USB Buskoppler

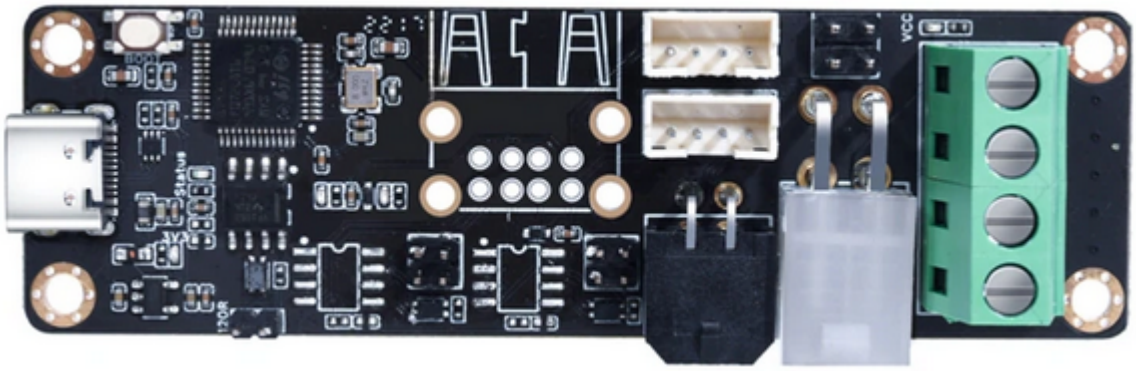
Dies ist das zweite Video zu meiner Klipper CAN Reihe. In diesem Video erkläre ich euch wie USB Bus Koppler funktionieren und wie man sie einrichtet. Zudem wird erklärt, wie ihr neue Firmware kompilieren könnt und wie ihr beim U2C / UTOC Board die extra “USB”-Hardware verwenden könnt.

YouTube Video #49

Übersicht

Auswahl einer Liste mit verfügbaren USB Buskopplern

BTT U2C



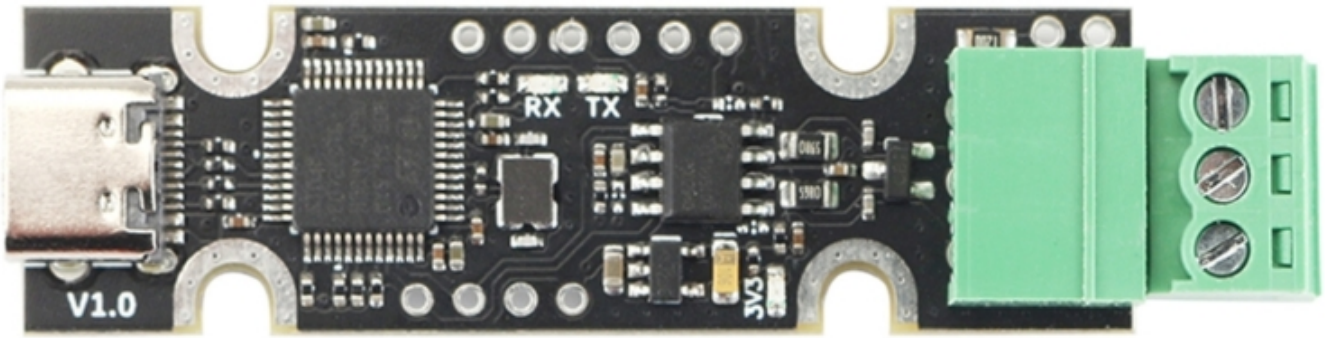
U2C 1.x

- Prozessor Typ : **STM32F072**
- candleLight Firmware kompilieren
- Klipper kompilieren
- https://github.com/bigtreetech/U2C

