hiveScale

Documentation

Document Version History

Author: Dirk Fischer, engineer@schwedenhof.net, www.schwedenhof.net

Version	Date	Comment
1.0	2021-03-22	Initial - DFi

Content

Overview

HiveScale is a weight data logger for beehives (or other objects). The logging interval is configurable, typically set to 5minutes. Logged data is transmitted wireless to a server over HTTP POST method, using a GPRS modem. The upload interval is configurable, typically set to 24 hours (in order to save energy and extend battery life time). Main design goal was a simple and cost efficient power supply. With a 6V, 4.5Ah battery (smaller than a beverage can) the scale can be operated for over one year without recharge.

This document is an incomplete collection of information about the project, intended for users and developers who want to implement the scale by themselves. Technical knowledge is required. The author my answer questions, but does not generally provide support.

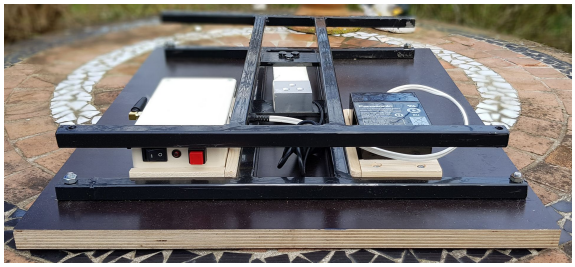
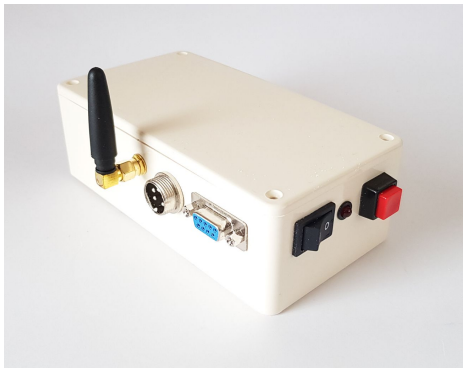
The firmware of hiveScale is published under GNU General Public License v3.0. It is hosted on github: https://github.com/schwedenhof/hiveScale
A STM32CubeMX and STM32CubeIDE project is part of the distribution, as well as documentation.

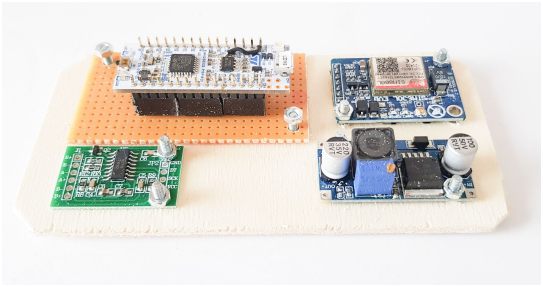
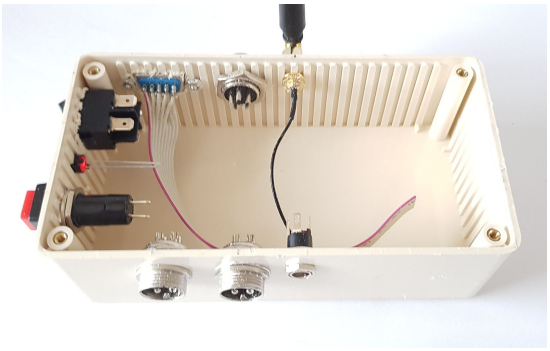
Please contribute to the project! There are tasks for hardware and software developers. Check the ToDo List for ideas. If you are interested, please contact the author: Dirk Fischer, engineer@schwedenhof.net, www.schwedenhof.net

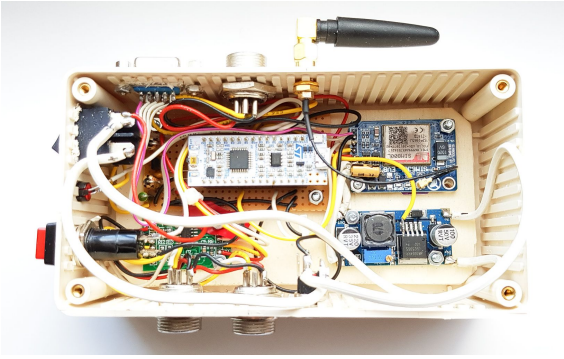
Features

- Scale with weight logger for beehives
  - configurable (e.g. 5 minutes) weight interval ⟶ e.g. 290 measurements a day, limited by available SRAM of Microcontroller
  - optional 2nd channel for 2 scales at one controller
  - based on HX711 24bit ADC
- Battery driven, no mains power supply required, 6V-12V input
- Low Power Design
  - allows simple, low maintenance and cost-efficient power supply
- Battery voltage monitoring, transferred together with weigth data
- Optional temperature sensor interface for DS18B20 sensor
- Configurable (e.g. once a day) data transfer to a private server or public clouds like ThingSpeak, limited by power budget (battery capacity)
- No WIFI required, collected data is transmitted via GPRS modem SIM800 module
- HTTP POST method is used for data transmission. Data may be received and stored by a PHP script on a private server. Alternatively a public cloud service like Thingspeak may be used.
- User interface: 1 status LED, 1 user button, UART terminal interface CLI - command line interface
- Cost-efficient, total cost: starting at €90
  - electronics, incl. housing ~€40 ... €60
  - load cell ~€15 ... €60
  - mechanics ~ €15 ... €60
  - 6V, 4,5Ah Battery ~ €10

Images

	Assembled frame with main components • Controller with STM32 MCU • load cell • battery • metal frame • wooden board – bottom • wooden board – top is missing
	Assembled frame with main components • Controller • load cell • battery • metal frame • wooden board - bottom • wooden board – top is missing
	Controller interfaces from left to right: • GPRS Modem Antenna • DS18B20 plug • Diagnosis Port UART TTL • power switch • Status LED • Control Button

	Controller interfaces from left to right: • power switch • Status LED • Control Button • Load cell plug channel A • Load cell plug channel B • Power Supply plug
	Controller interfaces from left to right: • Load cell plug channel A • Load cell plug channel B • Power Supply plug
	Controller base board with 4 modules • STM32 MCU board NUCLEO-L031K6 (white) • HX711 weight sensor module (green) • SIM800 modem (blue, upper right) • LM2596 DC/DC power supply for SIM800 modem (blue, lower right)
	Empty controller housing

	Controller housing with base board and wired components

ToDo

Both, hardware and software are working and doing their job robustly. However there are many ideas for improvement.

