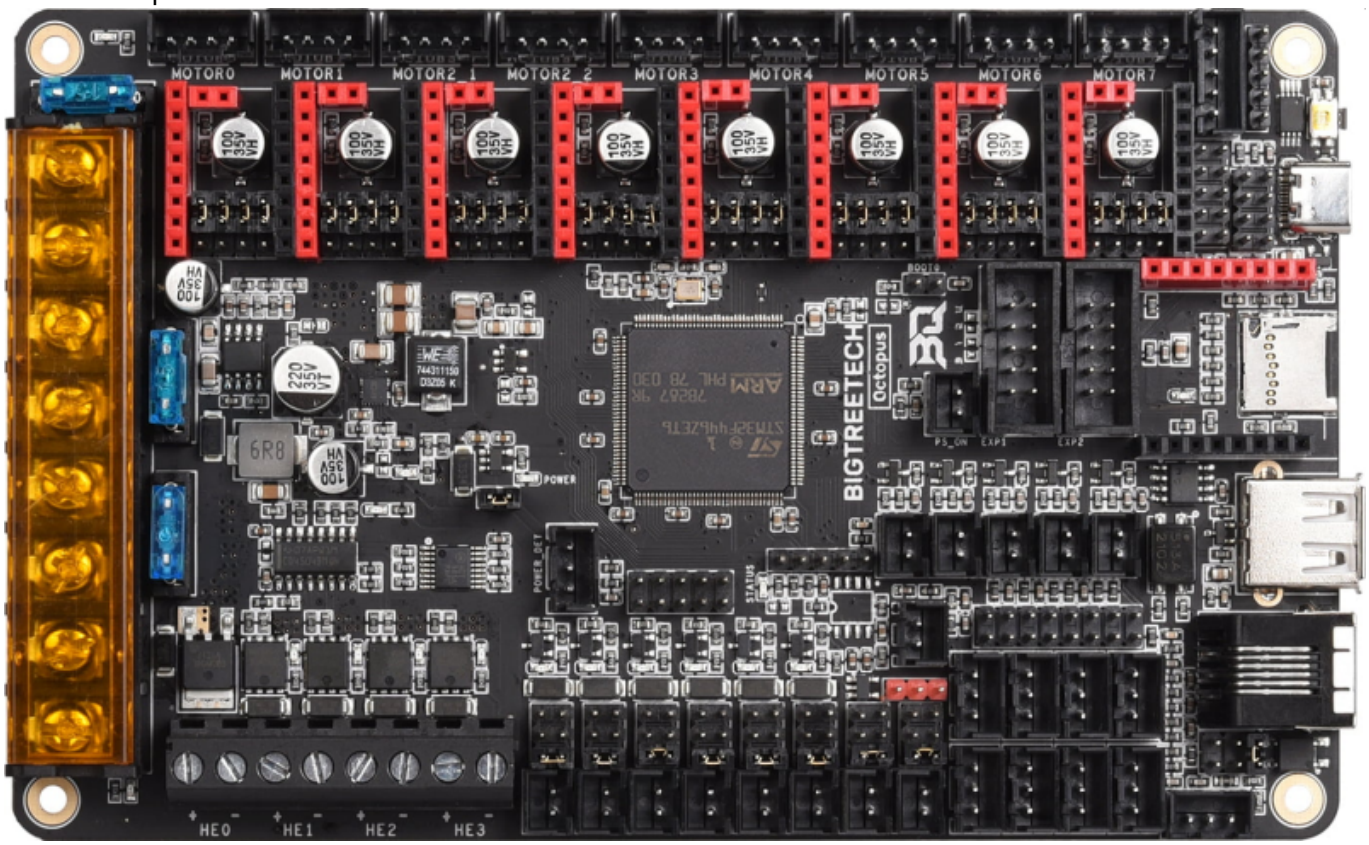
BTT Octopus [Pro] (CAN-Bridge)

Schrittweise Anleitung, um das BTT **Octopus [Pro]** Board über die **USB/CAN Bridge** in Betrieb zu nehmen.

