User Manual - S.USV solutions

Compatible with Raspberry Pi, UP Board and Tinker Board
Revision 2.0 | Date 15.03.2017

Table of Contents

1 Functions

S.USV solutions is an advanced power supply additional module for single-board computers, with the main focus on the uninterruptible power supply.
The module also provides additional functions in order to optimize the operation of the single-board computer by the user.

The S.USV solutions are fully functional plug & play solutions. The power supply of the S.USV solutions occurs directly through the GPIO power pins of the single-board computer and therefore uses a common voltage source, thus no additional cabling or power supply needed. In addition, the module is equipped with a LiPo battery. An integrated boost switching power converter covers the necessary voltage range, thereby the single-board computer shut down safely in case of misconduct and prevent data loss.
The “advanced” version also provides a power input for the extended voltage range of 7 – 24 volts (solar cells, automotive applications, etc.).

- HAT compliant UPS Modules
- Compatible with Raspberry Pi (S.USV pi), AAEON UP Board (S.USV UPs) and ASUS Tinker Board (S.USV tinker)
- Adapter solution for Pi Model A and B
- Uninterruptible power supply
- Plug & Play
- Monitoring – System (V/mA)
- Intelligent software solution including mobile application
- Built-in LiPo battery (300mAh) with intelligent charging function
- Battery Management Controller
- Battery Monitoring System
- Real Time Clock with Battery Back-Up
- Time-controlled on and off switching of the single-board computer
- Supply Switch (Power on and off buttons / File safe shutdown)
- LED – Status display
- Bootloader for Live – Firmware updates
- Power input with extended voltage range of 7-24 volts

2 Technical Specification

	S.USV pi		S.USV up	S.USV tinker
	basic	advanced	UPs	tinker
Plug & Play	√	√	√	√
Power Supply	5 Volts/2500 mA	5 Volts/2500 mA	5 Volts/2500 mA	5 Volts/2500 mA
Extended Voltage Range	x	7-24 Volts/3500 mA	7-24 Volts/3500 mA	7-24 Volts/3500 mA
Secondary output power	5 Volts/3500 mA	5 Volts/3500 mA	5 Volts/3500 mA	5 Volts/3500 mA
Interfaces	I²C	I²C	I²C	I²C
ID EEPROM	√	√	x	√
Monitoring - System	√	√	√	√
LiPo - Battery	√	√	√	√
Battery Management	√	√	√	√
Battery Monitoring	√	√	√	√
Real Time Clock	√	√	√	√
Supply Switch	√	√	√	√
Compatibility	Pi 3, Pi 2, Pi A+, Pi B+	Pi 3, Pi 2, Pi A+, Pi B+	All UP boards	All tinker boards
Dimensions	65x56,5 mm	65x56,5 mm	65x56,5 mm	65x56,5 mm

	300 mAh	3000 mAh
Normal Voltage	3.7 V	3.7 V
Working Voltage	3.0 - 4.2 V	3.0 - 4.2 V
Capacity	300mAh	3000mAh
Internal Impedance	≤60mΩ	≤30mΩ
Constant Charge | Discharge Current	2C | 15C	1C | 2C
Working Temperature	charge: 0-45°C; discharge: -20-60°C	charge: 0-45°C; discharge: -20-60°C
Connector	JST-PH-2P	JST-PH-2P
Leadwire	UL1571#28 / 50mm	UL1571#28 / 50mm
Wrapping Way	blue PVC	blue PVC
Line Drawn Out Direction	Center	Center
Dimensions	30.0 x 20.0 x 6.7 mm	60.0 x 50.0 x 9.0 mm

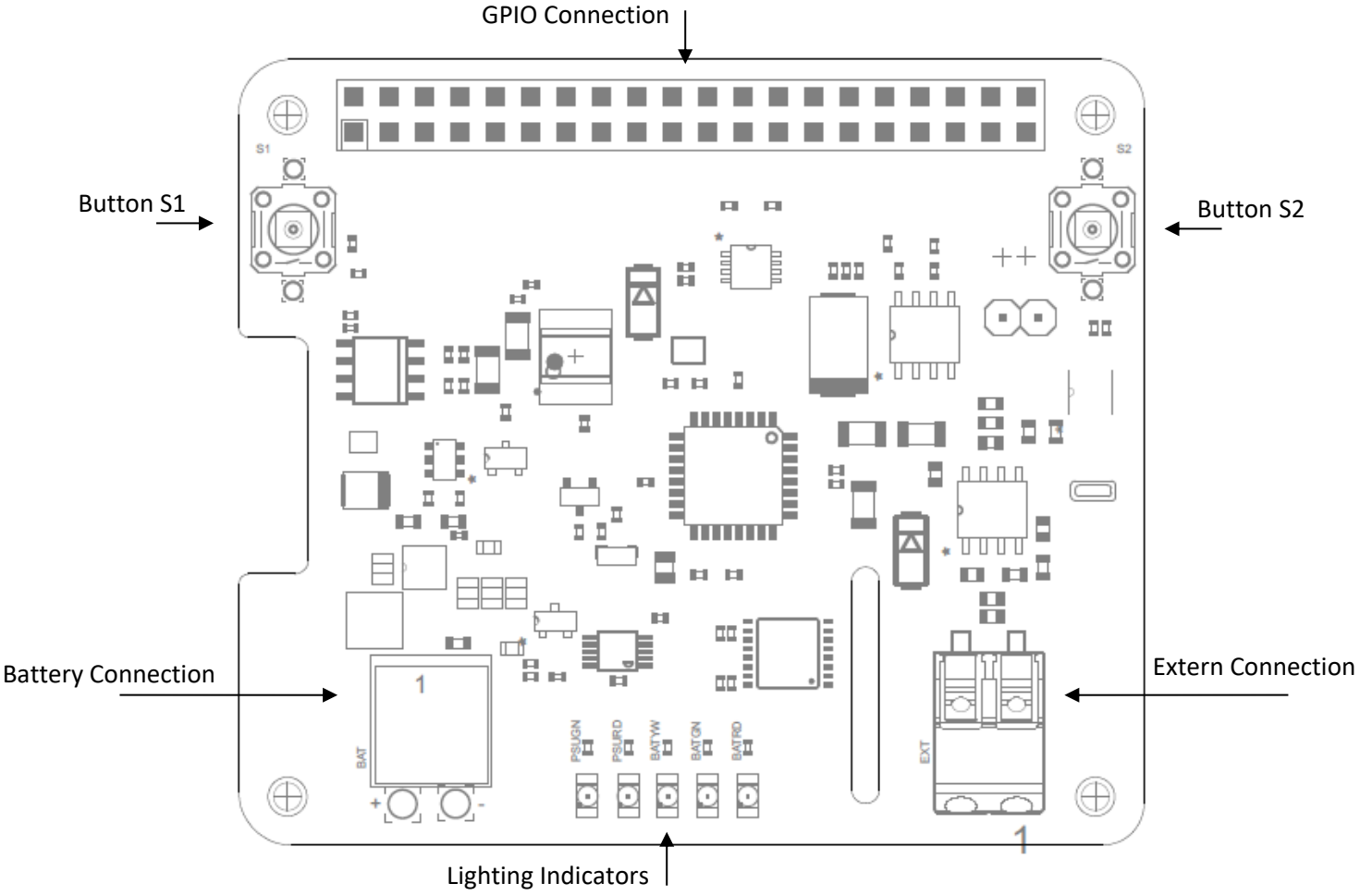
Bridging times during power failure (battery operation):

The following table gives an insight into the overall performance with respect to the back-up time of the S.USV in case of power failure. Measurements were taken at a remaining battery capacity of 70-80%.

