

Inbetriebnahme eines Steuerrechners für eine Photovoltaik-Insel mit einem lokalen Datenspeicher

Inhalt

Komponenten.....	4
Betriebssystem aufspielen	4
Rpiboot.exe	4
Raspi vorbereiten:	5
Rpiboot.exe ausführen	5
Pi Imager.....	6
Raspi booten.....	7
SSH-Verbindung.....	7
Stromversorgung.....	8
Netzteil für Hutschiene.....	8
Stromversorgung des Raspi-IO-Board	8
Stromversorgung der SATA-Festplatte.....	9
Einrichten der Festplatte.....	10
Prüfen, ob die Festplatte verfügbar ist mit	10
Partitionierung und Formatierung der Festplatte.....	11
Variante 1	11
Variante 2	11
Dateisystem EXT4 erstellen	12
Festplatte mounten.....	12
Verzeichnis erstellen und Rechte vergeben	12
Variante 1 – manuelles Mounten.....	12
Variante 2 – automatisches Mounten bei jedem Start	12
Test-Datei erstellen	13
Mount rückgängig machen.....	13
Samba und Freigaben -Zugriff auf die Festplatte von Windows aus	14
SMB- Manual anzeigen.....	14
Status der Samba-Dienste abfragen.....	14
Samba-Komponenten installieren.....	14

Freigaben in die Konfig-Datei eintragen.....	15
Samba-Dienste neu starten.....	15
User für Login beim Aufruf der Freigaben unter Windows.....	16
MariaDB.....	17
Installation Version 10.5.....	17
Sicherstellen, dass der Service nach dem Systemstart ausgeführt wird.....	17
Konfigurieren.....	18
Wo liegen die Konfig-Dateien?.....	18
Data-Dir verschieben.....	18
Dienst stoppen	18
Verzeichnis auf der Festplatte anlegen und Rechte vergeben.....	18
Konfiguration korrigieren	18
Bestehendes Verzeichnis verschieben	19
Dienst neu starten.....	19
Remotezugriff ermöglichen.....	19
Mariadb-Service neu starten.....	19
Weiteren User anlegen	19
Anmelden	19
Datenbank erstellen	19
Rechte für Datenbank vergeben	19
Test des Remote-Zugriffs von HeidiSql auf MariaDb von Windows aus.....	20
Log	20
Abmelden	20
Python	21
Installation.....	21
Python-Connector für die mariadb	21
Script timergesteuert ausführen	21
GPIO-Aktoren und -Sensoren	22
Aktoren: 3-Kanal-Relais-Board	22
Sensoren.....	22
https://github.com/grasmax/AcOnOff/blob/main/script/gh_gpiointest.py	23
Übergang von der virtuellen in die reale Welt.....	23
SSH-Verbindung einrichten	24
Dienste und Startreihenfolge	25
5 Arten.....	25
systemd verstehen	26

Besonderheit bei mariadb	27
Hier hatte Require=mnt... nichts gebracht.....	27
Alle Unit-Dateien auflisten	27
Alle startup-Zeiten aller Units auflisten	27
Log-Dateien	27
Montage auf der Hutschiene.....	28
Weitere nützliche Befehle	29
Anzeige von Prozessen, z.B. ssh	29
Ausschalten	29
Neustart.....	29
Temperatur.....	29

Komponenten

- Raspberry Pi CM4IO Board
- CM4001032 Raspberry Pi Compute Module 4, 1GB-RAM, 32GB-eMMC, BCM2711, ARM Cortex-A72
- (rasberry pi os lite (32bit) v11, 0,4 GB – getestet aber nicht produktiv)
- raspberry pi os (32bit) v11, 0.9 GB
- IO CREST JMB582 2 Port SATA III PCI-e 3.0 x1 Non-RAID Controller Karte Jmicro Chipsatz SI-PEX40148 (<https://github.com/geerlingguy/raspberry-pi-pcie-devices/issues/64>)
- 2TB Sata NAS Harddrive WD20EFZX
- 3x Relais Board - Raspberry Pi GPIO Erweiterung
- Renkforce 5x switch 5VDC
- Meanwell Duo Netzteil 66W 12V/5V

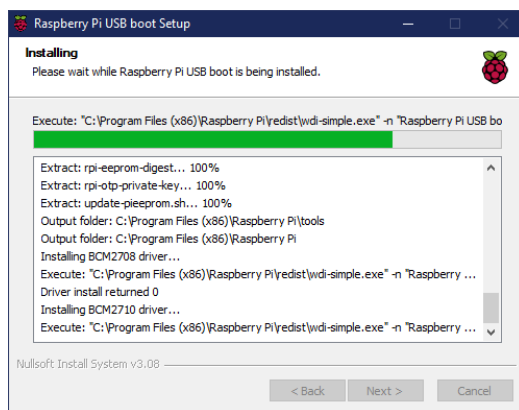
Betriebssystem aufspielen

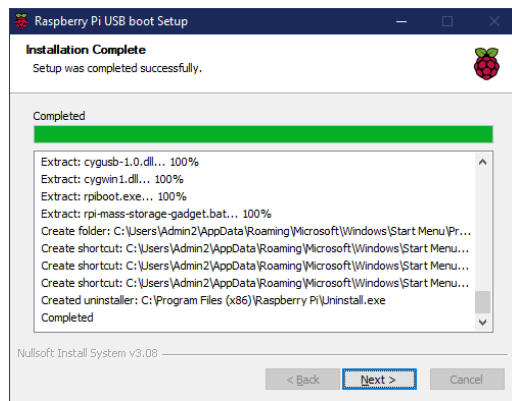
Rpiboot.exe

<https://core-electronics.com.au/guides/how-to-flash-write-raspbian-os-onto-raspberry-pi-compute-module-4-cm4/>