**BTT Octopus Board **



YouTube Video #148



Hinweise

- **SBC** bedeutet in der Anleitung **S**ingle **B**oard **C**omputer. Also meistens wohl ein Raspberry Pi.
- Es wird davon ausgegangen das auf dem SBC Klipper und MainSail eingerichtet ist.
- Ein Zugang zum SBC über SSH ist notwendig!

- Wenn `dmesg -HW` einen Fehler bringt, einfach `dmesg -Hw` verwenden.
- Der SD-Slot ist bei diesem Controller komplett überflüssig 😎

Boards

Die Boards gibt es generell mit verschiedenen Controllern. Beim Flashen also besonders darauf achten!

- STM32F446ZE + 12 MHz crystal
- STM32F429ZG + 8 MHz crystal
- STM32F407ZE
- STM32FH723ZE + 25Mhz crystal

Octopus 1.0

- erste Version

Octopus 1.1

- Korrekturen in der bedruckten Board Bescriftung (fan, spi3, uart2)
- Geändertes IC für die Spannungsversorgung

Octopus Pro
(https://www.reddit.com/r/BIGTREETECH/comments/qyx2uq/had_trouble_finding_the_differences_between/)

- Je nach Controller 1MB Flash
- bis zu 8 HV Schrittmotor Treiber
- Extra Beeper
- 12-60V Motor Input (anstelle 12-35V)
- Schrittmotor Input Level 5V (anstatt 3,3V)
- Onboard MAX31865 Chip
- Beim Pro 1.1 wurden einige Pins geändert (Siehe https://github.com/bigtreetech/BIGTREETECH-OCTOPUS-Pro/blob/master/README.md)

Verisonen 1.1

- Haben einen Fix dass das Hotend nicht bei einem Flashvorgang sofort voll aufheizt.

Board Unterschiede Normal / PRO

- Andere Position für den Raspi Anschluss
- geändertes Power Detect am PRO
- Spannungsauswahl an den Treibern
- ggf. anderer CAN Connector (kein RJ11 sondern XH2.54*2Pin)

Verkabelung

Stromversorgung

- Der Jumoer ist entweder mit POWER oder auf der Rückseite mit V_BUS / 5V gekennzeichnet.
- ! Bei Versorgung über den POWER Anschluss (also mit >= 12V am Eingang) muss der Jumper gezogen werden !
- **Firmware flashen**
  - Das Board wird **OHNE Versorgungsspannung betrieben (nur USB)**.
    Den Jumper auf Position POWER (oder V_BUS / 5V) setzen.
  - Das Board wird **mit >= 12V betrieben**.
    Den Jumper entfernen!

CAN Bus Anschluss

- Wer den CAN Bus überprüfen will, kann im **ausgeschalteten Zustand** den Buswiderstand mit einem Ohmmeter messen. Es müsste zwischen CAN H und CAN L ca. 120Ω ergeben wenn das Board nichts angeschlossen hat am CAN Interface. Mit angeschlossener CAN Hardware müssen es ca 60Ω sein.