U2C 2.x

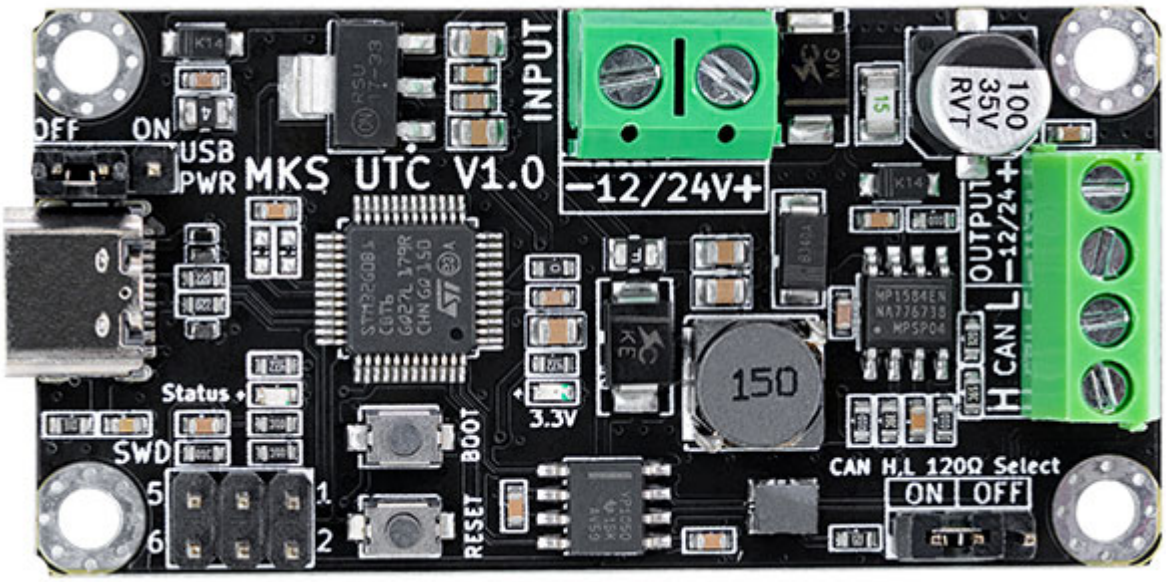
- Prozessor Typ : **STM32G0B1**
- candleLight Firmware kompilieren
- Klipper kompilieren
- Informationen zum Update Problem https://github.com/Esoterical/voron_canbus/tree/main/can_adapter/BigTreeTech%20U2C%20v2.1
- alternative Firmware für U2C 2.1 https://github.com/marckleinebudde/candleLight_fw/tree/multichannel
  `make nucleo_g0b1re_fw`
- alte Firmware Infos https://github.com/meteyou/KlipperMisc/blob/master/BigTreeTech-U2C/README.md https://github.com/meteyou/docs/blob/main/docs/btt-u2c/candlelight.md
- neue Firmware Infos https://github.com/docgalaxyblock/KlipperMisc/tree/main/CAN/BigTreeTech-U2C

UCAN



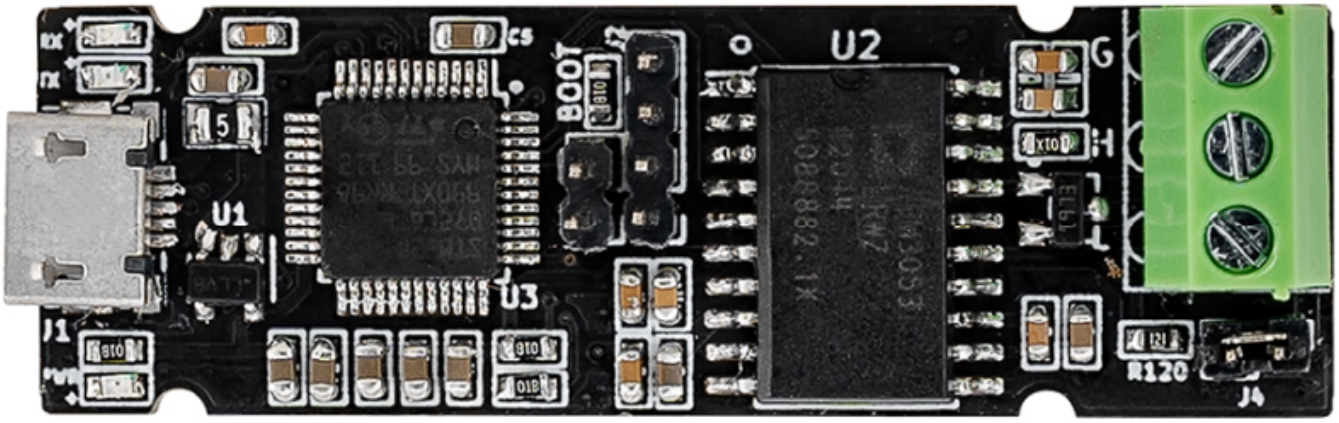
- Prozessor Typ : **STM32F072**
- candleLight Firmware kompilieren
- Klipper kompilieren
- leider keine weiteren Board-Infos zu finden von Fystec