	Power consumption +/-	300 mAh	3000 mAh
Raspberry Pi A+	500mA	45 minutes	11 hours
Raspberry Pi B+	500mA	30 minutes	6-8 hours
Raspberry Pi 2B	500mA	30 minutes	6-8 hours
Raspberry Pi 3B	500mA	30 minutes	6-8 hours
AAEON UP Board	500mA	30 minutes	4-5 hours
ASUS Tinker Board	500mA	30 minutes	4-5 hours

(Note: The measured runtimes in relation to the measured power consumption is not a guarantee and may deviate depending on the peripheral.)

2.1 Overview



2.2 Performance

- **Battery Connection:** Connector for connecting the supplied LiPo battery.
  Battery Connector: (Würth Elektronik 620 002 113 322)
- **Extern Connection:** Connector for connecting extended voltage range (7-24V).
- **GPIO PORT:** GPIO Connection to the single-board computer.
- **Button S1:**
  - **Raspberry Pi / Tinker Board:** Press for reboot or hold at least 3 seconds for shutdown.
  - **UP Board:** Press the button for a shutdown or hold it for at least 3 seconds for a restart. Press the button when the power is off to power up the single board computer.
- **Button S2:**
  - **Raspberry Pi / Tinker Board:** Press for booting the single-board computer.
- **Lighting Indicators:** LED status indicator for the S.USV.

2.3 Lighting Indicators

LED	*Indication*
PSU GREEN (Blinking)	*Startup - initialization of the S.USV firmware*
PSU GREEN	*Power Supply Unit is online (Voltage present)*
PSU RED	*Power Supply Unit is offline (Voltage loss) – Battery Powering is online*
BAT YELLOW	*Charging Circuit Online – Battery is charging*
BAT GREEN	*Charging Circuit Online – Battery is fully charged*
BAT RED	*Charging Circuit Offline – Battery is missing or corrupt*
BAT RED (Blinking)	***Charging Circuit Offline – Remaining battery capacity in the critical area***

3 Installation Guide

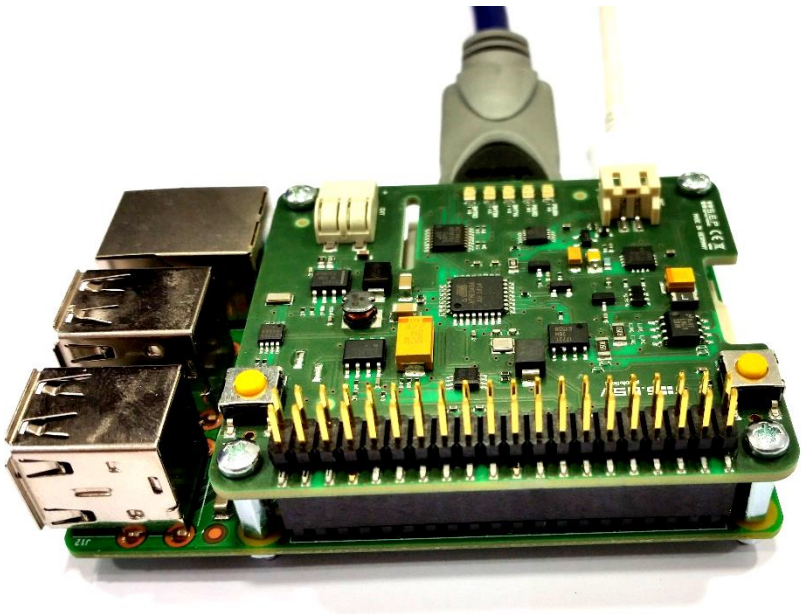
3.1 Hardware

In initial operation we recommend to fully charge the supplied battery to ensure full functionality. Furthermore, we recommend a PSU with at least 2 amps to operate the single-board computer.

In the following steps the commissioning of the S.USV will be described again in detail:

3.1.1 Commissioning S.USV

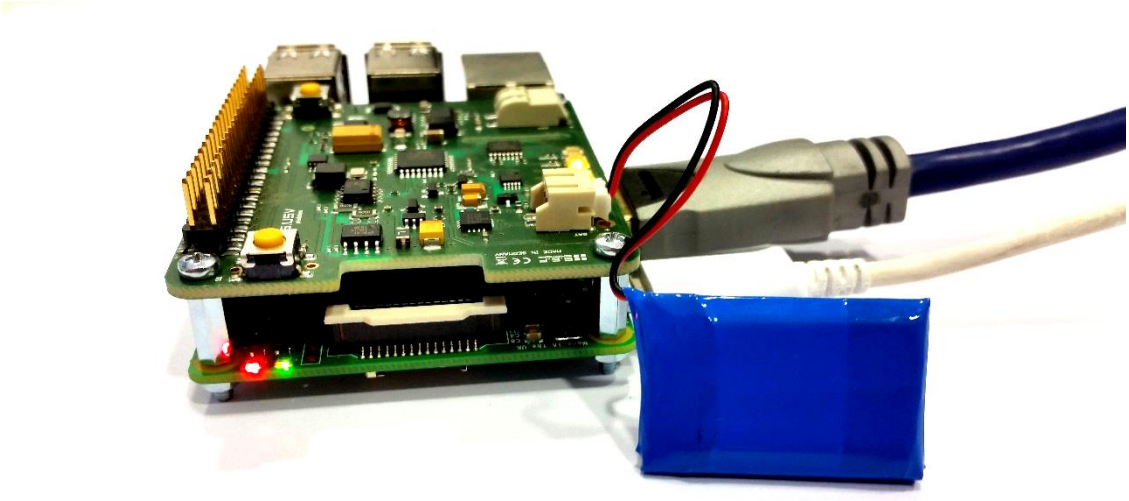
- The primary power supply of the S.USV occurs via the GPIO port - Pin 2 (+5V) of the single-board computer. Please now connect the board as shown on the single-board computer to establish the necessary connection and secure it with the included mounting kit.



3.1.2 Connecting the battery

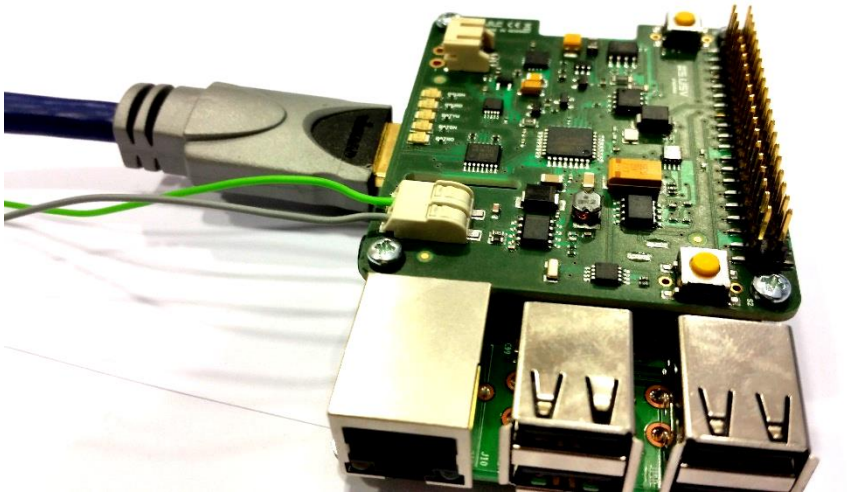
- For initial operation please make sure that the included LiPo-battery is plugged into the provided JST connector on the front side of the board.

  A start-up of the system without a plugged-in battery can lead to damage.



