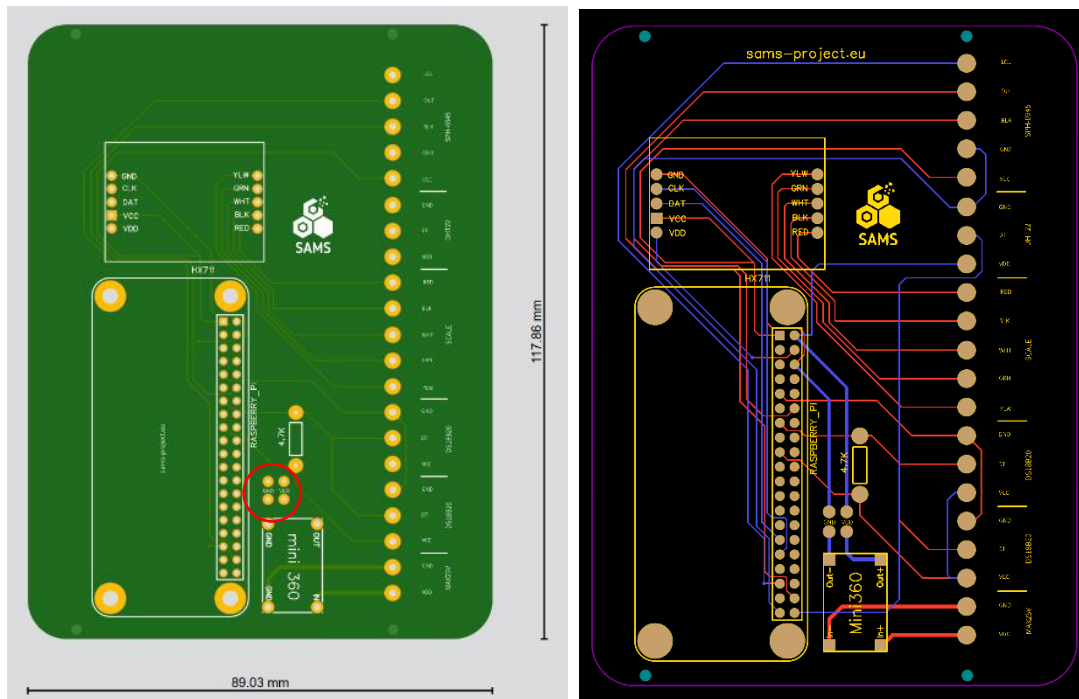


Figure 1: Top view of a ready to print circuit layout for the SAMS HIVE Monitoring System (marked red: bridge connection to use 12V input without Witty Pi) (left); Top and bottom view of PCB sketch (right)

labeled. The Raspberry Pi and Hx711 are connected using breakaway female headers (Figure 2) and the corresponding male headers. This enables a non-destructive disassembly of the components from the PCB if necessary. To be able to place the Witty Pi on the Raspberry Pi, breakaway male headers (2.54 mm): 1x40-pin, straight, double-sided are used. For the Hx711 only single-sided male header 1x5-pin headers are used. Screw terminal blocks (Figure 4) can be used to connect the external sensors, alternatively cables can be soldered to PCB. If the system is powered by a 12V power source (e.g. photovoltaic system) but the Witty Pi is not used, the two bridges next to the DC/DC converter must be connected to each other to provide a permanent power supply (Figure 1, red mark). To operate the DS18B20 temperature sensors, a 4.7k Ω resistor must be soldered to the designated position. If a RGB LED for status indication want to be used, it can be soldered directly to the bottom pins of the breakaway female header on the Raspberry Pi position on PCB (Figure 3, 5).

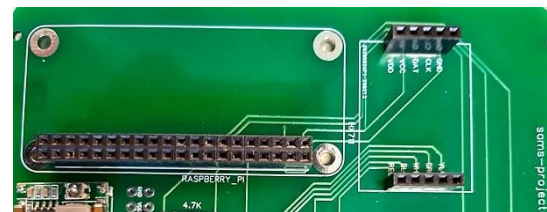


Figure 2: PCB female header

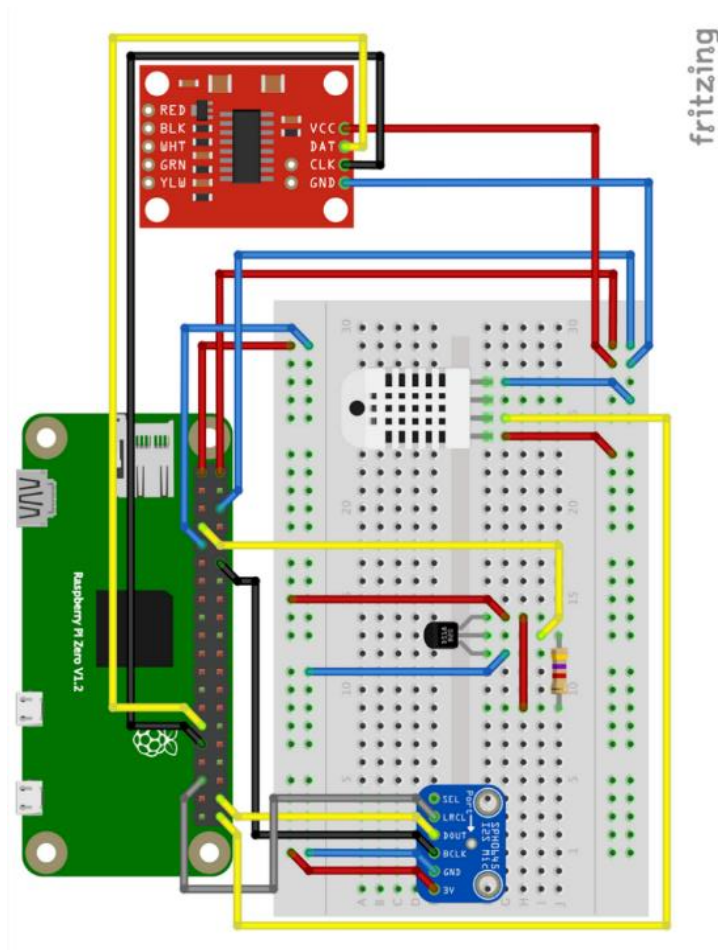


Figure 3: Circuit plan and wiring

Raspberry Pi wiringA/D Converter HX711

VCC to 5V
 DAT to Pin29/ GPIO 5
 CLK to Pin31/ GPIO 6
 GND to GND

Microphone SPH0645

LRCL to Pin35/ GPIO 19
 DOUT to Pin38/ GPIO 20
 BCLK to Pin12/ GPIO 18
 GND to GND
 3V to 3V

Temperature DS18B20

VDD to 3V
 DAT to Pin7/ GPIO 4
 GND to GND
 (4,7kΩ VDD and DAT)

Humidity DHT22

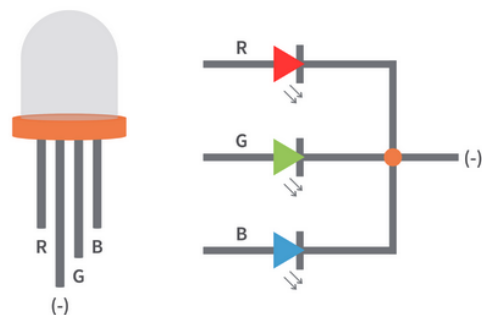
VDD to 5V
 DAT to Pin40/ GPIO 21
 GND to GND

RGB LED

Green to GPIO 9
 Red to GPIO 10
 (40 Ω in between)
 GND to GND
 Blue to GPIO 11



Figure 4: Screw terminal blocks (blue)

Figure 5: RGB LED Cathode Pinout
(www.circuitbread.com)

2.3 Case and Cables

A case for the PCB can be produced with common 3D printers (Figure 6). The SAMS HIVE prototype case was printed with a Zmorph VX with ABS filament. It can be equipped with cable ducts for load cell, connectors for power and flat cable as well as RGB LED for status indication and is easy to install (Figure 7, 8). The connection to the sensors is via 14-conductor flat cable with female 2x7 IDC connector cable (Figure 9). The case is optional and can be replaced by other common cases. For instance, instead of the 3D printable SAMS HIVE case (Figure 6), a small conventional food storage container can also be used as chassis for the PCB with its components and the corresponding cable connections. This is low cost, easy to obtain and waterproof. The material is suitable for the easy installation of connectors. For the plug connections of power, sensor frame and scale, holes have to be drilled into the case. The entry holes of the plug connection should be sealed with sealing compound or adhesive. D-Sub 9-pin connectors (Figure 10) can also be chosen for the sensor frame (with microphone, temperature and humidity sensor). As a connection for photovoltaic power a 12V DC power plug (Figure 8) is recommended.



Figure 6: CAD Sketch of SAMS HIVE Case



Figure 7: 3D printed SAMS HIVE Case

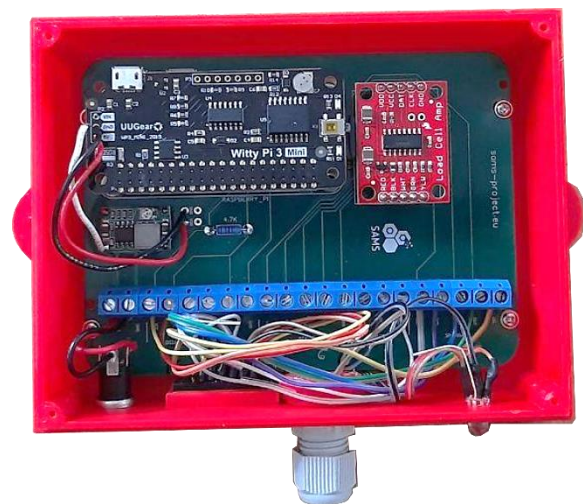


Figure 8: Opened 3D printed SAMS HIVE Case with PCB and components

To connect the sensor frame cables with 9-Pin D-Sub correctly, you can use figure 11 and the coloured cable code (1 – red, 2 – yellow, 3 – blue, 4 – white, 5 – black, 6 – orange, 7 – violet, 8 – brown, 9 – green). Figure 13 shows the open case and PCB with step-down converter, A/D converter and Raspberry Pi as well as screw terminal block (blue) for easy cable wiring without solder connection to PCB. The plug connections for sensor frame (D-Sub 9 plug) and photovoltaic power (DC power plug) are connected to the screw block. There are also cable ducts for the humidity sensor (DHT22) and the scale (H30A load cell). For the sensors a separate case was developed (Chapter 2.4)



Figure 9: 14-conduct flat cable with IDC connection



Figure 10: D-Sub 9-Pin Plug for Sensorframe connection

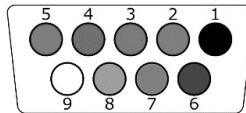


Figure 11: 9-Pin D-Sub pinout (backside)



Figure 12: DC power plug

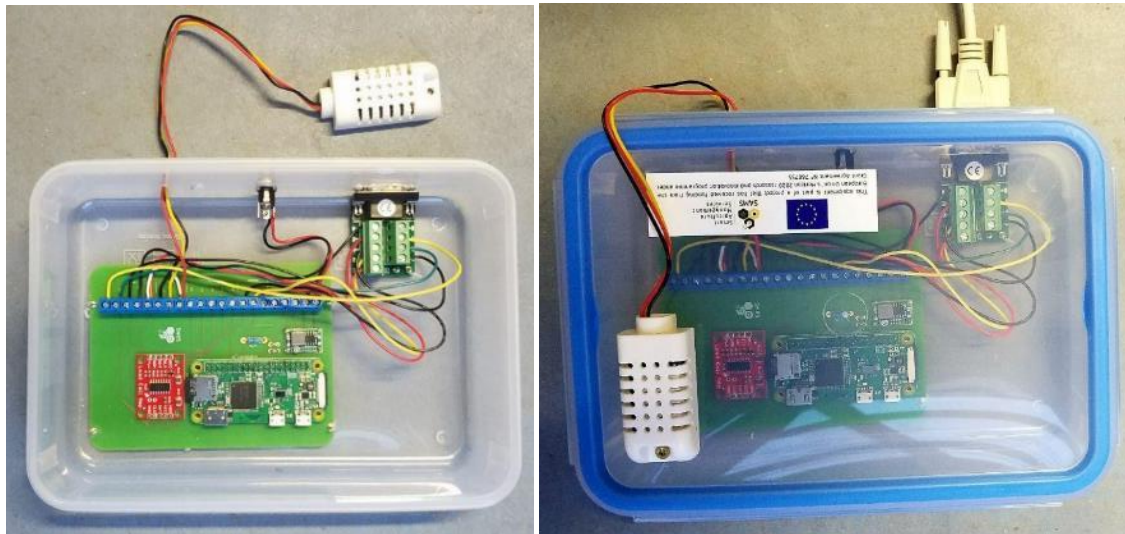


Figure 13: SAMS HIVE Monitoring System with sensor for outside temperature and humidity and cable

2.4 Sensor frame and placement

The DHT22, DS18B20 and SP0645 sensors for temperature, humidity and acoustic can be installed in the 3D printable polymer case (Figure 14). The case can be fitted on both sides with fine wire mesh (ideal mesh width < 0.2mm) to protect the sensors from contamination by wax and propolis. The sensor frame can then be placed centrally on a broodframe and screwed to the corresponding backplate on the other side of the broodframe. Ensure that a deepening (in size of the sensor frame) is scraped into the existing wax before the sensor frame is placed there, to keep the Beespace from the surface of the sensor frame to the next brood frame. The sensor frame is connected via flat cable to HIVE case (Figure 15). As an alternative to the polymer case, a wooden frame can be built and protected with wire mesh. For reference purposes both

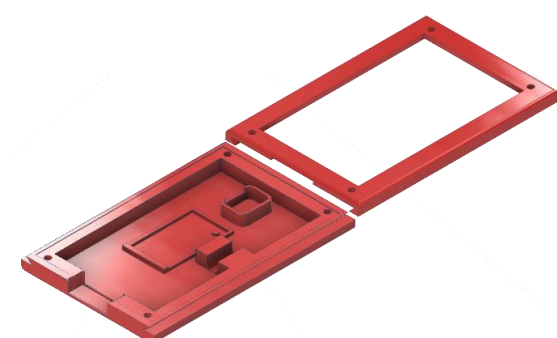


Figure 15: CAD sketch of 3D printable SAMS Sensor case

sensors DHT22 and DS18B20 can be placed in the sensor frame. However, only one temperature sensor in the sensor frame is required. The other one can be installed as an outdoor temperature sensor. We recommend to use DS18B20 as outside temperature sensor. Up to 10 additional DS18B20 temperature sensors can also be connected parallel to the DS18B20 interfaces on the PCB. However, attention must be paid to the respective sensor ID and this must be assigned accordingly when setting up the nodes in the DW. Therefore connect the sensors one by one and register them in the DW. It is recommended to have at least one sensor for outside temperature per apiary. Wire the cables to the sensors in the sensor frame according to Figure 9 and Figure 4 with connectors. If you choose a flat cable, connect the IDC connector according to figure 4 and the sketch in appendix (Wannenstecker 14, IDC 2x7). Figure 16 and Figure 17 show the positioning in the brood frame, on the left sensor case with wire mesh, on the right sensor case with printed polymer mesh, which has proved to be less robust against propolisation after tests.

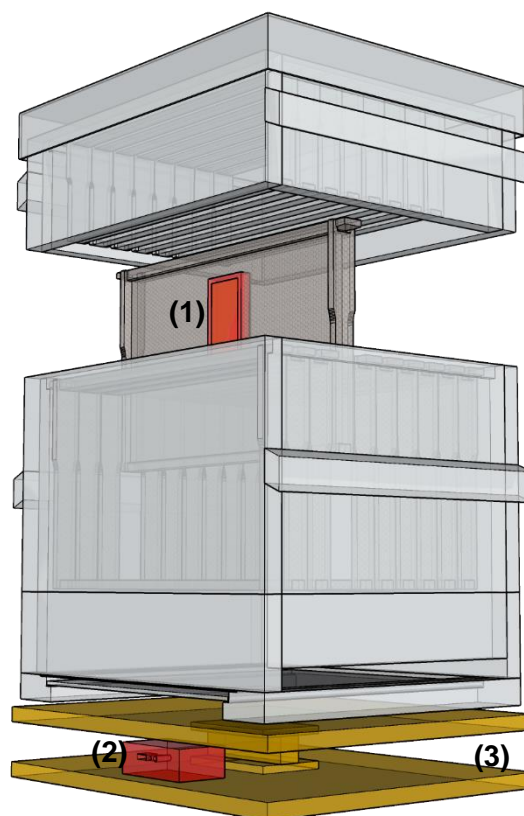


Figure 14: Placement of SAMS HIVE System in common beehive. (1) Sensor frame in broodframe, (2) HIVE case, (3) scale unit



Figure 16: Sensor case with wire mesh installed in regular brood frame



Figure 17: Sensor case with printed grid installed in regular brood frame

2.5 Scale unit

The scale unit consists of a load cell (BOSCHE H30A) and a metal plate on each side. The metal plates and the spacer are made of aluminium. Each metal plate is screwed onto the load cell with a spacer in between. Figure 18 shows the dimensions of the metal plate and the spacer with the appropriate hole dimensions. The outer dimensions and holes are not fixed and can be varied. The thickness of the material is 10mm for the metal plate and 6mm for the spacers. The holes in the metal plates shown in Figure 18 are used to attach the load cell (7mm) and a wooden plate to the scale unit (with M10 thread). The wooden plates should be of sufficient thickness to be able to take the maximum weight of the beehive. Figure 19 shows the dimensions of the load cell and the attachment of the metal plates and spacers. The cables of the load cell are color-coded and must be assigned to the appropriate labelled screw terminals for connection to the PCB (Figure 4).