Makerbase UTC



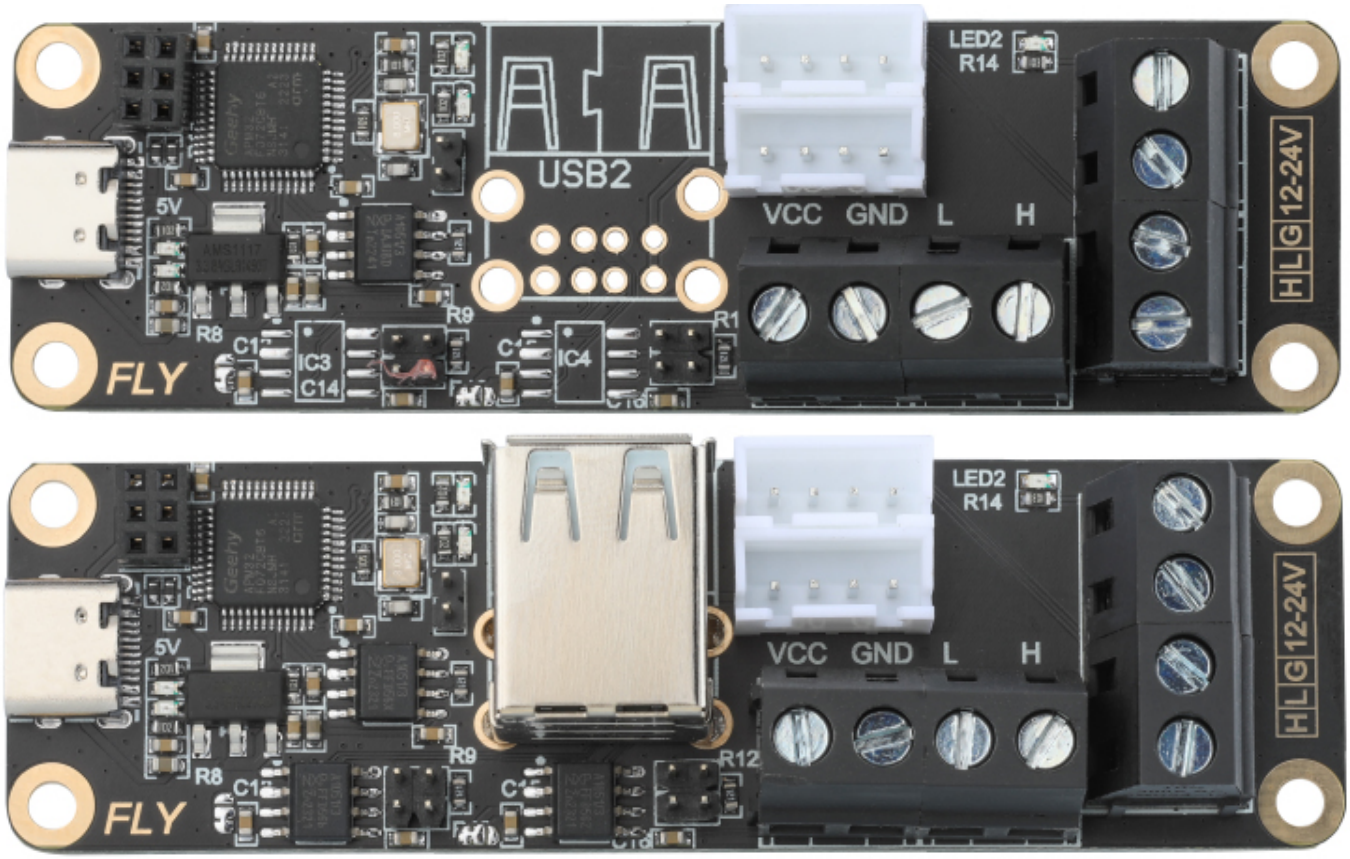
- Prozessor Typ : **STM32G0B1**
- candleLight Firmware kompilieren
- Klipper kompilieren
- https://github.com/makerbase-mks/MKS-THR36-THR42-UTC
- https://github.com/Esoterical/voron_canbus/tree/main/can_adapter/Makerbase%20UTC%201.0

