Bootloader Katapult

Katapult (ehemals CanBoot) wurde ursprünglich entwickelt, um über CAN-Bus Klipper Updates einzuspielen. Es wurde aber erweitert um die Möglichkeit auch mittels USB bzw. Seriell zu flashen. Somit ist Katapult ein sehr universell einsetzbarer Bootloader. Er ermöglicht ein Firmware Update (wie z.B. Klipper), ohne dass man am Druckerboard selber Hand anlegen muss. Also kein SD-Karten jonglieren, keine Jumper stecken, keine Taster drücken.

Das Video / diese Anleitung befasst sich nur mit STM32, RP2040 und da auch nur mit USB und CAN Bus! Serielles Flashen sollte am Ende ähnlich funktionieren wie die USB Variante. Die LPC176x Controller werden hier nicht behandelt da sie kaum noch Verwendung finden.

> **Hinweise:**
> 1. SBC meint im folgenden Text immer Single Board Computer und dürfte bei den meisten ein Raspberry Pi sein.
> 2. Wenn ihr Katapult installiert, wird der Original Bootloader gelöscht!

Github Repo zu Katapult
https://github.com/Arksine/katapult

YouTube Video #66



Warum Katapult?

- Es ist die einzige Möglichkeit, ein Druckerboard über CAN-Bus direkt zu flashen. Es muss als nicht extra ein USB-Kabel angeschlossen werden, um eine neue Firmware wie Klipper aufzuspielen.
- Mit Katapult entfällt im Normalfall jegliches “Hand anlegen” am Druckerboard (Jumper setzen, Taster drücken, …).
- Es wird keine SD-Karte mehr für ein Firmware Update benötigt.
- Generell vereinfacht Katapult den Updateprozess. Gerade bei älteren Boards bzw. Boards mit STM32F1 Controllern ist oft ein direktes Flashen (z.B. mittels DFU) nicht möglich.

Hardware erkennen

Man sollte im Vorfeld zwei wichtige Informationen ermitteln. Welcher Controller wird verwendet (und welche Einstellungen brauche ich dafür) und ob es ein Board mit oder ohne DFU Möglichkeit ist (DFU betrifft nur STM32 Boards).

Controller

- Der Controller lässt sich schnell ermitteln. Auf dem Board gibt es immer einen großen Chip mit vielen Pins. Darauf findet man den Typ wie z.B. STM32F446, STM32F103, RP2 (auch zu erkennen an dem Raspberry Pi Logo).
- Zudem gibt es für jedes Board meist eine Beschreibung oder eine Github Seite mit weiteren Informationen. Hier bekommt man dann meist auch die Informationen wie z.B. 'Clock Reference', 'Communication interface', ... Diese Informationen brauchen wir zum Erstellen von Katapult wie auch Klipper!
- Beispiel: https://github.com/FYSETC/FYSETC-SPIDER
  - Am Ende der Seite findet man alle Einstellungen für `make menuconfig`
  - Wichtig sind immer 'Micro-controller Architecture', 'Processor model', 'Clock Reference', 'Communication interface'
  - Der Bootloader Offset ergibt sich beim Erstellen von Katapult - dazu später mehr.

DFU Möglichkeit

- DFU (Device Firmware Upgrade) gibt es nur bei STM32 Controllern.
- Es ist ein spezieller Modus, in dem die Firmware (Katapult oder Klipper) direkt über USB auf den Chip übertragen werden kann.
- Ob der DFU Mode verfügbar ist, lässt sich meist daran festmachen, ob ein extra Jumper (oder Taster) vorhanden ist. Dieser Jumper ist meist mit `BT0`, `Boot0` oder `B0` gekennzeichnet. Oft fehlt dieser Jumper bei älteren Boards oder kleinen Boards mit STM32F1 Controller.
  Sollte euer Board einen solchen Jumper besitzen könnt ihr im Grunde auf Katapult verzichten, wenn das Board per USB angeschlossen ist. Schaden tut Katapult aber nicht bei solchen Boards.