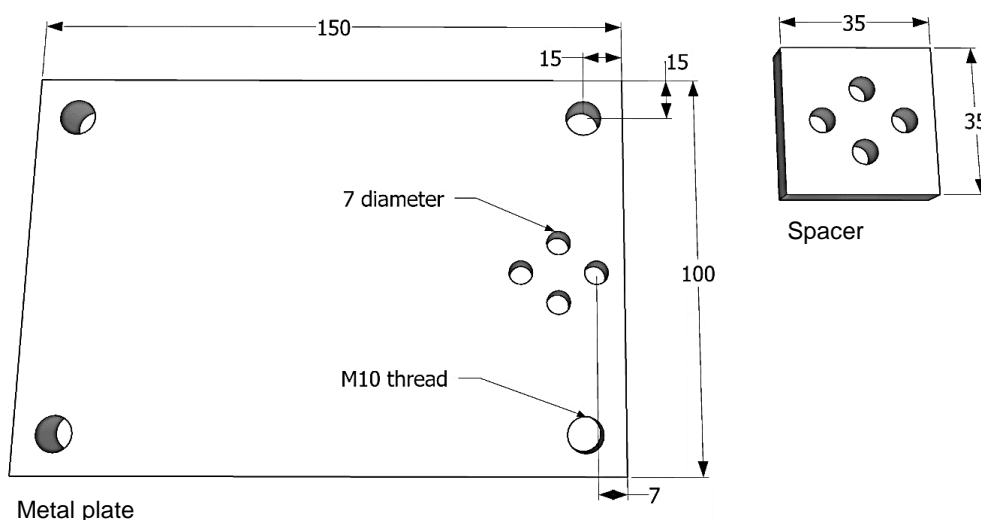


Figure 18: Measurements of metal plate and spacer for scale setup

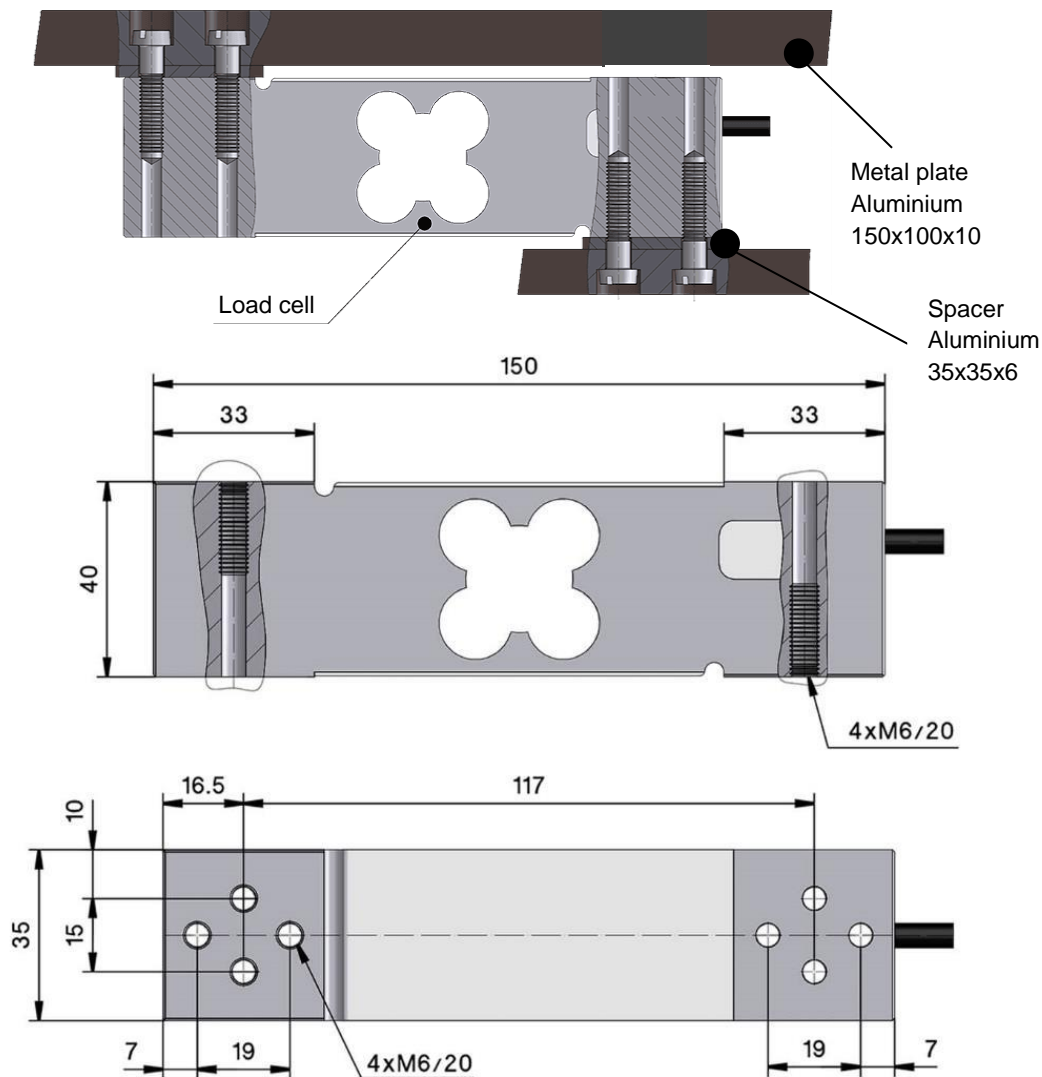


Figure 19: Measurements and Setup of Load cell (bosche.eu)

Another variant for the load cell setup comes from the beelogger project. Construction plans are available on the website <https://beelogger.de>. The advantage of this construction is an even more stable construction. In the SAMS project we decided to use only one load cell to reduce costs. The function is not negatively affected. Figure 20 shows the construction with one load cell on each side. The base frame consists of aluminium L-profiles which are held at the appropriate distance by wooden boards.



Figure 20: Scale unit with load cells H30A

2.6 WiFi Router adaption

In order to use the mobile WiFi GSM router E5330 from HUAWEI for the mobile data transmission it should be adapted. For this purpose two contact points must be connected. This will cause the router to start automatically as soon as it receives power. Therefore the two contact points on the PCB have to be bridged with a wire and soldering points as shown in Figure 20, below. To access the PCB, the four screws of the case (see Figure 20, top left) can be opened with a Torx screwdriver

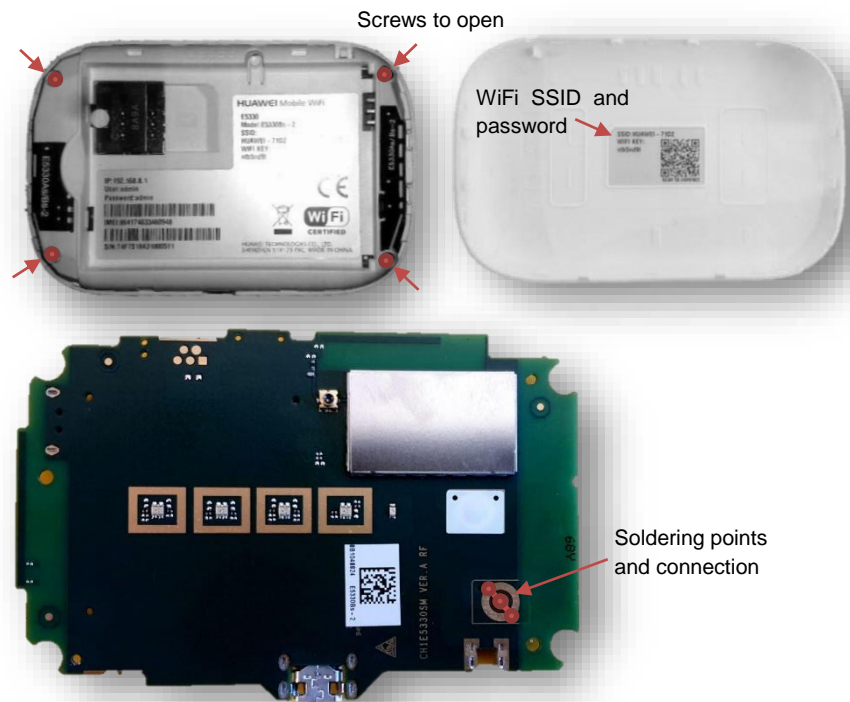


Figure 22: WiFi Router adaption for auto-switch-ON function (left); Access data (top right)

size 5. To reduce the power consumption to a maximum a dummy can be used instead of a battery or the contact GND on the router can be connected to the middle pin using a 4.7kΩ resistor. To power the router, connect it via USB cable to the microUSB port of Raspberry Pi of the HIVE system. To change settings on the mobile WiFi router, a connection must be established via web browser. Enter the IP address of the router in the browser address line: 192.168.1.8. Then log in and confirm with user name: admin and password: admin (see figure 21, top right). In the basic settings the option Fast boot can now be activated. In the WiFi settings, the automatic shutdown can be set to 10 minutes. If the WiFi setup of the SAMS system is not done via BerryLAN App but via wpa_supplicant (Appendix VII), the IP address of the system can also be read out via the tab Statistics in the user interface of the WiFi router. This is later necessary to calibrate the SAMS system (Chapter 3.5). The SSID and password for the WiFi connection to the router can be found on the inside of the router cover (see Figure 20). Don't forget to place a standard SIM card with at least 500 MB data volume/ month in the slot to run the router online.

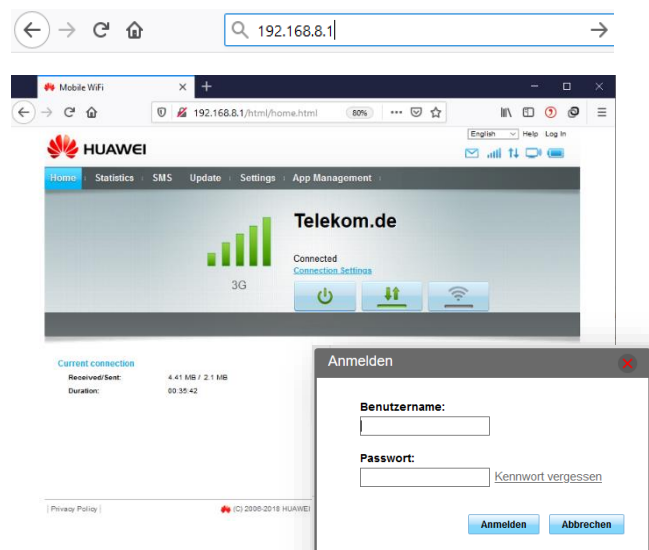


Figure 21: WiFi Router access

2.7 Witty Pi power management

If the system is not connected to the power grid, but is powered by a photovoltaic system for instance, we recommend using the Witty Pi 3 Mini. In order to reduce energy consumption, the Witty Pi will disconnect the Raspberry Pi from the power supply between measurements. In addition to other features (Appendix V, Witty Pi manual), the Witty Pi can be used to set very individual measuring intervals (Chapter 3.4). To be able to use the Witty Pi, its 5V and GND contacts must be connected to the VDD and GND contacts on the PCB using a cable connection as shown in Figure 23 (black and red cable).

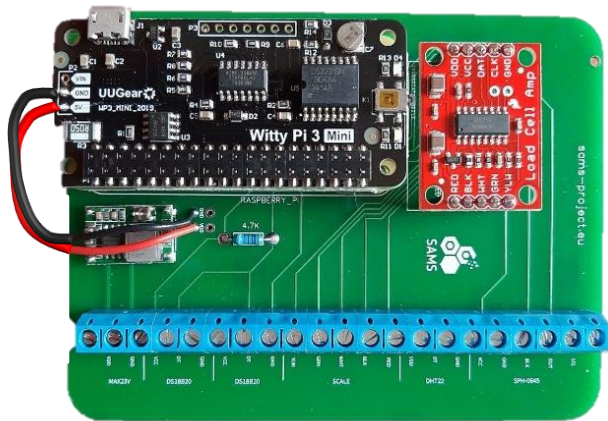


Figure 23: Witty Pi Power Connection (red and black cable connected to DC/DC converter)

If you want to use power monitor feature you need to connect another contact (DCIn) to the Witty Pi (VIN) as shown in Figure 24. This allows to monitor the charge state of the photovoltaic

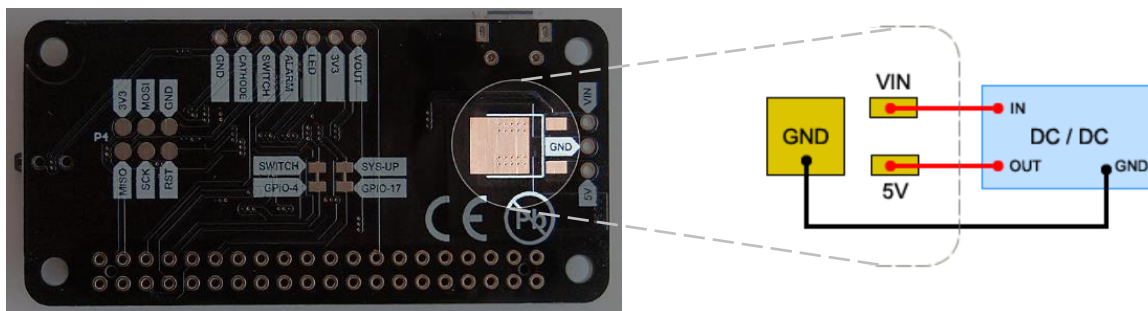


Figure 24: Optional Witty Pi Power Connection for detailed power monitoring

battery online. If the debug mode in online configuration is activated, the current voltage value of the DC input, i.e. the photovoltaic system, is posted in the DW after each restart of the system.

The Witty Pi is now plugged onto the RaspberryPi and connected to the PCB via cable connection for power supply (Figure 23). Afterwards the power supply (max. 23V) can be connected via the screw terminal blocks (blue - see marking "max23V" on the PCB). For the first start the button on the right side of the Witty Pi must be pressed. This default setting can be changed in the Witty Pi menu, so that the system will start automatically when it receives power (Chapter 3.4). Before the system can be started, a micro-SD card with a corresponding image needs to be plugged into the SD-slot of the Raspberry Pi (Chapter 3.1).

3. Software

3.1 Image installation

To install the SAMS software, the latest [SAMS image file](#) can be downloaded from the SAMS project website and placed on a standard micro-SD card. To install the image on the micro-SD card you can use the open source software [Win32 Disk Imager](#). Therefore insert the micro-SD card into the computer drive and check the drive name with your filemanager. Then select the appropriate drive and the downloaded image file in the software Win32 Disk Imager and write it to the micro-SD card (see Figure 25). After the process is done, the micro-SD card can be plugged in the slot of the Raspberry Pi and it can be started (Chapter 2.7). The image file contains the Raspberry Pi OS Raspbian Stretch as well as the software for the [SAMS HIVE system](#), the software for Witty Pi 3 Mini (Chapter 3.4) and the open source application BerryLan (Chapter 3.2). The SAMS software can also be installed on an existing Raspbian version (see Appendix II).

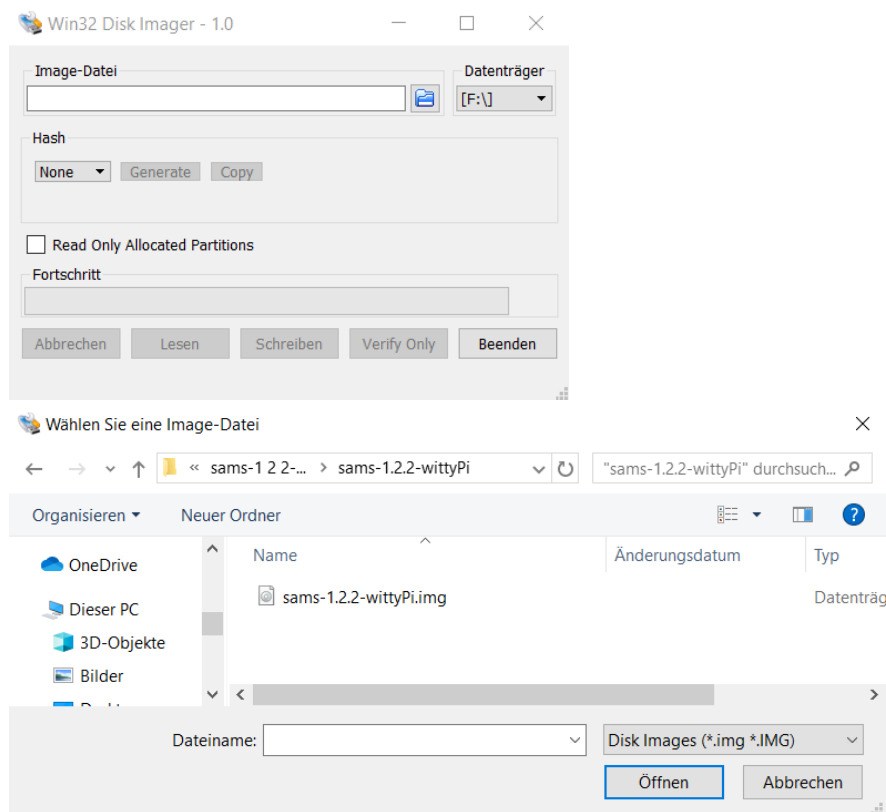


Figure 25: SAMS Image installation on microSD-Card with software w32 DiskImager

3.2 WiFi Setup

To calibrate the HIVE system, a WiFi connection between a router (Chapter 2.3) and the Raspberry Pi is required. The [BerryLAN App](#) on a current smartphone can be used for this purpose (Figure 26). It is already pre-installed on the SAMS image and only needs to be installed on the smartphone. To use the app, Bluetooth must be activated on the smartphone. Afterwards a Bluetooth connection can be established with the Raspberry Pi and in the following steps the WiFi can be selected in order to set up the app. Afterwards, the IP address of the Raspberry Pi will be displayed. The IP address can be used to calibrate the system via a web browser (Chapter 3.4) or connect via SSH (Chapter 3.3).

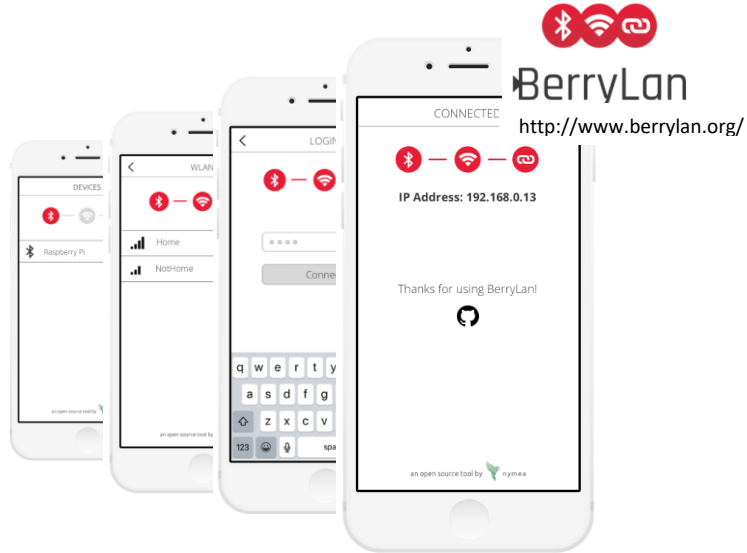


Figure 26: BerryLAN

3.3 Raspberry Pi Setup

If you wish to change the settings of the Raspberry Pi, you can establish a SSH connection to the Raspberry Pi using the [PuTTY software](#). Your computer needs to have connection to the same router as the Raspberry Pi with your SAMS system. As shown in Figure 27, simply enter

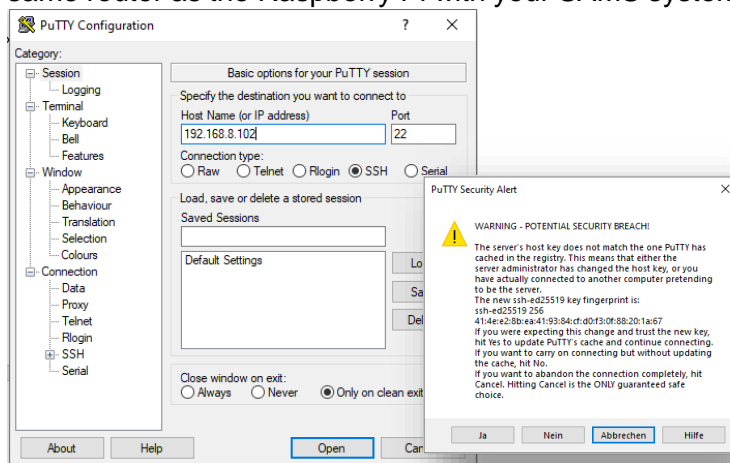


Figure 27: SSH Software PuTTY

the IP address of the Raspberry Pi (Chapter 3.2) in the field and confirm by pressing "Open". The following window can be confirmed with "Yes". A terminal (Figure 28) will open and you can start working on the Raspberry Pi after login with default user and password (Appendix V). For basic settings, the configuration-tool can be opened with the command "sudo raspi-config". For

example, the time zone can be changed (Figure 29), which is only necessary if system is used offline. Furthermore the storage medium can be expanded with the function "Expand Filesystem" (Figure 30). This frees up unused memory and is useful in case the Internet connection is too bad to send data or is completely down and data is stored on the internal microSD card.