Normal

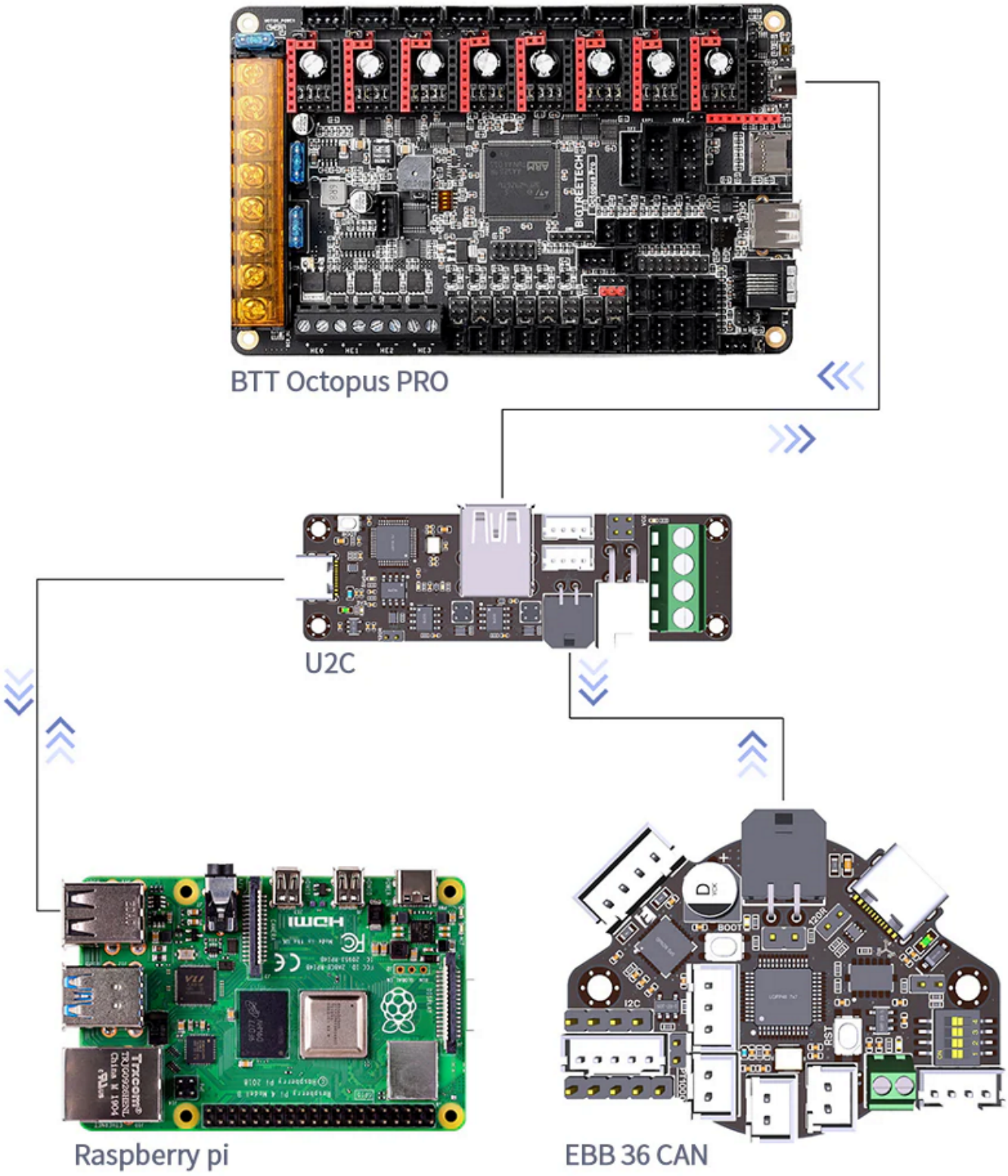
- Die normalen Board verwendewn einen RJ11 Anschluss für das CAN Interface
- die 120Ω sind immer vorhanden und können nicht per Jumper weggenommen werden

PRO

- Das CAN Interface ist ein XH2.54*2Pin Anschluss
- Die 120Ω können per Jumper selektiert werden

CAN über USB

Es gibt diese wüste Verkabelung über USB bei BTT. Ich halte da ehrlich gesagt nichts von. Technisch mag das funktionieren, aber verwirrender kann ein Aufbau für den Endanwender kaum sein.
Wenn ich die Wahl hätte würde ich immer eine direkte CAN Verkabelung vorziehen!



