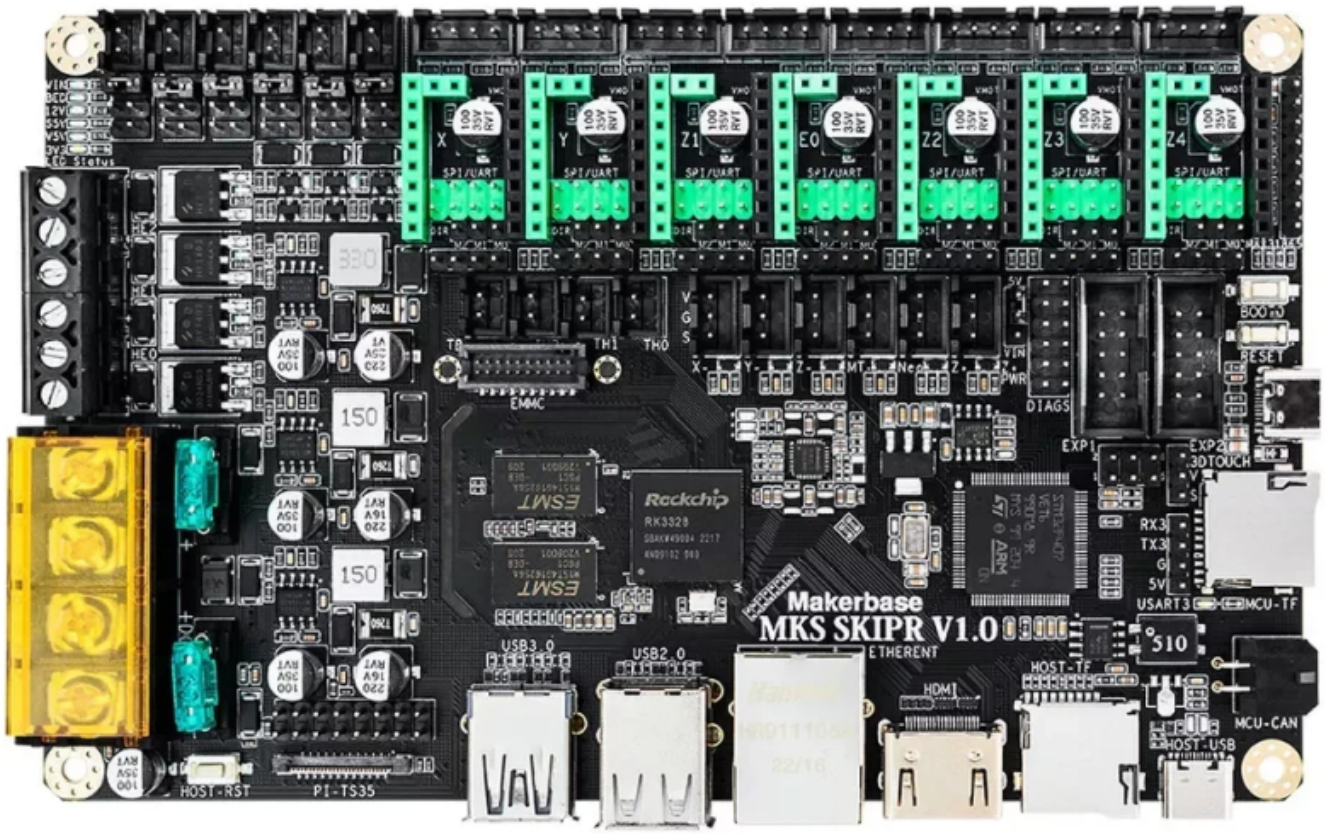
MKS SKIPR



YouTube Video #111

Hinweise

- **SBC** bedeutet in der Anleitung **S**ingle **B**oard **C**omputer. Im Falle des SKIPR ist das ein onboard 4-kern RK3328 mit 1GB Ram.
- Es wird davon ausgegangen, dass auf dem SBC Klipper und MainSail eingerichtet wird.
- Ein Zugang zum SBC über SSH ist notwendig!