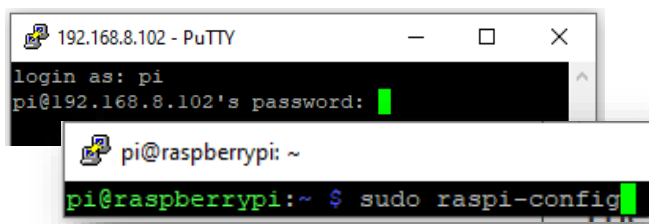


Figure 28: SSH Software PuTTY

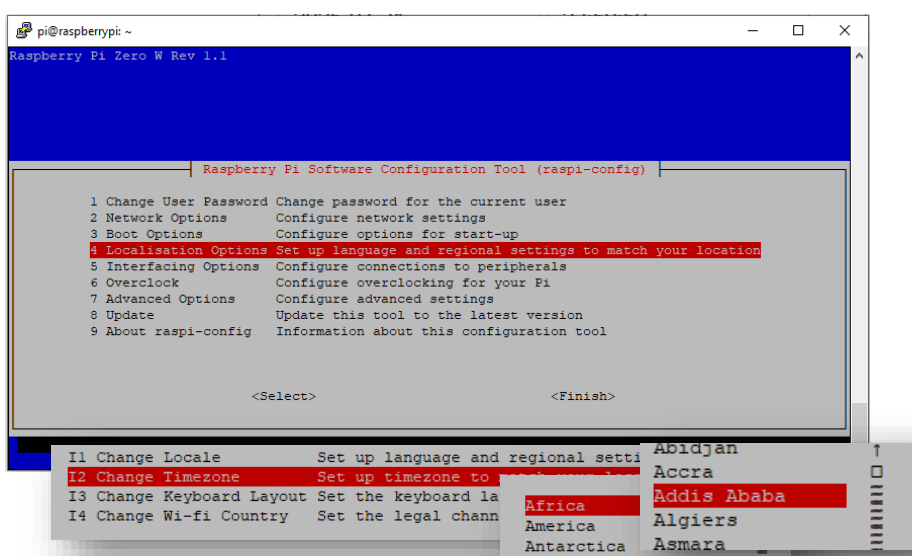


Figure 29: Set Timezone

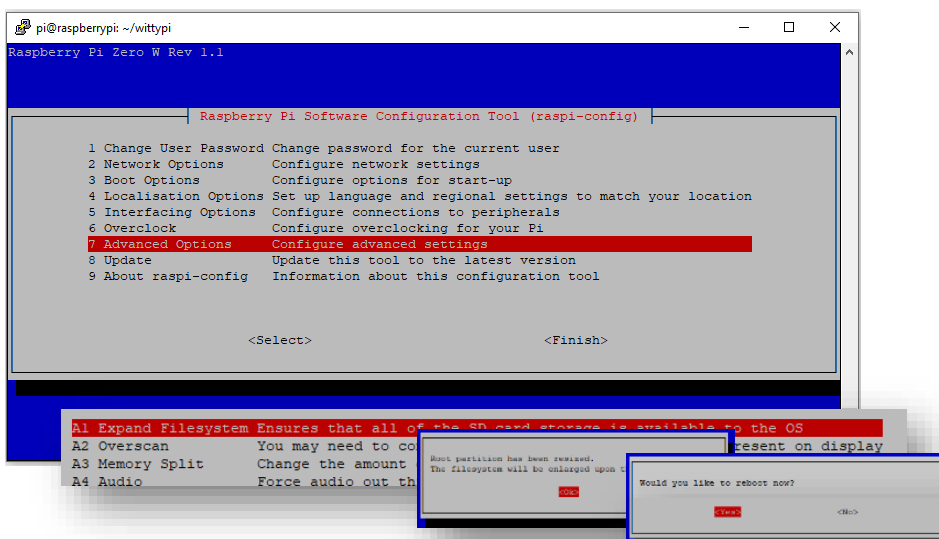


Figure 30: Expand Filesystem

3.4 Witty Pi Setup

The Witty Pi 3 mini brings several benefits. It provides a clean shutdown and startup via physical button if needed. It also keeps the time with an external RTC clock. Furthermore it enables the system to startup or shutdown at a precise time. If this function is required a schedule script must be created first. The easiest way to do this is to use the [Witty Pi Schedule Script Generator online](#) at the producer's website (Figure 31 and Appendix V). We recommend to use the Witty Pi default script for normal HIVE system and the Witty Pi router script to use HIVE system with mobile router (Appendix IV). There are two options to activate the script:

Option 1: A *config* file must be created or adapted in the SAMS DW, this has to be done in the *HW Config* tab (Chapter 4.2). The code of the schedule script is added to the end of the entry. The format must be the *YAML* text format (Figure 32).

Option 2: Using SSH connection (Chapter 3.3) a new file should be created in the folder *wittypi/schedules/* with the extension *.wpi* (Figure 33). The code of the schedule script is copied into this and the file can be saved and closed with Ctrl + X and confirmation with Y (Figure 33). Then change to the */wittypi/* folder and start the Witty Pi software (a bash script) with the command `sudo ./wittyPi.sh` (see Figure 34). There you can select and activate the previously created script under the menu item 6 Choose schedule scripts (see Figure 34).

To stop the script you have two ways: Either delete the *schedules.wpi* file in the */wittypi/* folder. Or select in the Witty Pi menu under item 10 Reset data the sub item 6 Perform all and stop all script activities.

When creating a schedule script, please consider if you want the Raspberry Pi to shut down automatically or if you want Witty Pi to shut down after a scheduled time. For use with Witty Pi schedule script it is recommended that you let Raspberry Pi do the shutdown itself. Therefore the line `auto_shutdown` in the Config has to be set to 1 (Figure 32). This will shut down the Raspberry Pi after successful

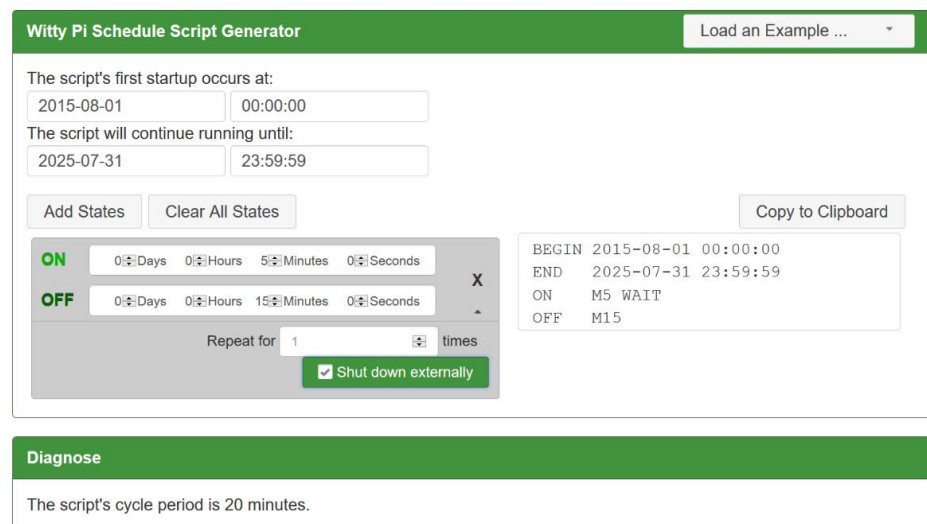


Figure 31: Witty Pi Schedule Script Generator
(<http://www.uuegear.com/app/wittypi-scriptgen/>)

measurement, upload of data and update if necessary. In addition the option Shut down externally must be marked in the Script Generator (Figure 31). This will create a WAIT entry in the script and the Witty Pi will only switch on the Raspberry Pi in a scheduled operation. This feature allows a further reduction of power consumption. For further features please read the [Witty Pi 3 Mini manual](#) (Appendix V).

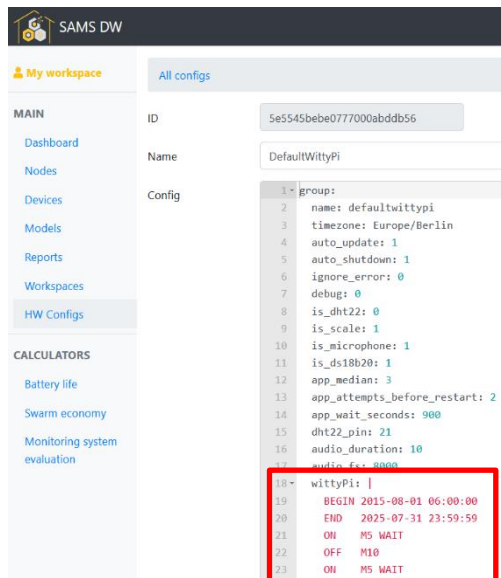


Figure 32: Data Warehouse HW Config

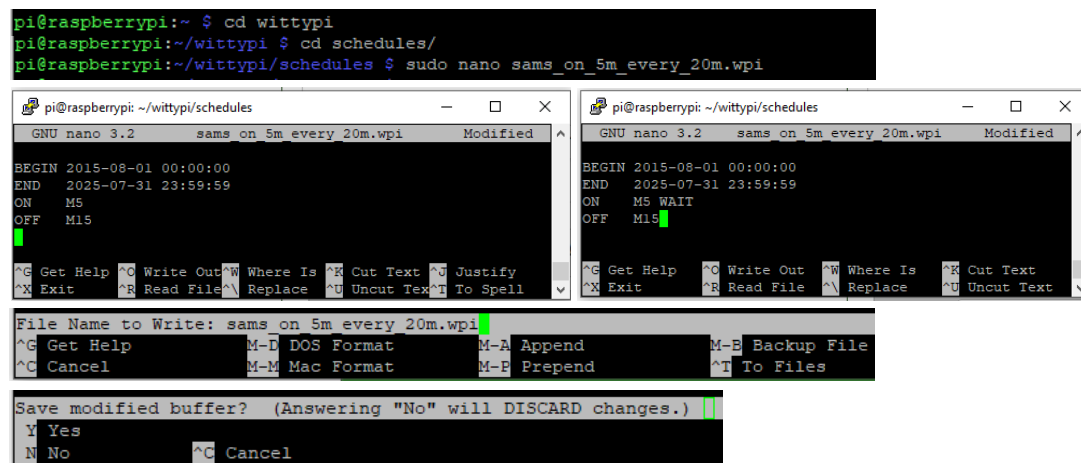


Figure 33: Create Witty Pi Schedule Script via SSH Terminal

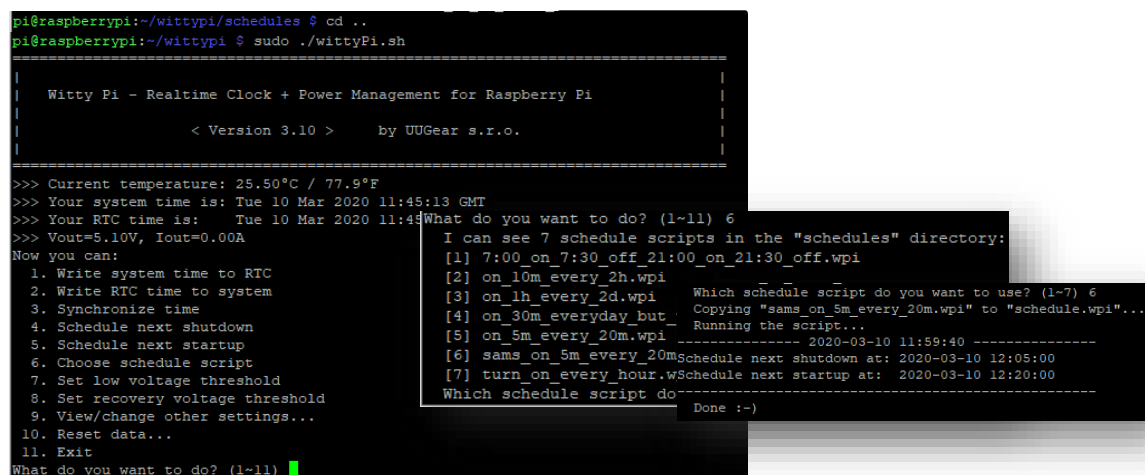


Figure 34: Start Witty Pi Software (bash script)

3.5 SAMS Software Calibration

Before you start the first measurement with the SAMS HIVE system, the system must be set up. The user-related credentials for data transfer to the DW must be entered and the scale must be calibrated once before the first use. To get HIVE system credentials for data transmission sign up in DW (Figure 35). To set up the system the IP address of the Raspberry Pi (Chapter 3.2) must be entered in the address line of the browser, followed by port 5000 as in the example in figure 36, above (e.g. 192.168.8.102:5000). If the connection is correct, you should now be prompted to enter the corresponding credentials in the ID and Secret row and confirm with SAVE (Figure 36). Then follow the prompts to calibrate the scale and finish the setup with a restart. The system will then start measuring and sending data to the DW. In the Settings menu, the Client ID, the WiFi connection quality, the free available memory on the microSD card and the config settings can be viewed. In addition, credentials and the scale calibration can be reset. Furthermore, the scale can be tared with the TARE button. If it is necessary to check the functionality of the sensors via the interface, the function Sensor Test can be used or the address line can be extended by /api (e.g. 192.168.8.102:5000/api). This shows the currently recorded sensor data. This is useful, for example, for a first check when measuring offline without a working internet.

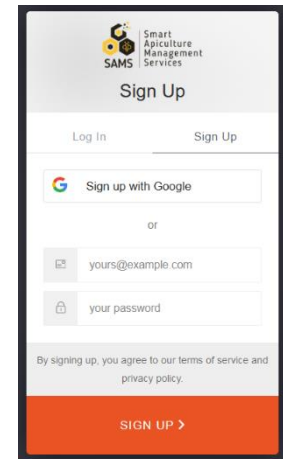


Figure 35: Sign up in Data Warehouse to get your HIVE system credentials

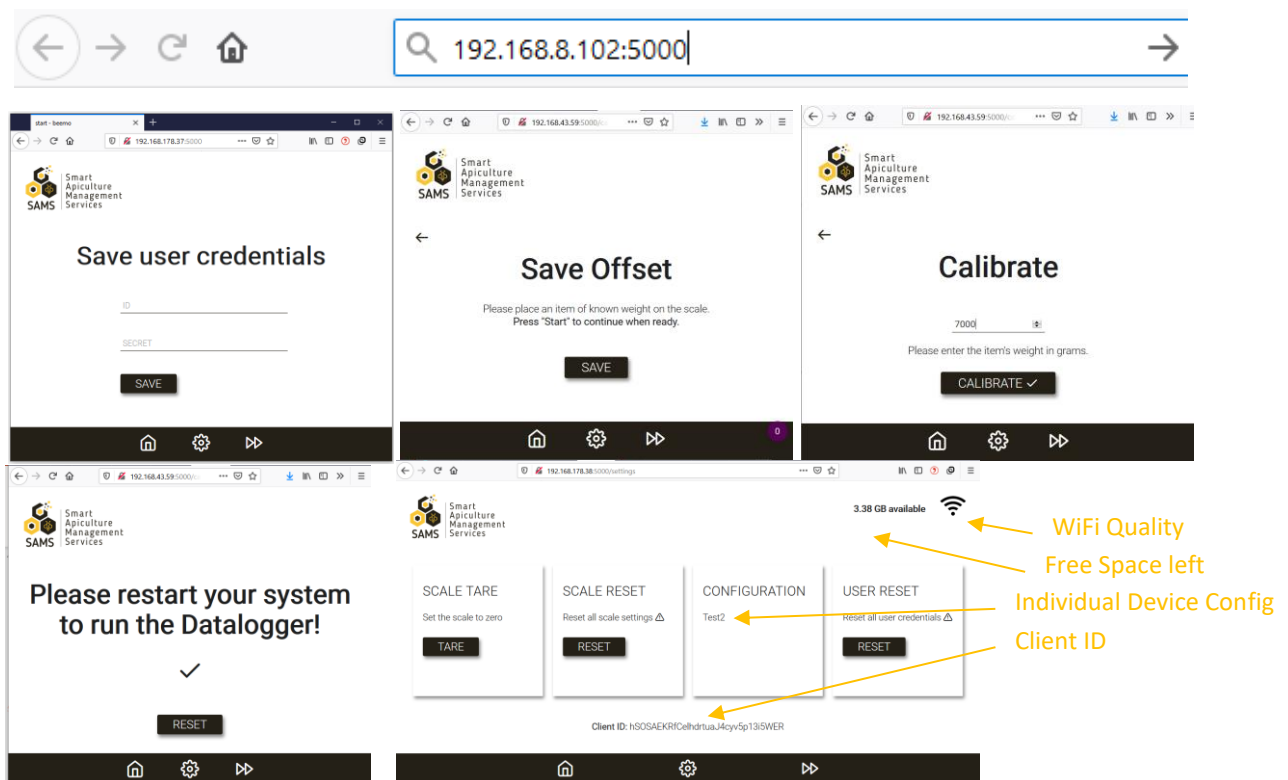


Figure 36: SAMS system software interface

3.6 Features

Besides the basic functions, the SAMS HIVE System offers additional features. The basic functions are sensor data recording of temperature, humidity, weight and acoustics, as well as WiFi data transmission to the DW. For the acoustic data the audio recordings are converted into FFT spectral data. The conversion of the audio data is mainly done for reasons of data protection and reduction of the data volume. Advanced features include:

- **Online Configuration:** Key parameters such as measurement interval or asymmetric measurement schedules can be adjusted in the DW for specific user groups or individual HIVE systems (Chapter 4.2)
- **Debug mode:** In debug mode (Figure 37, Chapter 4.3), information about the voltage supply of the photovoltaic battery, the signal strength of the WiFi, errors, name of the selected online configuration, version number of the app, update process, reboot if necessary, warnings if sensor should be checked, error if access denied (check credentials) is shown.
- **Automatic update process:** After each measurement, the system is checked for new updates and, if necessary, the latest firmware version is automatically downloaded and installed (Figure 37).
- **Calibration and Setting User Interface (WebApp):** A dedicated WebApp was developed for the calibration of the HIVE system (Chapter 3.5)
- **Self Test Interface:** A Self Test Interface is offered for functional testing after calibration (Chapter 3.5).
- **Status display:** Various status messages can be displayed via an RGB LED in the case. The function can be activated via the Online Configuration.
- **Offline Mode:** In offline mode all datasets are stored on the SD card and can be read from it. Either manually via ssh or via USB interface and the customized UNILV firmware (Report 4.3)

2020-06-27 20:31	debug	Config Name: hive_4
2020-06-27 20:31	debug	Start Application: 2.47
2020-06-27 20:25	debug	update from 2.46 to 2.47
2020-06-25 10:57	debug	Signal Strength: excellent

Figure 37: Display of messages in the data warehouse during the Debug mode with automatic update

3.7 Process

In the following, the software process after the configuration and the start of the measuring process is briefly described in words. A software flow chart illustrates it additionally Figure . This process describes a complete measuring interval. The distance of such measuring intervals can be adjusted by using the online configuration in the field 'app_wait_seconds' or the energy management with wittyPi 3 mini.