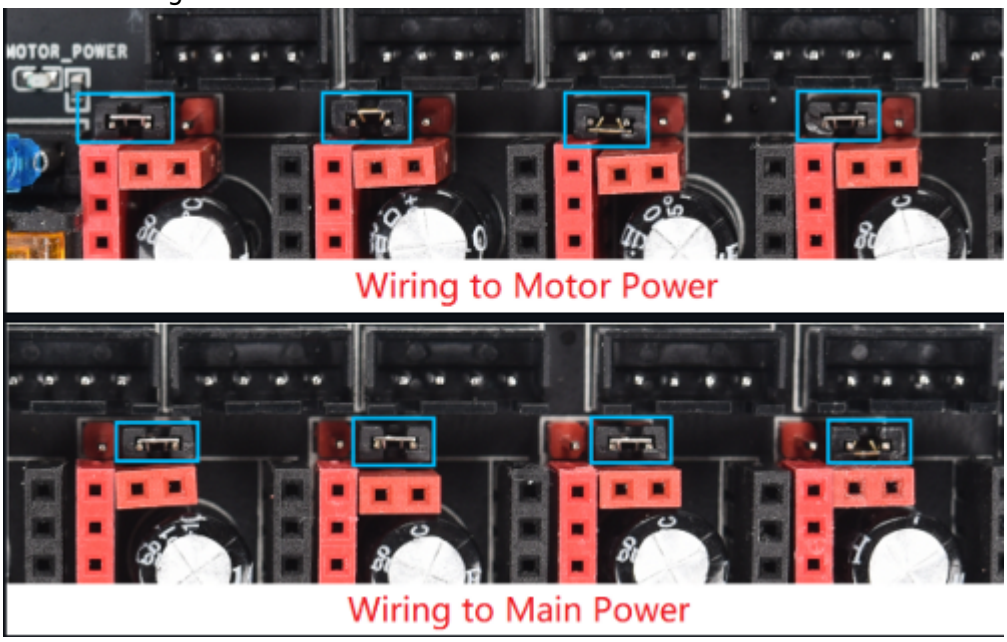
Versorgung Raspberry Pi

- Ein Raspberry Pi kann direkt über das Board versorgt werden. Ein passendes Kabel liegt nicht bei.
- Über diesen Anschluss wird auch gleich TX / RX verbunden mit dem Pi. Damit wäre ein Betrieb über UART möglich.

> **Hinweis**
> Hier werden nur 5V geliefert. Neue Raspberry Pi brauchen aber eher 5.1V. Es kann also mitunter zu Spannungsproblemen beim Pi kommen (Pi4, Pi5)

60V Anschluss (PRO)

- Das PRO Board kann bis 60V betrieben werden (mit den korrekten Stepper Treibern)
- Es muss ausgewählt werden wo der Motorstrom her kommen soll:



Bootloader sichern

Eine Sicherung lohnt nicht. Das Board wird mit Marlin ausgeliefert und kann schnell komplett neu geflasht werden.

Bootloader findet man hier

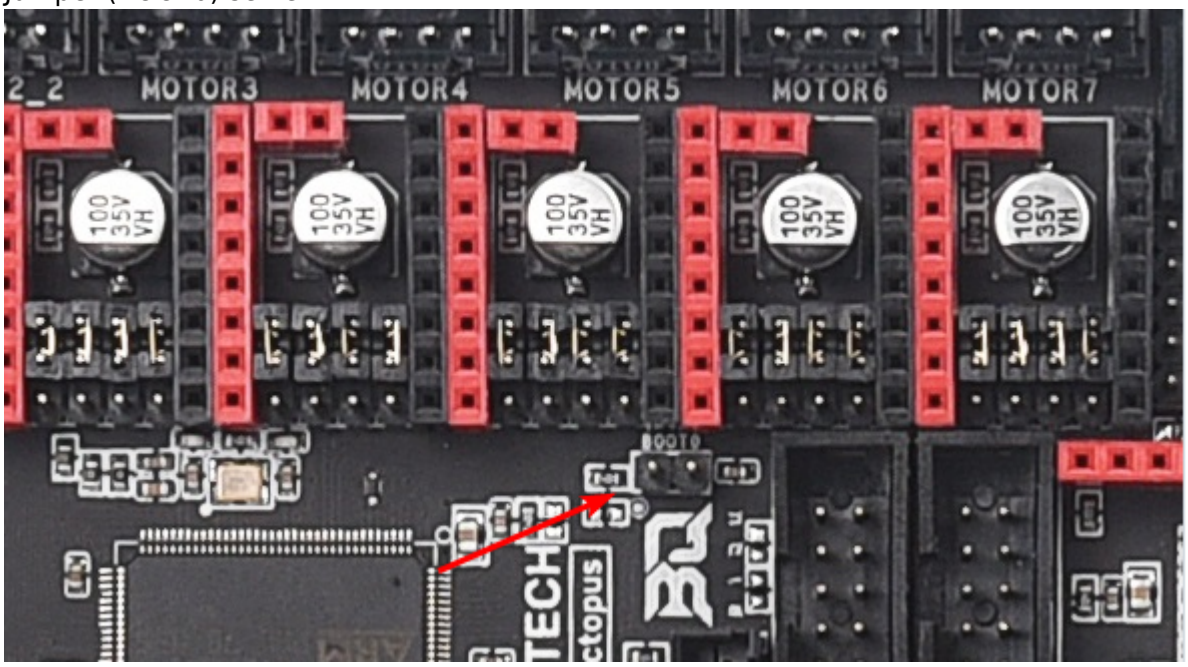
- https://github.com/GadgetAngel/BTT_SKR_13_14_14T_SD-DFU-Bootloader
- Für das PRO H723 habe ich den Bootloader mal extrahiert
  Bootloader und Orgiginal Firmware