3.1.3 Connecting the external power supply

- For usage of the wide-range input the shown terminal block is located on the front side of the board. Please note here the +/- marking on the board in order to avoid a short circuit.



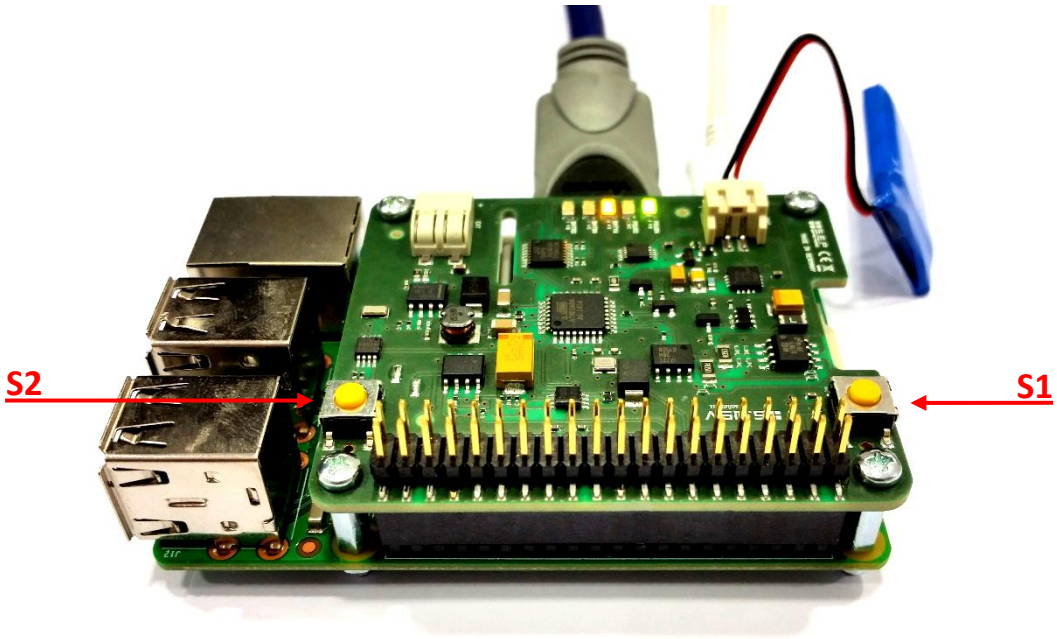
3.1.4 Using the buttons

- **Raspberry Pi / Tinker Board:**

  For Power on, Power off and restart of the single-board computer, the following buttons are available:

  - **S1** : Power Off / Reboot
  - **S2** : Power On

  Both buttons have corresponding vias, thus they can be performed at any point of a housing.



- **UP Board:**

  To switch on and off, as well as to restart the UP board, connect the S1 push button via the enclosed PicoBlade cable with **CN33 - Power Button Wafer** of the UP board.

  - **S1** : Power Off / Reboot

3.1.5 GPIO – Port

- For the power supply and data transmission of the S.USV following GPIO - Pins in use:

  - **Pin #02**: DC Power +5V – Power supply
  - **Pin #03**: GPIO 02 (SDA1, I²C) – I²C data line
  - **Pin #05**: GPIO 03 (SCL1, I²C) – I²C clock line
  - **Pin #13**: GPIO 27 (GPIO_GEN2) – Monitoring S.USV
  - **Pin #27:** ID_SD (I²C ID EEPROM) – ID data line
  - **Pin #28:** ID_SC (I²C ID EEPROM) – ID clock line

3.2 Software

3.2.1 Raspberry Pi - Raspbian

- To install and configure the Raspberry Pi, we recommend the Quick Start Guide of Raspberry Pi directly:

  https://www.raspberrypi.org/help/quick-start-guide/

- To install the operating system, we recommend the Image Installation Guides of Raspberry Pi:

  https://www.raspberrypi.org/documentation/installation/installing-images/

- The Image of Raspbian operating system can be found on the following page:

  https://www.raspberrypi.org/downloads/

3.2.2 UP Board – Ubilinux

- To install and configure the UP Board, we recommend the Quick Start Guide:

  https://up-community.org/wiki/Setup

- To install the operating system, we recommend the Image Installation Guides of Ubilinux:

  https://up-community.org/wiki/Installing_ubilinux

- The Image of Ubilinux operating system can be found on the following page:

  https://up-community.org/downloads/ubilinux

3.2.3 Tinker Board – TinkerOS

- To install and configure the Tinker Board, we recommend the Quick Start Guide:

  https://www.asus.com/de/Motherboards/TINKER-BOARD/HelpDesk/

- To install the operating system, we recommend the Image Installation Guides of TinkerOS:

  https://www.asus.com/de/Motherboards/TINKER-BOARD/HelpDesk/

- The Image of TinkerOS operating system can be found on the following page:

  https://www.asus.com/de/supportonly/TInker%20Board2GB/HelpDesk_Download/

3.2.4 I2C

The ID EEPROM contains data that identifies the board, tells the single-board computer how the GPIOs need to be set up and what hardware is on the board. This allows the S.USV *solutions* modules to be automatically identified and set up by the software at boot time including loading all the necessary drivers.

- The communication between the S.USV and the single-board computer happens via the I2C - interface, please activate and check this first using the following steps:

1. First you have to install the relevant I2C-Tools to be able to see which devices are connected to your single-board computer. To do this, you have to enter the following commands in the Terminal to install the i2c-tools utility:

   sudo apt-get install python-smbus

   ```
   pi@raspberrypi ~ $ sudo apt-get install python-smbus
   Reading package lists... Done
   Building dependency tree
   Reading state information... Done
   ```

   sudo apt-get install i2c-tools

   ```
   pi@raspberrypi ~ $ sudo apt-get install i2c-tools
   Reading package lists... Done
   Building dependency tree
   Reading state information... Done
   ```

2. Once, the steps above are done, reboot the system by using the command ***sudo reboot***

3. Now, when you log in, you can use the following command to see all the connected devices:

   sudo i2cdetect -y 1

   ```
   pi@raspberrypi ~ $ sudo i2cdetect -y 1
        0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
   00:          -- -- -- -- -- -- -- -- -- -- -- -- 0f
   10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
   20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
   30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
   40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
   50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
   60: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
   70: -- -- -- -- -- -- -- --
   ```

   This shows, that two I2C-Adresses are in use – 0x0F for the S.USV and 0x68 for the Real Time Clock located on the S.USV.

3.2.5 S.USV

1. Please download the provided debian package from our download section and save it to any local memory address on your single-board computer.

2. To install the Debian package on your single-board computer switch to directory in which the debian package is located and perform the following commands in the command line.

   sudo tar -xvf susvd-en-x.x-all.tar (to unzip the tar file)

   sudo dpkg -i susvd-en-x.x-all.deb (to install the debian package)

```
pi@raspberrypi ~ $ cd /home/pi/S.USV-Install/
pi@raspberrypi ~/S.USV-Install $ ls
susvd-en-1.1-all.deb
pi@raspberrypi ~/S.USV-Install $ sudo dpkg -i susvd-en-1.1-all.deb
```

3. The S.USV Client and Daemon are now fully installed and ready for use.
   (The installed files are located in the following path: ***/opt/susvd***)

4. In case of a successful installation, change to the directory ***/opt/susvd*** and run the following command as superuser to start the daemon and ensure proper work of the S.USV (Refer to section 4 for a description of all the commands):

   sudo ./susvd -start

```
pi@raspberrypi ~ $ cd /opt/susvd
pi@raspberrypi /opt/susvd $ sudo ./susvd -start
S.USV Daemon started..
pi@raspberrypi /opt/susvd $
```

   The correct start of the Daemon will be shown in the console output.