1. if debug is on: send following debug information:
 - Config version
 - Config Name
 - Signal strength
 - Voltage (vOut and vIn from wittyPi if installed)
2. check if system has valid access token
3. write online status (true or false)
4. if online: synchronise configuration with online config if exists
5. check if ignore_error
6. append sensors from config list
7. start get sensor data
8. if sensor data valid, try to send this data
9. if status code 200: (data accepted) then the system jump to step 9)
10. if status code 400: delete the dataset
11. if status code not 200 or 400, the system save the dataset (refer to save feature) and switch to offline mode
12. if sensor data is corrupt, send error message to DWH (**[SENSOR] failed!**) and delete the dataset
13. try to post log files to DWH
14. if response 200: delete the dataset
15. if response 400: delete dataset (redundant to point 7.)
16. if not response: return false
17. if response from logfiles are false: add one failed attempt
18. if failed attempts > app_attempts_before_restart (in config): send error message to DWH (**Too many errors: reboot system!**)
19. check for update and if auto_update true:
20. pull the new update.py
21. write update file to recognize the new update after restart
22. restart
23. the system will recognize the new update and execute the new update.py
24. restart
25. if auto_shutdown: shutdown the system
26. wait app_wait_seconds before begin the new cycle

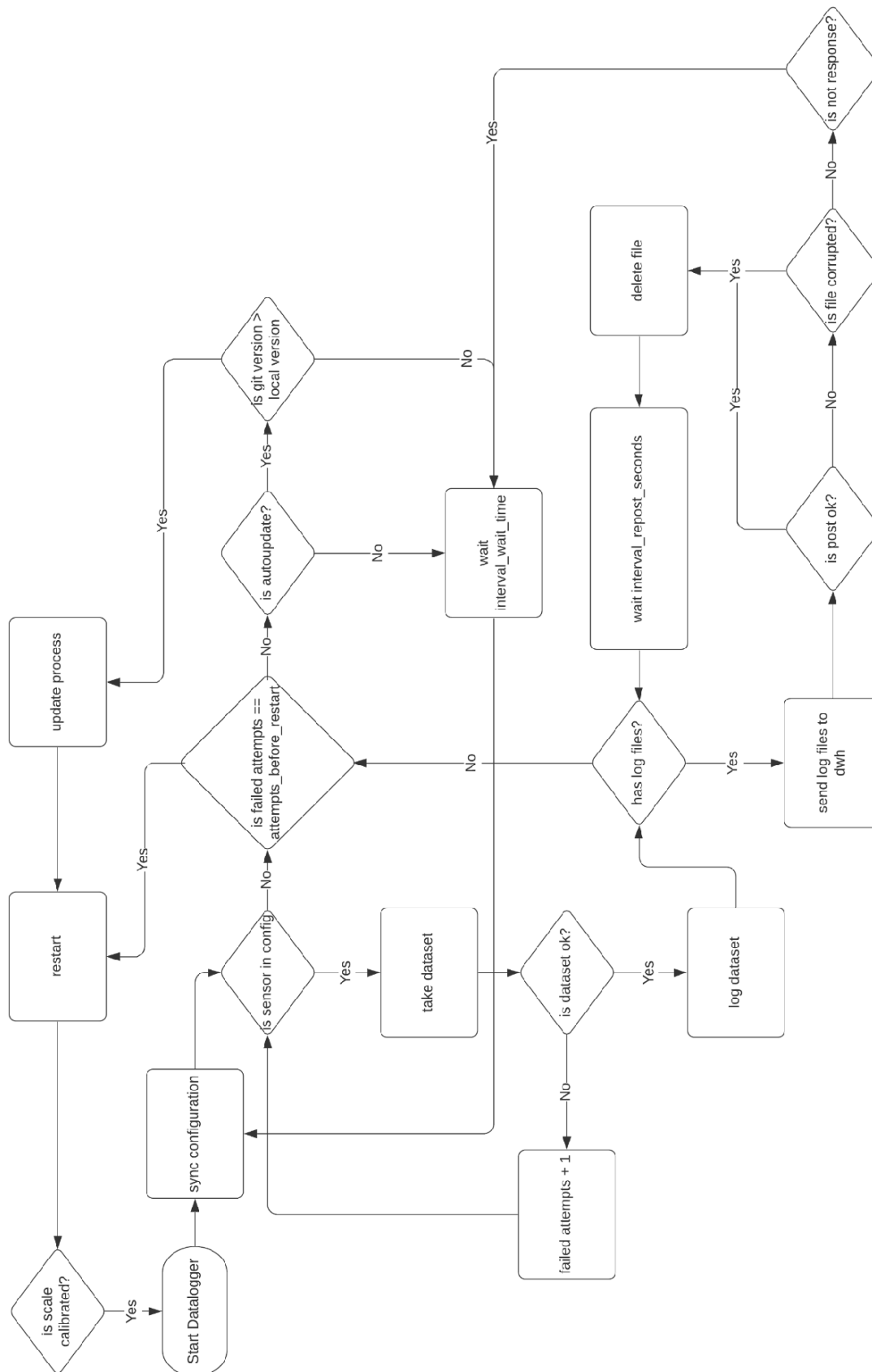


Figure 38: Software flow chart

3.8 Automatic error control

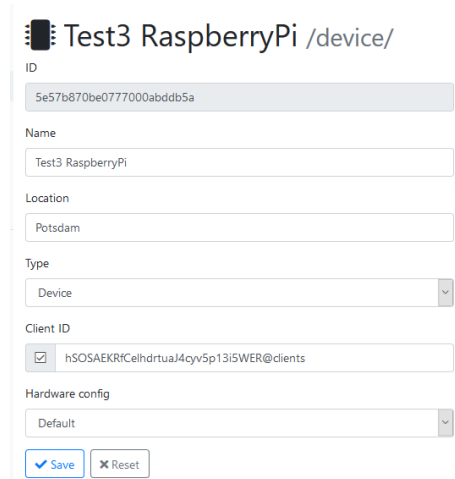
Sensors are checked for errors within each measurement routine. Errors are logged in the internal log and sent to the DW if the debug function is active (Chapter 4.3). Similarly, after an adjustable number of failed attempts, defective sensors are automatically switched off and a warning is sent. In the following, the warnings and their function are shown:

- `Failed to initialize [SENSOR]:` Sensor is broken or the wiring is not correct
- `Cannot get Scale data: [ERROR MESSAGE]:` Something is wrong with the wiring of the scale or the load cell is broken
- `Failed to insert log files: [ERROR MESSAGE]:` Writing the log file failed
- `Failed to read log files: [ERROR MESSAGE]:` Reading the log file failed
- `Failed to post log files: [ERROR MESSAGE]:` Exception while posting the Log file. Broken pipe or similar
- `[SENSOR] failed:` Sensor is broken or sends no valid data (Similar to "Failed to initialize [SENSOR]")
- `Too many errors: reboot system!:` Failed attempts of the System is \geq "attempts_before_restart" in the config. This happens, if a Sensor failed. The system remembers the sensor and excludes it at the next restart

4. Data Warehouse

4.1 Setup

To receive sensor data in the DW, the device must be configured as a node. The sensors are mapped in this node. The mapping is used to route incoming data stream to proper node. The best way is to use the register button to set a new device in the Nodes tab and enter the corresponding client ID. If further sensors need to be mapped, you can use the instructions in the [DW manual](#) (Appendix V). In the Device details of each device in the nodes tap a specific Hardware Config can be selected if it should differ from the default settings (Figure 39). It is recommended to select a default setting, depending on the configuration with or without Witty Pi feature. Custom Config can be created or changed in the HW Config tab.



The screenshot shows a configuration form for a device named 'Test3 RaspberryPi'. The form includes the following fields:

- ID:** 5e57b670be0777000abddb5a
- Name:** Test3 RaspberryPi
- Location:** Potsdam
- Type:** Device
- Client ID:** hSOSAERKRCelndrtuaJ4cyv5p13i5WER@clients
- Hardware config:** Default

At the bottom of the form, there are two buttons: 'Save' and 'Reset'.

Figure 39: Hardware Config selection

4.2 Online configuration (HW Configs)

The Online Config allows specific settings to be adjusted online for the SAMS HIVE system. Before each measurement, the system checks if any changes have been made in the configuration and updates it if relevant. Therefore it is necessary to select the appropriate Config in the nodes as described in chapter 4.1. Appendix VII contains an example of the default settings for use with Witty Pi. Figure 40 shows the entry of a Config. The following settings can be made so far:

- Line 2: Name of the configuration
 - Line 3: Automatic software update
 - Line 4: DHT22 sensor connected
 - Line 5: Balance connected
 - Line 6: Microphone connected
 - Line 7: DS18B20 connected
 - Line 8: Number of values for calculating the median within one measurement
 - Line 9: Distance between the individual measurements for one sensor in seconds
 - Line 10: Waiting time in seconds until the next upload attempt if previous attempt failed
 - Line 11: Number of upload attempts
 - Line 12: Number of unexpected failures until restart
 - Line 13: Waiting time in seconds until the next measuring cycle
 - Line 14: Raspberry Pi Pin connected to the DHT22 sensor
 - Line 15: Audio recording time in seconds for FFT
 - Line 16: Sampling rate in Hz
- more lines are added for the Witty Pi script (Chapter 3.4)

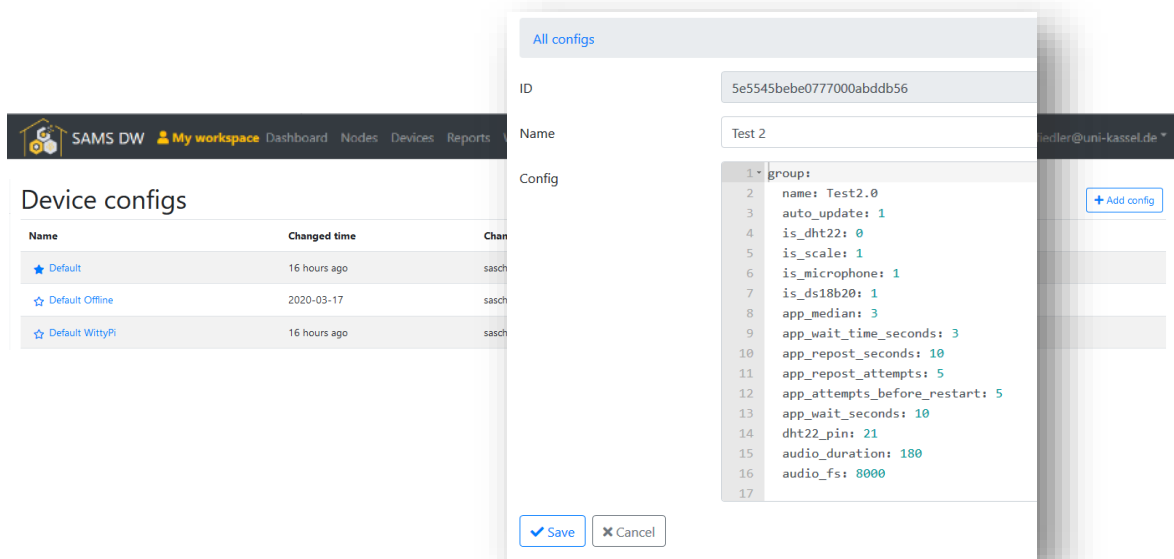


Figure 40: Device Config adaption

4.3 Report

The data can be displayed graphically under the Report tab in the DW. Figure 41 and Figure 42 show the data collection of temperature and weight in the same period over 3 days. This clearly shows the function of the Witty Pi Default script. While during the day from 6 am to 6 pm continuous measurements are taken every 15 minutes, at night only four measurements are taken, from 12 pm to 1 am. Under the Devices tab in the DW the current upload events (Figure 43), the device log (Figure 44) and errors can be displayed. Certain error codes are available (Figure 45).

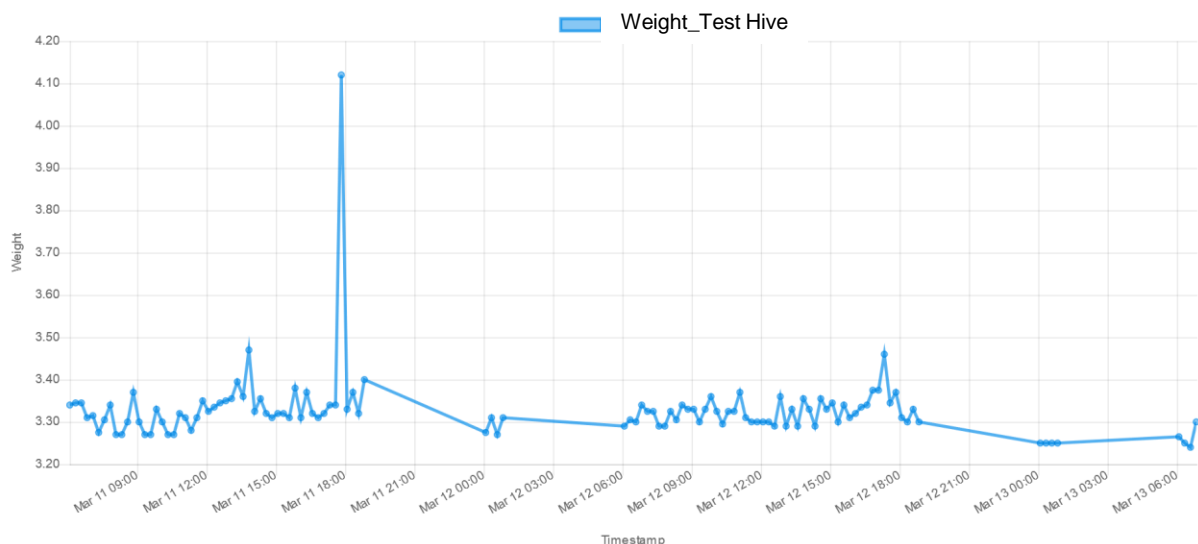


Figure 41: DW Report on Weight (11.03. 6:00 - 13.03. 07:00)

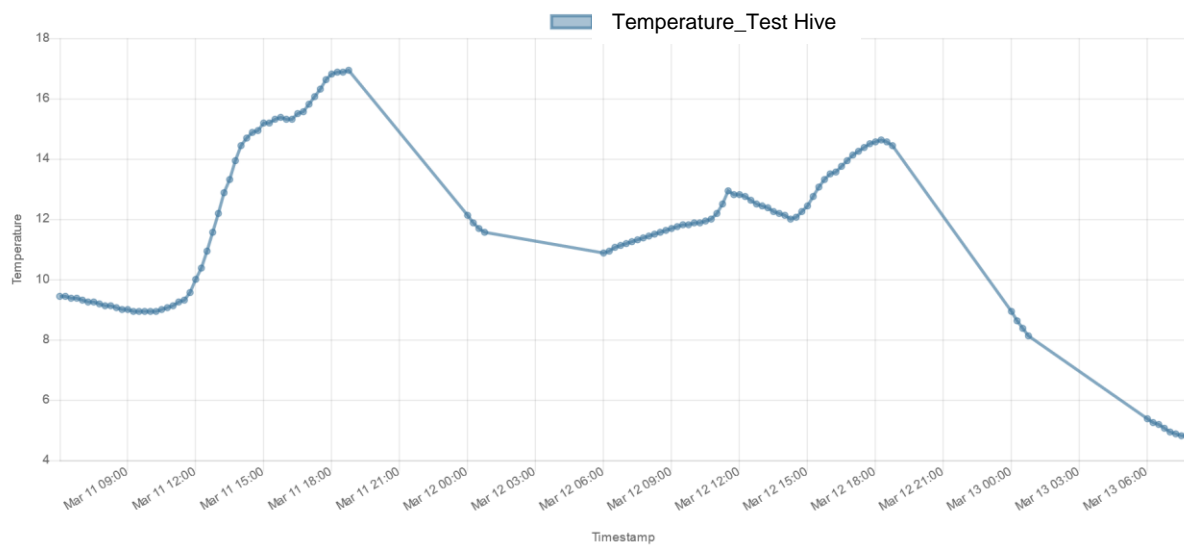
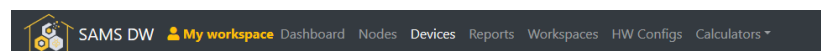


Figure 42: DW Report on Temperature (11.03. 6:00 - 13.03. 07:00)



All devices / Test3 RaspberryPi

Test3 RaspberryPi /events/

Device logs | All events

Date	Source ID	Result
7 minutes ago	scale-hSOSAERfCelhdrtuaJ4cyv5p13i5WER	✓ Ok
8 minutes ago	audio-hSOSAERfCelhdrtuaJ4cyv5p13i5WER	✓ Ok
11 minutes ago	ds18b20-1-hSOSAERfCelhdrtuaJ4cyv5p13i5WER	✓ Ok
11 minutes ago	ds18b20-0-hSOSAERfCelhdrtuaJ4cyv5p13i5WER	✓ Ok

Figure 44: Events Log

Device logs | All events

Date	Level	Message
13 minutes ago	debug	Signal strength: excellent
13 minutes ago	debug	Start application: 1.4

Figure 45: Device Log

```
.log("Start application. Version: {}".format(
.log("Error in Application: {}".format(e), "
.log("Take dataset error: {}".format(e), "er
.log("Try to send log files.", "debug")
.log("Dataset posting failed: {} | Log datas
.log("Sensor: {} not available".format(senso
.log("Too many errors: reboot system!", "err
.log("Failed attempts: {} times".format(self
.log("App crashed: reboot system in 120 seco
.log("Failed to initialize DHT22: {}".format
.log("Failed to initialize scale: {}".format
.log("Failed to initialize DS18B20: {}".form
.log("Cannot get FFT Data", "error")
.log("DS18B20 does not work properly", "warn
.log("Cannot get DS18B20 Data", "error")
.log("Cannot get DHT22 Data", "error")
.log("Cannot get Scale Data", "error")
.log("Space available: {} GB".format(round(g
.log("No more space available: {} GB".format
.log("Failed to insert log files: {}".format(e
.log("Failed to read log files: {}".format(e),
.log("File corrupted! Delete file", "warning
.log("Failed to post log files: {}".format(e),
.log("Cannot get scale data: {}".format(e), +
```

Figure 43: Error Codes

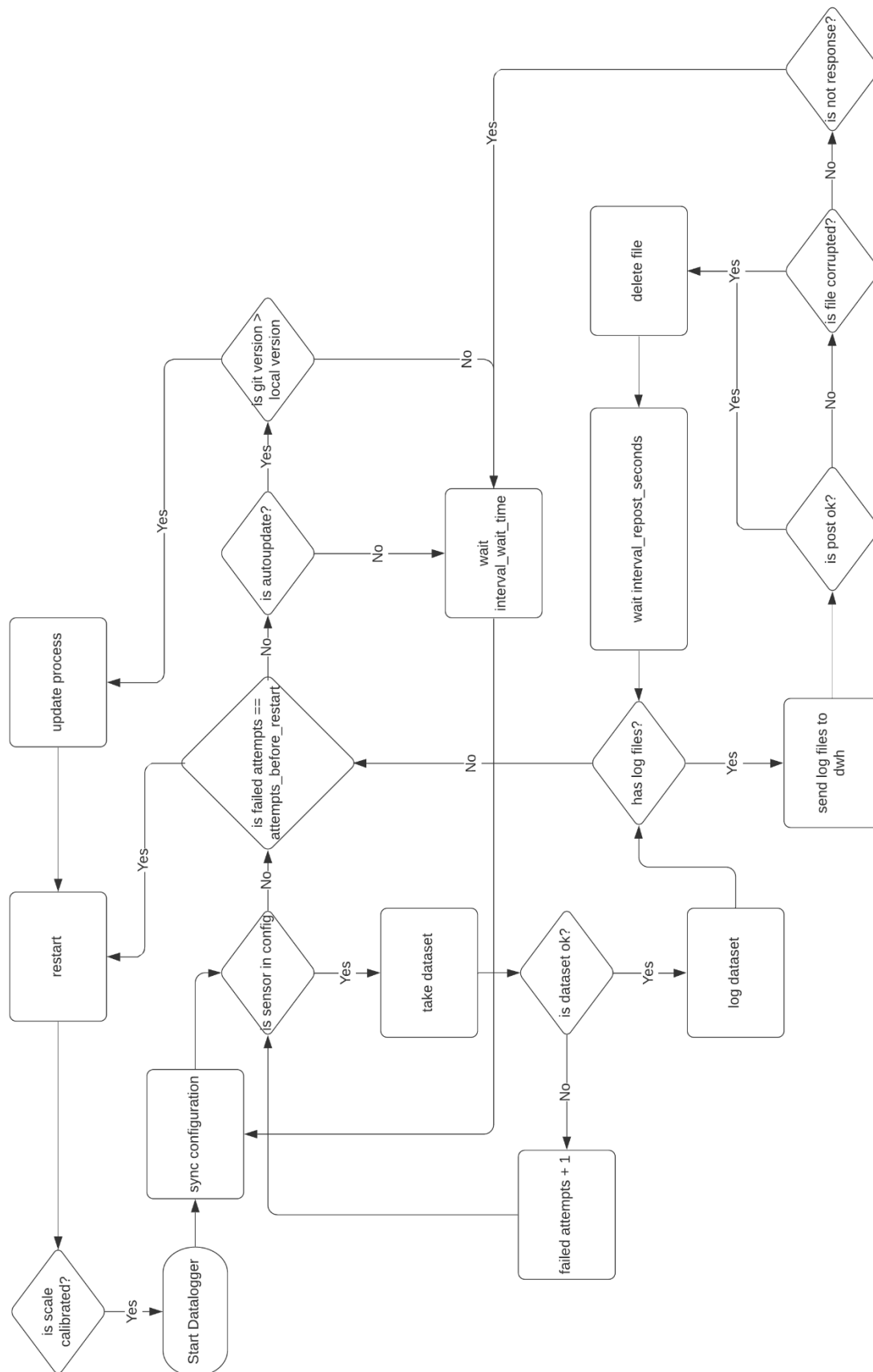
5. Protocol

To log errors [Github Issues](#) should be used. In order to support further development of the DSS algorithm, bee activity should be logged using [Hivelog](#) and the logs should be retained.