Vorgehen Flashen

- Wer sein Board das erste mal mit Klipper einrichtet muss die folgenden Schritte durchgehen:
  - DFU-Modus aktivieren
  - Katapult
  - Port ermitteln
  - Klipper
  - SBC
- Wer das Board schon nach dieser Anleitung eingerichtet hat kann das Klipper Update so durchführen …
  - Update

DFU Modus

- Board ausschalten / stromlos machen
- Jumper (BOOT0) setzen



- Stromversorgung wählen (**nicht beides nutzen**)
  - **Achtung** : Hauptstromanschluss und gesetzter 5V Jumper können zu einem Kurzschluss auf 5V führen !
  - Wenn das Board **extern mit Strom** versorgt wird, den Jumper `POWER` oder `V_BUS / 5V` **entfernen**.
  - Wenn das Board über **USB** versorgt wird, den **Jumper setzen**.
- In der Konsole `dmesg -Hw` starten und dann das Board per USB an schließen und es sollte in etwa so eine Meldung kommen :

```
[Oct 3 08:00] usb 1-1.4: new full-speed USB device number 5 using
xhci_hcd
[  +0.089949] usb 1-1.4: not running at top speed; connect to a high
speed hub
[  +0.006004] usb 1-1.4: New USB device found, idVendor=0483,
idProduct=df11, bcdDevice= 2.00
[  +0.000045] usb 1-1.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[  +0.000013] usb 1-1.4: Product: DFU in FS Mode
[  +0.000009] usb 1-1.4: Manufacturer: STMicroelectronics
[  +0.000008] usb 1-1.4: SerialNumber: 306D35693231
```

Katapult flashen

> **Hinweis:**
> 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`
  Hier den richtigen Controller auswählen!
  - STM32**F446**

```
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F446)  --->
    Build Katapult deployment application (Do not build)  --->
    Clock Reference (12 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
(PA13)  Status LED GPIO Pin
```

  - STM32**F429**

```
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F429)  --->
    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
```

  - STM32**F407**

```
    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
```

  - STM32F**H723**

```
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32H723)  --->
```

```
    Build Katapult deployment application (Do not build)  --->
    Clock Reference (25 MHz crystal)  --->
    Communication interface (USB (on PA11/PA12))  --->
    Application start offset (128KiB 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
(PA13)  Status LED GPIO Pin
```

    - Die Status LED bei den F429 und F407 könnte auch PA13 sein. Lässt sich aber nicht verifizieren. Am besten testen 🙂
    - **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 !)
  `sudo 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
- Den Jumper BOOT0 entfernen !
- Das Board einmal resetten
    - Reset-Taste (oberhalb vom USB-C Anschluss) drücken
    - oder das Board einmal stromlos machen
- Die Status LED vom Board sollte jetzt blinken (Die LED kann ja noch Board an unterschiedlichen Stellen sein!)