3.2.5.1 Register

Command	Array - Field	Returned Value	Mode	Comment
0xD0	0	Command	Read	0xD0
	1	Voltage in [mV]	Read	first 8 bit
	2		Read	last 8 bit
0xD1	0	Command	Read	0xD1
	1	Power Extern [mA]	Read	first 8 bit
	2		Read	last 8 bit
0xD2	0	Command	Read	0xD2
	1	Power Battery [mA]	Read	first 8 bit
	2		Read	last 8 bit
0xD3	0	Command	Read	0xD3
	1	Battery Voltage [mV]	Read	first 8 bit
	2		Read	last 8 bit
0xD4	0	Battery Status	Read	0 = Charge 1 = Full 2 = Failure
0x35	0	Command	Read	0x35
	1	Charge Status	Read	1 = Activated <0 = Deactivated
	2	Charge Current	Read	0 = 1000mA 1 = 500mA 2 = 300mA
0x22	0	Command	Read	0x22
	1	Firmware	Read	Major Version
	2	Firmware	Read	Minor Version
	3	Model	Read	0 = Advanced 1 = Basic
0x45	0	Command	Read	0x45
	1	Power Status	Read	0 = Secondary 1 = Primary

Command	Parameter	Mode	Comment
0x37	0 = 1000mA 1 = 500mA 2 = 300mA	Write	Change charging current
0x41	i2c-Address	Write	Change i2c-Address
0x29	-	Write	Activate Charging circuit
0x27	-	Write	Deactivate Charging circuit

3.2.6 RTC – Real Time Clock

The integrated real-time clock is a useful addition to the single-board computer. For existing Ethernet connection the current time synchronizes via the NTP (Network Time Protocol) service. In various scenarios the connection to a network is not possible. This may be the case in the car, at a solar or wind turbine or even when using the single-board computer in the control cabinet. With the use of the real time clock, the system time is kept up to date even when there is no network connection.

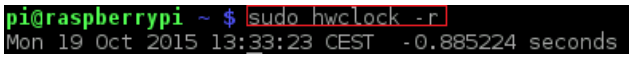
The ID EEPROM on the board configures the RTC module automatically.

(Note: Refer to point 2 to set up the current time.)

- Please verify that the chip module is seen by running ***sudo i2cdetect -y 1*** at the command line. The DS1308 Real Time Clock should be located at the I2C address ID #68.

1. Now check the time on the RTC device using:

   sudo hwclock -r

   If this is the first use of the RTC, it will report back Jan 1 2000.

   Please configure now the current time and then conform with:

   sudo hwclock -w

   to write the system time to the RTC module.

3.2.6.1 Timed switching on and off

In various scenarios, the single-board computer is a hard-wired module of an overall component. Partial accessibility suffers from a variety of conditions, sometimes it also deliberately avoided.

In many cases, the single board computer should be woken up at a defined time, retrieve various sensor information, pass them according to then automatically go back to the shutdown.

Also for the application as a media center, the timed switching on and off is a useful addition.

In order to extend these applications through a useful function, we implemented the time-controlled switching on and off. The user can easily transfer the desired times for startup and shutdown in the config, everything else is controlled by the integrated RTC chip and related software.

For more information on configuring and using the function, see chapter 4.2.1 Client Options.

4 Client Software

Communication between S.USV and the single-board computer via the I2C interface at the address 0x0F.
In principle the software package of the S.USV consist of two tools:

1. The S.USV Daemon (susvd), which monitors and controls the S.USV by constantly reading the S.USV status and reacting on several events. The S.USV Daemon will be started once and is running in the background.

2. The S.USV Client (susv), which gives the user the possibility to get and see the actual status of the S.USV as well as to control the S.USV, e.g. activating or deactivating the charging circuit. The S.USV-Client is also responsible for editing the config-variables of the S.USV Daemon, eg. Shutdown timer.

4.1 S.USV – Daemon

The S.USV daemon is responsible for monitoring and controlling the S.USV in conjunction with the single-board computer.

The S.USV daemon creates a log in the file: ***/var/log/susvd.log***

The following sections will be shown the individual options.

4.1.1 Daemon Configuration

To configure the S.USV Daemon change to the directory ***/opt/susvd*** and execute the following commands as superuser:

sudo ./susv -timer <time in seconds>

(Default value = 10)

This value indicates how long the system continues to run before the filesafe shutdown will be initiated by the S.USV after the voltage supply is switched to battery.

Values “>=0” are possible.

```
pi@raspberrypi ~ $ sudo ./susv -timer 60
**********************************
*                                *
* S.USV pi solutions             *
* www.s-usv.de                   *
*                                *
* Model: Advanced                *
* Firmware Version: 1.3          *
* Software Version: 1.3          *
*                                *
* Mail notification: Enabled     *
*                                *
* Timed Boot: Enabled            *
* Boot time: 07:00:00            *
*                                *
* Timed Shutdown: Enabled        *
* Shutdown time: 18:00:00        *
*                                *
* Sun Jan 10 09:00:10 2016       *
*                                *
**********************************
*                                *
* Shutdown timer set to: 60      *
*                                *
* Please restart S.USV Daemon    *
*                                *
**********************************
```

This function can be disabled by the value “-1”.

In this case, the single-board computer remains running, until the Battery Capacity reaches 10%. From that point on, the S.USV will automatically perform a filesafe shutdown in order to prevent the battery from getting damaged.

sudo ./susv -auto <0/1>

(Default value = 1)

This value determines the starting behavior of the S.USV Daemon. The value "1" activates the autostart, the value "0" disables the autostart.

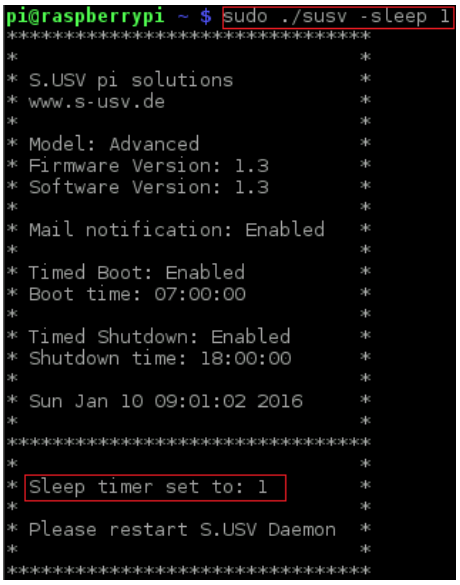
```
pi@raspberrypi ~ $ sudo ./susv -auto 1
**********************************
*                                *
* S.USV pi solutions             *
* www.s-usv.de                   *
*                                *
* Model: Advanced                *
* Firmware Version: 1.3          *
* Software Version: 1.3          *
*                                *
* Mail notification: Enabled     *
*                                *
* Timed Boot: Enabled            *
* Boot time: 07:00:00            *
*                                *
* Timed Shutdown: Enabled        *
* Shutdown time: 18:00:00        *
*                                *
* Sun Jan 10 09:00:31 2016       *
*                                *
**********************************
*                                *
* Autostart enabled              *
*                                *
* Please restart S.USV Daemon    *
*                                *
**********************************
```

If the autostart is disabled, please notice that you have to start the S.USV-Daemon manually in order for the S.USV to work correctly.

sudo ./susv -sleep <time in seconds>

(Default value = 1)

This value determines the repetition in which the S.USV Daemon monitors and controls the voltage output of the S.USV. Values “>=0” are possible.