Hardware

- Design of a mainboard PCB to host the MCU Nucleo board, modules and connectors for the external plugs
- Or design of a full custom single PCB with MCU, HX711, GPRS modem and power supply. In this case another MCU derivative, like STM32L071KZ, with more memory could be used to gain freedom for more software functionality. STM32L031K6 features only 32KB flash memory, which is a little bit tight.

Software

The hiveScale application is not finalized yet. It may be regarded as a prototype firmware. There are many things to fix, tidy up, complete and to add.

In somehow prioritized order:

1. auto app start after 30 minutes run mode
2. swarm alarm email or SMS
3. Send daily summary eMail at midnight with
   absolute weight, weight delta, battery voltage
4. improve HX711 measurement routine with mean value calculation
5. Implement startup detection and error handling for SIM800 wakeup, instead of a fixed waiting time after power on. Depending on the local GPRS network, startup time may vary.
6. Support 2nd weight channel logging
7. Tidy re-implementation of UART handler, adding synchronization mechanisms (e.g. IRQ locking)
8. Some codesize optimization measures were done already, e.g. tiny printf.c implementation (https://github.com/mpaland/printf), however there might be more potential for optimization.
9. Support DS18B20 temperature sensor. Hardware prepared already, but software is missing.
10. Add support for ThingSpeak Cloud, in addition to private server upload
11. Improve data presentation on webserver with modern javascript libraries

12. Setup a modern dashboard on server side, based on a framework like grafana
13. Add permanent scale mode: hiveScale stays in active mode and uploads weight in short time intervals. This mode can be used to weight honey frames during harvest
14. PC GUI for commissioning → remanent data file editor
15. add SMS service in addition to server upload

For users

Steps to setup a device

1. assemble controller electronics
2. assemble scale hardware, including mechanics and battery
3. download firmware file into flash of microcontroller via ST-Link USB
4. download configuration (remanent data) into flash of microcontroller via ST-Link USB
   alternatively, all configuration data can be set via UART terminal
5. prepare a webserver for data reception and presentation. This task is not covered in this document

Required software tools

- STM32 ST-Link Utility – flash programmer
- TeraTerm - terminal

Firmware download

Use STM32 ST-Link Utility Software from STMicroelectronics to program the firmware file "GitHub/hiveScale/Release/hiveScale.bin" into flash memory of the microcontroller at memory address 0x0800 0000.

The Nucleo board features an integrated ST-Link USB/SWD adapter. You can connect a Micro USB cable directly to the board. No external ST-Link is required

Configuration data download (remanent data)

Use STM32 ST-Link Utility Software from STMicroelectronics to program the example remanent data file "GitHub/hiveScale/remanent_data.bin" into flash memory at address 0x0808 0000.

You may edit and adapt the remanent data to your needs with a hex editor, before downloading the file into the flash memory.

Remanent data can also be set and stored via UART terminal commands.

The Nucleo board features an integrated ST-Link USB/SWD adapter. You can connect a Micro USB cable directly to the board. No external ST-Link is required

Commissioning via UART terminal

For commissioning, a PC/Laptop with USB interface is required. It must be connected to the UART Service Interface of hiveScale via an USB/TTL cable (e.g. FTDI). A terminal program, like TeraTerm, can be used to communicate with hiceScale and configure parameters and to calibrate the scale.

Set configuration parameters (remanent data)

The SIM800 modem requires communication parameters (server address, etc.). The scale needs to be calibrated. Log interval can be configured, etc. All these parameters must be set and stored permanently in the device. Remanent data is stored inside the integrated EEPROM at memory address 0x0808 0000.

For setting parameters use the terminal command "`setp`".
Syntax: `setp parameter_number value`

After setting all required parameters, call the terminal command "`rems`" to store the remanent data permanently in EEPROM.

The terminal command "`remi`" print all remanent data to the console.

Som parameters like "scale tara", "scale calibration" and the battery voltage monitor calibration "ADC low" and "ADC high" can be set by using dedicated terminal commands or the user button. This is necessary because determining of these parameters require reference measurements.

The strings "APN" and "URL" are related to the HTTP POST communication. APN is the network access point, for example "web.vodafone.de". It depends on your SIM card provider. The appendix of this document contains a list of APNs for various providers.
The URL is the internet domain of your webserver plus the name of the PHP script, which receive the logged data and stores it e.g. in a CSV file or a database on the webserver. Example: "www.myserver.com/datastore.php?scale=garden".
Check the chapter SIM800 Evalboard on page 25 for an example PHP script.

After setting and storing the parameters, remanent data can be read from EEPROM and stored in a binary file on a PC, using the ST-Link Utility. Vice versa, a prepared remanent data file on a PC can be programmed into the EEPROM, using the ST-Link Utility software.

Parameter Number	Data	Type	Valid Values	Terminal cmd	Default Value
20	Scale tara	uint32_t		tara	
21	Scale calibration	uint32_t		cali	
1	Wakeup Period [seconds]	uint32_t	1-86400		300 = 5 minutes
10	Upload Alarm Hour	uint8_t	0-23		23
11	Upload Alarm Minutes	uint8_t	0-59		55
12	Upload Period [hours]	uint8_t	0-23		0 = once per day
30	ADC low	uint32_t		batl	

31	ADC high	uint32_t		bath	
32	Vbat low	uint32_t			5
33	Vbat high	uint32_t			7
40	APN	szString	35 chars		
41	URL	szString	120 chars		

```
#define APP_RD_MAX_APN_STRING_LEN 35
#define APP_RD_MAX_URL_STRING_LEN 120

typedef struct APP_REMANENT_DATA_Ttag{
	uint32_t ulTara;
	uint32_t ulCali;

	uint32_t ulWakeupPeriod;

	uint8_t bAlarmHrs;
	uint8_t bAlarmMin;
	uint8_t bUploadHoursIncrement;
	uint8_t bReserved;

	uint32_t ulAdcHigh;
	uint32_t ulAdcLow;
	uint32_t ulVoltLow;
	uint32_t ulVoltHigh;

	char szApn[APP_RD_MAX_APN_STRING_LEN];
	char szPrvUrl[APP_RD_MAX_URL_STRING_LEN];

} APP_REMANENT_DATA_T;
```

Tara and calibration of scale

Remove all objects from the scale platform, call terminal command "`tara`".

Put a 10kg reference load onto the scale platform and call the terminal command "`cali`".

After setting tara and calibration, call the terminal command "`rems`" to store the calibration values permanently in EEPROM.