Port ermitteln

- Den USB Stecker abziehen
- `dmesg -HW` starten und den USB Stecker wieder anstecken

```
pi@Pi4Test:~/katapult $ dmesg -HW
[Oct 3 08:16] usb 1-1.4: USB disconnect, device number 8
[  +0.453418] usb 1-1.4: new full-speed USB device number 9 using
xhci_hcd
[  +0.099931] usb 1-1.4: New USB device found, idVendor=1d50,
idProduct=6177, bcdDevice= 1.00
[  +0.000048] usb 1-1.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[  +0.000014] usb 1-1.4: Product: stm32h723xx
[  +0.000011] usb 1-1.4: Manufacturer: katapult
[  +0.000010] usb 1-1.4: SerialNumber: 35003B001951313234353230
[  +0.056130] cdc_acm 1-1.4:1.0: ttyACM0: USB ACM device
[  +0.000162] usbcore: registered new interface driver cdc_acm
[  +0.000012] cdc_acm: USB Abstract Control Model driver for USB modems
and ISDN adapters
```

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

Klipper flashen

- `cd ~/klipper`
- `make menuconfig`
  Hier den richtigen Controller auswählen!
  - STM32**F446**

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F446)  --->
    Bootloader offset (32KiB bootloader)  --->
    Clock Reference (12 MHz crystal)  --->
    Communication interface (USB to CAN bus bridge (USB on
PA11/PA12))  --->
    CAN bus interface (CAN bus (on PD0/PD1))  --->
    USB ids  --->
(1000000) CAN bus speed
[*] Optimize stepper code for 'step on both edges' (NEW)
()  GPIO pins to set at micro-controller startup (NEW)
```

  - STM32**F429**

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F429)  --->
    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 PD0/PD1))  --->
    USB ids  --->
(1000000) CAN bus speed
[*] Optimize stepper code for 'step on both edges' (NEW)
()  GPIO pins to set at micro-controller startup (NEW)
```

  - STM32**F407**

```
[*] 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 PD0/PD1))  --->
    USB ids  --->
(1000000) CAN bus speed
```

```
[*] Optimize stepper code for 'step on both edges' (NEW)
()  GPIO pins to set at micro-controller startup (NEW)
```

- STM32F**H723**

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32H743)  --->
    Bootloader offset (128KiB bootloader)  --->
    Clock Reference (25 MHz crystal)  --->
    Communication interface (USB to CAN bus bridge (USB on
PA11/PA12))  --->
    CAN bus interface (CAN bus (on PD0/PD1))  --->
    USB ids  --->
(1000000) CAN bus speed
()  GPIO pins to set at micro-controller startup (NEW)
```

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

```
pi@Pi4Test:~/klipper $ make -j4 flash FLASH_DEVICE=/dev/ttyACM0
  Creating symbolic link out/board
  Building out/autoconf.h
  Building hid-flash
...
  Preprocessing out/src/generic/armcm_link.ld
  Building out/compile_time_request.o
Version: v0.13.0-253-g96c3ca160
  Linking out/klipper.elf
  Creating bin file out/klipper.bin
  Flashing out/klipper.bin to /dev/ttyACM0
Entering bootloader on /dev/ttyACM0
Device reconnect on
/sys/devices/platform/scb/fd500000.pcie/pci0000:00/0000:00:00.0/0000:01
:00.0/usb1/1-1/1-1.4/1-1.4:1.0
/usr/bin/python3 lib/canboot/flash_can.py -d /dev/serial/by-
path/platform-fd500000.pcie-pci-0000:01:00.0-usb-0:1.4:1.0 -f
out/klipper.bin

Attempting to connect to bootloader
CanBoot Connected
Protocol Version: 1.1.0
Block Size: 64 bytes
Application Start: 0x8020000
MCU type: stm32h723xxv0.0.1-106-g399e50e
Flashing '/home/pi/klipper/out/klipper.bin'...

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

Write complete: 1 pages
```

```
Verifying (block count = 679)...