Makerbase CANable-MKS



- Prozessor Typ : **STM32F072**
- candleLight Firmware kompilieren
- Klipper kompilieren
- https://github.com/makerbase-mks/CANable-MKS

Mellow Fly UTOC



- Prozessor Typ : **STM32F072**
- candleLight Firmware kompilieren
- Klipper kompilieren
- https://github.com/Mellow-3D/Klipper-CAN-Toolboards
- https://github.com/Esoterical/voron_canbus/tree/main/can_adapter/Mellow%20Fly%20UTOC-1%20and%20UTOC-3
- eine vorkompilierte Firmware findet sich hier https://mellow-3d.github.io/files/utoc_firmware.bin

CAN Druckerboard

Auch normale Druckerboard oder Kopfboards können als USB Koppler verwendet werden. Wichtig ist dabei folgendes:

- Das Board muss getrennte Pins für CAN und USB aufweisen
- Das Board sollte im Idealfall schon einen Transceiver besitzen (Das trifft auf alle Kopfboards zu - kaum aber auf normale Druckerboards)

Ein EBB36 oder EBB42 kann also recht problemlos dafür verwendet werden, um einen USB Bus Koppler zu ersetzen - auch wenn dabei natürlich eine Menge Funktionen von dem Board ungenutzt bleiben.

Das ganze funktioniert deshalb, weil Klipper in dem Fall als **USB to CAN bus bridge** eingesetzt wird.

Linux Tools

- in der Linux SSH Konsole folgendes ausführen:
  `tail -f /var/log/kern.log`
- Gerät anstecken über USB
- Ergebnis:

```
pi@TestPi:~ $ tail -f /var/log/kern.log
Apr 30 10:38:51 TestPi kernel: [ 7313.123004] usb 1-1.4: new full-speed
USB device number 7 using dwc_otg
Apr 30 10:38:51 TestPi kernel: [ 7313.256404] usb 1-1.4: New USB device
found, idVendor=1d50, idProduct=606f, bcdDevice= 0.00
Apr 30 10:38:51 TestPi kernel: [ 7313.256442] usb 1-1.4: New USB device
strings: Mfr=1, Product=2, SerialNumber=3
Apr 30 10:38:51 TestPi kernel: [ 7313.256458] usb 1-1.4: Product:
candleLight USB to CAN adapter
Apr 30 10:38:51 TestPi kernel: [ 7313.256471] usb 1-1.4: Manufacturer:
bytewerk
Apr 30 10:38:51 TestPi kernel: [ 7313.256483] usb 1-1.4: SerialNumber:
0050004A5542501720393839
Apr 30 10:38:51 TestPi kernel: [ 7313.258646] gs_usb 1-1.4:1.0:
Configuring for 1 interfaces
```

- Alternativ auch `sudo dmesg -Wtd` aufrufen und Gerät per USB anstecken und
- auch `lsusb` liefert einen CAN Adapter:

```
pi@TestPi:~ $ lsusb
Bus 001 Device 007: ID 1d50:606f OpenMoko, Inc. Geschwister Schneider
CAN adapter
```

Netzwerk einrichten

Damit der Adapter im Betriebsystem auch erkannt wird muss eine Netzwerk Interface Konfiguration (`/etc/network/interfaces.d/can0`) angelegt werden.

- `sudo nano /etc/network/interfaces.d/can0`
- folgenden Inhalt in der Datei einfügen :

```
auto can0
iface can0 can static
    bitrate 500000
    up ifconfig $IFACE txqueuelen 1024
```

- Editor mit **STRG + x → dann Y → dann Enter** verlasen
- System neu starten mittels
  `sudo reboot`
- Nach dem Reboot sollte der Befehl `ip a` ein CAN Interface listen :

```
can0: <NOARP,UP,LOWER_UP,ECHO> mtu 16 qdisc pfifo_fast state UP group
```

```
default qlen 1024
    link/can
```

- Wichtig dabei ist **state UP**. Wird hier ein DOWN angegeben, ist das Interface nicht betriebsbereit!

txqueuelen

“txqueuelen” ist eine Einstellung für die Übertragungswarteschlange (englisch: Transmission Queue Length) eines Netzwerkinterfaces in Linux-basierten Betriebssystemen.