Stromversorgung

- Im Betrieb wird das Board mit 12V oder 24V versorgt (Anschluss POWER + -)
- Das Board lässt sich auch mit der normalen Stromversorgung (über den POWER Anschluss) flashen.

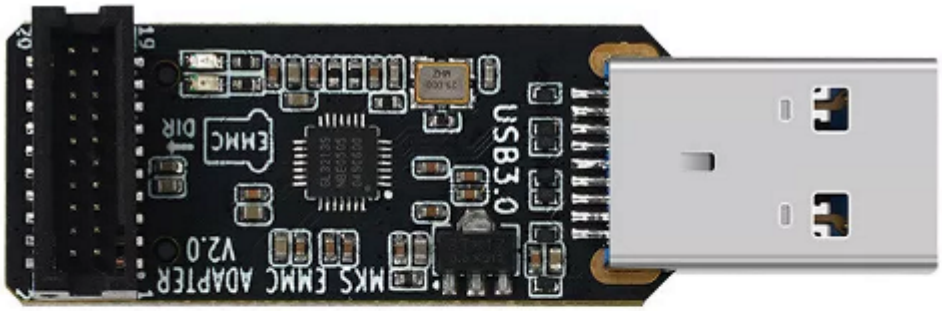
Verkabelung

- Stromversorgung über 24V / GND
- CAN Anschluss über die schwarze (Molex?) Buchse rechts unten (zwischen den USB-C Ports)
    - **ACHTUNG** CAN funktioniert nur mit extra USB-Kabel!

OS Setup

Image schreiben

- https://github.com/redrathnure/armbian-mkspi/releases
    - Armbian-unofficial_24.2.0-trunk_Mkspi_bookworm_current_6.6.17.img.xz
- Am einfachsten mit Raspberry Pi Imager (https://www.raspberrypi.com/software/)
- Alternativ mit Rufus (https://rufus.ie/de/)
- Für das Schreiben auf den eMMC braucht man einen Adapter



Dabei wird das gleiche Image und das gleiche Tool verwendet!

LAN Konfig

- Der LAN Anschluss steht auf DHCP und sollte sofort funktionieren.
- Wird WLAN Aktiviert über die Template Datei ist der LAN Anschluss erstmal deaktiviert!

WLAN Konfig

- WLAN Adapter TL-WN725N (https://www.tp-link.com/de/home-networking/adapter/tl-wn725n/)
- `armbian_first_run.txt.template` umbenennen in `armbian_first_run.txt`

```
#Networking:
FR_net_change_defaults=1

# Enable WiFi or Ethernet.
#   NB: If both are enabled, WiFi will take priority and Ethernet will
be disabled.
FR_net_ethernet_enabled=1
FR_net_wifi_enabled=1

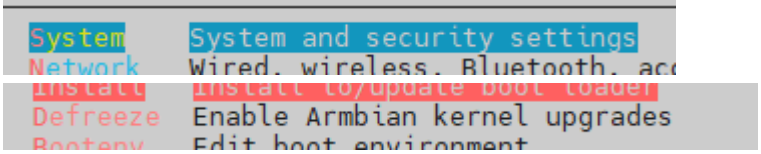
#Enter your WiFi creds
#   SECURITY WARN: Your wifi keys will be stored in plaintext, no
encryption.
FR_net_wifi_ssid='SSID'
FR_net_wifi_key='PASS'

#   Country code to enable power ratings and channels for your country.
eg: GB US DE | https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2
FR_net_wifi_countrycode='DE'
```

erster Start & Login

- Nach dem ersten Booten gibt es nur den User **root** mit Passwort **1234**.
- Es wird dann ein neuer User angelegt, den man danach auch als Standard verwenden sollte!
- Als Sprache am besten en/us festlegen.
- Wenn das initial Setup durch ist kann man sich über `su - <USER>` mit dem neuen User anmelden.