In case of a standalone commissioning, without a PC, the terminal commands "`tara`", "`cali`" and "`rems`" can be activated by a certain user button usage. Refer to the respective section in this document.

The 10kg reference load should be determined with a precise scale as accurate as possible, in order to assure correct weight results. A bucket filled with water may be a good reference load.

Battery voltage monitor calibration

Similar to the scale calibration, the battery voltage monitoring ADC must be calibrated.

Set power supply voltage to a Vlow reference, like 5V and call the terminal command "`batl`".

Set power supply voltage to a Vhigh reference, like 7V and call the terminal command "`bath`".

After setting Vbat refercne values, call the terminal command "`rems`" to store the calibration values permanently in EEPROM.

In case of a standalone commissioning, without a PC, the terminal commands "batl", "bath" and "`rems`" can be activated by a certain user button usage. Refer to the respective section in this document.

The actual Vlow and Vhigh voltage must be entered in the configuration parameters Vbat low and Vbat high via the terminal command "`setp`". In the template remanent data file 5 and 7 is set as default. No float values are allowed, only integers

Terminal commands

After power on, the device remains in STOP mode (to save power) and wakes up periodically just for a short moment to perform a weight measurement.
In order to activate terminal usage, the device must be set to RUN state. RUN state is activated by pressing the user button. In RUN mode, the status LED starts blinking.
Execution of terminal commands do not change the system state. The system remains in RUN state. The only way to leave run mode and enter normal logging state, is execution of the terminal command "`app`" or using the user button as described in the respectibe section, which starts the application and transitions to STOP state.

UART Parameter	Value
Speed	115200
Data	8
Parity	none
Stopbits	1
Flow Controller	none
Termination Character Transmit	NL+CR
Termination Character Receive	CR

	Terminal command	Function
TERM_HELP	help	Print a list of terminal commands
TERM_BLINK_ON	bln1	Switch on LED blinking

TERM_BLINK_OFF	bln0	Switch off LED blinking
TERM_RTC	rtc	Print current RTC time and date
TERM_AT	at	Issue SIM800 AT command e.g. "`at+cclk?`" to fetch network time SIM800 must be in ON state
TERM_BAT	bat	Perform a battery voltage measurement
TERM_BAT_LOW	batl	Calibrate Vbat low (5V)
TERM_BAT_HIGH	bath	Calibrate Vbat high (7V)
TERM_HX	hx	Perform a weight measurement and store the result in the logging buffer
TERM_TARA	tara	Set scale to 0kg - tara
TERM_CALI	cali	Calibrate scale to reference weight of 10kg
TERM_PUSH	push	Uplaod logged data to server SIM800 must be in ON state
TERM_ALARM	alrm	Print configured alarm time at alarm time, the log buffer is uploaded to server
TERM_SETPARAM	setp	Set remanent data parameters usage e.g.: "`setp 10 23`" set alarm hour to 23 refer to list of remanent data elements
TERM_SETTIME	sett	Fetch time from network and set RTC SIM800 must be in ON state
TERM_STARTAPP	app	Start the application, goto to stop mode
TERM_SIM800_ON	on	Switch on SIM800 modem
TERM_SIM800_OFF	off	Switch off SIM800 modem
TERM_BUFLIST	bufl	List logged data without deleting data
TERM_BUFFLUSH	buff	List and clear logged data
TERM_BUFCLR	bufc	Clear log buffer
TERM_BUFINFO	bufi	Print number of elements in buffer
TERM_REMINFO	remi	List remanent data (configuration)
TERM_REMLOAD	reml	Load remanent data (configuration)
TERM_REMSTORE	rems	Store remanent data into EEPROM

User Button

After power on, the device remains in STOP mode (to save power) and wakes up periodically just for a short moment to perform a weight measurement. Status LED remains off in STOP mode.
In order to activate user button and terminal usage, the device must be set to RUN state. RUN state is activated by pressing the user button. In RUN mode, the status LED starts blinking.

Specific commands can be executed by pressing the user button for a specific number of status LED blink pulses

Number of LED blink pulses	command
2	“app” - start normal operation, enter STOP mode
5	“tara” scale tara -set to 0kg
6	“cali” scale calibration to 10kg
7	“batl” Vbat low – set supply volateg to 5V
8	“bath” Vbat high – set supply volateg to 7V
9	“rems” store the calibration values (remanent data) in EEPROM

Status LED

The Status LED indicates the device status

LED	Status
off	Normal operation – STOP mode (or now power supply connected)
“slow” blink period	RUN mode, terminal and user button active, SIM800 modem is powered off
“fast” blink period	RUN mode, terminal and user button active, SIM800 modem is powered on
Flashing	ERROR: Network time could not retrieved after power on, no GPRS network found

Data upload to a webserver

The device measures the weight in a configured time interval, e.g. every 5 minutes, and stored the data inside the SRAM memory. At least once a day or more often, the collected data must be send to a webserver for permanent storage in a CSV file or database on the server.

The URL of the webserver internet domain and "receiver PHP script" is set in the URL parameter. The PHP script receives the logged data with the HTTP POST method and stores it in a CSV file or database on the webserver. Example: "www.myserver.com/datastore.php?scale=garden".

Example PHP script "datastore.php" on server "www.myserver.com"

```
<?php

$device = "default";
$device = $_GET['device'];

$data = file_get_contents("php://input");
$method=getenv('REQUEST_METHOD');

$my_file = $device.'.csv';
$handle = fopen($my_file, 'a') or die('Cannot open file:  '.$my_file);
fwrite($handle, $data);
fclose($handle);

echo "received and stored data in file $my_file";
?>
```

file content "garden.csv"