Die txqueuelen-Einstellung bestimmt die maximale Anzahl von ausgehenden Netzwerkpaketen, die in der Warteschlange des Interfaces gespeichert werden können, bevor sie tatsächlich übertragen werden. Eine höhere txqueuelen-Einstellung kann die Leistung des Netzwerkinterfaces verbessern, indem sie die Übertragung von Paketen beschleunigt, insbesondere bei hoher Netzwerklast. Allerdings kann eine zu hohe txqueuelen-Einstellung auch zu einer höheren Latenzzeit und zu einem erhöhten Speicherbedarf führen. Die optimale Einstellung hängt von den spezifischen Anforderungen des Netzwerks und der Anwendung ab, die das Interface nutzt.

Die Angaben zu txqueuelen variieren von Anleitung zu Anleitung. Oft finden sich **Werte im Bereich von 256-1024**. Viel höher sollte man sicher auch nicht gehen.

Netzwerk Restart

Wenn der USB Stecker vom Buskoppler abgezogen wird, geht der CAN Bus sofort offline. Mittels `ip a` ist er dann auch nicht mehr sichtbar. Um nicht immer das ganze System neu zu starten kann nach dem erneuten Verbinden des Buskopplers folgender Befehl genutzt werden:
`sudo systemctl restart networking.service`

Firmware

Als Firmware für die USB Bus Koppler kann entweder candleLight oder Klipper (USB to CAN bus bridge) eingesetzt werden. Grundsätzlich würde ich eher auf die candleLight Version gehen. Denn die ist speziell für den Einsatz als USB Bus Koppler entwickelt worden. Zudem unterstützt sie (Stand Mai 2023) auch mehr Hardware als Klipper.

Ein weiterer Nachteil von Klipper sind nötige Firmware Updates. Es kommt von Zeit zu Zeit vor das ein Klipper Update relevant ist. Dann müsste auch der Buskoppler wieder mit einer Klipper Firmware upgedatet werden.

Ein Nachteil bei candleLight ist derzeit das es zwei Versionen gibt. Eine für den STM32F072 und eine für den STM32G0B1. Es gibt Bestrebungen diese beiden Versionen zusammenzuführen, aber wann das genau passiert bleibt abzuwarten. **Also Obacht beim Firmware kompilieren und installieren!**

Hinweis zu candleLight
Leider gibt es nicht wirklich viele brauchbare Informationen welche candleLight Firmware nun genau genutzt werden muss. Wenn man candleLight kompiliert (ohne extra Parameter) entstehen 8-10

verschiedene Firmware Versionen. In der Erklärung zum Firmware kompilieren zu STM32F072 und STM32G0B1 gibt es jeweils eine (wenn auch unvollständige) Liste welcher Typ genutzt werden sollte. Wenn man das selber rausfinden möchte hier ein paar Tips:

- im Quellcode in der Datei `candleLight_fw/CMakeLists.txt` findet sich schon mal eine Liste für jeden µController (Beispiel):

```
########## generate list of targets.
# the "_fw" part is appended automatically
set(TGTF042_LIST "cantact" "canalyze" "canable" "usb2can" "cannette")
set(TGTF072_LIST "candleLight" "CANable_MKS" "CONVERTDEVICE_xCAN01")
set(TGTF407_LIST "STM32F4_DevBoard")
set(TGTG0B1_LIST "budgetcan")
```

  Hier ist ersichtlich, welche Version für einen bestimmten Controller verfügbar ist.