Kernel Freeze

- `sudo armbian-config`
- 

Updates & Tools

- `sudo apt update && sudo apt upgrade -y && sudo apt install -y git silversearcher-ag wavemon hexedit sudoku tcpdump iptraf mc htop dcfldd`

```
nano usbutils ranger tldr ncdu can-utils multitail fd-find lsof x11vnc terminator minicom cutecom joystick jstest-gtk i2c-tools speedtest-cli iotop && mkdir -p ~/.local/share && tldr -u
```

- `sudo apt autoremove -y modem* cups* pulse* avahi* triggerhappy*`
- Fix für Python3
  `sudo apt install python3-pip python3-serial`

dfu-util Fix

- Als root anmelden
  `su -`
- `cd /etc/udev/rules.d`
- udev Regel laden und passend ablegen
  `wget https://raw.githubusercontent.com/wiieva/dfu-util/refs/heads/master/doc/40-dfuse.rules -O 40-dfuse.rules`
- **Als root abmelden**
  `exit`
- `sudo usermod -aG plugdev $USER`
- Einmal neu starten damit alles aktiv wird
  `sudo reboot`

Link : https://github.com/wiieva/dfu-util/blob/master/doc/40-dfuse.rules

Klipper & Co mit kiauh

- `cd ~ && git clone https://github.com/th33xitus/kiauh.git && ./kiauh/kiauh.sh`
- 1 → Install
  - 1 → Klipper
    - Number of Klipper instances to set up (default=1): 1
    - Create example printer.cfg? (Y/n): Y
  - 2 → Moonraker
    - Create example moonraker.conf? (Y/n): Y
  - 3 → Mainsail
    - Download the recommended Mainsail-Config? (Y/n): Y

Linux MCU

- `cd ~/klipper/`
- `sudo cp ./scripts/klipper-mcu.service /etc/systemd/system/`
- `sudo systemctl enable klipper-mcu.service`
- `make menuconfig`

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (Linux process)  --->
```

```
()  GPIO pins to set at micro-controller startup
```

- `sudo usermod -a -G tty $USER`
- `sudo service klipper stop`
- `make flash -j4`
- `sudo systemctl start klipper-mcu`

Klipper für die MCU

> **Hinweis:**
> Für beide Fälle wird ein USB-C Kabel gebraucht!

DFU-Modus

- Das USB Kabel anschließen und zwar an den rechten USB-C Port unter den beiden Tastern.
- `dmesg -Hw`
- Die Boot 0 Taste (obere) gedrückt halten, die Reset-Taste (untere) kurz drücken und dann die Boot 0 Taste loslassen.
- Jetzt sollte im Terminal folgendes erscheinen :

```
[Sep29 08:31] usb 5-1: USB disconnect, device number 2
[  +0.704472] usb 5-1: new full-speed USB device number 3 using ohci-
platform
[  +0.229032] usb 5-1: New USB device found, idVendor=0483,
idProduct=df11, bcdDevice=22.00
[  +0.000019] usb 5-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[  +0.000008] usb 5-1: Product: STM32  BOOTLOADER
[  +0.000007] usb 5-1: Manufacturer: STMicroelectronics
[  +0.000006] usb 5-1: SerialNumber: 398C33723232
```

  Wichtig ist hier **STM32 BOOTLOADER**

- STRG + C um dmesg zu beenden

UART Variante

- Für die UART Variante braucht es keinen Bootloader!
- `cd ~/klipper && make menuconfig`

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F407)  --->
    Bootloader offset (No bootloader)  --->
    Clock Reference (8 MHz crystal)  --->
    Communication interface (Serial (on USART1 PA10/PA9))  --->
(250000) Baud rate for serial port
()  GPIO pins to set at micro-controller startup (NEW)
```

- `make -j4`
- `dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D ~/klipper/out/klipper.bin`

CAN Variante

- Für CAN muss das Board mit einem extra USB-C-Kabel erweitert werden! Und zwar der USB-C Port rechts bei den beiden Tastern an einen USB-Port neben dem Netzwerkstecker. **Ohne dieses extra Kabel kriegt man CAN nicht zum laufen!** Der Controller ist leider nur mit der UART an den SBC verbunden. Und über UART bekommt man kein CAN Interface abgebildet.