usbboot/win32/rpiboot_setup.exe runtergeladen und installiert nach

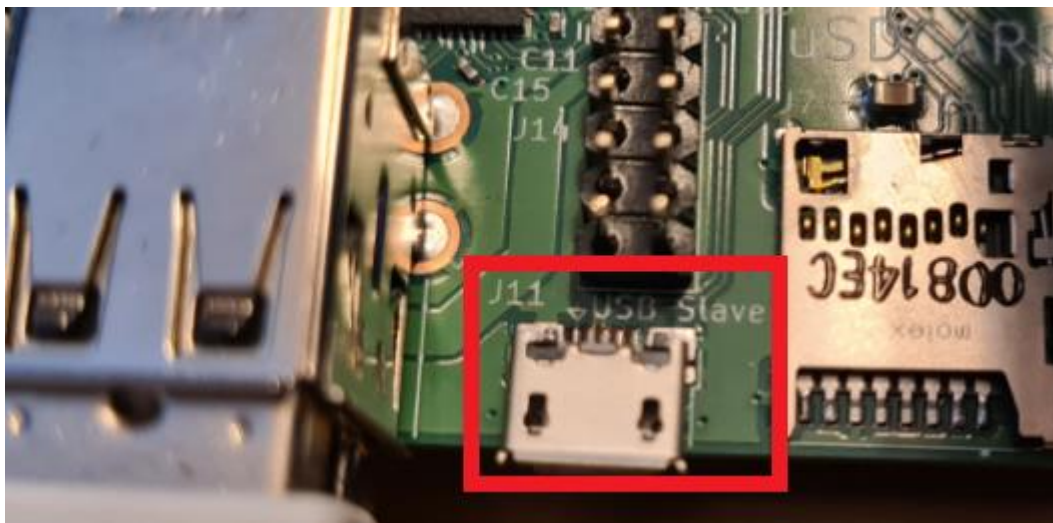
C:\Program Files (x86)\Raspberry Pi\rpiboot.exe





Raspi vorbereiten:

USB-Slave-Port des IO-Boards mit einem Micro-Usb/USB-Kabel mit dem Windows-PC verbinden. Achtung: nicht jeder USB-Port am Windows-PC ist geeignet!



Brücke/Jumper1 setzen:



Strom einschalten

Rpiboot.exe ausführen

Wenn Endlosschleife „Loading embedded: bootcode4.bin“, dann anderen USB-Port benutzen.

<https://github.com/raspberrypi/usbboot/issues/72>

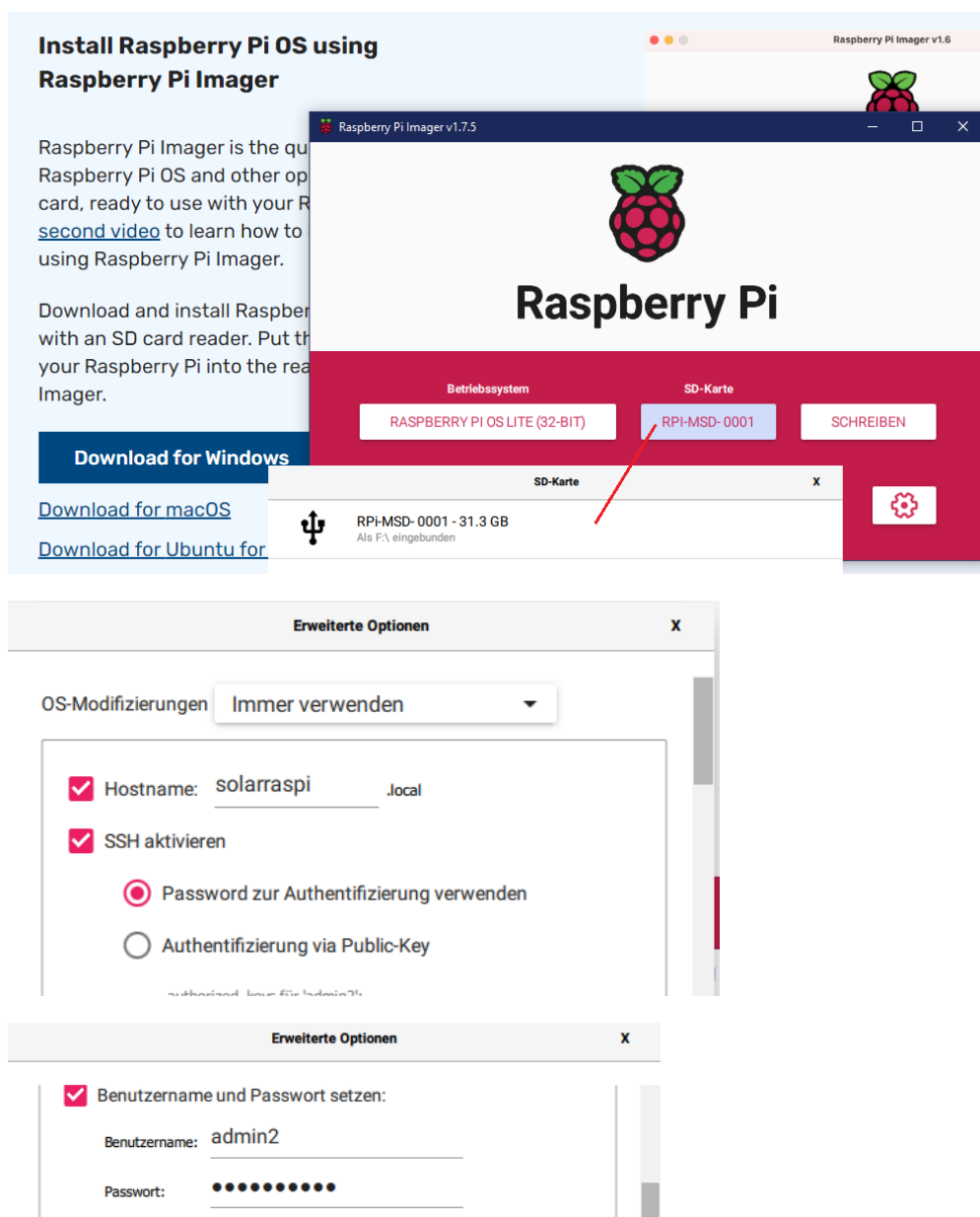
```
Eingabeaufforderung
Loading embedded: bootcode4.bin
Loading embedded: bootcode4.bin
Loading embedded: bootcode4.bin
Loading embedded: bootcode4.bin

C:\Program Files (x86)\Raspberry Pi>rpiboot.exe
RPiBOOT: build-date Dec 16 2022 version 20221215~105525 1afa26c5
Waiting for BCM2835/6/7/2711...
Loading embedded: bootcode4.bin
Sending bootcode.bin
Successful read 4 bytes
Waiting for BCM2835/6/7/2711...
Loading embedded: bootcode4.bin
Failed to claim interface
Loading embedded: bootcode4.bin
Second stage boot server
Cannot open file config.txt
Cannot open file pieeprom.sig
Loading embedded: start4.elf
File read: start4.elf
Cannot open file fixup4.dat
Second stage boot server done

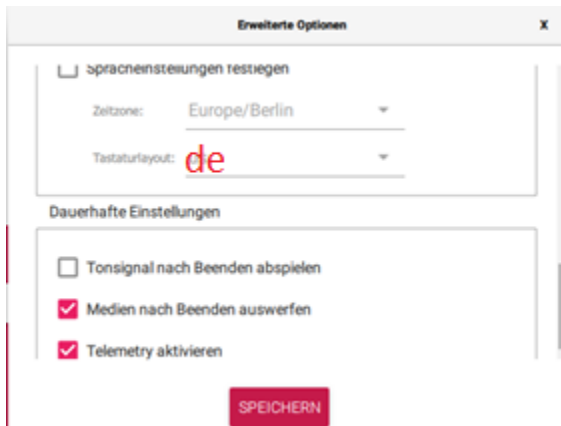
C:\Program Files (x86)\Raspberry Pi>
```