Annexes

- I. Software flowchart**
- II. Prepair SD Card**
- III. GPIO places Raspberry Pi**
- IV. Witty Pi schedule script**
- V. Bug report**
- VI. WiFi wpa_supplicant.conf**
- VII. HW Config default**
- VIII. Scale setup**
- IX. Quick Start Guide**
- X. Manual for ESP8266 based system set up**

I. Software flow chart



II. Prepair SD Card

Update and Upgrade

```
sudo apt update
```

```
sudo apt-get upgrade
```

Extend Filesystem

```
sudo raspi-config
```

→ Advanced Options

→ Expand
Filesystem...

Install ftp server

```
sudo apt-get install vsftpd
```

- Edit ftp server config

```
sudo nano /etc/vsftpd.conf
```

- Uncommend following lines:

```
anonymous_enable=NO
```

```
local_enable=YES
```

```
write_enable=YES
```

```
local_umask=022
```

- Add this lines:

```
user_sub_token=$USER
```

```
local_root=/home/$USER/sams_system
```

Create sams_system directory:

```
mkdir /home/pi/sams_system /create directory/
```

```
chmod a-w /home/pi/sams_system /set access rights/
```

Setup Microphone

```
sudo nano /boot/config.txt
```

Uncomment `dtparam=i2s=on`

```
sudo nano /etc/modules
```

Add `snd-bcm2835` on its own line

- reboot

```
lsmod | grep snd
```

to confirm the modules are loaded

```
pi@raspberrypi:~$ lsmod | grep snd
snd_soc_bcm2835_i2s      20480  0
regmap_mmio             16384  1 snd_soc_bcm2835_i2s
snd_soc_core            188416  1 snd_soc_bcm2835_i2s
snd_compress            20480  1 snd_soc_core
snd_pcm_dmaengine       16384  1 snd_soc_core
snd_bcm2835             24576  1
snd_pcm                 98304  4 snd_pcm_dmaengine,snd_soc_bcm2835_i2s,snd_bcm2835,snd_soc_core
snd_timer               32768  1 snd_pcm
snd                     73728  7 snd_compress,snd_timer,snd_bcm2835,snd_soc_core,snd_pcm
```

```
sudo apt-get update
```

```
sudo apt-get install rpi-update
```

```
sudo rpi-update
```

- reboot

```
sudo apt-get install git bc libncurses5-dev bison flex libssl-dev
```

```
sudo wget https://raw.githubusercontent.com/notro/rpi-source/master/rpi-source -O /usr/bin/rpi-source
```

```
sudo chmod +x /usr/bin/rpi-source
```

```
/usr/bin/rpi-source -q --tag-update
```

```
rpi-source --skip-gcc
```

- this may take a while!! (Code coverage for fuzzing (KCOV) [N/y/?] (NEW) Enter this)

```
sudo mount -t debugfs debugfs /sys/kernel/debug
```

```
git clone https://github.com/PaulCreaser/rpi-i2s-audio
```

```
cd rpi-i2s-audio
```

```
sudo nano my_loader.c
```

Change: `.platform = „3f203000.i2s“`

and `.name = "3f203000.i2s"` to:

```
.platform = „20203000.i2s“
```

```
.name = „20203000.i2s“
```

```
make -C /lib/modules/$(uname -r )/build M=$(pwd) modules
```

```
sudo insmod my_loader.ko
```

Verify:

```
lsmod | grep my_loader
```

```
dmesg | tail
```

Take to autostart:

```
sudo cp my_loader.ko /lib/modules/$(uname -r)
```

```
echo 'my_loader' | sudo tee --append /etc/modules > /dev/null
```

```
sudo depmod -a
```

```
sudo modprobe my_loader
```

- **reboot**

```
sudo apt-get install python3-pip
```

Install pip3

Clone or Download web app

[GitHub - sams-project/sams-app](#)

- upload the project to the ftp
- directory install requirements from project

```
pip3 install -r requirements.txt
```

- **install RPi.GPIO**

```
sudo apt-get install rpi.gpio
```

- **install lib atlas (workaround for numpy)**

```
sudo apt-get install libatlas-base-dev
```

- **install Adafruit DHT**

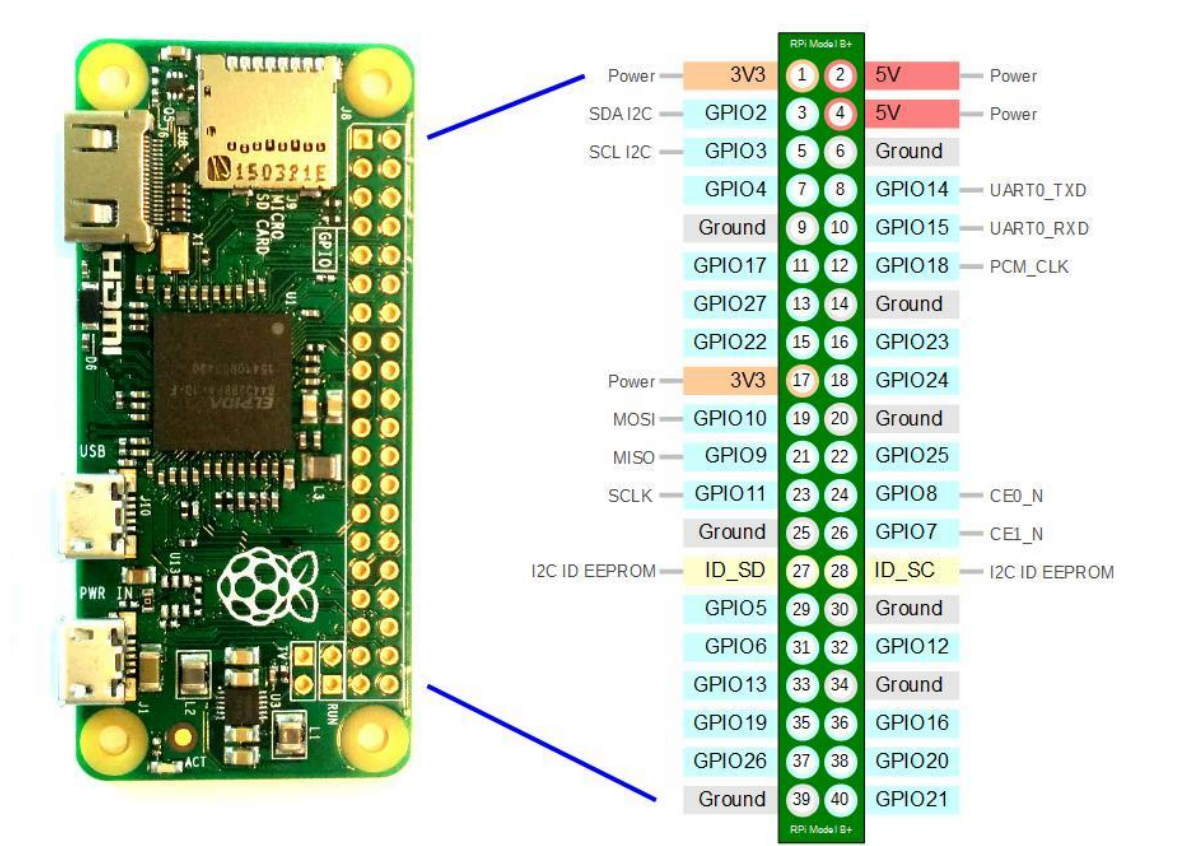
```
sudo pip3 install Adafruit_DHT
```

```
pip3 install sounddevice
```

```
sudo apt-get install libportaudio2
```

install sounddevice

III. GPIO places Raspberry Pi



(<https://raspberrypi.stackexchange.com/questions/83610/gpio-pinout-orientation-raspberrypi-zero-w>)

IV. Witty Pi Schedule Scripts **Measurement Interval – SAMS default**

Planning of the Witty Pi ON and OFF cycles to reduce energy consumption (within 24 hours)					Schedule Script Witty Pi Mini 3 /Witty Pi/schedules/ sams_default.wpi	
					BEGIN 2015-08-01 06:00:00 END 2025-07-31 23:59:59	
Hour	Minutes	Cycle	Time	State		
6:00	5	1	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	2	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	3	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	4	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
7:00	5	5	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	6	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	7	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	8	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
8:00	5	9	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	10	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	11	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	12	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
9:00	5	13	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	14	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	15	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	16	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
10:00	5	17	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	18	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	19	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	20	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
11:00	5	21	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	22	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	23	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	24	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	
12:00	5	25	5 On	ON	M5 WAIT	
	15		10 Off	OFF	M10	
	20	26	5 On	ON	M5 WAIT	
	30		10 Off	OFF	M10	
	35	27	5 On	ON	M5 WAIT	
	45		10 Off	OFF	M10	
	50	28	5 On	ON	M5 WAIT	
	60		10 Off	OFF	M10	

13:00	5	29	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	30	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	31	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	32	5 On	ON	M5 WAIT
	60		10 Off	OFF	M10
14:00	5	33	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	34	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	35	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	36	5 On	ON	M5 WAIT
	60		10 Off	OFF	M10
15:00	5	37	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	38	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	39	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	40	5 On	ON	M5 WAIT
	60		10 Off	OFF	M10
16:00	5	41	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	42	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	43	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	44	5 On	ON	M5 WAIT
	60		10 Off	OFF	M10
17:00	5	45	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	46	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	47	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	48	5 On	ON	M5 WAIT
	60		10 Off	OFF	M10
18:00	5	49	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	50	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	51	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	52	5 On	ON	M5 WAIT
	19:00 - 23:00		5h 10min	370	Off
0:00	5	53	5 On	ON	M5 WAIT
	15		10 Off	OFF	M10
	20	54	5 On	ON	M5 WAIT
	30		10 Off	OFF	M10
	35	55	5 On	ON	M5 WAIT
	45		10 Off	OFF	M10
	50	56	5 On	ON	M5 WAIT
	1:00 - 5:00		5h 10min	370	Off

Measurement Intervall - Witty Pi 3 Mini – Schedule Script – SAMS default router

Planning of the WittyPi ON and OFF cycles to reduce energy consumption (within 24 hours)					Schedule Script WittyPi Mini 3 /wittypi/schedules/sams_default_router.wpi						
BEGIN 2015-08-01 06:00:00 END 2025-07-31 23:59:59											
Hour	Minute s	Cycl e	Tim e	Stat e							
6:00	10	1	10	On	ON M10 WAIT	13:00	10	29	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	2	10	On	ON M10 WAIT		25	30	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	3	10	On	ON M10 WAIT		40	31	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	4	10	On	ON M10 WAIT		55	32	10	On	ON M10 WAIT
	60		5	Off	OFF M5		60		5	Off	OFF M5
7:00	10	5	10	On	ON M10 WAIT	14:00	10	33	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	6	10	On	ON M10 WAIT		25	34	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	7	10	On	ON M10 WAIT		40	35	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	8	10	On	ON M10 WAIT		55	36	10	On	ON M10 WAIT
	60		5	Off	OFF M5		60		5	Off	OFF M5
8:00	10	9	10	On	ON M10 WAIT	15:00	10	37	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	10	10	On	ON M10 WAIT		25	38	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	11	10	On	ON M10 WAIT		40	39	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	12	10	On	ON M10 WAIT		55	40	10	On	ON M10 WAIT
	60		5	Off	OFF M5		60		5	Off	OFF M5
9:00	10	13	10	On	ON M10 WAIT	16:00	10	41	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	14	10	On	ON M10 WAIT		25	42	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	15	10	On	ON M10 WAIT		40	43	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	16	10	On	ON M10 WAIT		55	44	10	On	ON M10 WAIT
	60		5	Off	OFF M5		60		5	Off	OFF M5
10:00	10	17	10	On	ON M10 WAIT	17:00	10	45	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	18	10	On	ON M10 WAIT		25	46	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	19	10	On	ON M10 WAIT		40	47	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	20	10	On	ON M10 WAIT		55	48	10	On	ON M10 WAIT
	60		5	Off	OFF M5		60		5	Off	OFF M5
11:00	10	21	10	On	ON M10 WAIT	18:00	10	49	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	22	10	On	ON M10 WAIT		25	50	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	23	10	On	ON M10 WAIT		40	51	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	24	10	On	ON M10 WAIT		55	52	10	On	ON M10 WAIT
	60		5	Off	OFF M5		19:00 - 23:00	5h 5min	37 0	Off	OFF H5 M5
12:00	10	25	10	On	ON M10 WAIT	0:00	10	53	10	On	ON M10 WAIT
	15		5	Off	OFF M5		15		5	Off	OFF M5
	25	26	10	On	ON M10 WAIT		25	54	10	On	ON M10 WAIT
	30		5	Off	OFF M5		30		5	Off	OFF M5
	40	27	10	On	ON M10 WAIT		40	55	10	On	ON M10 WAIT
	45		5	Off	OFF M5		45		5	Off	OFF M5
	55	28	10	On	ON M10 WAIT		55	56	10	On	ON M10 WAIT
	60		5	Off	OFF M5		1:00 - 5:00	5h 5min	37 0	Off	OFF H5 M5

IV. Links, Accounts, Passwords

Links:

Latest SAMS Image v. 2.47

(https://sams.science.itf.llu.lv/images/sams_image_2.47.zip)

W32 Diskimager

(<https://www.heise.de/download/product/win32-disk-imager-92033>)

SAMS PCB Gerber File

(<https://github.com/sams-project/sams-app/blob/master/gerber-sams-project.zip>)

BerryLAN App

(<https://play.google.com/store/apps/details?id=io.guh.berrylan>)

puTTY

(<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>)

SAMS Data Warehouse

(<https://sams.science.itf.llu.lv/>)

SAMS Data Warehouse manual

(<https://sams-project.eu/wp-content/uploads/2020/02/DW-data-sending-guide.pdf>)

SAMS Software and Error log on Github

(<https://github.com/sams-project/sams-app/issues>)

UUGear Witty Pi 3 Mini manual

(http://www.uugear.com/doc/Witty_Pi3Mini_UserManual.pdf)

UUGear Witty Pi Schedule Script Generator online:

(http://www.uugear.com/app/Witty_Pi-scriptgen/)

Protocol Hivelog

<https://www.hivelog.dk/>

Accounts & Passwords:

Mobile WiFi GSM Router (HUAWEI E5330).

(find default password on the backside of the covers lid)

Raspberry Pi. Login: pi, Password: samsrocks

V. Bug report

WiFi Connection via BerryLAN App: Problem: No SAMS device or, in the next step, no WiFi device is displayed. Reason: unknown. Solution: Restart the app and the Raspberry Pi, switch on GPS as well as switch off and on again Bluetooth on the smartphone. If possible, also switch off other Bluetooth and WiFi devices in the surrounding area.

Outlier values of weight data: Problem: Measured values sometimes jump several hundred grams within two measurements. Cause: Instability of the balance and movement due to wind and missing temperature compensation could cause the problem. Solution: Implement temperature compensation. Change the scale design. For example, use two to four load cells per weighing unit to increase stability and reduce vibration. Software-based error elimination.

Witty Pi status LED: The red LED is the indicator of output voltage (between +5V and GND pins in GPIO header), while the white LED is driven by the firmware and it could have different meanings. When the power is connected, the white LED blinks, and red LED stays off.

If you tap on the button, Witty Pi gives power to your Raspberry Pi, and both LEDs are on. After the boot Witty Pi's software sends out SYS_UP signal by pulling up GPIO-17 pin for a moment, the firmware received that signal and turns off the white LED.

When you tap the button while Pi is on, the white LED will light up, indicating the shutdown procedure has been started. After the shutdown is finished, the TXD pin will go LOW and Witty Pi will detect that, and cut the power after a delay. The white LED will also be turned off when the power is cut, and return to the blinking state.

If you see both LEDs are on, and Pi is actually offline, that most probably because your Pi was stuck at the shutdown procedure. If your Pi can not finish the shutdown, the TXD pin will never go LOW and Witty Pi will not cut the power (leaving both LEDs on). You may check the system log and see if there was something wrong during the shutdown.

When you see both LEDs are on, you can use multimeter to measure the voltage between TXD and GND. If it is still about 3.3V, then it is probably a shutdown error.

Witty Pi WAIT function: The function of premature shutdown using the WAIT syntax in the Witty Pi schedule script after finishing the measurement routine using the activated function `auto_shutdown = 1` in the online configuration in DW does not work in offline mode

VI. WiFi wpa_supplicant.conf

Instead of using BerryLAN, a file called wpa_supplicant.conf can be created in the boot directory of the SD card to set up WiFi on the Raspberry Pi. The WiFi is set up automatically the first time you start the device. The contents of the file are as described below.

```
country=ET
ctrl_interface=DIR=/var/run/wpa_supplicant
GROUP=netdev
update_config=1
network={
    ssid="SSID"
    psk="password"
    key_mgmt=WPA-PSK
}
```

VII. HW Config default

group:

name: defaultwittypi

timezone: Europe/Berlin

auto_update: 1

auto_shutdown: 1

ignore_error: 0

debug: 1

is_dht22: 1

is_scale: 1

is_microphone: 1

is_ds18b20: 1

app_median: 2

app_attempts_before_restart: 3

app_wait_seconds: 900

dht22_pin: 21

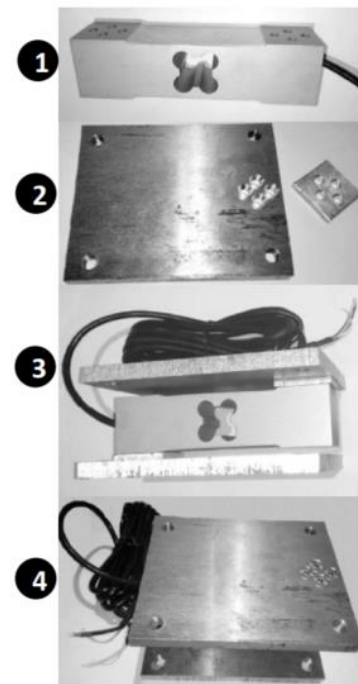
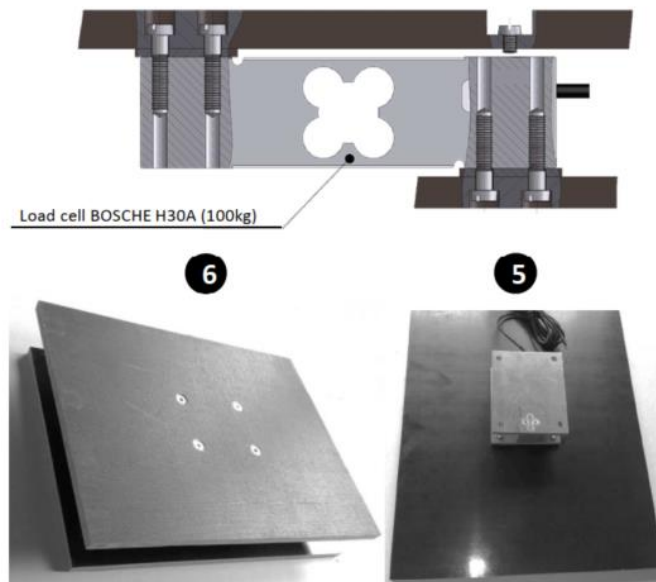
audio_duration: 10

audio_fs: 8000

wittyPi: |

[...] (Appendix IV)

VIII. Scale setup



IX. Quick Start Guide

You can also download the SAMS Quick Start Guide as .pdf file on our [Github page](#):

SAMS Monitoring System

Quick Start Guide

Universität Kassel
Sascha Fiedler
sascha.fiedler@uni-kassel.de
<https://sams-project.eu>
<https://github.com/sams-project>
Version 1

Abstract

The SAMS Quick Start Guide outlines the steps for getting the SAMS Monitoring System up and running. The guide is split into sections. Each section contains step-by-step instructions. Make sure to follow the steps with the guide to avoid any problems during your first time with SAMS. For a more detailed description, please refer to the Detailed Operating Instructions [\[here\]](#)

Contents

1	System Setup	2
1.1	System case	2
1.2	Hardware Construction.	2
2	Sensor Setup	2
2.1	Sensor frame	2
2.2	Hardware construction	2
3	Software Setup	3
3.1	What you need	3
3.2	Start System	3
3.3	Set up Wifi	3
3.4	Calibrate system	3
3.5	Change Settings on RaspberryPi	3
3.6	Set up Node in Data Warehouse (DW)	3
3.7	Check Data Stream in DW.	3
3.8	Report bugs	4
3.9	Space for Notes	4

1 System Setup

1.1 System case

3D printed:	download the case.stl file [here] download the case lit.stl file [here] prepare *.gcode files with Slicer-Software for your 3D-printer and print we recommend ABS filament
DIY case:	instead of the 3d printed case you can also create your own case, make sure your system is protected against environmental influences we recommend a food storage container
→ Chapter 2.3	You can find detailed information in the main manual in chapter 2.3 Case and Cables

1.2 Hardware Construction

soldering:	get your PCB gerber file [here] solder the components to the designated slots of the PCB wire and/or solder the Screw terminal blocks with the power supply plug, the plug for the sensor frame and the scale
assembly:	screw the finished system into the housing assemble the plugs
→ Chapter 2.1, 2.2	You can find detailed information in the main manual in chapter 2.1 Components and 2.2 Circuit plan and Printed Circuit Board (PCB)

2 Sensor Setup

2.1 Sensor frame

3D printed:	download the sensorframe frontplate.stl file [here] download the sensorframe middleplate.stl file [here] download the sensorframe backplate.stl file [here] prepare *.gcode files with Slicer-Software for your 3D-printer and print we recommend ABS filament
DIY frame:	instead of the 3d printed frame you can also create your own frame make sure your frame is suitable for your hive we recommend to pay attention to the beespace
→ Chapter 2.4	You can find detailed information in the main manual in chapter 2.4 Sensor Placement.