Katapult Bootloader

- Damit man später das Board noch updaten kann, ist es ratsam den Bootloader Katapult zu installieren.
- Katapult muss auch nur 1x installiert werden. Bei einem Update ist das nicht mehr nötig.
- Katapult wird **über USB** (DFU-Mode) eingerichtet!
- Katapult laden wenn noch nicht vorhanden, sonst in den Katapult Ordner wechseln
  `[ ! -d "$HOME/katapult/" ] && cd ~ && git clone https://github.com/Arksine/katapult && cd katapult || cd ~/katapult`
- `make menuconfig`
  - ```
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F407)  --->
    Build Katapult deployment application (Do not build)  --->
    Clock Reference (8 MHz crystal)  --->
    Communication interface (USB (on PA11/PA12))  --->
    Application start offset (32KiB offset)  --->
    USB ids  --->
() GPIO pins to set on bootloader entry
[*] Support bootloader entry on rapid double click of reset button
[ ] Enable bootloader entry on button (or gpio) state
[ ] Enable Status LED
    ```
  - **Wichtig:** Hier wird als Communication Interface USB ausgewählt, nicht CAN!
  - Sonst ist später kein Update möglich!
- Katapult kompilieren
  `make -j4`
- Katapult flashen (das Board muss im DFU Mode sein !)
  `dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D ~/katapult/out/katapult.bin`
  - Wichtig ist am Ende `File downloaded successfully` bei der Ausgabe im Terminal
- Das Board einmal resetten
  - Reset Taste (Reset) drücken
  - oder das Board einmal stromlos machen
    ```

Port finden

- `dmesg -Hw` aufrufen
- Reset Taste am Board drücken

```
[  +0.733491] usb 5-1: new full-speed USB device number 4 using ohci-
platform
[  +0.235102] usb 5-1: New USB device found, idVendor=1d50,
idProduct=6177, bcdDevice= 1.00
[  +0.000047] usb 5-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[  +0.000026] usb 5-1: Product: stm32f407xx
[  +0.000020] usb 5-1: Manufacturer: katapult
[  +0.000020] usb 5-1: SerialNumber: 3C0055000E51323236333739
[  +0.004590] cdc_acm 5-1:1.0: ttyACM0: USB ACM device
```

  - Wir brauchen die Information mit **tty...** also in diesem Fall **ttyACM0**
  - STRG+C drücken, um dmesg zu beenden

Klipper mit USB/CAN Bridge

- `cd ~/klipper`
- `make menuconfig`

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F407)  --->
    Bootloader offset (32KiB bootloader)  --->
    Clock Reference (8 MHz crystal)  --->
    Communication interface (USB to CAN bus bridge (USB on
PA11/PA12))  --->
    CAN bus interface (CAN bus (on PB12/PB13))  --->
    USB ids  --->
(1000000) CAN bus speed
()  GPIO pins to set at micro-controller startup
```

- Klipper kompilieren und flashen (über USB / seriell!)
`make -j4 flash FLASH_DEVICE=/dev/ttyACM0`

```
pi@mkspi:~/klipper $ make -j4 flash FLASH_DEVICE=/dev/ttyACM0
  Creating symbolic link out/board
  Building out/autoconf.h
  Compiling out/src/sched.o
...
  Compiling out/src/stm32/hard_pwm.o
  Preprocessing out/src/generic/armcm_link.ld
  Building out/compile_time_request.o
Version: v0.12.0-102-g9f41f53c
  Linking out/klipper.elf
  Creating hex file out/klipper.bin
```

```
  Flashing out/klipper.bin to /dev/ttyACM0
Entering bootloader on /dev/ttyACM0
Device reconnect on
/sys/devices/platform/soc/5200000.usb/usb2/2-1/2-1.4/2-1.4:1.0
/usr/bin/python3 lib/canboot/flash_can.py -d /dev/serial/by-
path/platform-5200000.usb-usb-0:1.4:1.0 -f out/klipper.bin

Attempting to connect to bootloader
CanBoot Connected
Protocol Version: 1.0.0
Block Size: 64 bytes
Application Start: 0x8020000
MCU type: stm32h723xx
Flashing '/home/biqu/klipper/out/klipper.bin'...

[##################################################]

Write complete: 1 pages
Verifying (block count = 477)...

[##################################################]

Verification Complete: SHA = 381BC7BAE3D7B8717F7169CEDB8EA08E4D59A4CA
CAN Flash Success
```