Pi Imager

Pi Imager installieren und ausführen: eMMC des CM\$ erscheint unter „SD-Karte“ zur Auswahl:



Tastatur auf „de“ einstellen! Das erspart y/z-Ärger bei z.B. Passwörtern!



Raspi booten

Strom ausschalten

USB-Kabel abziehen

Jumper-Stecker auf dem IO-Board entfernen

LAN-Kabel anstecken

Feste IP-Adresse im Router konfigurieren

Maus, Tastatur und Bildschirm anstecken

PCIe-*2Sata-Board einstecken und mit Festplatte verbinden

Strom einschalten

Wenn Tastatur und Maus nicht funktionieren:

Usb-Kabel steckt noch am Slave-Port des IO-Boards.

Texas_Dave – vor 2 Jahren

Nur ein Kommentar, um zu verhindern, dass jemand anderes in die gleiche Grube fällt wie ich. Ich habe diese Zeile zu config.txt hinzugefügt, indem ich sie über J11 verbunden und als USB-Speichergerät booten ließ. Sobald ich es in Windows sehen konnte, habe ich config.txt bearbeitet. Aber egal was, USB-Maus und -Tastatur würden nie funktionieren. Schließlich wurde mir klar, dass J11 immer noch über das Micro-USB-Kabel mit Strom versorgt wurde. Ich erinnere mich an den Schaltplan, der dadurch einen Schalter umlegt, der die USB-Signale vom HUB zu J11 umleitet. Von J11 getrennt und alles in Ordnung! Dabei gingen nur 20 Minuten verloren. :-) Nicht ungewöhnlich für unbekannte Hardware. Lektion gelernt. Ich hoffe, das hilft jemand anderem, nicht 20 Minuten oder mehr zu verschwenden.

SSH-Verbindung

mit z.B. putty testen

SSH-Einstellungen geändert (in der Hoffnung, dass dann der Login auch mit root funktioniert)

```
sudo nano /etc/ssh/sshd_config
```

u.a. PAMAuth aus

login mit root war trotzdem nicht möglich, zum Glück funktionierte der andere Login noch...

Stromversorgung

Netzteil für Hutschiene

Die Stromversorgung für das Raspi-IO-Board und die Festplatte kommt von einem

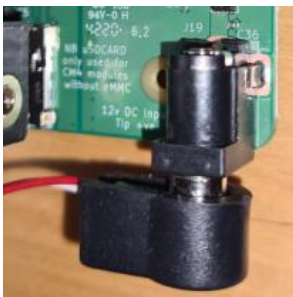
Meanwell Duo Netzteil 66W 12V/5V

Com/V1: 5VDC, Com/V2: 12 VDC



Stromversorgung des Raspi-IO-Board

Die Stromversorgung erfolgt über einen 5,5x2,1mm Hohlstecker gewinkelt



Stromversorgung LAN-Switch

Winkelstecker (3,4?)3,5mm, am besten das von Renkforce mitgelieferte Kabel benutzen.

Hier wurde ein Multi-Stecker verwendet:

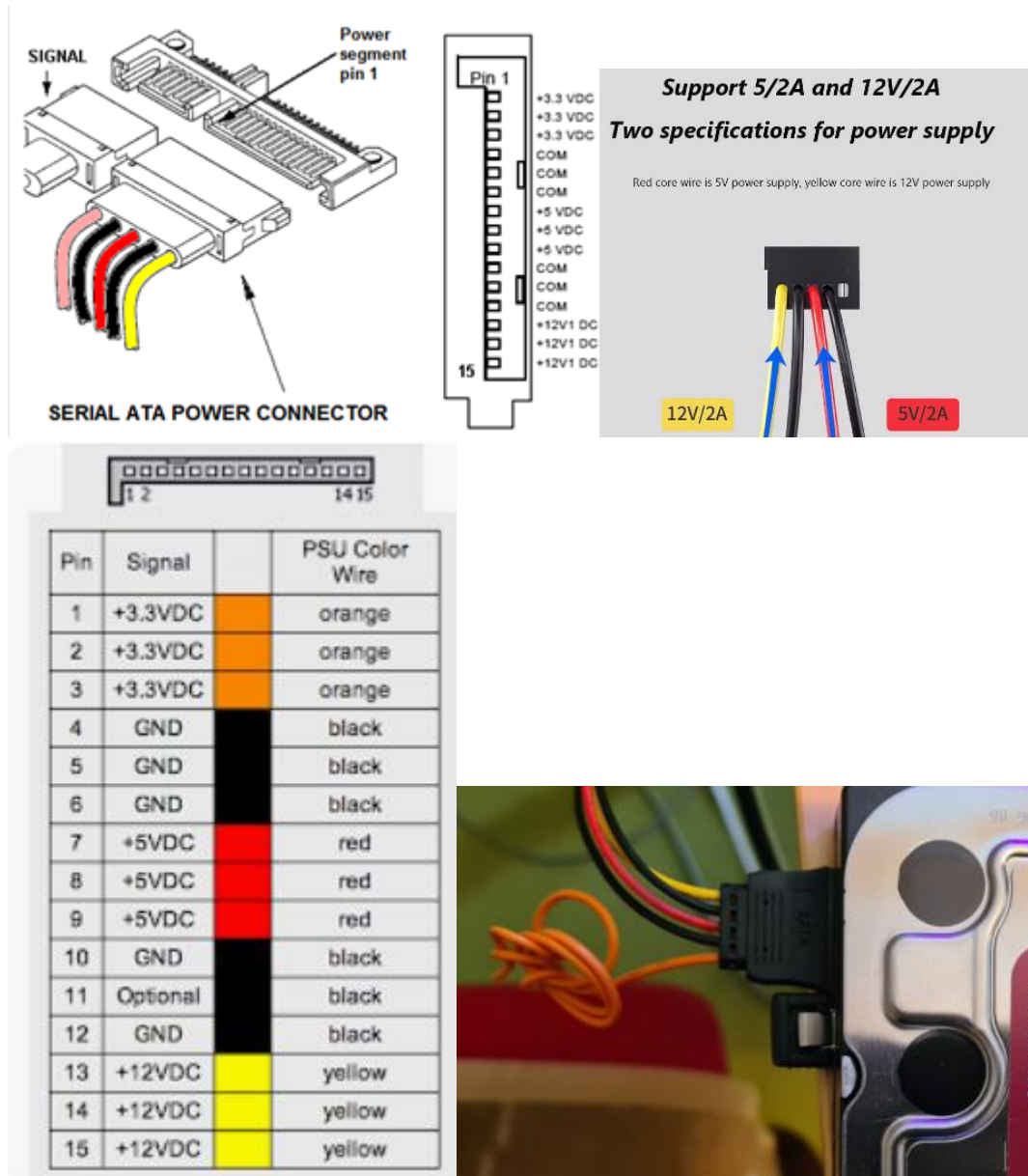


Stromversorgung der SATA-Festplatte

Wie kommen die 5V/12V an die Festplatte?

12V/0,45A über schwarz und gelb

5V/0,6A über schwarz und rot



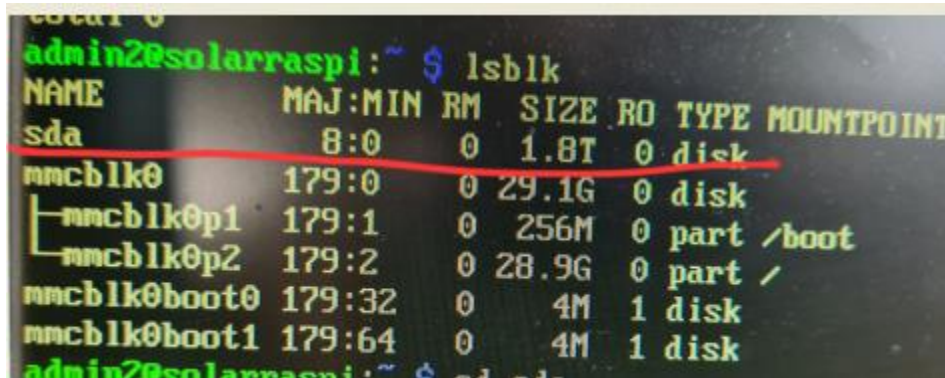
September 2023: der im Bild dargestellte Stecker wurde durch eine abgewinkelte Version ersetzt, um unter die Gehäuseabdeckung zu passen.

Einrichten der Festplatte

Prüfen, ob die Festplatte verfügbar ist mit

<https://www.elektronik-kompodium.de/sites/raspberry-pi/2102191.htm>

lsblk



```
admin2@solarrraspi:~$ lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0    0   1.8T  0 disk
mmcblk0             179:0    0   29.1G  0 disk
├─mmcblk0p1          179:1    0    256M  0 part /boot
└─mmcblk0p2          179:2    0   28.9G  0 part /
mmcblk0boot0        179:32    0     4M   1 disk
mmcblk0boot1        179:64    0     4M   1 disk
```

Partitionierung und Formatierung der Festplatte

Sudo apt-get update

Variante 1

Sudo apt install gparted - I läuft nur in OS-Version mit Desktop!

Variante 2

GPT- Partition erstellen mit fdisk

sudo fdisk /dev/sda

m (help)

g (create GPT partition table)...created

W

```
admin2@solarrraspi: ~$ sudo fdisk /dev/sda
Welcome to fdisk (util-linux 2.36.1).
Changes will remain in memory only, until you decide to write the
changes to disk. Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xad8b89fa.

Command (m for help): m

Help:
DOS (MBR)
a toggle a bootable flag
b edit nested BSD disklabel
c toggle the dos compatibility flag

Generic
d delete a partition
l list known partition types
n add a new partition
p print the partition table
t change a partition type
u verify the partition table
i print information about a partition

Misc
m print this menu
u change display/entry units
x extra functionality (experts only)

Script
I load disk layout from sfdisk script file
O dump disk layout to sfdisk script file

Save & Exit
w write table to disk and exit
q quit without saving changes

Create a new label
g create a new empty GPT partition table
G create a new empty SGI (IRIX) partition table
o create a new empty DOS partition table
s create a new empty Sun partition table

Command (m for help): g
Created a new GPT disklabel (GUID: B780C602-69EB-B14F-92DF-3483685CEB7B)

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

```
admin2@solarrraspi: ~$ sudo fdisk -l /dev/sda
Disk /dev/sda: 1.82 TiB, 2000398934016 bytes, 3907029168 sectors
Disk model: WDC WD20EFZX-68A
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: B780C602-69EB-B14F-92DF-3483685CEB7B
```

Dateisystem EXT4 erstellen

`sudo mkfs.ext4 /dev/sda`