4.1.2 Daemon Controlling

To control the S.USV Daemon change to the directory ***/opt/susvd*** and execute the following commands as superuser:

sudo ./susvd -start

Starts the S.USV Daemon Service and its configuration.

sudo ./susvd -stop

Stops the S.USV Daemon Service.

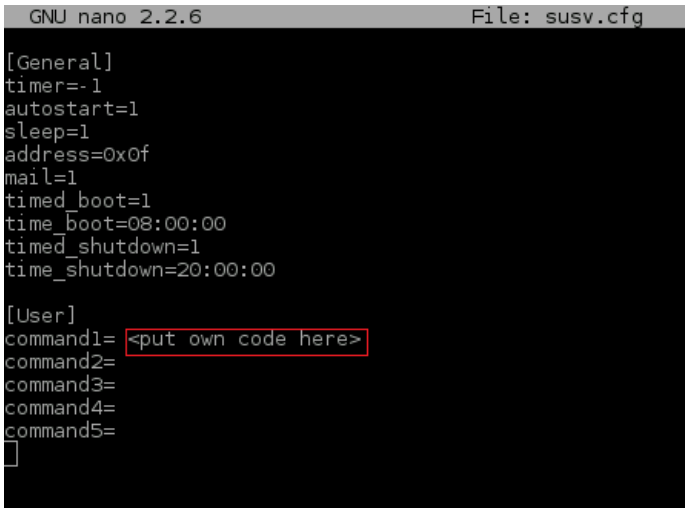
sudo ./susvd -restart

Restarts the S.USV Daemon Service.

4.1.3 User Shutdown Script

Within the S.USV config it is possible to include various commands or full scripts. These are carried out in accordance with the shutdown by the daemon, so the user has the option to configure the shutdown process according to his wishes and conditions.

Change to the directory ***/opt/susvd*** and edit the associated config file ***sudo nano susv.cfg***



4.2 S.USV – Client

The S.USV Client allows the user state monitoring and function control of the S.USV.

The following sections will be shown the individual options.

4.2.1 Client Options

To control the S.USV Client change to the directory ***/opt/susvd*** and execute the following commands:

./susv -help

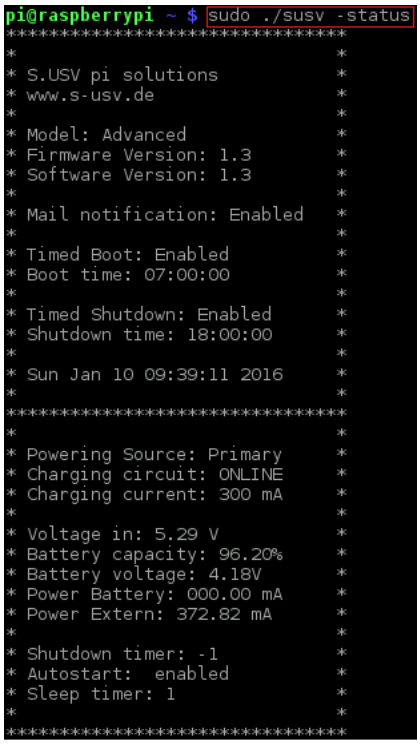
This command shows all possible options.

Overview and notification:

./susv -status

Read S.USV status.

This command allows to read the S.USV status. Here of all available modes are indicated as well as the current powering source and its power consumption.



./susv -mail <1/0>

(Default value = 0)

Enable / Disable notification via e-mail.

With activated notification via e-mail, a corresponding notification to the configured address is transmitted in case of power loss.

Change to the directory ***/opt/susvd/scripts*** and edit the corresponding python script with ***sudo nano mail.py***

```
  GNU nano 2.2.6                File: mail.py

import smtplib
from email.MIMEMultipart import MIMEMultipart
from email.MIMEText import MIMEText
from email.MIMEBase import MIMEBase
from email import encoders

fromaddr = "YOUR EMAIL"
toaddr = "EMAIL ADDRESS YOU SEND TO"

msg = MIMEMultipart()

msg['From'] = fromaddr
msg['To'] = toaddr
msg['Subject'] = "SUBJECT OF THE EMAIL"

body = "TEXT YOU WANT TO SEND"

msg.attach(MIMEText(body, 'plain'))

filename = "susvd.log"
attachment = open("/var/log/", "rb")

part = MIMEBase('application', 'octet-stream')
part.set_payload((attachment).read())
encoders.encode_base64(part)
part.add_header('Content-Disposition', "attachment; filename= %s" % filename)

msg.attach(part)

server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(fromaddr, "YOUR PASSWORD")
text = msg.as_string()
server.sendmail(fromaddr, toaddr, text)
server.quit()
```

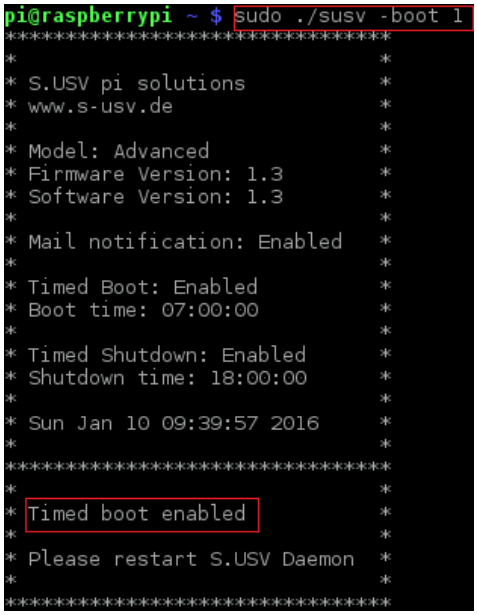
Timed switching on and off:

./susv -boot <1/0>

(Default value = 0)

Enable / Disable time-controlled activation.

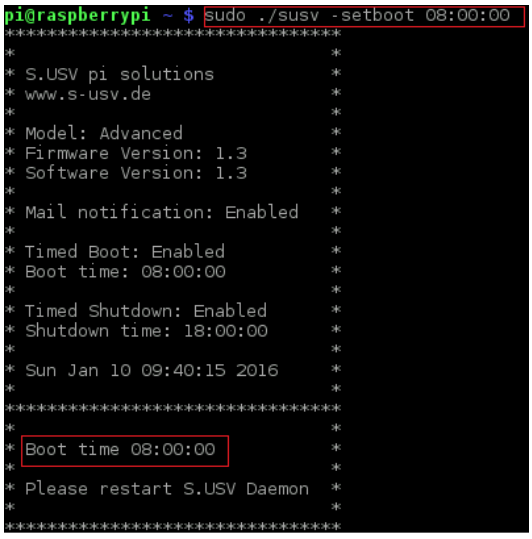
With this command the timed switching on the single-board computer is configured. The value „1“ stands for the activation, the value „0“ for the deactivation.



./susv -setboot <HH:mm:ss>

Configure time for the timed switching.

Use this command to configure the desired time to ramp up the single-board computer for the defined time.

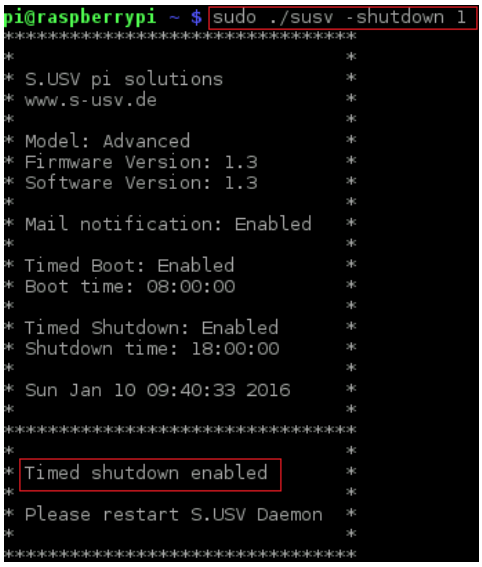


./susv -shutdown <1/0>

(Default value = 0)

Enable / Disable time-controlled activation.