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

Verification Complete: SHA = 624529675027BC607A142B6E6DC44ABF81D3BD50
CAN Flash Success
```

- Board einmal resetten
- kurzer Test mit `lsusb` → Geschwister Schneider CAN adapter sollte erscheinen

```
pi@Pi4Test:~/klipper $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 006: ID 1d50:606f OpenMoko, Inc. Geschwister Schneider
CAN adapter
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

SBC

- ❗ Seit Bookworm gibt es eine neue CAN Konfiguration! ❗
  Ich habe dazu auch ein extra Video gemacht
  https://www.drklipper.de/doku.php?id=klipper_faq:can:neue_can_konfiguration
- Im aktuellen **MainSail OS** ist das schon fertig eingebunden. Ihr braucht also an dieser stelle nur `sudo reboot`
  Und danach sollte euer board gefunden werden.
- Bei **anderen Images** (Vorausgesetzt >= Bookworm) könnte ihr die Konfig so anlegen:
  - Konfigurationsdateien erstellen
    `sudo nano /etc/systemd/network/25-can.network`

    ```
    [Match]
    Name=can*
    [CAN]
    BitRate=1M
    [Link]
    RequiredForOnline=no
    ```

    NOTE RestartSec=0.1s füht zu Interface Problemen ab Kernel 6.12.47

- Konfigurationsdateien erstellen
  `sudo nano /etc/systemd/network/99-can-link.link`

```
[Match]
OriginalName=can*
[Link]
TransmitQueueLength=128
```

- Dienst aktivieren
  `sudo systemctl enable --now systemd-networkd`
- `sudo udevadm control --reload-rules && sudo udevadm trigger`

- Reboot erforderlich
  `sudo reboot`
- Prüfen
- `ip link show can0` → Zeigt qlen 128
- `networkctl status can0` → Zeigt Bitrate 1M
- Interface auf UP setzen
  `sudo ip link set can0 up type can bitrate 1000000`

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@Pi3Test:~/klipper $ ~/klippy-env/bin/python
~/klipper/scripts/canbus_query.py can0
Found canbus_uuid=c57dc3b70ff0, Application: Klipper
Total 1 uuids found
```

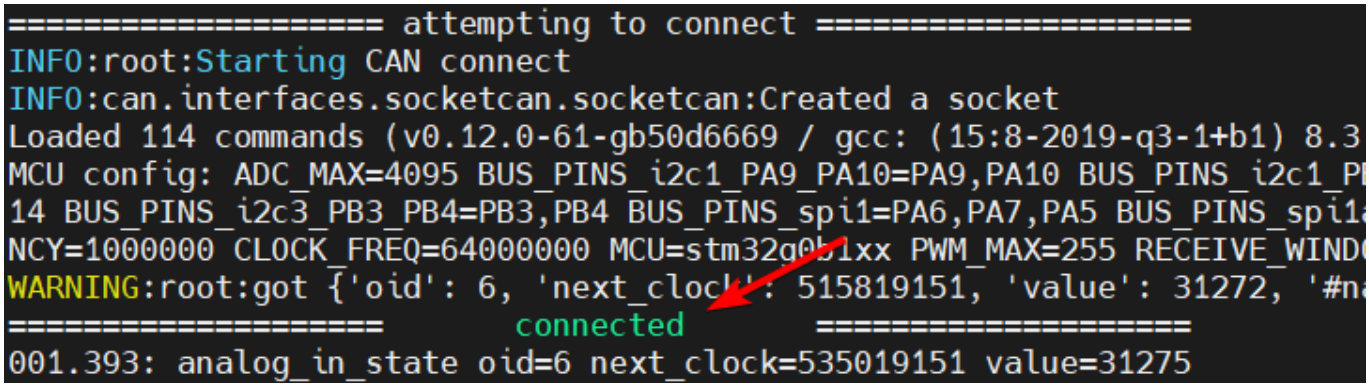
  - Die **UUID** (canbus_uuid=**c57dc3b70ff0**) notieren !
  - Wird bei diesem Schritt kein Board gefunden, hilft oft ein Reset am Board (entweder über Reset Taster oder 1x Strom weg und wieder dran)

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 c57dc3b70ff0`

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.



Konfiguration

- `cd ~/printer_data/config`
- **Beispiel Konfiguration Octopus 1.1**
  `cp ~/klipper/config/generic-bigtreetech-octopus-v1.1.cfg printer.cfg`
- **Beispiel Konfiguration Octopus PRO 1.0**
  `cp ~/klipper/config/generic-bigtreetech-octopus-pro-v1.0.cfg printer.cfg`
- **Beispiel Konfiguration Octopus PRO 1.1**
  `cp ~/klipper/config/generic-bigtreetech-octopus-pro-v1.1.cfg printer.cfg`
- `nano ~/printer_data/config/printer.cfg`