Katapult einrichten

- `cd ~ && git clone https://github.com/Arksine/katapult && cd katapult`
- `make menuconfig`
  - Hier braucht ihr jetzt die Informationen zum Controller. Beispiel Spider Board:

```
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F446)  --->
    Build Katapult deployment application (32KiB bootloader)  --->
    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
```

    - `Micro-controller Architecture` ist der Controller Type - also z.B. STM32 oder RP2040
    - `Processor model` ist die Variante - also z.B. STM32F446
    - `Build Katapult deployment application` muss dann aktiviert werden, wenn per SD-Karte gerflsht werden soll.
    - `Clock Reference` ist die Geschwindigkeit vom Onboard Taktgeber
    - `Communication interface` legt fest wie ihr mit dem Board kommunizieren wollt
    - `Application start offset` legt fest wo im Flash vom Controller später die Firmware (z.B. Klipper) startet. Nehmt hier am besten immer den kleinsten Wert - reicht im NOrmalfall. Und achtet darauf, dass unter `Build Katapult deployment application` der gleiche Wert gesetzt ist wenn ihr die Option nutzt!
    - USB `ids` lassen wir unangetastet
    - '' GPIO pins to set on bootloader entry'' hier kann man Pins eintragen, die beim Board Start einen bestimmten Status (gesetzt / nicht gesetzt) haben müssen. Einige ältere STM32 Boards brauchen das.
    - `[*] Support bootloader entry on rapid double click of reset button` diese Option sollte immer aktiviert sein. Denn so könnt ihr das Board auch mit dem Reset Taster in den Katapult Boot Modus bringen. Nennen wir es “Notanker” 🙂
    - `[ ] Enable bootloader entry on button (or gpio) state` und `[ ] Enable Status LED` braucht man im Grunde nie
- `make`
    - Damit wird Katapult kompiliert und liegt danach im Ordner `out\katapult.bin`.

Katapult flashen

Im Folgenden wird beschrieben, wie der Bootloader Katapult auf den Controller geflasht wird.
Das Flashen von Klipper erfolgt im nächsten Schritt!

> **Hinweise**
> 1. Man sollte nach dem Flashen von Katapult das Board 1x resetten.
> 2. Katapult wird immer über einen der folgenden Wege geflasht (USB, SD, ST-Link). Nur die eigentliche Firmware (z.B. Klipper) kann später über den CAN Bus geflsht werden!

STM32

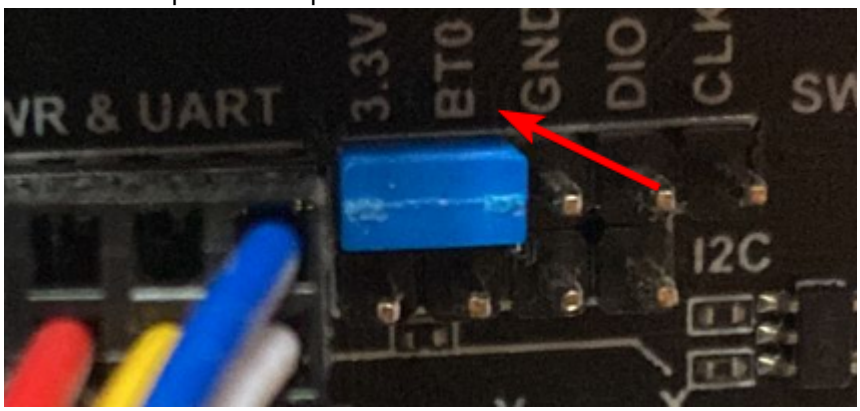
Bei den STM32 Controllern gibt es drei brauchbare Wege, um Katapult zu flashen. Der einfachste Weg wäre über den DFU Modus, der aber leider nicht bei allen Boards genutzt werden kann (siehe dazu Hauch Hardware erkennen → DFU Möglichkeit). Installation mittels SD-Karte wäre die zweite Option die (bis auf die SD-Karte) ohne extra Hardware auskommt. Die letzte Option wäre über einen ST-Link Programmieradapter.

DFU-Mode

> **Hinweis**
> Funktioniert nur über USB !

Wie schon erwähnt muss das Board dafür über einen extra Jumper (selten auch ein Taster) verfügen.