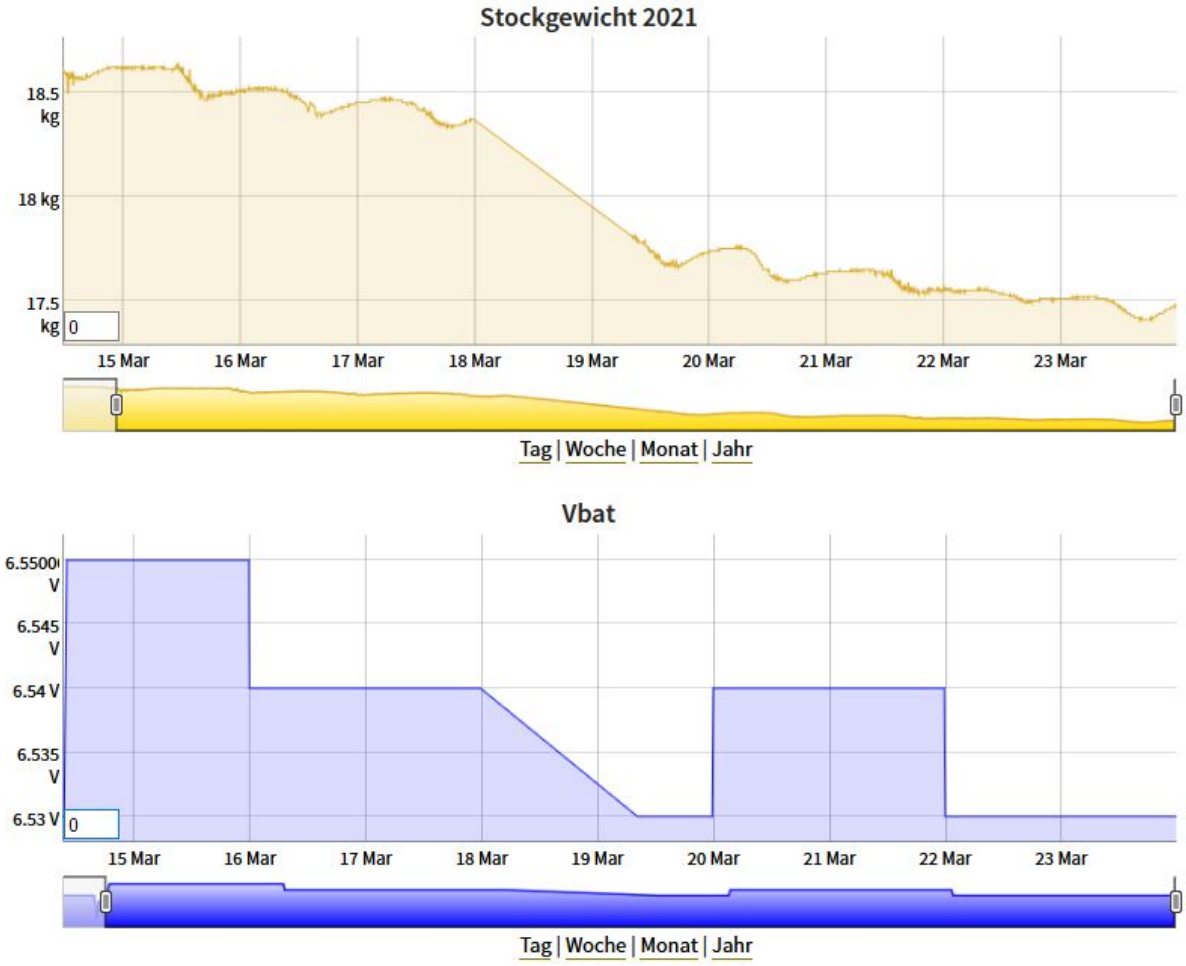
```
2021-01-09T11:03:00,21.34,6,54
2021-01-09T11:08:00,21.33,6,54
2021-01-09T11:13:00,21.32,6,54
2021-01-09T11:18:00,21.32,6,54
…
```

row format of *.csv file
dateTtime,weight[kg],Vbat[volt]
YYYY-MM-DDTHH:MM:SS,weight[kg],Vbat[volt]

The SIM800 module supports only HTTP, not the secure, encrypted HTTPS protocol.

You may use a public cloud like Thingspeak instead of your own webserver for this purpose. Or you may setup a dashboard framewrok like grafana.com.

The webserver can display the stored data from the CSV file or the database on a webpage. This task is not covered in this document. For an example implementation, using the "dygraph" framework, check www.schwedenhof.net/imkerei/stockwaage



Pause weight data logging

If the beekeeper works on the hive, you may want to pause the weight data logging to prevent false weight data. For this purpose, just enter RUN mode by pressing the user button once. The status LED starts blinking. After you finished work on the hive, press the user button for a time duration of 2 LED blink pulses. This is the command to start the application and enter STOP mode. The LED stops blinking and is off.

No automatic return from RUN mode to weight logging is implemented yet, so don't forget to leave RUN mode by pressing the user button at the end of your work.

For Developers

The firmware of hiveScale is published under GNU General Public License v3.0. It is hosted on github: https://github.com/schwedenhof/hiveScale

Please contribute to the project! There are tasks for hardware and software developers. Check the ToDo List for ideas. If you are interested, please contact the author: Dirk Fischer, engineer@schwedenhof.net, www.schwedenhof.net

The package contains

- STM32CubeMX project
- STM32CubeIDE project
- pre-compiled firmware file "hiveScale.bin"
- example (template) configuration file "remanent_data.bin"
- various related documentation files

Required Software Tools

- STM32CubeMX – GUI for MCU configuration and code generator
- STM32CubeIDE – development environment
- STM32 ST-Link Utility – flash programmer
- TeraTerm - terminal

Device Operation States

State	Actions	Next State
STOP	• switch off LED • … switch peripherals, IOs, etc. into modes with low powers consumption • enter low power stop mode • periodical wakeup (e.g. every 5 minutes) for weight measurements • periodic alarms (e.g. one a day) for data upload	•
MEASURE	• 0	•
ALARM	• switch on power for modem (SIM800 module) • wait 45sec, check modem availability, do	•

	error handling • push complete ringbuffer data to server, using http POST • switch off modem	
WAKEUP	• 0	•
RUN	• wait for incoming terminal commands via USART2 • execute terminal commands	• stays in RUN • STOP when executing terminal command “app”
	•	•