SBC einrichten

- Interface einrichten
  Achtung : die Bitrate von 1000000 muss auch in der Board Firmware eingestellt werden!
- `sudo nano /etc/network/interfaces.d/can0`

```
 allow-hotplug can0
 iface can0 can static
    bitrate 1000000                                    # Ensure it's
the same as selected for MCU firmware.
    up ip link set $IFACE txqueuelen 128               # ... please
use this version instead.
    #up ip link set $IFACE txqueuelen 1024 restart-ms 200 # Bit more
aggressive configuration
```

- Editor mit STRG + X → Y → Enter beenden
- Testen mit `ip a`
  `can0: <NOARP,UP,LOWER_UP,ECHO> mtu 16 qdisc pfifo_fast state **UP** group default qlen 1024`
- Sollte das Interface auf DOWN stehen hilft meist ein
  `sudo systemctl restart networking.service`
  oder ein
  `sudo ip link set can0 up type can bitrate 1000000`

oder ein
`sudo reboot`

Can Query

> **Hinweis**
> Die folgenden Schritte setzen natürlich voraus, das der CAN Bus korrekt im Vorfeld eingerichtet wurde!

Wenn das Board über CAN verbunden ist, dann kann man mit den folgenden Schritten prüfen, ob Katapult geflasht wurde:

- Klipper Dienst stoppen
  `sudo systemctl stop klipper.service`
- `~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0`
  Wenn ein Board gefunden wird, dann sollte folgende Ausgabe erscheinen:

```
pi@mkspi:~$ ~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py
can0
Found canbus_uuid=baf76f0ff03d, Application: Klipper
Total 1 uuids found
```

  - Die **UUID** (canbus_uuid=**baf76f0ff03d**) notieren !
  - Wird bei diesem Schritt kein Board gefunden, hilft oft ein Reset am Board (entweder über den Reset Taster oder 1x Strom weg und wieder dran)

kurzer Test

Ob das Board korrekt mit Klipper läuft, lässt sich mit folgendem Befehl schnell testen:
`~/klippy-env/bin/python ~/klipper/klippy/console.py -c can0 baf76f0ff03d`

Der Pfad am Ende muss natürlich mit dem übereinstimmen, was ihr im vorherigen Schritt ermittelt habt!

Wenn ihr ein **connected** am Anfang des Textes seht, ist das Board richtig geflasht.

```
=================== attempting to connect ====================
INFO:root:Starting CAN connect
INFO:can.interfaces.socketcan.socketcan:Created a socket
Loaded 114 commands (v0.12.0-61-gb50d6669 / gcc: (15:8-2019-q3-1+b1) 8.3
MCU config: ADC_MAX=4095 BUS_PINS_i2c1_PA9_PA10=PA9,PA10 BUS_PINS_i2c1_P
14 BUS_PINS_i2c3_PB3_PB4=PB3,PB4 BUS_PINS_spi1=PA6,PA7,PA5 BUS_PINS_spi1
NCY=1000000 CLOCK_FREQ=64000000 MCU=stm32g0b1xx PWM_MAX=255 RECEIVE_WIND
WARNING:root:got {'oid': 6, 'next_clock': 515819151, 'value': 31272, '#n
====================       connected       ====================
001.393: analog_in_state oid=6 next_clock=535019151 value=31275
```

Konfiguration

- `cd ~/printer_data/config`

- **Beispiel Konfiguration MKS SKIPR**
  `wget https://raw.githubusercontent.com/makerbase-mks/MKS-SKIPR/refs/heads/main/klipper%20firmware/skipr%20to%20voron%20V2.4.cfg -O printer.cfg`
- `nano ~/printer_data/config/printer.cfg`
  - **UART Konfig**

    ```
    [mcu]
    # The hardware use USART1 PA10/PA9 connect to RK3328
    serial: /dev/ttyS0
    restart_method: command
    ```

  - **CAN Konfig**

    ```
    [mcu]
    canbus_uuid: baf76f0ff03d
    ```

  - Hier muss die CAN UUID eingetragen werdem die vorher ermittelt wurde!
- Den Editor mit STRG + X → Y → Enter verlassen
- Klipper starten
  `sudo systemctl start klipper.service`

Klipper Update

UART

- Das Update wird per USB-Kabel eingespielt und es muss vorher der DFU Modus aktiviert werden!
- Ein Update im UART Modus ist letztlich ein neues Flashen von Klipper. Also genauso wie hier beschrieben.