```
mkfs 1.46.2 (28-Feb-2021)
Found a gpt partition table in /dev/sda
Proceed anyway? (y,N) y
Creating filesystem with 488378646 4k blocks and 122101760 inodes
Filesystem UUID: 11fb3855-d4b8-4f02-a0a9-49af140350dc
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
    102400000, 214990848

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting information: done
```

Festplatte mounten

Verzeichnis erstellen und Rechte vergeben

`sudo mkdir /mnt/wd2tb`

`sudo chmod 777 -R /mnt/wd2tb`

Variante 1 – manuelles Mounten

Achtung nicht das media- sondern das mnt-Verzeichnis benutzen:

`sudo mount /dev/sda /mnt/wd2tb`

Variante 2 – automatisches Mounten bei jedem Start

ID der Festplatte ermitteln:

```
sudo blkid
```

```
/dev/sda: UUID="11fb3855-d4b8-4f02-a0a9-49af140350dc" BLOCK_SIZE="4096" TYPE="ext4"
```

ID in die fstab-Datei eintragen:

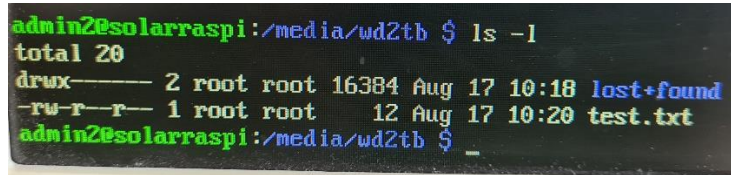
```
sudo nano /etc/fstab
```

diese Zeile hinzufügen

```
UUID=11fb3855-d4b8-4f02-a0a9-49af140350dc /mnt/wd2tb ext4 defaults,auto,users,rw,nofail,x-  
gvfs-name=wd2tb 0 0
```

Test-Datei erstellen

test.txt erstellt mit sudo nano



```
admin2@solarraspi:/media/wd2tb $ ls -l
total 20
drwx----- 2 root root 16384 Aug 17 10:18 lost+found
-rw-r--r-- 1 root root    12 Aug 17 10:20 test.txt
admin2@solarraspi:/media/wd2tb $
```

Mount rückgängig machen

Umount /mnt/wd2tb

Wenn umount fehlschlägt, dann mit

sudo lsof /mnt/wd2tb

ermitteln, welche Prozesse auf das Verzeichnis zugreifen, z.B. die Samba-Freigaben. Diese Prozesse vor umount beenden.

Samba und Freigaben -Zugriff auf die Festplatte von Windows aus

SMB- Manual anzeigen

`man smb.conf`

Status der Samba-Dienste abfragen

`sudo service smbd status`

`sudo service nmbd status`

Dienste laufen, wenn ‚active‘ angezeigt wird.

Samba-Komponenten installieren

`sudo apt-get install samba samba-common`

ggf auch smbclient installieren

Freigaben in die Konfig-Datei eintragen

```
sudo nano /etc/samba/smb.conf
```

```
[SambaTmp]
```

```
comment = Freigabe fuer TMP
```

```
path = /tmp
```

```
read only = no
```

```
[SambaEtc]
```

```
comment = Freigabe fuer etc
```

```
path = /etc
```

```
read only = yes
```

```
[SambaLib]
```

```
comment = Freigabe fuer lib
```

```
path = /lib
```

```
read only = no
```

```
[SambaVar]
```

```
comment = Freigabe fuer var
```

```
path = /var
```

```
read only = yes
```

```
[SambaWd2Tb]
```

```
comment = Freigabe fuer die 2TB NAS-Platte
```

```
path = /mnt/wd2tb
```

```
read only = no
```

```
[SambaHome]
```

```
comment = Freigabe fuer etc
```

```
path = /home
```

```
read only = yes
```

Samba-Dienste neu starten

```
sudo service smbd restart
```

```
sudo service nmbd restart
```


User für Login beim Aufruf der Freigaben unter Windows

<https://www.elektronik-kompendium.de/sites/raspberry-pi/2007071.htm>

Wenn man sich unter Windows mit einer Samba-Freigabe eines Raspi verbinden möchte, wird man aufgefordert, Nutzer und Passwort einzugeben.

Dieser Nutzer muss im Raspi bereits existieren und hier noch einmal mit einem „SMB-Passwort“ registriert werden:

```
sudo smbpasswd -a <user>
```

zweimal das Passwort eingeben

Ggf noch mal die die Dienste neu starten (s.o.)

Und schon funktionieren die Freigaben 😊

MariaDB

Installation Version 10.5

<https://raspberrypi.com/install-mariadb-raspberry-pi/>

```
sudo apt update
```

```
sudo apt upgrade
```

```
sudo apt install mariadb-server
```

Prüfen, wohin installiert wurde:

```
sudo find / -name "*mariadb*"
```

Sicherstellen, dass der Service nach dem Systemstart ausgeführt wird

Fehler:

Leider startet die mariadb nicht immer. Oft kommt dieser Fehler

Can't create test file /mnt/wd2tb/mariadb/datadir/solarraspi.lower-test

Lösungsversuche:

1. Dazu muss der Raspi so konfiguriert werden, dass er beim Booten wartet, bis die Netzwerkverbindung steht. Das geht mit
Sudo raspi-config

1 .. system options

S6 .. network at boot

```
x
x Would you like boot to wait until a network connection x
x is established? x
x x
x <Yes> <No> x
x x
m
```

- ## 2. @reboot-Eintrag in der crontab

Hat nichts gebracht

- ### 3. Abhängigkeit definieren

```
sudo nano /lib/systemd/system/mariadb.service
```

Hat leider nichts gebracht:

[Unit]

Requires=mnt-wd2tb.mount

Aber das: ohne Requires!

```
[Unit]
Description=MariaDB 10.5.19 database server
Documentation=man:mariadb(8)
Documentation=https://mariadb.com/kb/en/library/systemd/
After=network.target mnt-wd2tb.mount
```

Evt. braucht man das noch: **sudo systemctl daemon-reload**

Konfigurieren

`sudo mysql_secure_installation`

Passwort für root vergeben

Weitere Einstellungen:

Switch to unix_socket authentication [Y/n]: n

Remove anonymous users? [Y/n] y

Disallow root login remotely? [Y/n] n

Remove test database and access to it? [Y/n] y

Reload privilege tables now? [Y/n] y

Wo liegen die Konfig-Dateien?