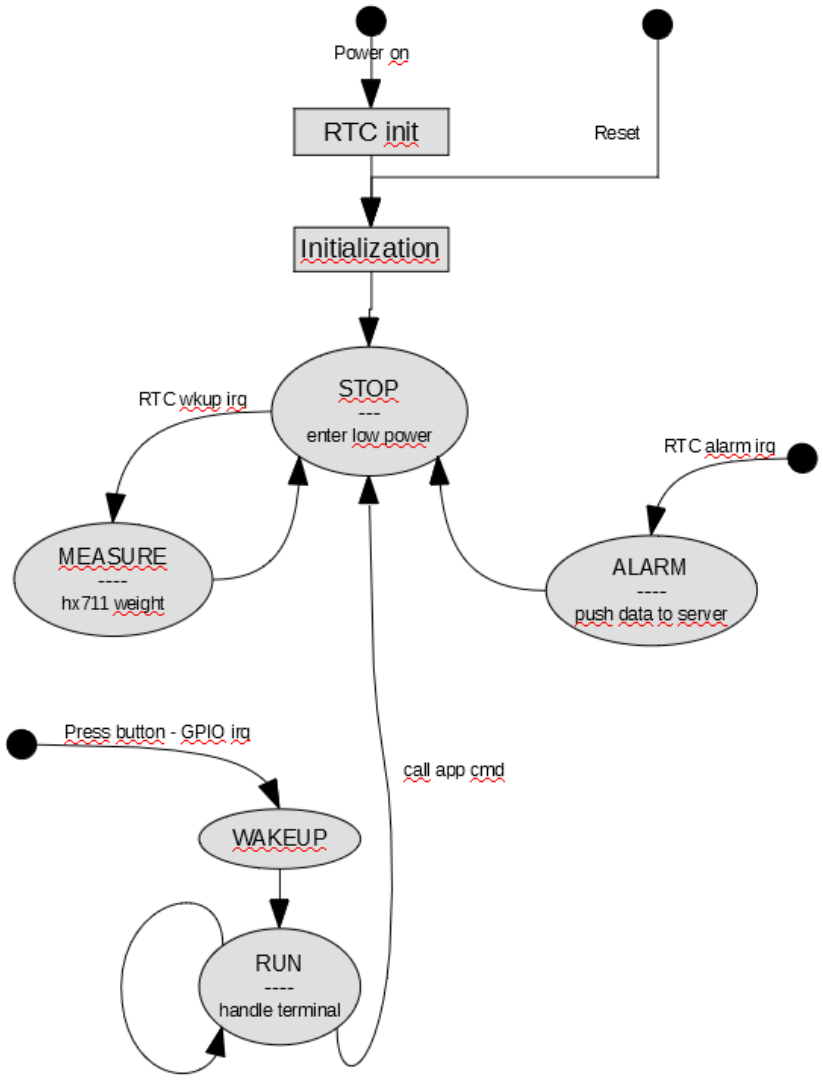


Figure: State Diagram

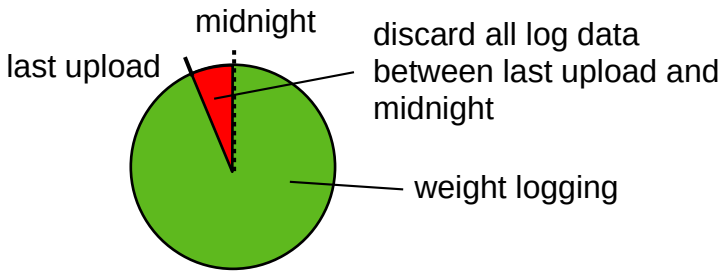
Initialization

1. read remanent data
2. switch on modem and fetch time from network
3. set RTC alarm
4. set wakeup interval
5. start "app" ⟶ go to stop mode

Logging

Only time and weight is buffered internally, not the date, in order to save the limited SRAM ressources.

"Todays" Date is added during data upload to the server. Therefore all remaining data, not uploaded, is discarded at midnight (date change).
Assumption: After last upload, until midnight, no further data is logged. i.e. the last upload should happen sharp before midnight, e.g. 23:55



Vbat Monitoring

R1=365kOhm
R2=97kOhm

I=26uA @12V, VAdc=2.52

I=14uA @6.5V, VAdc=1.37

Interrupts

Interrupt	Callback Handler Function	Purpose
timer2	HAL_TIM_PeriodElapsedCallback	Time base for user interface button input and LED output
timer21	HAL_TIM_PeriodElapsedCallback	Time base for LPUART1 RX timeout

usart2	HAL_UART_RxCpltCallback	UART RX one byte received
lpuart1	HAL_UART_RxCpltCallback	UART RX one byte received
GPIO exti	HAL_GPIO_EXTI_Callback	Wakeup from stop mode button pressed
RTC alarm A	HAL_RTC_AlarmAEventCallback	Once a day wakeup alarm for data transfer to server via modem

UART

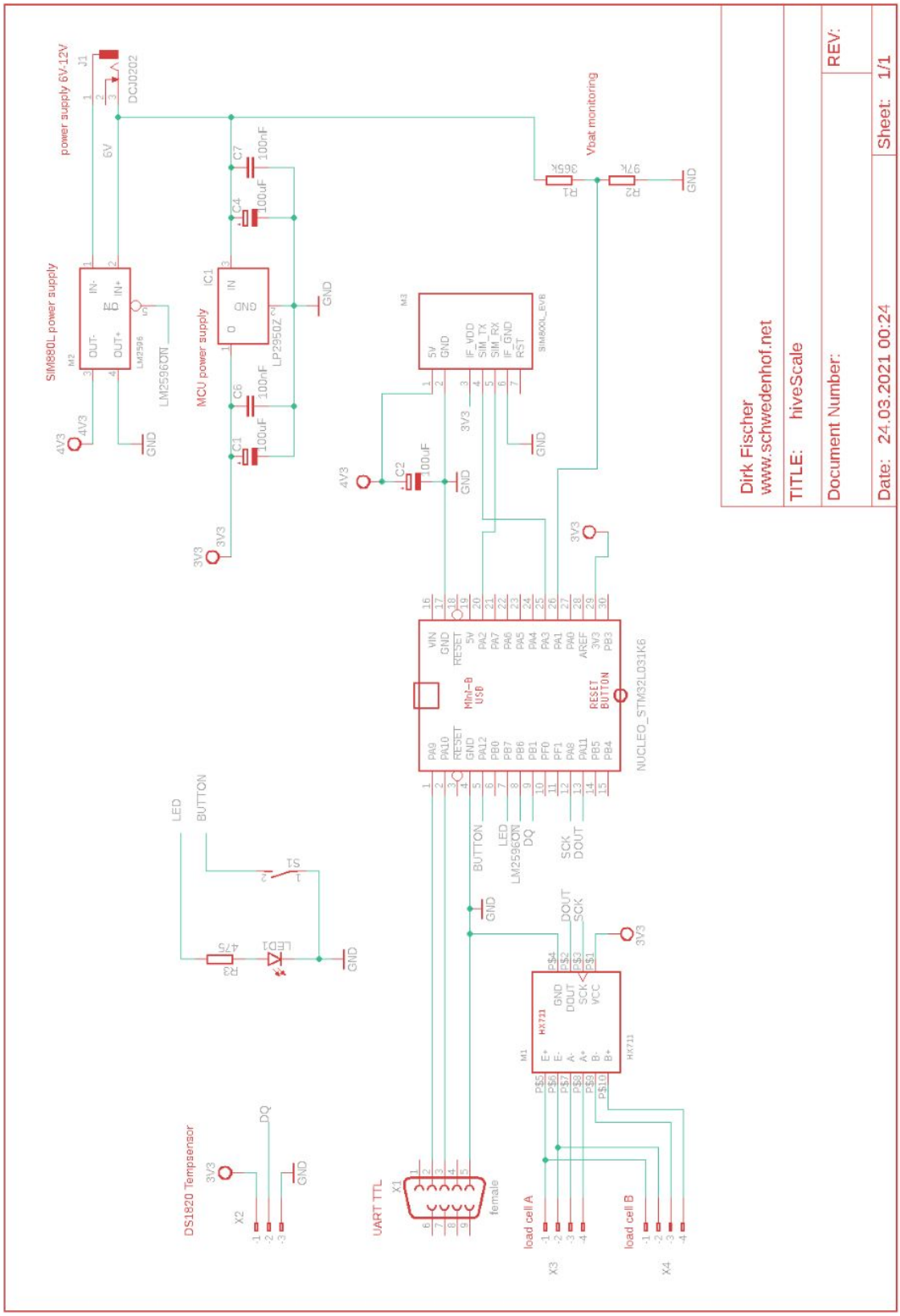
Two UARTs are utilized: USART2 for terminal operation with a PC, LPUART1 for communication with SIM800 Modem device for IP communication. Both UARTs are operated in interrupt mode.