CAN

> **Hinweis:**
> Das Klipper Update wird über USB eingespielt! Über den CAN-Bus ist ein Update nicht möglich wenn das Board als USB/Can Bridge arbeitet.

- Klipper Dienst stoppen
  `sudo systemctl stop klipper.service`
- Alle CAN UUID's ermitteln
  `grep canbus_uuid ~/printer_data/config/* -n`

  ```
  pi@mkspi:~$ grep canbus_uuid ~/printer_data/config/* -n
  /home/biqu/printer_data/config/BTT_EBB.cfg:10:canbus_uuid: 44d860c9632b
  /home/biqu/printer_data/config/printer.cfg:162:canbus_uuid:
  fa5ad324b369
  ```

- Das Skipr Board per flshtool.py resetten. Welche UUID das Skipr Board hat kann man bei

mehreren Busteilnehmern leider nicht ohne weitere erkennen.
`~/klippy-env/bin/python ~/katapult/scripts/flashtool.py -i can0 -u <Skipr UUID> -r`

```
pi@mkspi:~$ ~/klippy-env/bin/python ~/katapult/scripts/flashtool.py -i
can0 -u fa5ad324b369 -r
Sending bootloader jump command...
Bootloader request command sent
Flash Success
```

    - Die Status LED sollte jetzt anfangen zu blinken
- Den Port ermitteln
  `dmesg |tail -n 10`

```
pi@mkspi:~$ dmesg |tail -n 10
[  296.579125] gs_usb 2-1.4:1.0 can0: Couldn`t shutdown device
(err=-19)
[  296.959524] usb 2-1.4: new full-speed USB device number 5 using
ehci-platform
[  297.190002] usb 2-1.4: New USB device found, idVendor=1d50,
idProduct=6177, bcdDevice= 1.00
[  297.190028] usb 2-1.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[  297.190034] usb 2-1.4: Product: stm32h723xx
[  297.190040] usb 2-1.4: Manufacturer: katapult
[  297.190045] usb 2-1.4: SerialNumber: 1E004300105131323634343O
[  297.208551] cdc_acm 2-1.4:1.0: ttyACM0: USB ACM device
[  297.208666] usbcore: registered new interface driver cdc_acm
[  297.208673] cdc_acm: USB Abstract Control Model driver for USB
modems and ISDN adapters
```

Wie immer brauchen wir die tty... Angabe. In diesem Fall ist is **ttyACM0** wie man in der drittletzten Zeile sehen kann.

- `cd ~/klipper`
- `make menuconfig`
  → Die Einstellungen sind dieselben wie oben unter CAN angegeben.
- Klipper flashen
  `make -j4 flash FLASH_DEVICE=/dev/ttyACM0`
  Den ermittelten Port halt am Ende ggf. anpassen.
- Klipper starten
  `sudo systemctl start klipper.service`

Sonstiges

Diese Punkte sind nicht immer Bestandteil vom YouTube Video, aber nützlich 😎

STM32 Temperatur

Der interne Temperatur Sensor des STM32 kann mit folgendem Konfig Schnipsel ausgelesen werden:

```
[temperature_sensor Skipr]
sensor_type              : temperature_mcu
sensor_mcu               : mcu
```

ADXL345 (Input Shaper)

Eine Beschreibung gibt es hier:
https://github.com/redrathnure/armbian-mkspi?tab=readme-ov-file#adxl345spi-usage

Links

- https://github.com/makerbase-mks/MKS-SKIPR
- https://github.com/redrathnure/armbian-mkspi
- Original Images
  https://drive.google.com/drive/folders/1tTuSvF9OL2qtPXElau8YOXn2sWbdxa9e
- https://www.kis3d.de/MKS-Makerbase.html