```
admin2@solarrraspi:/ $ cd /etc/mysql
admin2@solarrraspi:/etc/mysql $ ls -l
-bash: ls-: command not found
admin2@solarrraspi:/etc/mysql $ ls -l
total 24
drwxr-xr-x 2 root root 4096 Aug 24 08:41 conf.d
-rw----- 1 root root 544 Aug 24 08:42 debian.cnf
-rwxr-xr-x 1 root root 1731 Feb 10 2023 debian-start
-rw-r--r-- 1 root root 1126 Feb 10 2023 mariadb.cnf
drwxr-xr-x 2 root root 4096 Aug 24 08:42 mariadb.conf.d
lrwxrwxrwx 1 root root 24 Aug 24 08:41 my.cnf -> /etc/alternatives/my.cnf
-rw-r--r-- 1 root root 839 Feb 8 2021 my.cnf.fallback
admin2@solarrraspi:/etc/mysql $
```

Data-Dir verschieben

Leider kann das Datenverzeichnis bei der Installation nicht angegeben werden und liegt nun im eMMC-Speicher des CM4-Moduls und muss noch auf die Sata-Platte verschoben werden.

<https://www.digitalocean.com/community/tutorials/how-to-change-a-mariadb-data-directory-to-a-new-location-on-centos-7>

<https://www.tecmint.com/change-default-mysql-mariadb-data-directory-in-linux/>

Dienst stoppen

`sudo service mysql stop`

Verzeichnis auf der Festplatte anlegen und Rechte vergeben

`mkdir /mnt/wd2tb/mariadb/datadir`

`sudo chmod 777 -R /mnt/wd2tb`

Konfiguration korrigieren

`sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf`

Eintrag ändern: datadir = /mnt/wd2tb/mariadb/datadir

Bestehendes Verzeichnis verschieben

Mit Raspi-Dateiexplorer gescheitert.

```
sudo mv /var/lib/mysql/* /media/wd2tb/mariadb/datadir/ ok
```

Dienst neu starten

```
sudo service mysql start
```

Remotezugriff ermöglichen

```
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

bind-address auf 0.0.0.0 ändern

Dienst neu starten (s.u.)

Mariadb-Service neu starten

```
sudo service mariadb restart
```

Weiteren User anlegen

Nur möglich, wenn man als root angemeldet ist:

```
sudo mysql -uroot -p
```

pwd: hier eingeben

```
CREATE USER 'master'@'%' IDENTIFIED BY 'raspi';
```

Anmelden

```
sudo mysql -uroot -p
```

oder

```
sudo mysql -umaster -p
```

pwd: hier eingeben

Datenbank erstellen

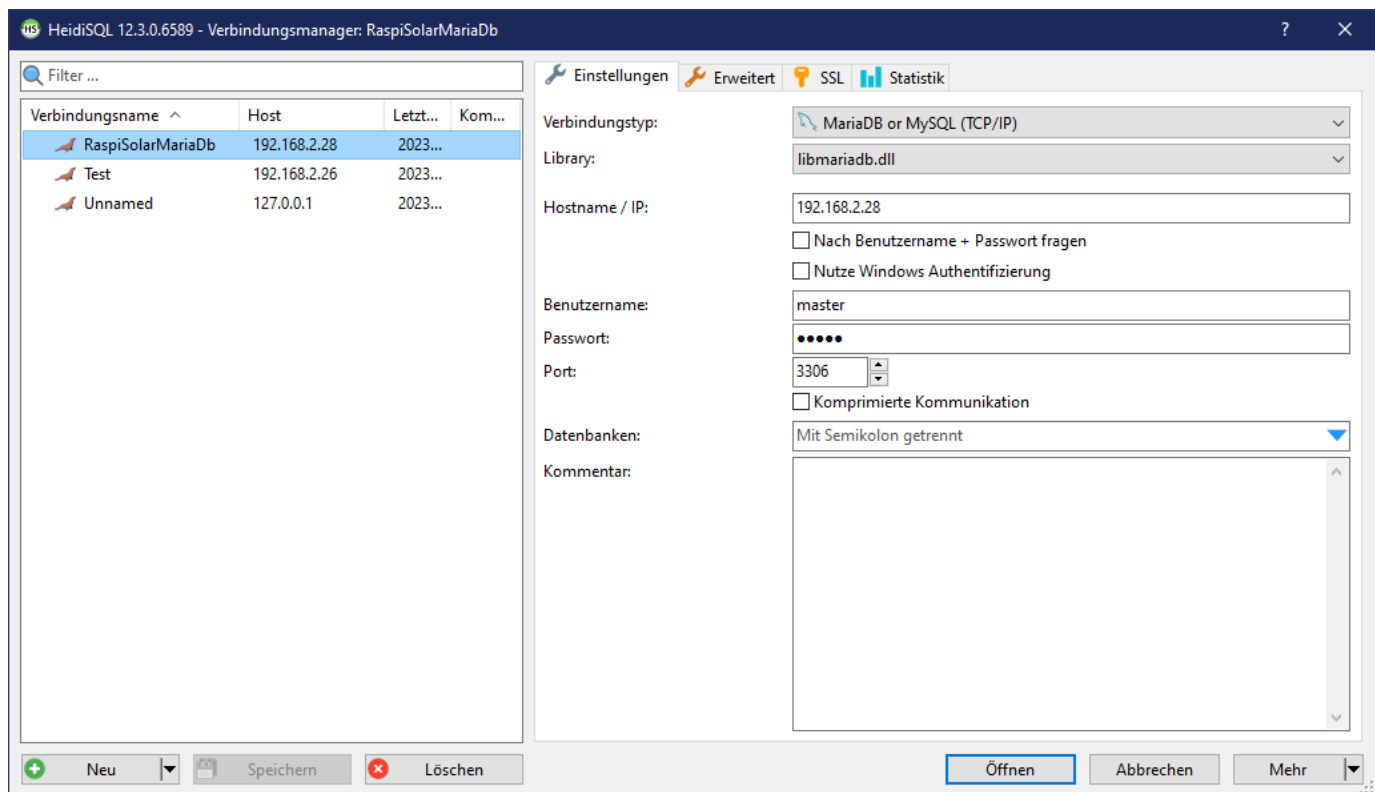
```
create database solar2023;
```

Rechte für Datenbank vergeben

```
GRANT ALL PRIVILEGES ON solar2023.* TO 'master'@'%';
```

```
FLUSH PRIVILEGES;
```

Test des Remote-Zugriffs von HeidiSql auf MariaDb von Windows aus
Ohne Probleme:



Log

Nicht aktiviert, weil es alles sehr langsam machen soll.