RX interrupt
RX timeout interrupts

UART reception states

```
    APP_RX_ONGOING = 0,
    APP_RX_NL,
    APP_RX_OK,
    APP_RX_ERROR,
    APP_RX_BUFFER_FULL,
    APP_RX_ENDOFFRAME,
    APP_RX_TIMEOUT
```

Schematics


power supply 6V-12V
DCJ0202
6V
SIM980L power supply
LM2596DN
MCU power supply
LP2950Z
Vbat monitoring
365K
97K
SIM800L_EVB
NUCLEO_STM32L031K6
Mini-B USB
RESET BUTTON
LED
BUTTON
LM2596DN
DQ
SCK
DOUT
HX711
load cell A
load cell B
DS1820 Tempsensor
UART TTL
female
Dirk Fischer
www.schwedenhof.net
TITLE: hiveScale
Document Number:
REV:
Date: 24.03.2021 00:24
Sheet: 1/1

Hardware components

STM32 Nuceo-Board

The Nucleo board L031K6 must be modified

Remove the following solder bridges

SB2, SB9, SB14, SB16, SB18

```
CN3.1 PA9 now VCOM via ST-Link RX SB2-off
CN4.5 PA2 originally on Nucleo: VCOM, USART2_TX via SB2
```

	CN3	STM32L031K6		CN4	
USART2_TX	1	PA9	Vin	1	x
USART2_RX	2	PA10	GND	2	x
x	3	NRST	NRST	3	x
x	4	GND	5V	4	x
button UI_INPUT	5	PA12	PA2	5	**UART1_TX - SIM800 RX**
	6	PB0	PA7	6	A7
user i/f LED	7	PB7 (PB6)	PA6	7	(IMPORTANT SB16 off, else PB6)
LM2596 ONn/OFF	8	PB6 (PB7)	PA5	8	*SPI1_SCK* (IMPORTANT SB18 off, else PB7)
DS1820 Temp	9	PB1	PA4	9	*SPI1_NSS*
32.768 OSC	10	PF0	PA3	10	**UART1_RX - SIM800 TX**
32.768 OSC	11	PF1	PA1	11	**A1 - VBAT Monitor**
HX711_SCK	12	PA8	PA0	12	
HX711_DOUT	13	PA11	AVDD	13	x
SPI1_MISO	14	PB5	3V3	14	x
SPI1_MOSI	15	PB4	PB3	15	LED3

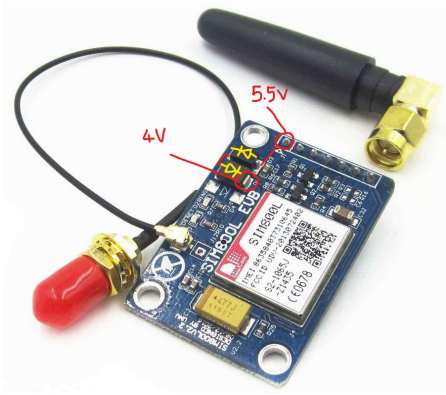
```
VCP_RX - virtual COM port RX - USART2.RX - PA15 (via STLink) or PA10 (via FTDI cable)
VCP_TX - virtual COM port TX - USART2.TX - PA9
```

On Nucleoboard L031K6, the VCOM is originally connected to USART2 on pins PA2->CN4.5 and PA15 (no external pin on Nucleo)
PA2->CN4.5 is the only option for UART1_TX. therefore USART2_TX must be resoldered to PA9 -> CN3.1
`With SB2, USRAT2_TX can be disconnected from ST-Link Chip`

SIM800 Evalboard

Modifications:

- Bypass input diods and adjust LM2596 output voltage to 4.3V
- add a 100uF cap to power supply



J1
1 5V – (LM2596 OUT+), but bypass 2 diods
2 GND -(LM2596 OUT-), but bypass 2 diods
3 VDD - 3V3
4 SIM_TX (grey) – LPUART1_RX PA3 CN4.10
5 SIM_RX (white) – LPUART1_TX PA2 CN4.5
6 GND - GND
7 RST - NC

network LED blink period
3sec: network connected
1sec: network search,
0,5sec:GPRS active
static on: RING LED

If the SIM800 Modul do cyclic resets during power on phase, input voltage drops occur and/or input voltage is too low, increase output voltage of DC/DC LM2596 in this case

The AT command "AT+CBC" can be used to monitor the VBAT voltage.
Take care to stay below SIM800L Vcc max of 4.4V

Document "SIM800_Hardware Design_V1.08.pdf"

The power supply range of SIM800 is from 3.4V to 4.4V. Recommended voltage is 4.0V. The transmitting burst will cause voltage drop and the power supply must be able to provide sufficient current up to 2A. For the VBAT input, a bypass capacitor (low ESR) such as a 100µF is strongly recommended.