2.2 Hardware construction

soldering:	solder the components (DHT22, DS18B20 and SP0645) to the plug
assembly:	assemble the sensors on their designated spots of the printed frame assemble a fine wire mesh between the frame and sliding cover screw the sensor frame onto one side- and the backplate onto the other side of a broodcomb
→ Chapter 2.4	You can find detailed information in the main manual in chapter 2.4 Sensor Placement.

3 Software Setup

3.1 What you need

Before you start: Have a 16 GB SD card ready
Download SAMS Software Image [here]
Install W32 Diskimager or similar and write image to SD card [here]
install BerryLan App on your smartphone [here]
provide Wifi login details
provide ID and secret: SIGN UP to get credentials here: [here]
precisely defined weight (>7kg)
Install PuTTY or similar SSH client [here]

3.2 Start System

start SAMS System: Insert SD card into Raspberry Pi of the SAMS system
connecting the SAMS system to power (starting)
wait ~ 30 sec for booting

3.3 Set up Wifi

BerryLan: after the boot process, start BerryLan on your smartphone
switch on Bluetooth, GPS and follow the setup to establish Wifi access
Note the IP address

3.4 Calibrate system

Start calibration interface: start the browser on your PC or Smartphone and enter the IP address
followed by port 5000 as follows: e.g. 192.168.1.38:5000. Follow the
calibration setup. Reboot.

3.5 Change Settings on RaspberryPi

SSH: establish SSH connection to Raspberry Pi. To do this, use PuTTY
or similar on your PC and connect to IP address (e.g. 192.168.1.38).
Terminal opens.
Access: use the following Login to get access to your Raspberry Pi:
Login: pi
Password: samsrocks
Extend SD card size: go to Terminal 'sudo raspi-config' - Advanced settings - Expand Filesystem.
Systemtime: Set your geographical time (if offline) via Terminal enter 'sudo raspi-
config' - Timezone

3.6 Set up Node in Data Warehouse (DW)

create Account: open the DW page in the browser [here] and set up the user or use existing
access. To create Device with ID, click on tab 'Node' and 'Register SAMS
Hive'. Then enter ID, description and location. Only one ID for each
Device.

3.7 Check Data Stream in DW

click: on tab 'Devices' – Last event
→ Chapter 4 You can find detailed information in the main manual in chapter 4 Data
Warehouse.

3.8 Report bugs

Get in touch: Please report bugs and ideas and log them [\[here\]](#) with the relevant parameters so that we can further adapt the software. Find a solution for your Problem [\[here\]](#). Thanks for your support and stay tuned [\[here\]](#)

3.9 Space for Notes

This image shows a single page of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

X. Manual for ESP8266 based system set up

This chapter describes the system set up hardware and software wise for ESP8266 based HIVE system.

ESP8266 and sensor connections in general

To measure temperature, humidity and weight following GPIO pins of the ESP8266 based boards with the sensors/ sensor modules need to be connected:

- DS18B20 temperature sensor:
 - VDD -> 3...3.3V
 - DQ -> GPIO14
 - GND -> GND
 - Resistor (2k...4.7k) between VDD and DQ pin
- DHT22 temperature/ humidity sensor:
 - VCC -> 3...3.3V
 - DAT -> GPIO12
 - GND -> GND
 - Resistor (2k...4.7k) between VDD and DQ pin
- A/D converter based on HX711 IC:
 - VCC -> 3...3.3V
 - SCK -> GPIO5
 - DT -> GPIO4
 - GND -> GND
 - E+ -> Red wire of Load cell
 - E- -> Black wire of Load cell
 - A- -> White wire of Load cell
 - A+ -> Green wire of Load cell

Since the ESP8266 board will be using deep sleep mode, GPIO16 (the “wake” pin) needs to be connected to the RST pin, otherwise, the microchip will not be able to “wake up” and run continuously. The connection of the ESP8266 itself is as follows:

- RST -> GPIO16

It also should be mentioned that different boards are naming the pins differently, but the original numbering (GPIOx) is the same for all boards. While the ESP8266 with an adapter board uses the GPIO numbering style, NodeMCU and D1 Mini boards use different style, by putting the letter “D” (meaning that these pins are digital) followed by a number. The single analog pin is numbered as “A0”. When connecting sensors to the according boards, follow the instructions, board pinouts described below. If using a completely different ESP8266 board (not described in this section) it is advised to look up the correct pin numbering scheme.

Hardware set-up

Using NodeMCU

The sensor connection diagram to NodeMCU is illustrated in Figure 1x. When using development platforms, like NodeMCU or D1 Mini there are not a lot of components required,

Only resistors (see Fig. 1x. R1 and R2) for correct sensor operation. These are pullup-resistors for DS18B20 temperature sensor and DHT22 temperature/ humidity sensor. The values for these resistors can be selected in the range from 2k Ω to 4.7k Ω . Some sensor modules have these resistors built-in already on the sensor board (usually the DHT22 modules). In this case, no extra pull-up resistor is necessary. If using multiple DS18B20 sensors with longer cables, the resistor value should be close to 2k Ω , since the cable length (introduces capacitance) might have impact on the data signal. In cases with long line 1-Wire networks, refer to according guidelines (<https://www.maximintegrated.com/en/design/technical-documents/tutorials/1/148.html>).

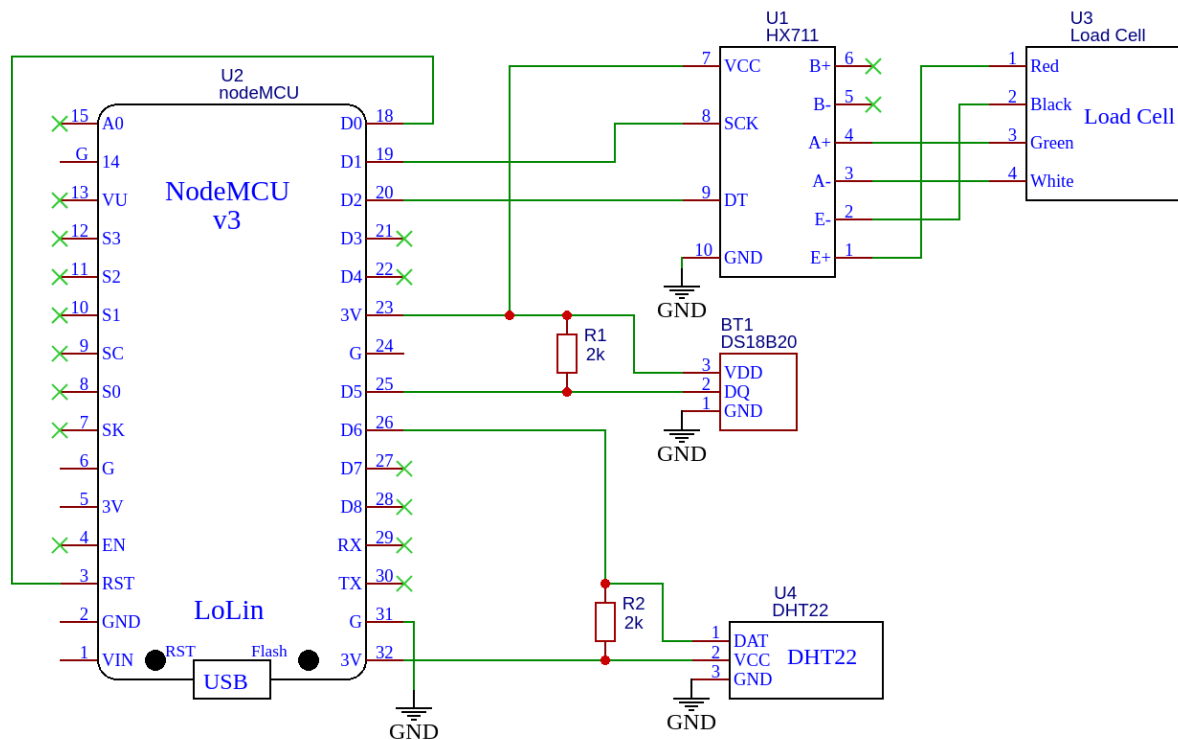


Figure 1x. NodeMCU and sensor connection diagram

To power NodeMCU two methods can be used - power from a battery pack via VIN and GND connection or by using a power bank and powering from a USB port. According to (<https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet>) and (<https://diyi0t.com/what-is-the-esp8266-pinout-for-different-boards/>) the VIN accepts voltages between 7 and 12V.

When powering the NodeMCU from a power bank via USB port, it should be mentioned, that during the deep sleep mode, the ESP8266 consumes very little power, therefore the power bank might shut down due to inactivity. This is also highly dependent on a power bank and the sleep time programmed within the NodeMCU.

Using ESP8266 with an adapter board

When the system is built around ESP8266 with an adapter board on a pcb, a lot more components are needed, e.g., two pushbuttons to ensure a convenient way to enter the ESP8266 microchip into bootloader mode (prepare the chip for software upload), low quiescent

current voltage regulator, voltage supervisor. To facilitate the building process of such a monitoring system, a PCB was designed (see Fig. 2x.).

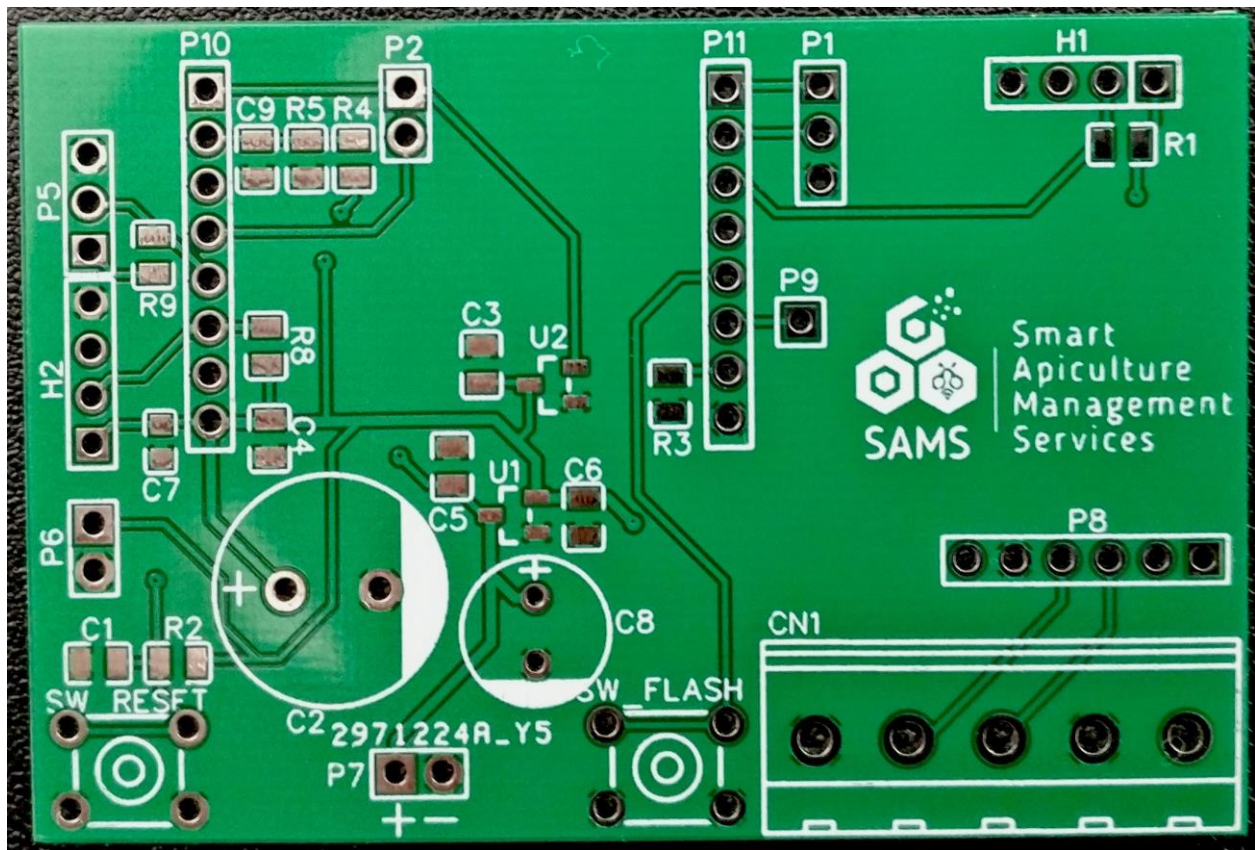


Figure 2x. PCB for monitoring system based on ESP8266 with an adapter board

Therefore, the components like, capacitors, resistors, pin headers, buttons have a dedicated place on the PCB and no extra wiring is required. Every component on the board is numbered and can be referred to the components's table (see Table 1x).

Table 1x. Parts list for PCB

Part Name and value	Designator	Footprint	Quantity
Voltage regulator: MCP1700T-3302E/TT	U1	SOT-23-3_L2.9-W1.6-P1.90-LS2.8-BR	1
Voltage supervisor: TPS3839G33DBZR	U2	SOT-23-3_L2.9-W1.3-P1.90-LS2.4-BR	1
Resistor: 1K	R3	C0805	1
Resistor: 2K	R8,R9	C0805	2
Resistor: 10K	R2,R1	C0805	2

Resistor: 82K	R5	C0805	1
Capacitor: 100nF	C7	C0603	1
Capacitor: 100nF	C4,C1	C0805	2
Capacitor (electrolytic): 47uF	C8	CAP-D8.0XF3.5	1
Capacitor (electrolytic): 2200uF	C2	CAP-TH_BD12.5-P5.00-D1.2-FD	1
Capacitor: 10nF	C9	C0805	1
Capacitor: 1uF	C5,C6,C3	C0805	3
Header-Female-2.54_1x2	P2	HDR-TH_2P-P2.54-V-M	1
HX711 (Header-Female- 2.54_1x4)	H1	DIP-1X4P2.54	1
HX711 (Header-Female- 2.54_1x6)	P8	HDR-6X1/2.54	1
ESP8266 (Header-Female- 2.54_1x8)	P10,P11	HDR-TH_8P-P2.54-V	2
PWR_IN (Header-Male- 2.54_1x2)	P7	HDR-TH_2P-P2.54-V-M	1
DHT22 (Header-Male- 2.54_1x4)	H2	HDR-TH_4P-P2.54-V	1
DS18_20 (Header-Male- 2.54_1x3)	P5	HDR-TH_3P-P2.54-V	1
UART (Header-Male- 2.54_1x3)	P1	HDR-TH_3P-P2.54-V	1
Load cell (WJ128V-5P-5.0- 14-00A)	CN1	CONN-TH_5P-P5.00_WJ128V-5P- 5.0-14-00A	1
Pushbutton: TS-1102-5016	SW_FLASH,SW_ RESET	KEY-6.0*6.0-2	2

To make the PCB more compact, SMD (surface-mount-device) type components were used. Additionally, a separate PCB was designed (see Fig 3x.) that uses through-hole components instead of the SMD.

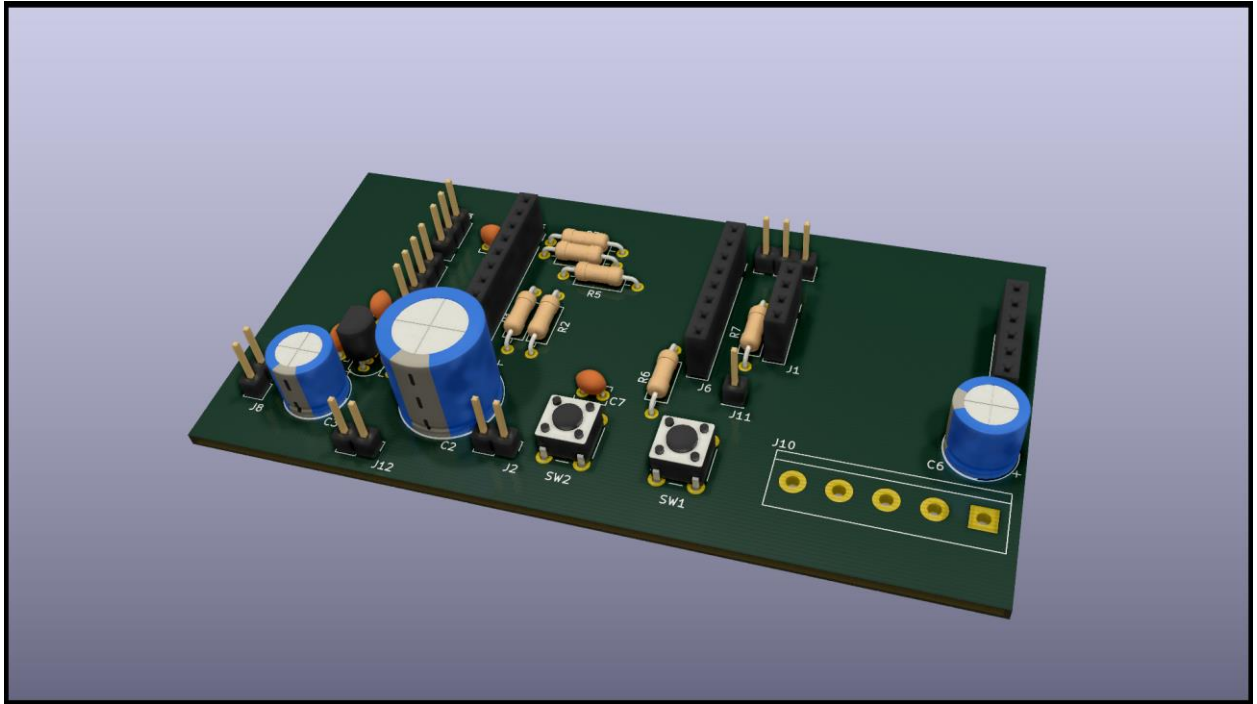


Figure 3x. 3D design of a PCB with through-hole components based on ESP8266 with an adapter board

Sensor connection to the PCB

Depending on the headers or sockets soldered on the PCB, the connection should look like depicted in Fig. 4x. Some load cells may provide 5 wires, where the 5th might be colored yellow or purple. This is the shield wire and helps remove noise when measuring the weight and should be connected to GND. Rest of the wires usually corresponds the following coloring scheme:

- Red: VCC
- Black: GND
- White: A- or O- or S-
- Green: A+ or O+ or S+

More information about load cell connection to the analog digital converter or load cell amplifier can be found at <https://learn.sparkfun.com/tutorials/load-cell-amplifier-hx711-breakout-hookup-guide/all>.