Abmelden

quit

Python

Installation

Das RaspberryPI OS (32bit), v11, bringt 3.9. bereits mit:

```
admin2@solarrraspi: ~  
admin2@solarrraspi:~ $ python  
Python 3.9.2 (default, Mar 12 2021, 04:06:34)  
[GCC 10.2.1 20210110] on linux
```

Python-Connector für die mariadb

Für

```
import mariadb
```

im python-Script wird der Connector gebraucht. Installation mit

```
pip install mariadb
```

Dabei wird automatisch versucht, bei 1.1.7 beginnend, eine funktionierende Version zu finden:

Rückwärts bis 1.1.2 brach die Installation mit Fehlern ab.

Aber 1.0.11 wurde dann installiert:

```
Downloading https://www.piwheels.org/simple/mariadb/mariadb-1.0.11-cp39-cp39-linux_armv7l.whl (198 kB 1.3 MB/s)  
Installing collected packages: mariadb  
Successfully installed mariadb-1.0.11
```

Script timergesteuert ausführen

Vorsicht! Crontab -r löscht alle Einträge aus der crontab!

Für Ausführung als root eintragen mit sudo crontab -e.

Für Ausführung als angemeldeter Nutzer: crontab -e

Beispiel: zwei Zeilen eingetragen, um ein shellscript zu starten:

```
# von 8-20 Uhr in Minute 50 das shell script ausfuehren
```

```
50 8-20 * * * sh /mnt/wd2tb/releases/meteoblue_forecast/mb_pvpro.sh
```

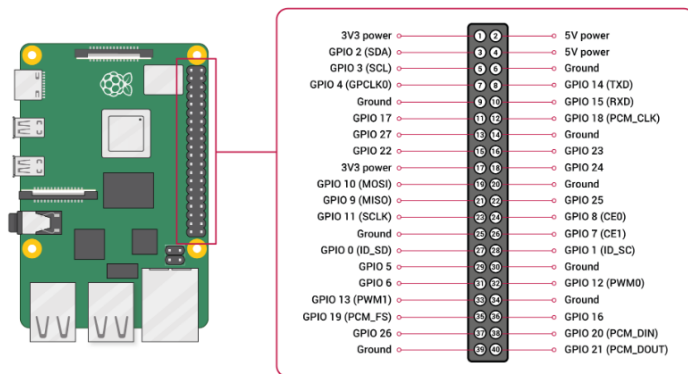
Beispiel für ein shell-Script:

```
cd /mnt/wd2tb/script/meteoblue_forecast
```

```
# Solarprognose von MeteoBlue holen und speichern
```

```
python mb_pvpro.py
```

GPIO-Aktoren und -Sensoren



Aktoren: 3-Kanal-Relais-Board

https://www.waveshare.com/wiki/RPi_Relay_Board



Channel No.	RPi Pin No.	wiringPi	BCM	Descriptions
CH1	37	P25	26	Channel 1
CH2	38	P28	20	Channel 2
CH3	40	P29	21	Channel 3

Testscripts zum Ansteuern der Relais siehe

https://github.com/grasmax/AcOnOff/blob/main/script/gh_gpiorelaytestXX.py

https://github.com/grasmax/AcOnOff/blob/main/script/gh_gpiorelaytest.py

Ab September 2023 wirts BCM-20/Relais 2 benutzt.

Sensoren

<https://www.elektronik-kompendium.de/sites/raspberry-pi/2006051.htm>

Wichtig: Die als Sensor geschalteten Pins müssen mit einem Pulldown-Widerstand versehen werden.

Dieser sollte 10-100 kΩ haben.

Alternativ kann der interne Pulldown aktiviert werden:

`GPIO.setup(p, GPIO.IN, GPIO.PUD_DOWN)` # den internen pulldown-Widerstand aktivieren

Ab September 2023 werden die internen Pulldowns benutzt.

Die Kabelfarben sind wie folgt:

Farbe	BCM-Pin	Überwachung von
Orange	3,3 V DC vom 3,3V-Pin links oben	
Gelb	17	
Grün	27	
Blau	22	Relais 2

Testscript

https://github.com/grasmax/AcOnOff/blob/main/script/gh_gpiointest.py

Übergang von der virtuellen in die reale Welt

Linkes Bild: Die GPIO-Pins sind mit Verteilerklemmen verbunden:

Die Klemmen 1-3 (v.l.) verbinden die IN-Pins mit den KM12-Modulen, die den Schaltzustand der Stromstoßschalter liefern.

Die Klemmen 4-6 (v.l.) verbinden die Relais des Relaisboards mit den Stromstoßschaltern.

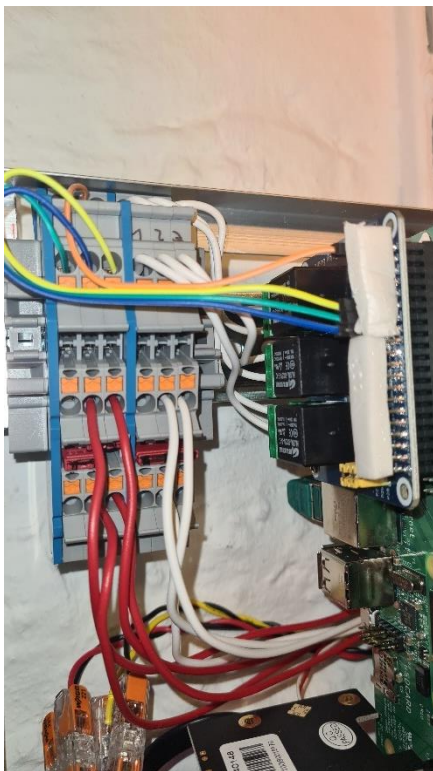
Rechtes Bild:

Die Klemmen 1-3 (v.l.) verteilen 5 VDC auf Sata-Platte und LAN-Switch.