Configure SIM800 before usage

1. activate network time tracking
   at+clts=1;&W
   power cycle required

   trackiong of network time enabled?
   Check with "at+clts?"
   fetch network time from SIM800
   at+cclk?
   answer: +CCLK: "21/01/22,08:22:09+04"

2. set fixed baud rate “19200“ in order to receive messages during SIM800 power on sequence
   at+ipr=19200;&W

Configure SIM card before usage

1. disable PIN
   currently the firmware does not transmit a PIN during SIM800 startup
   Therefore the PIN of the SIM card must be disabled
   You may use a mobile phone for this setting

HTTP POST “at” command sequence

The logged data is transferred and stored on the server side as a CSV file

```
ate0
at+sapbr=3,1,"Contype","GPRS"
at+sapbr=3,1,"APN","web.vodafone.de"  ← adapt to your provider
at+sapbr=1,1
at+httpinit
at+httppara="CID",1
at+httppara="URL","www.myserver.com/datastore.php?scale=garden" ← adapt to your server
at+httpdata=500,6000

DOWNLOAD 500 characters for the POST data, timeout after 6000msec

at+httpaction=1
at+httpterm
at+sapbr=0,1
```

EMAIL “at” command sequence

```
ate0
at+sapbr=3,1,"Contype","GPRS"
at+sapbr=3,1,"APN","web.vodafone.de"  ← adapt to your provider
at+sapbr=1,1
at+sapbr=2,1
at+emailcid=1
at+emailto=30
at+smtpsrv="12.345.678.89",587 ← adapt to your provider
at+smtpauth=1,"login","password" ← adapt to your account
at+smtpfrom="darth.vader@deathstar.net","Anakin Skywalker"
at+smtprcpt=0,0,"imperator@galaxy.net","Imperator"
at+smtpsub="Hi, how are you"
at+smtpbody=20

DOWNLOAD 20 characters for the mail body

at+smtpsend
at+sapbr=0,1
```

https://www.instructables.com/TCPIP-Connection-Over-GPRS-How-to-Send-Data-to-Ser/

LM2596 DC/DC power supply

Modifications:

- disconnect LM2596 pin5 ON/OFF from PCB
- connect LM2596 pin 5 ON/OFF to Nucelo CN3.8 – PB6

DS1820 Dallas Tempsensor

```
 bottom
.-------.   1 GND
\ o o o /   2 DQ
 `-----´    3 VDD  2V8 .. 5V5 (1/2°C accuracy: 4V3 .. 5V5)
  1 2 3
```

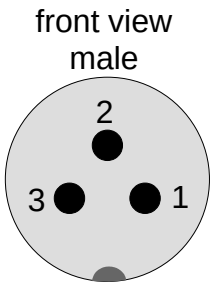


Figure: temperature probe plug in casing (male)

Plug pin	Nucleo pin	cable	DS18B20
1	CN4.14 - 3V3	red	3V3
2	CN3.9 - PB1	yellow	DQ Signal
3	CN4.2 - GND	black	GND

HX711

```
VDD 2V6 .. 5V5
SCK Tmin 0.4usec

HX711 pins                                    load cell "Anyload 108BA"
E+ AVdd                                       INPUT + (red)         pin 1 (red)
E- Gnd                                        INPUT - (black)
A- channel A negative analogue input INNA     OUTPUT - (white)
A+ channel A positive analogue input INPA     OUTPUT + (green)
```

```
HX711 pins    RASPBERRY PI          STM32 Nucleo 32
Vcc           J8.1     rot          CN4.14
Gnd           J8.6     blau         CN4.2
DOUT          J8.13    grün         CN3.13 port PA11
SCK           J8.12    orange       CN3.12 port PA8
```

Load cell plugs

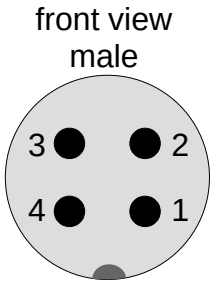


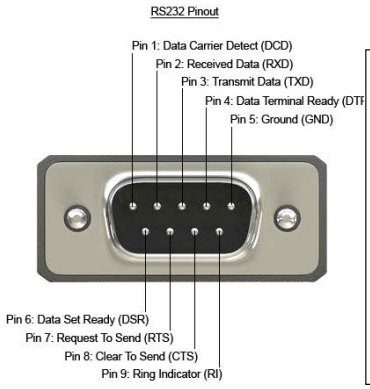
Figure: load cell plug in casing (male)

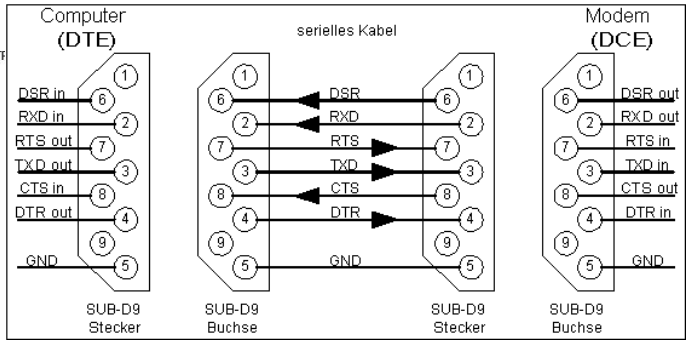
Plug pin	HX711 pin	cable	Load cell
1	E+	red	Vcc
2	E-	black	GND
3	A- / B-	white	Out-
4	A+ / B+	green	Out+

Diagnosis UART

FTDI USB/UART Cable

Red wire : 5V
Black wire : GND
White wire : RXD
Green wire : TXD
Yellow wire : RTS
Blue wire : CTS





FTDI cable (PC) - (DTE)			HiveScale - (DCE)		
Signal	colour	SUBD9 male	Signal	SUBD9 female	Nucleo
RXD (in)	white	2	RXD (out)	2	USART2 TX – CN3.1 – Port PA9
TXD 3V3 (out)	green	3	TXD (in)	3	USART2 RX – CN3.2 – Port PA10
GND	black	5	GND	5	GND CN3.4 - GND