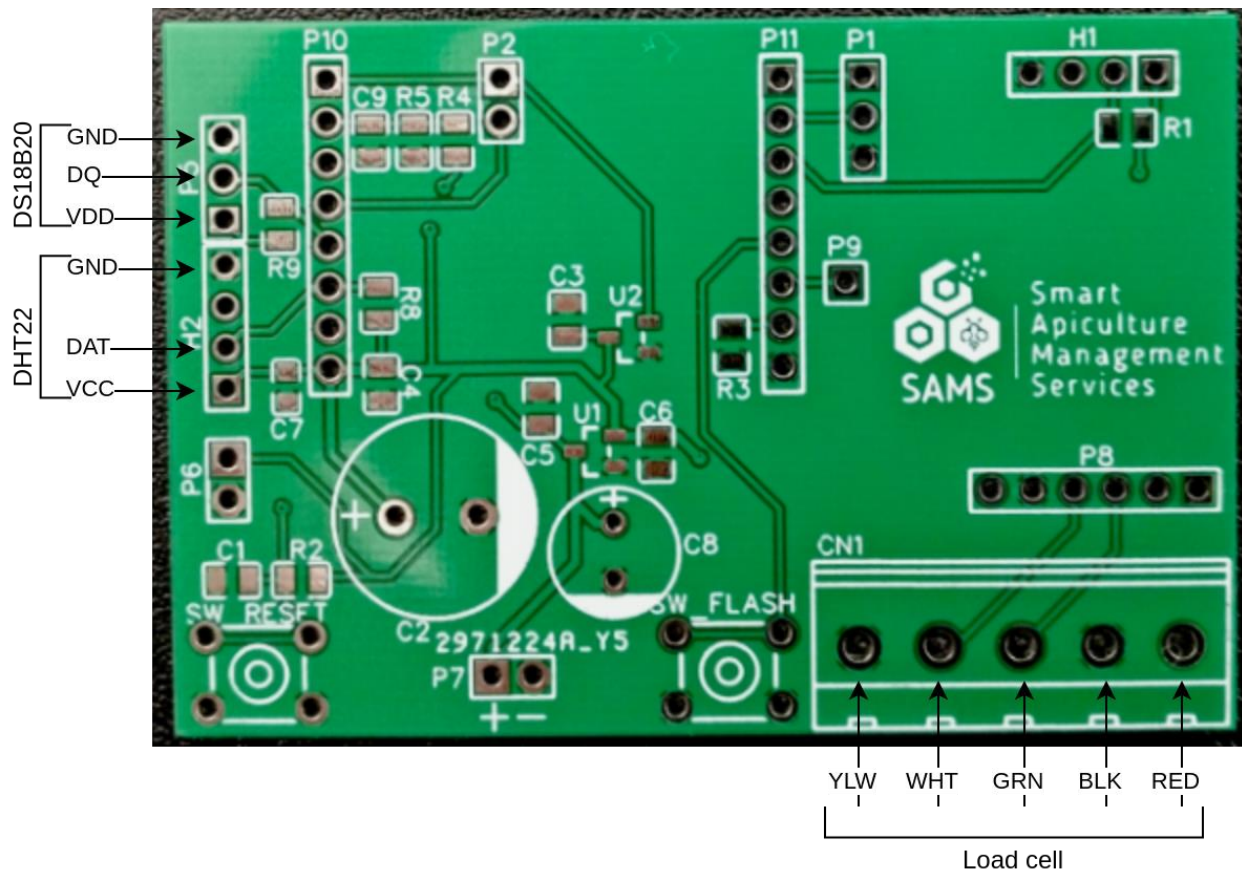


Figure 4x. Sensor connection

ESP8266 with an adapter board and A/D converter HX711 should be placed according to the Fig. 5x. They can be either soldered directly or plugged into a socket (that is soldered onto the PCB). Our recommendation is to solder sockets, so these modules could be easier replaceable if damaged.

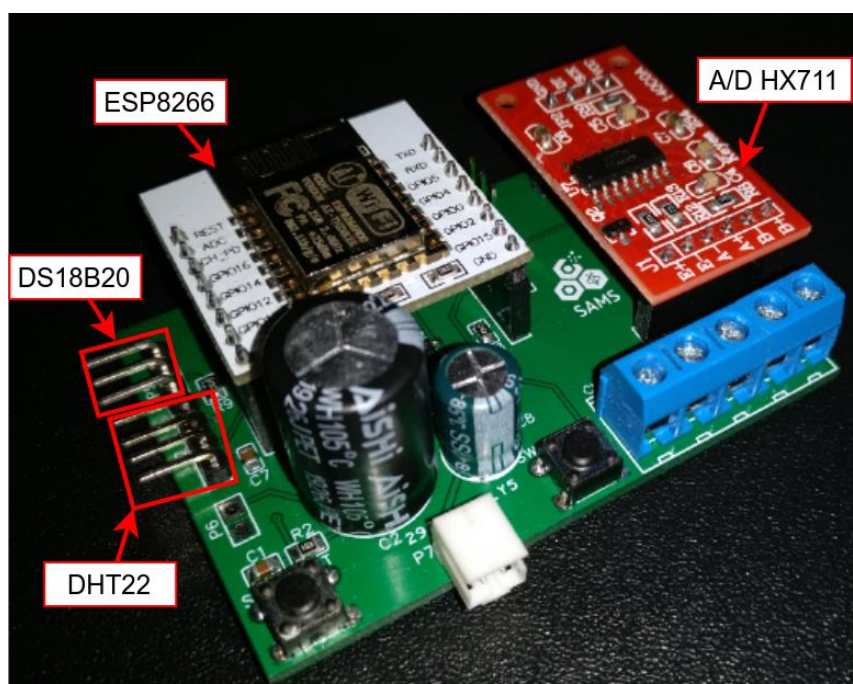


Figure 5x. Fully assembled ESP8266 based monitoring system

A casing for the PCB was also designed and 3D printed (see Fig. 6x.). The according 3D files will also be available publicly after the project ends.



Figure 6x. 3D case for ESP8266 based monitoring system

Software set-up

One of the solutions to upload software onto the ESP8266 microchip is to use the open-source Arduino IDE that can be downloaded from <https://www.arduino.cc/en/main/software>. By default, Arduino IDE does not support development for ESP8266 boards, therefore additional steps need to be taken to enable it:

- Open Arduino IDE select: `File -> Preferences` (see Fig. 7x).

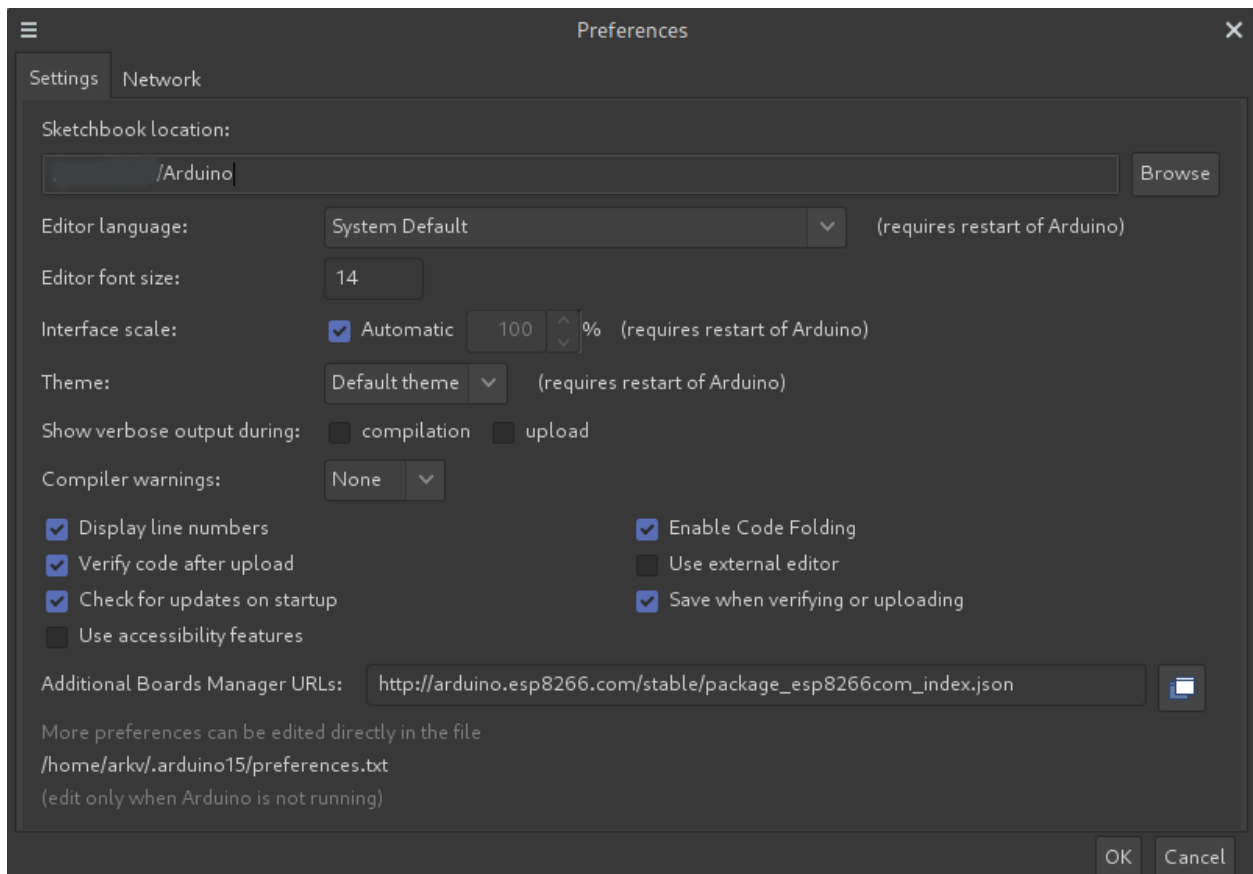


Figure 7x. Preferences window

- Locate field: “Additional Boards Manager URLs:” and press the icon on the right side (see Fig. 8x). After pressing the icon, a new window should appear (see Fig. 9x).

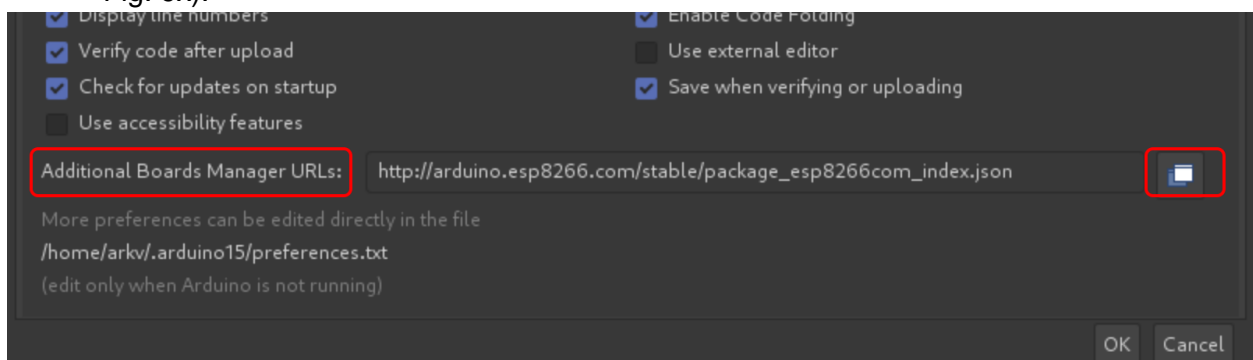


Figure 8x. Boards manager option

- In the text field of the window enter the following line and press “OK” button (see Fig. 9x):
http://arduino.esp8266.com/stable/package_esp8266com_index.json

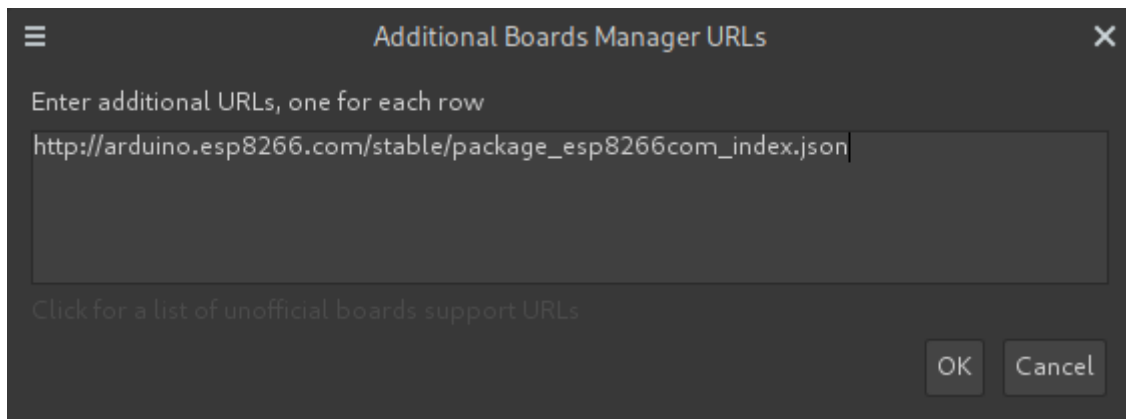


Figure 9x. Adding support for ESP8266 boards

- Close the opened preferences window. Next step is to install the necessary packages. To do so (see Fig. 10x.): Tools -> Board:"..." -> Boards Manager...

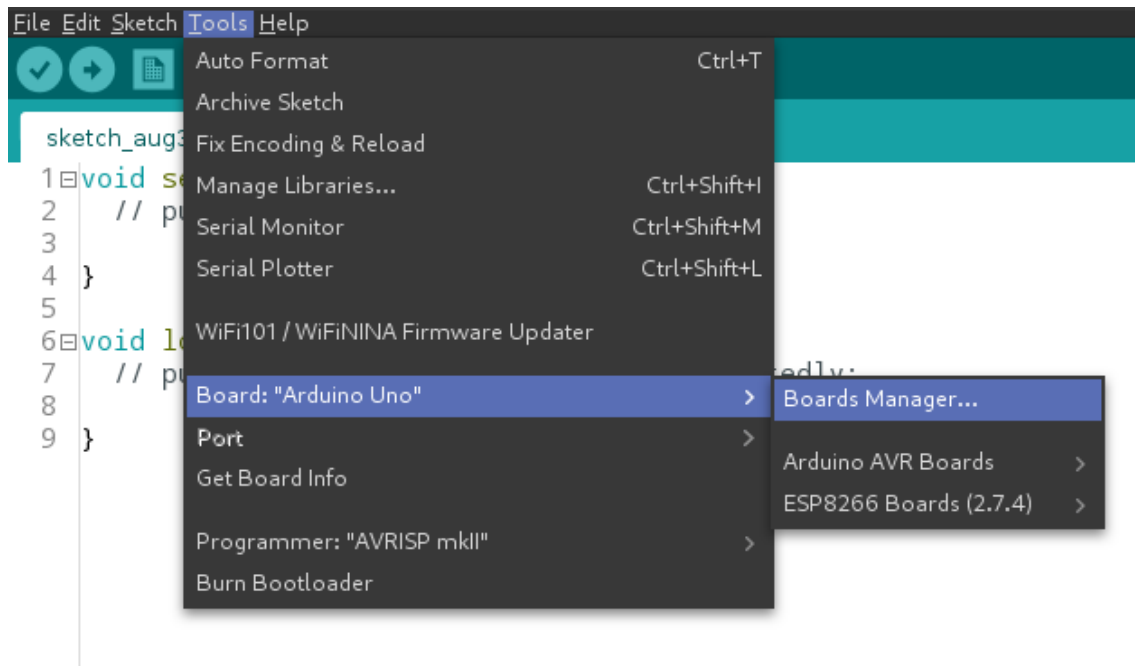


Figure 10x. Access to boards manager

- In the boards manager window's search field enter "esp8266" and press install under the "esp8266 by ESP8266 Community" (see Fig. 11x.).

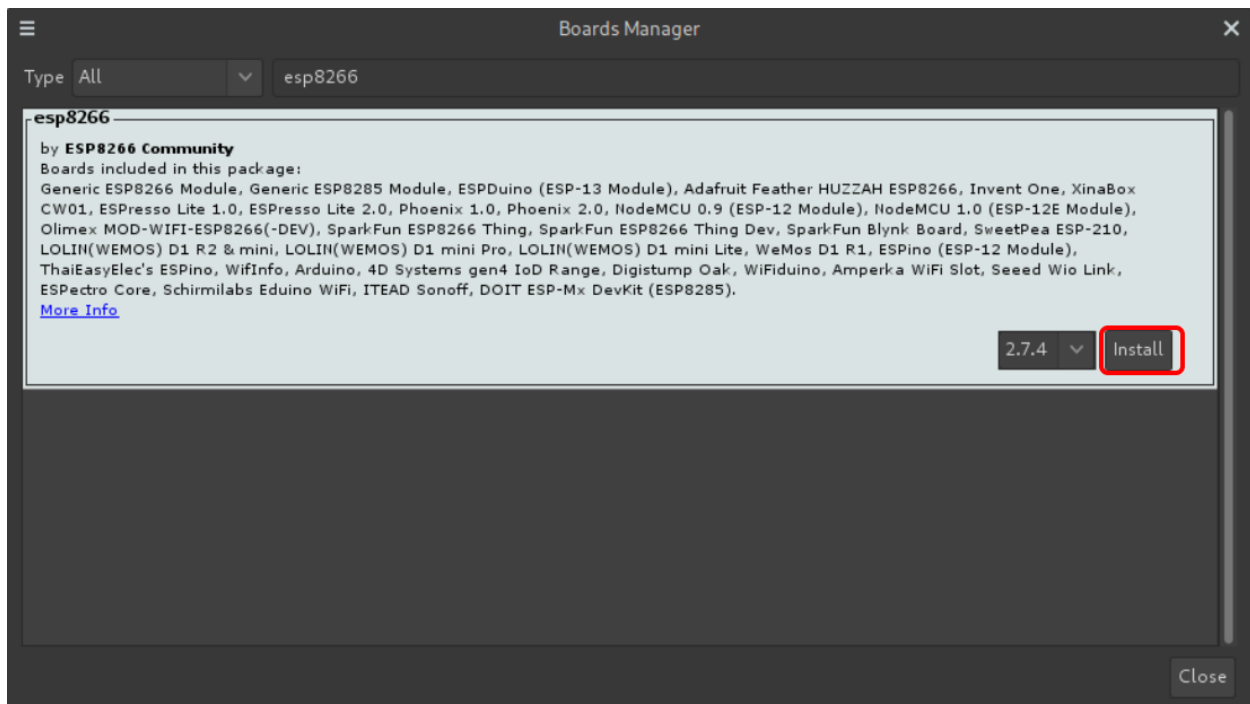


Figure 11x. Installation of ESP8266 boards

- In the Arduino IDE select: Tools -> Board -> ESP8266 Boards -> NodeMCU 1.0 (ESP-12E Module) (see Fig. 12x.).