Die Klemmen 4-6 (v.l.) verteilen 12 VDC auf Sata-Platte, Raspi-IOBoard und Stromstoßschalter.

KM12/Stromstsch links: September 2023: defekt

KM12/Stromstsch rechts: schaltet ab September 2023 48 VDC für den Schütz, der 230VAC für den MPII einschaltet.



Dienste und Startreihenfolge

5 Arten

<https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/>

1. `sudo nano /etc/rc.local`
2. `sudo nano /home/pi/.bashrc`
3. `cd /etc/init.d`
4. `sudo nano /lib/systemd/system/sample.service`

The permission on the unit file needs to be set to 644 :

```
sudo chmod 644 /lib/systemd/system/sample.service
```

```
[Unit]
```

```
Description=My Sample Service
```

```
After=multi-user.target
```

```
[Service]
```

```
Type=idle
```

```
ExecStart=/usr/bin/python /home/pi/sample.py
```

```
[Install]
```

```
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable sample.service
```

```
sudo reboot
```

5. crontab

<https://www.dexterindustries.com/howto/auto-run-python-programs-on-the-raspberry-pi/>

<https://www.linux-community.de/ausgaben/linuxuser/2018/07/handarbeit-2/>

Typ	Funktion
.service	Dienste starten, überwachen und stoppen
.device	Gerätedateien anlegen
.mount	Ein- und Aushängen von Mountpoints
.automount	automatisches Ein- und Aushängen von Mountpoints
.target	Gruppe von Units definieren
.timer	wiederkehrende Aufgaben definieren (ähnlich Cron)
.socket	Verbindungen zwischen Prozessen herstellen
.network	Netzwerke konfigurieren
.path	Service-Units abhängig von Änderungen ausführen

```
ls /lib/systemd/system/
```

[illegible]

Als letzten Schlüssel sehen Sie `Wants`, mit dem Sie optionale Abhängigkeiten bezeichnen. Eine harte Abhängigkeit dagegen kennzeichnen Sie mit `Require`. Startet der dort eingetragene Dienst nicht, versagt der Dienst, zu dem diese Unit gehört, seine Tätigkeit. Bei `Wants` startet er trotzdem. Wenn Sie sich nun fragen, wozu das zusätzliche `After` gut ist: Dessen Fehlen würde bedeuten, dass beide Units parallel starten, was in unserem Fall nicht sinnvoll wäre.

Besonderheit bei mariadb

Hier hatte Require=mnt... nichts gebracht.

Erst als der Mountpunkt für die Sata-Platte bei After eingetragen wurde, funktionierte der Start der mariadb stabil. Bei reboot und Neustart

Alle Unit-Dateien auflisten

`systemctl list-unit-files`

u.a.

<code>mnt-wd2tb.mount</code>	<code>generated</code>	<code>-</code>
<code>mariadb.service</code>	<code>enabled</code>	<code>enabled</code>
<code>mariadb@.service</code>	<code>disabled</code>	<code>enabled</code>
<code>mysql.service</code>	<code>alias</code>	<code>-</code>
<code>mysqld.service</code>	<code>alias</code>	<code>-</code>

Alle startup-Zeiten aller Units auflisten

`systemd-analyze blame`

sagt leider nichts über die boot-Reihenfolge

Log-Dateien

Stehen in `/var/log`.

Habe dafür eine Samba-Freigabe erstellt und alle Dateien lesbar gemacht:

```
sudo chmod 777 /var/log/ -R
```

Komischerweise taucht mariadb nicht mehr im boot.log auf, seit ich die Abhängigkeit vom Mountpunkt in After= nachgetragen habe.

Aber in daemon.log finden sich alle Ausschriften.

Aber leider kann so nicht mehr festgestellt werden, wo sich die mariadb in der boot-Reihenfolge einordnet.

Montage auf der Hutschiene

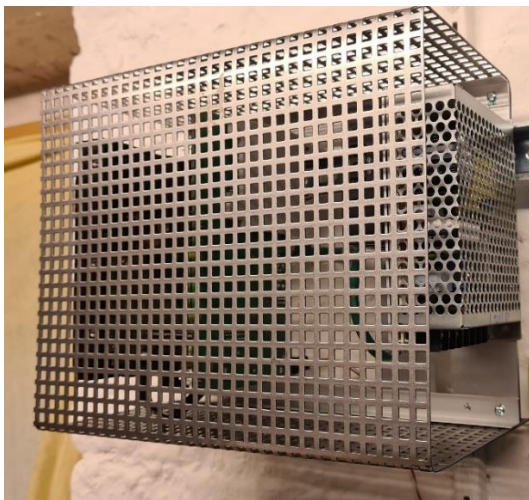
ALU-Flachprofile zugeschnitten, gebohrt und M3-Gewinde geschnitten.

Die ALU-Schreife mit den 45mm-Clips und den zu montierenden Geäten verschraubt.

<https://www.netzgeraet.de/hutschienenetzteile/tragschienen/5354/din-schienen-hutschienenclip-45mm-clip.html>

Einzel in die Hutschiene eingehängt und mit Winkelprofilen hinten oben und unten verschraubt.

Stahlblech, gelocht, 500x250, abgewinkelt und mit vier Schrauben an den Winkelprofilen befestigt.



Weitere nützliche Befehle

Anzeige von Prozessen, z.B. ssh

```
ps aux | grep ssh
```

```
whereis ssh-add
```

Ausschalten

```
systemctl poweroff -i
```

```
sudo shutdown -h now
```

Neustart

```
sudo shutdown -r now
```

```
sudo reboot
```

Temperatur

```
vcgencmd get_throttled
```

```
vcgencmd measure_temp
```