BOM – bill of materials

No				
1	NUCLEO-L031K6	11,-	RS Components	Link
2	SIM800L module incl. antenna	10,-	ebay	
3	HX711 module	2,50	ebay	
4	LM2596 DC/DC module	1,50	ebay	
5	LDO LP2950CZ-3.0 6V->3V	0,55	Pollin	Link
6	Einbau-Taster	0,60	Pollin	Link
7	LED 5mm, low current	0,07	Pollin	Link
8	LED Clipfassung	0,05	Pollin	Link
9	Buchsenleiste RM2,54	0,50	Pollin	
10	Gehäuse e.g. 150x80x50	6,-	RS Components	Link
11	Einbau-Hohlbuchse 5,5/2,5	0,70	Pollin	Link
12	2 x Buchse/Stecker 16mm 4pol for load cells	4,40	Pollin	Link
13	1 x Buchse/Stecker 16mm 3pol for tempsensor	2,-	Pollin	
14	Cable			
15	Grid circuid board RM2,54 70mmx40mm	1,-	Pollin	
16	DS18B20 stainless steel temp sensor proble	2,50	ebay	
17	Battery 6V 4.5Ah PANASONIC LC-R064R5PG	10,-	Pollin	Link
18	Quality Loadcell	55,-	Bosche	Link
19	Alternative, low cost loadcell	14,-	ebay	
20	2 x Siebdruckplatte 18mm 420mmx510mm			
21	Profilstahl 15mm ca. 3,5m			

Technical Data

Interfaces

Interface	Type	Notes
Load cell channel A	4 pin male	VCC output 3V
Load cell channel B	4 pin male	VCC output 3V
Temperature Probe	3 pin male	DS18B20 signals, VCC output 3V
Power Supply	Hohlbuchse 5,5/2,5	6V-12V, recommended 6V
UART Service Interface	Sub D9 female (DCE)	HiveScale RX max voltage level: 5V HiveScale TX high level: 3V Ftf pin type, VDD+4V max → 5V tolerant
Antenna Plug	Mini-BNC Plug female	
One Button, LED Human Machine Interface		

Current consumption

Mode	Description	Conditions	Current 6V supply	Duration in 24h	Consumption in 24h
1	Weighting mode every 5 minutes	Nucleo board run mode @3V, 16MHz LM2596 stdby, SIM800 off	7mA	12*24*1sec ~5min=0.085h	0.6mAh
2	Data transfer once a day	Nucleo board run mode @3V, 16MHz LM2596 12V→5.5V, SIM800 on	60mA	2min ~0,035h	2.1mAh
3	Normal idle mode >99% of day	Nucleo board stop mode @3V LM2596 stdby, SIM800 off	0.16mA	23.88h	3,85mAh
					~6.6mAh

Operation time

Self-discharge not considered in the calculation below. Real operation time is shorter.

Battery Type	Capacity	Operation Time
Motor Cycle Battery	4,5Ah, 6V	4500mAh/6.6mAh/day = 681 days = 1.9 years
Motor Cycle Battery	7Ah, 6V	7000mAh/6.6mAh/day = 1060 days = 2.9 years

Appendix

APN – Access Point Name

Anbieter	Zugangsdaten
1&1	APN: web.vodafone.de APN-Typ: default
A.T.U.Talk	APN: gprs.gtcom.de Benutzername: keins (leer lassen) Passwort: keins (leer lassen)
Aldi	APN: internet.eplus.de Benutzername: eplus Passwort: eplus
Aldi Talk / MEDIONmobile	APN: internet.eplus.de Benutzername: keins Passwort: keins
Ay Yildiz	APN: internet.eplus.de Benutzername: keins Passwort: keins
Base / E-Plus / Yourfone	APN: internet.eplus.de Benutzername: eplus Kennwort: eplus Zusätzlich muss Datenroaming aktiviert sein
BildMobile	APN: access.vodafone.de Benutzername: keins Passwort: keins
Blau.de	APN: internet.eplus.de Benutzername: blau Passwort: blau Tipp: Du kannst auch "Handy" an die 80 215 schicken, um die Daten zu erfahren
Callmobile / DeutschlandSim / Klarmobil / Mobilcom Debitel	*Im Netz von O2* **Siehe APN Einstellung O2** *Im Netz von Telekom* **Siehe APN Einstellung Telekom** *Im Netz von Vodafone* **Siehe APN Einstellung Vodafone**
Congstar	APN: internet.t-mobile Username: t-mobile

Anbieter	Zugangsdaten
	Passwort: tm
DiscoTel	APN: internet.t-mobile Benutzername: t-mobile Passwort: tm
EDEKAmobile	APN: data.access.de Benutzername: keins Passwort: keins
Fonic	APN: surfo2 ODER pinternet.interkom.de Benutzer: nicht notwendig (leer) Passwort: nicht notwendig (leer) MCC: 262 MNC: 07 APN Typ: default
Galeria Mobile	APN: gprs.gtcom.de Benutzername: keins Passwort: keins
Globus Mobile	APN: gprs.gtcom.de Benutzername: keins Passwort: keins
ja!mobil	APN: internet.t-mobile Benutzername: t-mobile Passwort: tm
Lidl Mobile	APN: pinternet.interkom.de Benutzername: t-mobile Passwort: tm
McSim	APN: web.vodafone.de Benutzername: keins Passwort: keins
n-tv GO	APN: gprs.gtcom.de Benutzername: keins Passwort: keins
Nettokom	APN: internet.eplus.de Benutzername: nettokom Passwort: nettokom
NettoKOM	APN: internet.eplus.de Benutzername: nettokom Passwort: nettokom
Netzclub	APN: pinternet.interkom.de Benutzername: leer Passwort: leer MCC: 262 MNC: 07
O2	APN (Zugangspunkt): internet

Anbieter	Zugangsdaten
	Benutzername: nicht notwendig Passwort: nicht notwendig Verbindungssicherheit: aus Zusätzlich muss Datenroaming aktiviert sein
Otelo	APN: data.otelo.de Benutzername: keins Passwort: keins
PennyMobil	APN: internet.t-mobile Benutzername: t-mobile Passwort: tm
Sparhandy	APN: internet.t-mobile Benutzername: t-mobile Passwort: tm
Tchibo Mobil	APN: webmobil1 MMSC: http://10.81.0.7:8002 MMS-Proxy: 82.113.100.8 MMS-Port: 8080 MCC: 262 MNC: 07
Telekom (T-Mobile)	APN: internet.t-mobile Benutzername: t-mobile Passwort: tm
Unitymedia Mobil	APN: internet.partner Benutzername: keins (leer lassen) Passwort: keins (leer lassen)
Vodafone	APN: web.vodafone.de
WhatsApp SIM	APN: internet.eplus.de Benutzername: eplus Passwort: gprs
WinSIM	*Im Netz von O2* **Siehe APN Einstellung O2** *Im Netz von Vodafone* **Siehe APN Einstellung Vodafone**