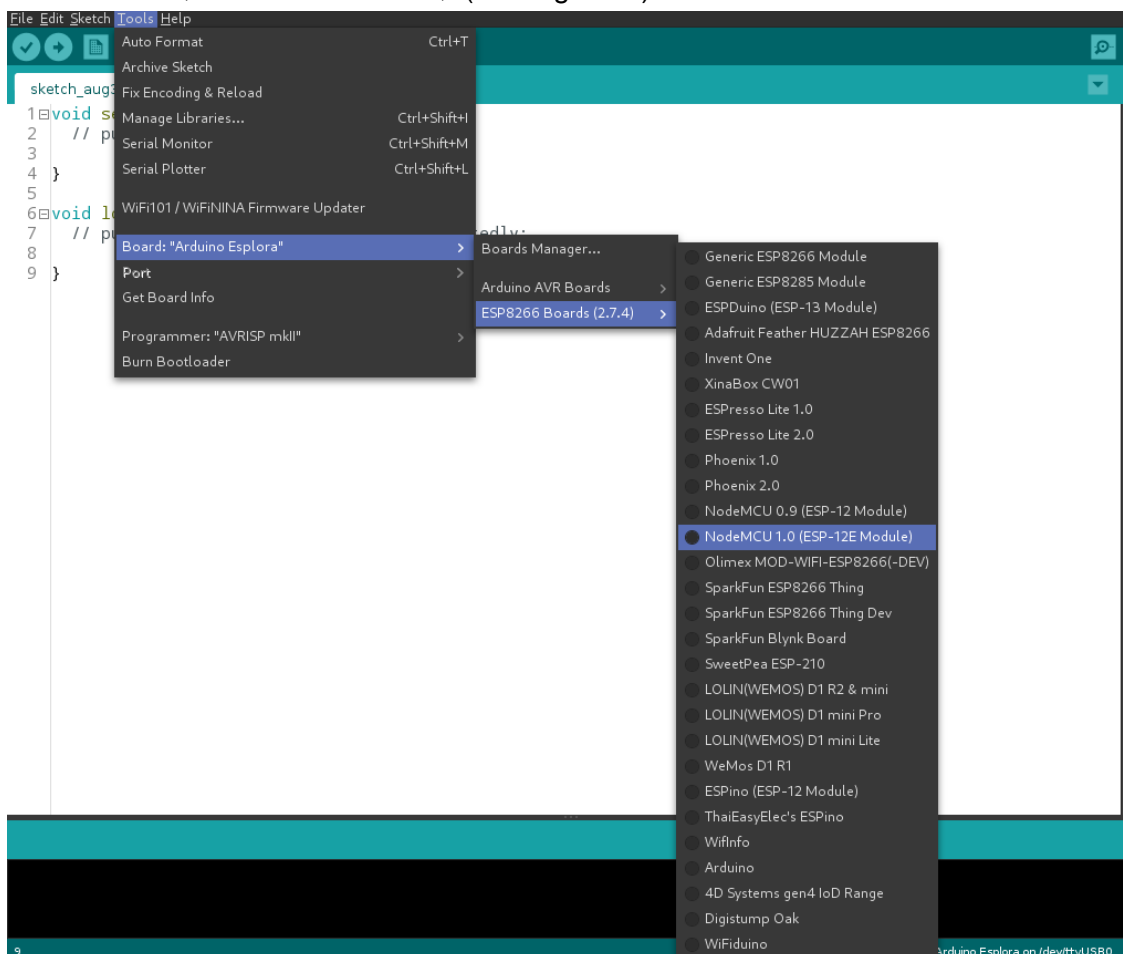


Figure 12x. Selection of NodeMCU board

- When the ESP8266 board (like NodeMCU, D1 Mini) is connected to the computer via USB port, under `Tools`, the port needs to be selected. The port number may be different on one's computer. On Windows operating systems, it should be named "COMx", where x is a number, greater than 1. After the proper port has been selected, the device is ready for software uploading.

Since some ESP8266 based boards (like Adafruit Huzzahh, ESP8266 with an adapter board) do not have a USB port, a USB to serial converter is needed, e.g., FTDI, MCP2200. These devices have "broken out" pins that are numbered as Tx, Rx, and GND. So the connection between the USB to serial converter and ESP8266 board is as follows:

- **Tx** of the USB to serial converter goes to **Rx** of the ESP8266 board;
- **Rx** of the USB to serial converter goes to **Tx** of the ESP8266 board;
- GND goes to GND.

The port in the Arduino IDE software will be numbered similarly like in the NodeMCU case. To upload the code to these ESP8266 devices requires putting them into bootloader mode, that can be done by holding down the FLASH button while pressing the RESET button on the PCB, where these pushbuttons are labeled as SW_FLASH and SW_RESET, accordingly (see Fig. 4x., Fig. 5x.).

ESP8266 based monitoring systems code uploading

The code for ESP8266 based monitoring system will be open-source and publicly available via github.

Library installation

For proper operation of the software some additional code libraries need to be added. These libraries are:

- DHT sensor library (needed to read DHT22 sensor's value);
- HX711 Arduino Library (needed to measure weight);
- RTCVars (needed to manage information saving in RTC memory).

To add these libraries, open: `Sketch -> Include Library -> Manage Libraries...`

A library manager window should appear (see Fig. 13x.).

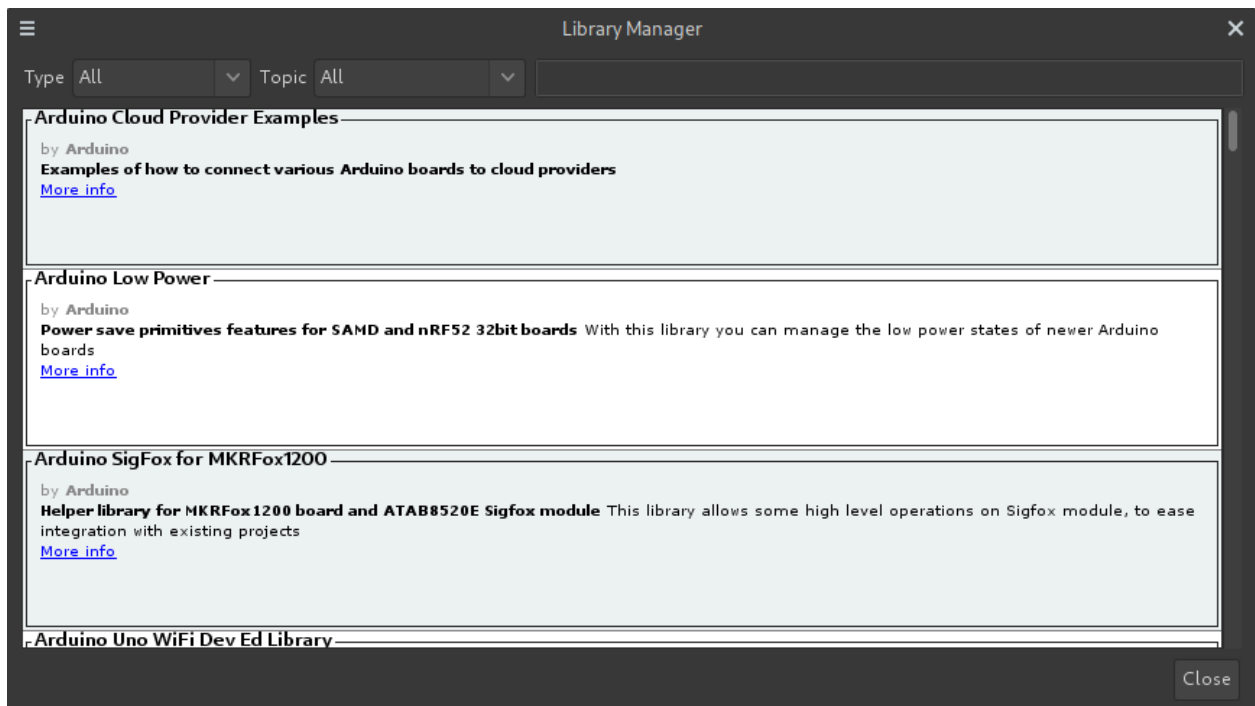


Figure 13x. Library manager window

To install needed libraries use the search bar, by inserting the name of the libraries.

- **DHT sensor library**

When installing this library, select “DHT sensor Library by Adafruit” (see Fig. 14x.). During the installation, user will be asked to install “Adafruit Unified Sensor” library, this library should be installed as well.

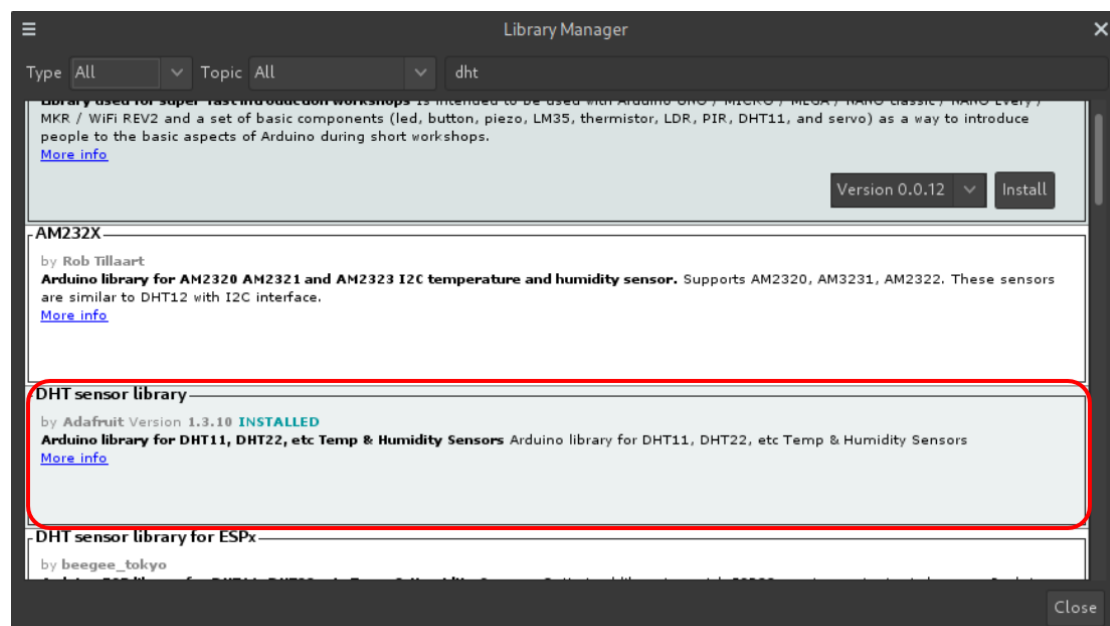


Figure 14x. Installation of DHT library

- **HX711 Arduino Library**

To install this library, enter “hx711” in the search bar and select “HX711 Arduino Library by Bogdan Necula, Andreas Motl” (see Fig. 15x.).

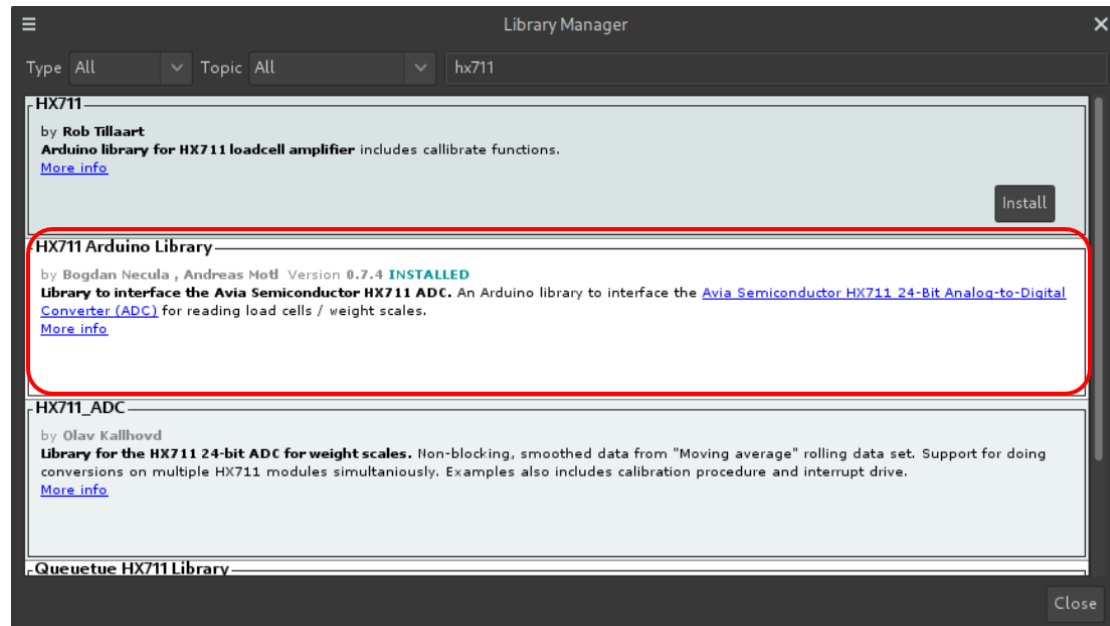


Figure 15x. Installation of HX711 library

- **RTCVars library**

To install this library, enter “rtcvvars” in the search bar and select “RTCVars by Lars Friedrichs” (see Fig. 16x.).

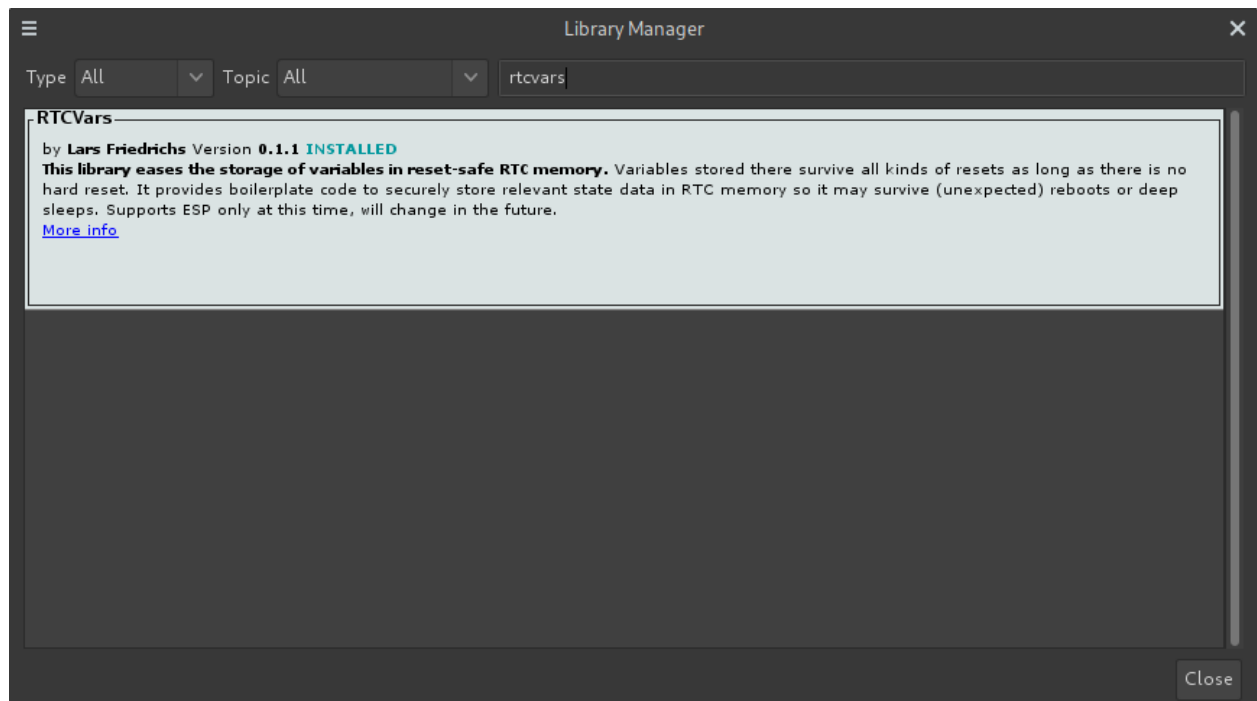


Figure 16x. Installation of RTCVars library

Software changes

The software that will be available after the project end, will need some adjustments for every ESP8266 board. These adjustments require changing the `client_id` and `secret`, and WiFi credential information (see Fig. 17x.). The `client_id` and `secret` must be acquired from SAMS DW maintainers (currently UNILV partners).

```
10 //Only change WiFi and auth0 credentials
11 //WiFi credentials
12 const char* ssid = "wifi ssid";
13 const char* pwd = "wifi password";|
14
15 //auth0
16 String client_id = "client id";
17 String secret = "secret";
18
```

Figure 17x. Credentials that need changing in the code

Other changes are related to scale calibration in order to obtain correct weight values. In the software these are called `scaleFactor` and `scaleOffset`. A separate program needs to be run first on the ESP8266 based system, to obtain these values. This program will also be available as open source via github. The load cell calibration software, when run, contains a menu like (see Fig. 18x.) interface that leads the user through the calibration process and at the end will show the `scaleFactor` and `scaleOffset` values.

```
1 - set known weight | current: 5000.00g;
2 - calibrate;
3 - measure;
q - show menu;
Choice (1-3):
```

Figure 18x. Load cell calibration menu

These are the following steps necessary to calibrate the load cell:

- To set calibration weight, enter **1** and press enter key. Input the calibration weight in grams, for example, if 20kg will be used then enter 20000 and press enter. By default, this is set to 5kg or 5000g.
- To calibrate the scale, enter **2** and press enter key. The following text should show up:
Reset

Place weight and enter 'w'!

- Put the calibration weight on the scales and enter the 'w' key. After the calibration process is done, the following text should show up:

scaleFactor: xxxxx

scaleOffset: yyyy

Calibration finished. Remove weight!

- Enter the xxxxx and yyyyy values in the corresponding places in the monitoring software. An example of `scaleFactor` 20.80 and `scaleOffset` 109910 is given in Fig. 19x.

```
42 |const float scaleFactor = 20.80;  
43 |const long scaleOffset = 109910;|
```

Figure 19x. Scale calibration value example

To upload the software onto ESP8266 board, press the arrow button (see Fig. 20x.) in the Arduino IDE software.

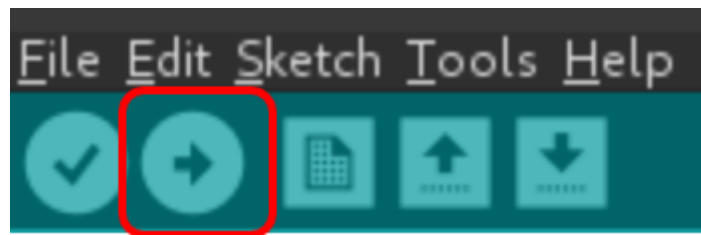


Figure 20x. Button to upload code to ESP8266 board

If there are no errors, the Arduino should show the following in the field at the bottom of the screen (see Fig. 21x.):

```
Done uploading.  
MAC: dc:4f:22:75:8d:cc  
Uploading stub...  
Running stub...  
Stub running...  
Configuring flash size...  
Auto-detected Flash size: 4MB  
Compressed 398128 bytes to 293260...  
Wrote 398128 bytes (293260 compressed) at 0x00000000 in 25.9 seconds (effective 123.2 kbit/s)...  
Hash of data verified.  
  
Leaving...  
Hard resetting via RTS pin...
```

Figure 21x. Upload complete message

Project website: www.sams-project.eu

Project Coordinator contact:

Stefanie Schädlich
Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ) GmbH
Wielinger Straße 52
82340 Feldafing, Germany
stefanie.schaedlich@giz.de



DISCLAIMER

Neither GIZ nor any other consortium member nor the authors will accept any liability at any time for any kind of damage or loss that might occur to anybody from referring to this document. In addition, neither the European Commission nor the Agencies (or any person acting on their behalf) can be held responsible for the use made of the information provided in this document.