- Welche Variante man dann nimmt - also z.B. CANable_MKS - kann man aus der Datei `candleLight_fw/include/config.h` entnehmen. Hier sind die Pin Konfigurationen und Chip Details hinterlegt. Das kann man ganz gut mit dem Schaltplan (sofern vorhanden) abgleichen.
- Dann kann man auch mal in die aktuelle Firmware schauen. Die kann man für einige Koppler aus dem Netz laden, oder man extrahiert die Firmware selber. In der Firmware finden sich manchmal textuelle Hinweise, was für das Kompilieren verwendet wurde: `gs_usb interface fw_0e1638b_2022-03-09 canable.io canable gs_usb canable firmware upgrade interface`
- Und zuletzt gibt es auf der Startseite von dem Github Projekt jeweils eine längere Liste mit Hinweisen, welches Board welchen Controller verwendet. Zusammen mit den oberen Hinweisen kann man dann auch oft herausfinden welche Kompile-Variante genutzt werden muss.

candelight (STM32F072)

- benötigte Pakete installieren
  `sudo apt install cmake gcc-arm-none-eabi`
- `cd ~`
- Repository von Github klonen
  `git clone https://github.com/candle-usb/candleLight_fw`
- `cd ~/candleLight_fw`
- Vorbereitungen für das Kompilieren (Create Toolchain)
  ```
  mkdir build
  cd build
  cmake .. -DCMAKE_TOOLCHAIN_FILE=../cmake/gcc-arm-none-eabi-8-2019-q3-update.cmake
  ```
- Kompilieren je nach verwendetem Adapter:
  - **U2C 1.x** → `make canable_fw` oder `make candleLight_fw`
  - **UCAN** → `make FYSETC_UCAN_fw`
  - **Makerbase CANable-MKS** → `make CANable_MKS_fw`
  - **Mellow Fly UTOC** → noch unbekannt Muss mit den verschiedenen F072 Optionen getestet werden!

candelight (STM32G0B1)

- benötigte Pakete installieren
  `sudo apt install cmake gcc-arm-none-eabi`
- `cd ~`
- Repository von Github klonen
  `git clone --depth=1 -b stm32g0_support https://github.com/bigtreetech/candleLight_fw`
- `cd ~/candleLight_fw`
- Vorbereitungen für das Kompilieren (Create Toolchain)
  ```
  mkdir build
  cd build
  cmake .. -DCMAKE_TOOLCHAIN_FILE=../cmake/gcc-arm-none-eabi-8-2019-q3-update.cmake
  ```
- Kompilieren je nach verwendetem Adapter:
  - **U2C 2.x** → `make budgetcan_fw`
  - **Makerbase UTC** → `make budgetcan_fw`

Klipper (STM32F072)

Klipper kann ebenfalls als Firmware für den Buskoppler verwendet werden. Hierbei kommt der USB/CAN Bridge Mode zum Einsatz, der in der Compile Konfiguration ausgewählt werden muss. Die folgende Anleitung zeigt **exemplarisch** wie das **für das U2C V1.x** funktioniert. Für andere Boards muss das CAN Interface ggf. angepasst werden! Hier sollte der Schaltplan zu Rate gezogen werden um die richtige Konfig zu ermitteln für die USB und CAN Pins!

Es wird hier auf den Einsatz von CanBoot verzichtet. Das macht die Installation nur unnötig komplex.

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

```
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F072)  --->
    Bootloader offset (No bootloader)  --->
    Clock Reference (8 MHz crystal)  --->
    Communication interface (USB to CAN bus bridge (USB on PA11/PA12))  --->
    CAN bus interface (CAN bus (on PB8/PB9))  --->
    USB ids  --->
(500000) CAN bus speed
()  GPIO pins to set at micro-controller startup
```

  mit **q** und **y** beenden
- `make -j4`

Klipper (STM32G0B1)

> **Hinweis**
> Klipper kann bei diesem Controller noch nicht als USB/CAN Bridge verwendet werden. Die Pins PB5/PB6 stehen in der Klipper make Konfiguration nicht zur Auswahl!
> Es blebt also nur die candleLight Variante.
> Ein Pull Request bei Klipper ist dafür in Vorbereitung. Die Funktion sollte also in den nächsten Wochen / Monaten zur Verfügung stehen.