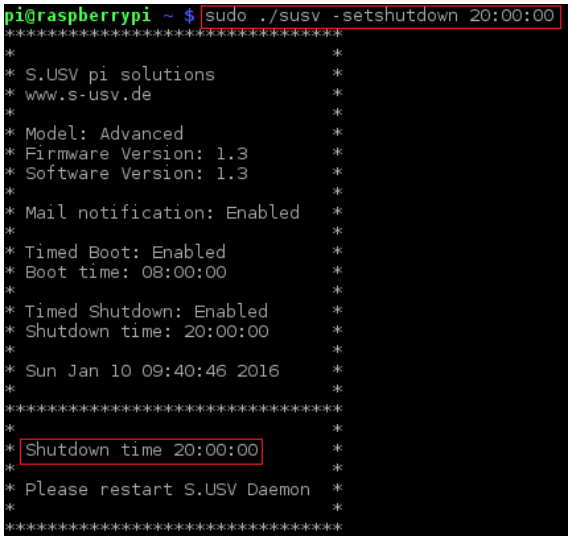
With this command the timed switching on the single-board computer is configured. The value „1“ stands for the activation, the value „0“ for the deactivation.



./susv -setshutdown <HH:mm:ss>

Configure time for the timed switching.

Use this command to configure the desired time to shutdown the single-board computer for the defined time.

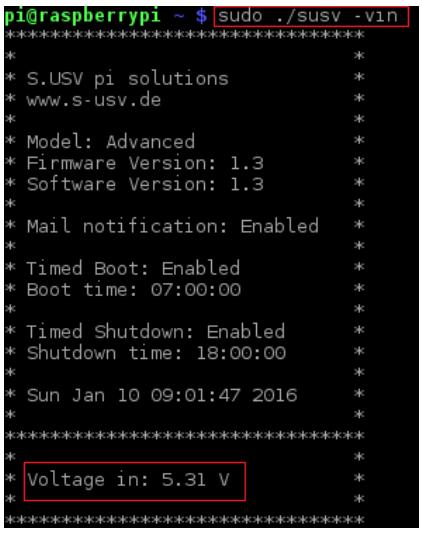


Power sources and supply:

./susv -vin [0]

Read input voltage.

With this command it is possible to check the current input voltage.

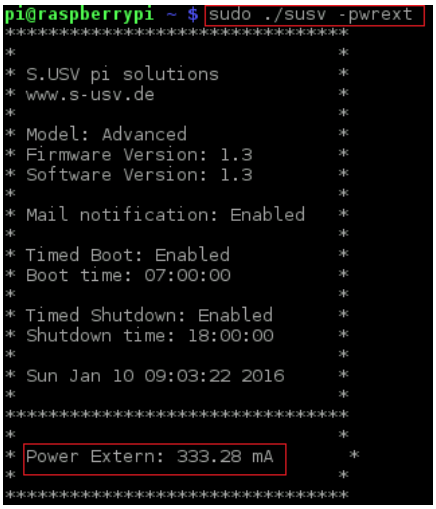


(Note: An attached „0“ to the command returns only the value.)

./susv -pwrext [0]

Read the external power consumption.

With supply via the external voltage input, the current external power consumption can be checked by this command.

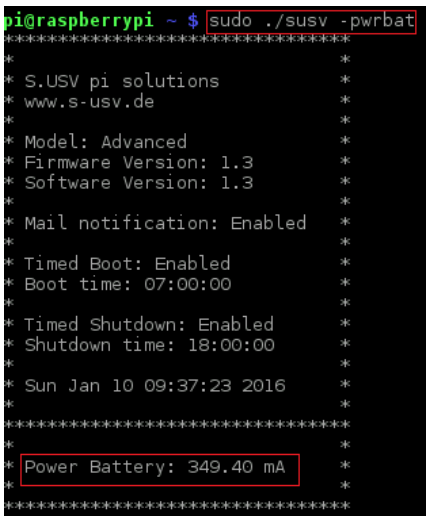


(Note: An attached „0“ to the command returns only the value.)

./susv -pwrbat [0]

Read the battery power consumption.

With supply via battery, the current battery power consumption can be checked by this command.



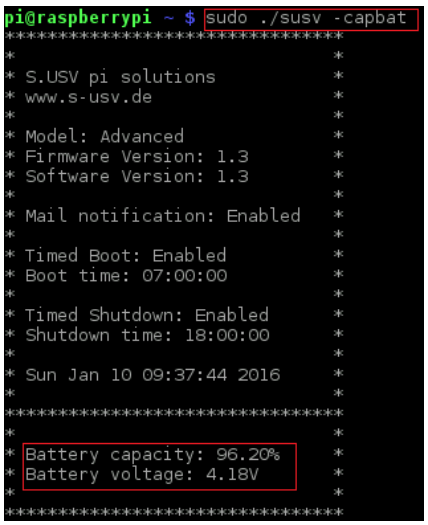
(Note: An attached „0“ to the command returns only the value.)

Battery and charging circuit:

./susv -capbat [0]

Read battery capacity.

This command allows to read the current battery voltage and the remaining battery capacity.



(Note: A remaining capacity of <25% is indicated by the LED BATRD. At a remaining capacity of <10% the single-board computer will automatically shut down.)

(Note2: An attached „0“ to the command returns only the value.)

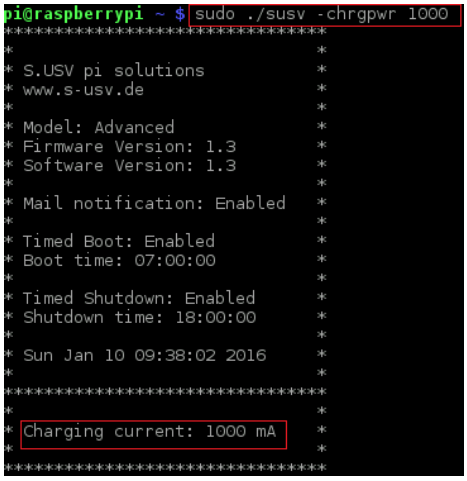
sudo ./susv -chrgpwr <300/500/1000>

(Please use this command as superuser)

Change the charge current for the battery.

Use this command to change the active charge current for the battery. In order to minimize the charging time following current strengths are available.

- 300mA
- 500mA
- 1000mA

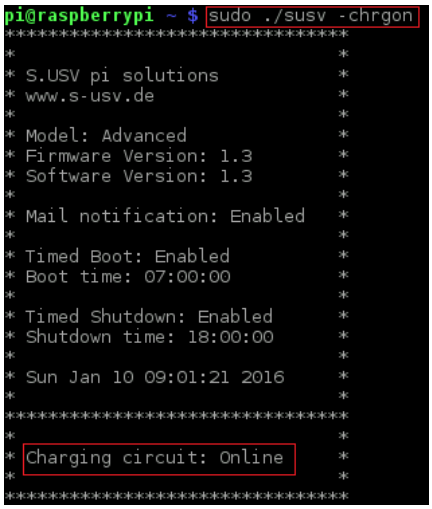


(Note: The configured charging current is stored in EEPROM and reloaded at startup.)

./susv -chrgon

Switch charging circuit on.