```
[mcu]
canbus_uuid: c57dc3b70ff0
#restart_method: command
```

  - Die Zeile mit `serial` löschen oder auskommentieren
  - Die Zeile mit `restart_method` löschen oder auskommentieren
  - Die Zeile mit `canbus_uuid` entsprechend mit der ermittelten UUID von oben anpassen
- Klipper starten
  `sudo systemctl start klipper.service`

Klipper Update

> **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@Pi3Test:~/klipper $ grep canbus_uuid ~/printer_data/config/* -n
/home/pi/printer_data/config/BTT_EBB.cfg:10:canbus_uuid: 44d860c9632b
/home/pi/printer_data/config/printer.cfg:30:canbus_uuid: c57dc3b70ff0
```

- Das Octopus Board per flshtool.py resetten. Welche UUID das Leviathan hat kann man bei mehreren Busteilnehmern leider nicht ohne weitere erkennen.

`~/klippy-env/bin/python ~/katapult/scripts/flashtool.py -i can0 -u <BOARD UUID> -r`

```
pi@Pi3Test:~/klipper $ ~/klippy-env/bin/python
~/katapult/scripts/flashtool.py -i can0 -u c57dc3b70ff0 -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@Pi3Test:~/klipper $ dmesg |tail -n 10
[76418.167383] IPv6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready
[76867.446711] usb 1-1.4: USB disconnect, device number 37
[76867.446933] gs_usb 1-1.4:1.0 can0: Couldnt shutdown device (err=-19)
[76867.787311] usb 1-1.4: new full-speed USB device number 38 using
dwc_otg
[76867.933716] usb 1-1.4: New USB device found, idVendor=1d50,
idProduct=6177, bcdDevice= 1.00
[76867.933741] usb 1-1.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[76867.933749] usb 1-1.4: Product: stm32f446xx
[76867.933755] usb 1-1.4: Manufacturer: katapult
[76867.933761] usb 1-1.4: SerialNumber: 350053000851313133353932
[76867.938929] cdc_acm 1-1.4:1.0: ttyACM0: USB ACM device
```

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

- `cd ~/klipper`
- `make menuconfig`
  → Die Einstellungen sind dieselben wie oben unter Klipper flashen 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 😎

ST-Link (SWD)

Das Board verfügt über einen SWD Port. Mit einem entsprechenden ST-Link kann das Board auch

direkt geflasht werden.

STM32 Temperatur

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

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

Links

- Github Repo
  https://github.com/bigtreetech/BIGTREETECH-OCTOPUS-V1.0
  https://github.com/bigtreetech/BIGTREETECH-OCTOPUS-Pro
- Schaltplan
  Octopus
  https://github.com/bigtreetech/BIGTREETECH-OCTOPUS-V1.0/blob/e8772c868988fae53ebecabf80e005a5ca8df50e/Hardware/BIGTREETECH%20Octopus%20-%20SCH.pdf
  Octopus Pro
  https://github.com/bigtreetech/BIGTREETECH-OCTOPUS-Pro/blob/e576e906a688485e6d588152b6e62532858e5d35/Hardware/BIGTREETECH%20Octopus%20Pro%20V1.1-sch.pdf
- Klipper Konfig
  https://raw.githubusercontent.com/bigtreetech/BIGTREETECH-OCTOPUS-V1.0/master/Firmware/Klipper/generic-bigtreetech-octopus.cfg
  https://raw.githubusercontent.com/bigtreetech/BIGTREETECH-OCTOPUS-V1.0/master/Firmware/Klipper/generic-bigtreetech-octopus-pro-v1.1.cfg

From:
https://drklipper.de/ - **Dr. Klipper Wiki**

Permanent link:
https://drklipper.de/doku.php?id=klipper_faq:flash_guide:stm32h723:btt_octopus_pro_can

Last update: **2025/10/03 18:48**