Flashen

Zum Flashen wird der DFU Mode verwendet. Dafür gibt es auf allen Boards immer einen BT0 oder Boot0 Taster. Um das Board in den DFU mode zu bekommen geht man wie folgt vor:

- Board vom USB-Port trennen
- BT0 bzw. Boot0 Taste drücken
- USB Verbindung wieder herstellen
- wenn alles geklappt hat, liefert ein `dmesg` in der Linux Konsole folgendes: **DFU in FS Mode**

```
[13393.886628] usb 1-1.4: new full-speed USB device number 7 using
dwc_otg
[13394.020063] usb 1-1.4: New USB device found, idVendor=0483,
idProduct=df11, bcdDevice= 2.00
[13394.020105] usb 1-1.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[13394.020122] usb 1-1.4: Product: DFU in FS Mode
[13394.020134] usb 1-1.4: Manufacturer: STMicroelectronics
[13394.020146] usb 1-1.4: SerialNumber: 208938835542
```

Das Board kann dann geflasht werden mit der passenden Firmware.

candleLight
Hier findet man im Ordner `~/candleLight_fw/build` die kompilierte Firmware. Je nach make Variante hat die Firmware Datei einem anderen Namen. Wurde die Firmware als Beispiel mit `make budgetcan` kompiliert, dann lautet der Dateiname der Firmware `budgetcan_fw.bin`. Der Befehl zum Flashen des Boards würde dann so aussehen:
`dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D ~/candleLight_fw/build/budgetcan_fw.bin`

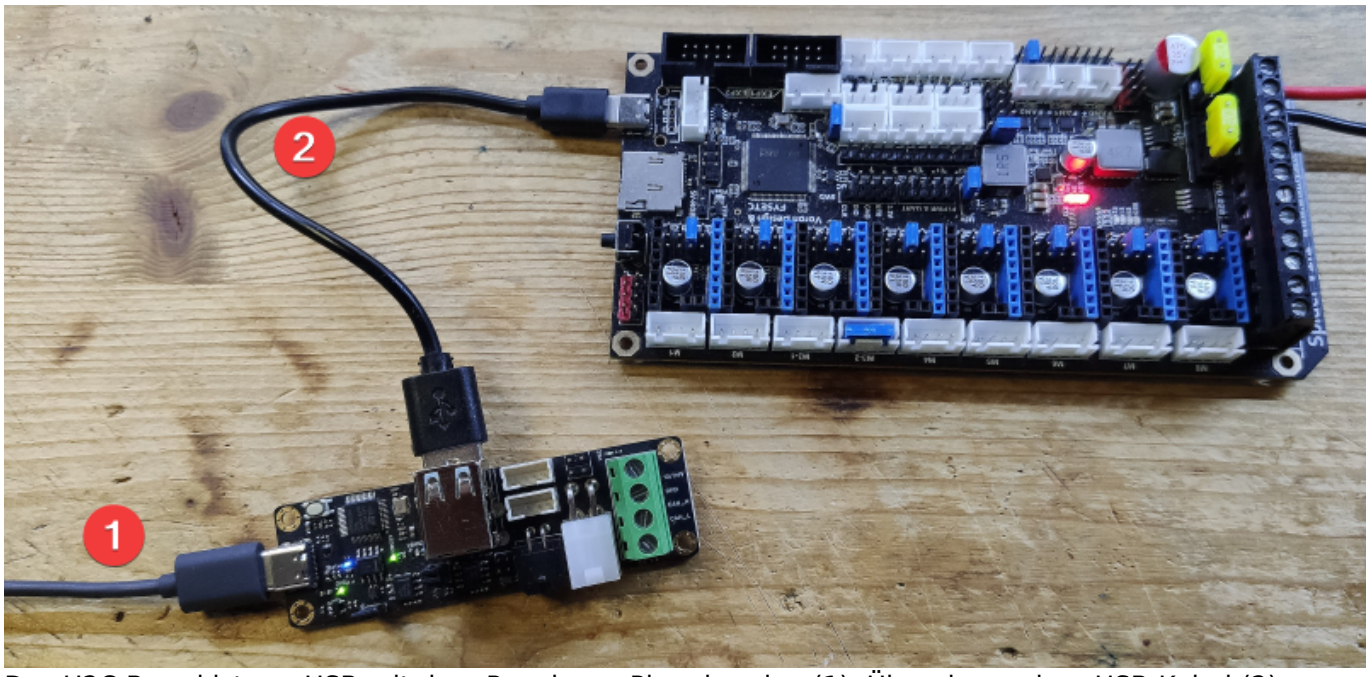
Klipper
Wurde Klipper als Firmware für den Buskoppler kompiliert lautet der Flash Befehl wie folgt:
`dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D ~/klipper/out/klipper.bin`

Hersteller Firmware
Wurde eine Hersteller Firmware aus dem Internet geladen lautet der Flash Befehl wie folgt:
`dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D <DATEINAME>`
<DATEINAME> muss dabei durch den Dateinamen der Firmware ersetzt werden.