Allows the manual switching of the battery's charging circuit.

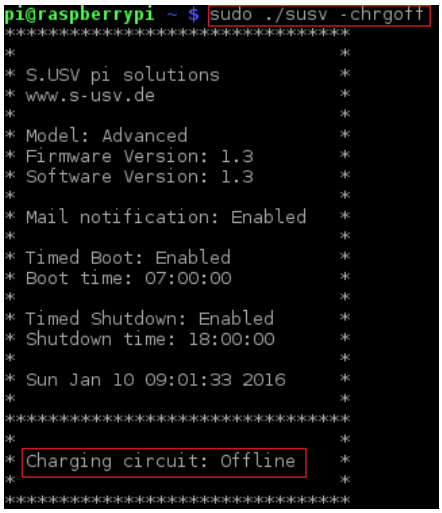


(Note: The configuration of the charging circuit is stored in EEPROM and reloaded at system startup.)

./susv -chrgoff

Switch charging circuit off.

Allows the manual shutdown of the battery's charging circuit.



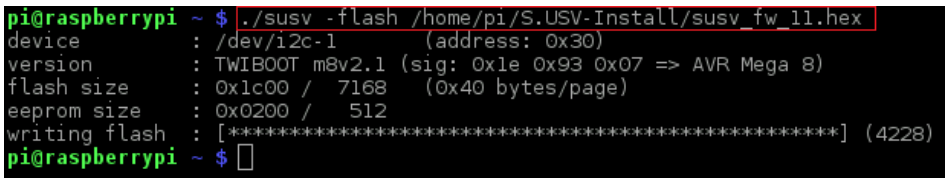
(Note: The configuration of the charging circuit is stored in EEPROM and reloaded at system startup.)

Firmware and addressing:

./susv -flash <path to HEX file>

Upgrade the firmware.

Use this command to upgrade the actual firmware.



(Note: Please restart the system after the flash process and re-initialize of the S.USV.)

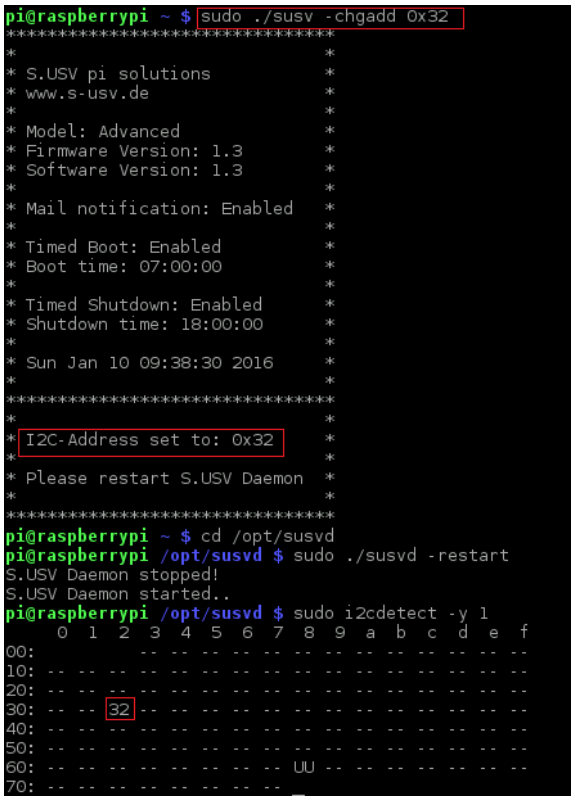
sudo ./susv -chgadd <0x..>

(Default address = 0x0f)

(Please use this command as superuser)

Change the I2C address of the S.USV.

To avoid potential compatibility issues, use this command to change the I2C address of the S.USV. Verify the address by typing ***i2cdetect -y 1*** in the terminal.



(Note: The configured I2C address is stored in EEPROM and reloaded at startup.)

5 Support

Please contact support@susv.de if you need further assistance.

5.1 Tutorials

5.1.1 RTC Configuration - SYSTEMD

The corresponding RTC module (DS1307) is automatically configured by the ID EEPROM and loaded into the kernel. Please use the following steps to use the RTC as a hardware clock:

1. Disable the fake-hwclock of the single-board computer, which can lead to complications with the RTC. To do this, use the following commands in the terminal:

   sudo apt-get -y remove fake-hwclock
   sudo update-rc.d -f fake-hwclock remove

2. After the fake-hwclock has been disabled, the original hardware clock script should be adjusted accordingly. Edit the ***/lib/udev/hwclock-set*** file and uncomment the following lines:

   #if [-e /run/systemd/system] ; then
   #exit 0
   #fi

3. Restart the system

   sudo reboot

   Then check the correct system time and write it into the RTC

   sudo hwclock -w

5.2 FAQ – Frequently Asked Questions

❖ **What are the advantages of the S.USV solutions modules?**

- During the work process on the single board computers, the operating system and user data are written continuously to the SD card. A sudden failure or interruption of the power supply during the writing of files can lead to loss and / or corruption of important data. The S.USV solutions bridges these voltage fluctuations and gives the single-board computers more stability in the supply due to the uninterrupted power supply. Furthermore, S.USV solutions has additional useful functions.

❖ **Are the S.USV solutions modules completely plug & play?**

- The S.USV solutions modules are intelligent plug & play solutions. No further configurations are necessary to use the main function of the uninterruptible power supply. The interface and communication between the single board computer and S.USV are automatically configured via the HAT EEPROM. A corresponding S.USV client and daemon is available for monitoring and controlling the modules.

❖ **Which single board computers are compatible with the S.USV solutions modules?**

- The S.USV solutions series currently consists of three different versions:
  - S.USV pi (Compatible with all Raspberry Pi models)
  - S.USV UPs (Compatible with all AAEON UP Boards)
  - S.USV tinker (Compatible with all ASUS Tinker Boards)

❖ **How does the power supply of S.USV work?**

- The power supply of the S.USV solutions comes directly via the GPIO power pins of the single board computer and thus uses a common voltage source, which means that no further cabling or power supply is necessary. The "Advanced" variant also offers a optional input for the extended voltage range of 7 - 24 V (solar cells, automotive area, etc.).

❖ **Which voltage supplies can be connected via the external input?**

- All voltage sources with a voltage range between 7-24V and a minimum output current of 2A can be used.

❖ **Which protective circuits have been implemented with regard to voltage supply**?

- To protect the external power supply, the S.USV module and the single-board computer from damage, the following precautions have been taken on both the primary and secondary side:
  - ZVD - Circuit
  - TVS - Circuit
  - PPTC Fuse
  - Power Monitoring System

❖ **What is the output power of the S.USV solutions modules?**

- The S.USV solutions modules provide the following output power:
  - Primary: 5V / 3.5A (+/- 5%)
  - Secondary: 5V / 3.5A (+/- 5%)

❖ **Which GPIO pins are used by the S.USV?**

- For the power supply and data transmission of the S.USV following GPIO - Pins in use:

  - **Pin #02**: DC Power +5V – Power supply
  - **Pin #03**: GPIO 02 (SDA1, I²C) – I²C data line
  - **Pin #05**: GPIO 03 (SCL1, I²C) – I²C clock line
  - **Pin #13**: GPIO 27 (GPIO_GEN2) – Monitoring S.USV
  - **Pin #27:** ID_SD (I²C ID EEPROM) – ID data line
  - **Pin #28:** ID_SC (I²C ID EEPROM) – ID clock line

❖ **How does the automatic switchover from primary to secondary supply mode work?**

- The input voltage of the entire system is cyclically monitored by the software and compared with an internally configured voltage reference. In order to meet the supply specifications of 5V +/- 5%, the voltage is automatically switched to the secondary mode (battery operation) if the voltage drops <4.75V. When the primary power supply returns, the S.USV automatically returns to the primary mode.