U2C / UTOC extra Hardware

Diese Boards haben extra 2 USB-Ports mitten auf dem Board. Es sei an dieser Stelle darauf hingewiesen, dass das keine realen USB-Ports sind! Vielmehr dienen diese Ports nur dazu, Druckerboards ohne Transceiver zu verbinden. Beide Boards haben dafür 2 zusätzliche Transceiver aufgelötet. Wenn man nun ein Board wie das Spider oder das Octopus verwendet, können diese über ein USB-Kabel mit dem Buskoppler Board verbunden werden. Über das USB-Kabel werden dann die CAN_RX und CAN_TX Leitungen vom µController übertragen. USB wird hier also nur für die Leitungen zweckentfremdet. Die Druckerboards werden dann mit einer Klipper Firmware kompiliert, wo die CAN

Leitungen die gleichen sind wie die USB Leitungen. Jetzt muss am Buskoppler ggf. noch der 120Ohm Widerstand gesetzt werden und vermutlich die VBUS Spannung (vorher am Druckerboard nachmessen, ob dort nichts anliegt !!).



Das U2C Board ist per USB mit dem Raspberry Pi verbunden (1). Über das andere USB-Kabel (2) werden nur die CAN Daten des µControllers übertragen.

Nachteile

- Lange Zuleitung für CAN_TX, CAN_RX
- USB-Kabel gibt es als reine Ladekabel → keine Verbindung
- Eigentlich unnötiger Aufwand … Warum nicht gleich USB verwenden, wenn man ein USB-Kabel dran hat?
- **Spannung messen am USB BUS VBUS1 & VBUS2** (wenn das Druckerboard noch nicht angeschlossen ist) könnten am USB Stecker vom Druckerboard anliegen. Dann hätte man einen Kurzen zwischen VBUS U2C und VBUS vom Druckerboard!
- 120Ohm Terminierung beachten!
- In Summe zu viele Anschlussmöglichkeiten. Verwirrende Ports wie die beiden weißen z.B.

Jumper

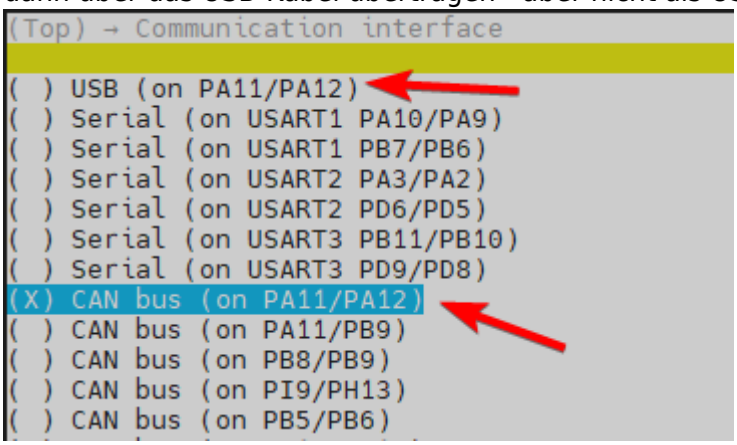
- der Jumper ganz links zum USB-C-Port ist die Terminierung für das U2C selber (wenn benötigt)
- unterer USB-Port (zur Platine) ist der mittlere Jumper Block (VBUS1)
- oberer USB-Port ist der Block zum grünen Anschluss hin (VBUS2)

CAN_OUT

Die CAN_OUT “Ports” bieten die Möglichkeit, 2 CAN Geräte ohne Transceiver anschließen zu können. Dafür müssen aber 2 Voraussetzungen erfüllt sein:

1. Die CAN Pins und USB Pins müssen übereinstimmen (Die CAN_RX und CAN_TX Daten werden

dann über das USB-Kabel übertragen - aber nicht als USB Device !). Beispiel Spider Board:



2. Die USB Pins müssen direkt mit der USB Buchse verbunden sein (µController → USB Buchse)

Links

- https://github.com/xbst/PiCAN/tree/master