❖ **What kind of batteries are used with the S.USV solutions modules?**

- Lithium polymer accumulators specially designed for the system are used. Each product is supplied with a 300mAh battery, a 3000mAh higher capacity battery can be purchased in our online shop.

  Refer to P.4 - Chapter 2 Technical Specifications for further battery characteristics. We generally advise against using accumulators with deviating characteristics.

❖ **Can batteries with higher capacity be used?**

- There is generally no capacity limitation regarding the use of lithium-polymer or lithium-ion accumulators. The current revision 2.0 was positively tested up to a capacity of 10,000mAh.

  For further data regarding battery characteristics, see page 4 - Chapter 2 Technical Specifications. We generally advise against using accumulators with deviating characteristics.

❖ **How does the battery charging circuit work on the S.USV solutions modules?**

- An intelligent Battery Management System has been implemented for all S.USV solutions products. The charging circuit was specifically designed for the use of lithium-polymer and lithium-ion batteries. The charging circuit divides the charge of the batteries into four charging cycles, controlled by each other, to charge the battery gently and thereby increase the service life. The batteries offered by us can be charged up to a charging current of 1A, the user can configure the charging current via the client software (300, 500 and 1000mA).

❖ **Which protective circuits have been implemented with regard to the voltage supply through the batteries?**

- The intelligent Battery Management System, as well as the Protection Circuit modules on the batteries offered by us, are protected by both the software and the hardware from damage (overvoltage, deep discharge and overheating). With a residual capacity of <25%, this is signaled by the status indicators, with a residual capacity of <10%, the system is led into the shutdown process in order to avoid a deep discharge. In secondary mode (battery operation), the charging circuit is automatically deactivated.

  Software and hardware work independently in this scenario, in order to protect the accumulators in any case from damage.

❖ **Can the S.USV solutions modules also be used in conjunction with other HATs?**

- A main focus of the S.USV solutions series lies in the modularity and variability of the system. All GPIO pins are connected 1:1 and can be re-used freely, only GPIO pin 13 (GPIO27) is reserved for the status monitoring by the S.USV.

  The following product pallets have been positively tested so far:

  - HiFi Berry
  - FHEM
  - HomeMatic
  - Enocean
  - RPI WWCAM

❖ **How does the automatic shutdown work?**

- With the automatic shutdown of S.USV, the measured input voltage is continuously compared with a reference value (voltage threshold). If the input voltage falls below the internally configured threshold of 4.75V, the S.USV automatically switches to the secondary mode (battery operation). The resulting shutdown process is handled in this mode, depending on the configured shutdown timer.

  For further information and configuration options, please refer to page 16 - Chapter 4.1.1 Daemon Configuration.

❖ **Is the system automatically restarting after the power supply is restored?**

- A return of the primary voltage immediately leads to a restart when the system is switched off. The system goes into the reboot process. Also during the shutdown process, a return of the primary voltage results in a direct reboot of the system upon completion of the shutdown operation.

❖ **How does the time-controlled switching of the S.USV solutions modules work?**

- The user can activate and configure the function of the time-controlled switching of the entire system. After successful configuration, the system can be put into the idle state at the configured shutdown time. The integrated real-time clock is evaluated in this state by appropriate firmware routines and responds to it depending on the configured boot time. If the configured time matches, the system automatically enters the boot process.

  For further information and configuration options, please refer to page 19 - Chapter 4.2.1 Client Options.

❖ **Is there the possibility of a preconfigured mail notification / alarming?**

- It is possible to activate the appropriate mail notification by the S.USV. For this purpose, the user saves his mail data into a predefined Python script, which is sent when the primary supply voltage is lost, in order to inform the customer about the failure of the supply source.

  For further information and configuration options, please refer to page 19 - Chapter 4.2.1 Client Options.

❖ **Is there the possibility of preconfigured user shutdown scripts?**

- Using the Config of the S.USV modules (susv.cfg in /opt/susvd), corresponding user shutdown scripts can be created by the customer, which are executed when the shutdown process is initiated.

  For further information and configuration options, please refer to page 19 - Chapter 4.2.1 Client Options.

❖ **How to communicate between S.USV and single board computer?**

- The communication between S.USV and single board computer happens via the I2C interface (3.3V level). The S.USV modules operate as slaves, so the I2C bus can be easily re-used over other peripherals.

  All available data such as input voltage, power consumption or remaining capacity of the battery can be further processed via corresponding I2C registers or directly via the preconfigured client commands. This allows the user to react independently to different events.

  For further information or configuration options, please refer to page 14 - Section 3.2.5.1 Register and page 19 - 4.2.1 Client Options.

5.3 Troubleshooting

❖ **HW/SW Compatibility**

- Please note the corresponding compatibility between hardware and software revision. The non-given functionality, as well as the faulty parameters, may result from corresponding incompatibility.

  Please use the 1.X firmware and software updates for the hardware revision 1.X. This also applies to the current hardware revision 2.X and the associated software updates.

❖ **I2C-Request timed out – I2C-Address change**

- First, check the entire I2C bus using the ***sudo i2cdetect -a -y 1*** command; the I2C address of the S.USV may have been moved to another address (for example, 0x01, 0x00).

- In this case, please manually transfer the address to the susv.cfg located in ***/opt/susvd*** and execute a restart of the daemon process

- If the I2C communication has been activated again, you can reset the I2C address back to the original 0x0f.

❖ **S.USV does not switch off completely**

- The problem indicates a faulty configuration of the S.USV daemon process.

  Please check first if it works correctly - Change to the directory ***/opt/susvd*** and execute the following command ***sudo ./susvd -status***

❖ **S.USV pi basic switches unexpectedly in secondary mode - Boot Cycle**

- Please note that the supply of various peripherals and activated charging circuit via the microUSB input of the single-board computer can lead to voltage drops on the GPIO power pins. The S.USV switches automatically to the secondary mode with an input voltage <4.75V and with the daemon process activated in the shutdown.

  Please proceed as follows to check the problem:

  - Shutdown the system
  - Removing the S.USV solutions module
  - Reboot the system and stop the S.USV Daemon process

- Restart the system with S.USV solutions module
- You can now measure the input voltage directly at the GPIO pins

In most cases the problem could be solved by a more powerful power supply with an output voltage of 5.25V. For systems with higher power consumption we recommend to realize the power supply via the external input of the S.USV, which can supply up to 5A.

❖ **S.USV ignores return of the primary power supply**

- In the secondary mode (battery operation) of the S.USV, current values are continuously determined and further processed in corresponding firmware programs. This allows the modules to recognize a return of the primary power supply, resulting in a significant reduction in power consumption through the secondary input. The intelligent software detects and responds to the performance of the primary power source and responds accordingly.

  If the S.USV module does not switch back to primary mode after the power source has returned, we recommend testing the scenario with more powerful power supplies.

❖ **Shutdown/Reboot Button S1 does not react**

- For the function of the S1 Shutdown/Reboot button, a correct working of the S.USV Daemon process is necessary, which processes the corresponding signals.

  Please check first if it works correctly - Change to the directory ***/opt/susvd*** and execute the following command from ***sudo ./susvd -status***

Olmatic GmbH / Marie-Curie-Straße 3 / 72202 Nagold
Tel. +49 (0) 74 52 - 92 99 96 - 0 / Fax +49 (0) 74 52 - 92 99 96 - 9
E-Mail info@s-usv.de / www.s-usv.de