Hier ein Beispiel vom Spider Board:



Dieser Jumper legt 3,3V auf den BT0 Pin. Das Board geht damit nach einem Reset in einen speziellen DFU Modus und mittels `dfu-util` kann dann Katapult geflasht werden.
Überprüfen kann man das mit `dfu-util -l`. Wenn das Board per USB z.B. an eurem Pi hängt, kriegt ihr dann folgende Ausgabe:

```
pi@Pi3Test:~/katapult $ dfu-util -l
dfu-util 0.9

Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.
Copyright 2010-2016 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WARRANTY
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/

Found DFU: [0483:df11] ver=2200, devnum=29, cfg=1, intf=0, path="1-1.4",
alt=3, name="@Device Feature/0xFFFF0000/01*004 e", serial="STM32FxSTM32"
Found DFU: [0483:df11] ver=2200, devnum=29, cfg=1, intf=0, path="1-1.4",
alt=2, name="@OTP Memory /0x1FFF7800/01*512 e,01*016 e",
serial="STM32FxSTM32"
Found DFU: [0483:df11] ver=2200, devnum=29, cfg=1, intf=0, path="1-1.4",
alt=1, name="@Option Bytes  /0x1FFFC000/01*016 e", serial="STM32FxSTM32"
Found DFU: [0483:df11] ver=2200, devnum=29, cfg=1, intf=0, path="1-1.4",
alt=0, name="@Internal Flash  /0x08000000/04*016Kg,01*064Kg,03*128Kg",
serial="STM32FxSTM32"
```

Anschließend könnt ihr Katapult flashen:
`dfu-util -R -a 0 -s 0x08000000:mass-erase:force -D ~/katapult/out/katapult.bin`
Jetzt wird erst der Speicher gelöscht und anschließend Katapult geflasht.

Nicht vergessen, den Boot0 Jumper wieder zu entfernen!

SD-Karte

Wenn euer Board keinen DFU Jumper hat, dann könnt ihr Katapult auch über die SD-Karte flashen. Aber um es gleich vorwegzunehmen … Manchmal braucht man bei der Methode Geduld und innere Ruhe 😉 Denn es wäre nicht das erste Mal, dass diese Methode mehr schlecht als recht funktioniert …

- Für das Flashen per SD-Karte braucht ihr eine kleine SD-Karte. Alles bis 4Gb sollte funktionieren, größere Karten machen öfters Probleme.
- Formatiert die Karte als FAT32.
- Bei `make menuconfig` kann man bei Katapult folgende Option setzen: `Build Katapult deployment application`. Setzt hier den gleichen Wert wie bei `Application start offset`.
- Mit `make` das Kompilieren starten
- Im Katapult Ordner wird eine extra Datei erzeugt: `~/katapult/out/deployer.bin`.
- Diese Datei müsst ihr von eurem SBC (z.B. euer Raspberry Pi) auf die SD-Karte im Hauptverzeichnis kopieren. (Geht super mit MobaXTerm)
- Die Datei umbenennen von `deployer.bin` in `firmware.bin`
- Die SD-Karte in das Board stecken und das Board mit Strom versorgen.
- Jetzt sollte in den nächsten 5-10 Sekunden Katapult geflasht werden.
- Man bekommt übrigens keinen Hinweis, ob und wann das Flashen abgeschlossen ist. Ggf. also mehrfach probieren.
- Sollte das Flashen geklappt haben wirdf meistens die Datei auf der SD-Karte in `FIRMWARE.CUR` umbenannt.

Man merkt schon an der Länge der Anleitung, dass Flashen über SD-Karte ein Graus ist, den man - wenn Katapult einmal läuft - nie wieder braucht 🙂

ST-Link

Für diese Variante wird ein ST-Link benötigt. Das ist ein Programmieradapter für STM32 Controller. Man bekommt ihn als China Clone bei eBay oder Amazon und die Teile sehen so aus:



Wenn gar nichts mehr funktioniert, bekommt man mit so einem Adapter fast jeden STM32 wieder flott 🙂

- Um den ST-Link zu nutzen müssen wir erst die Software `stlink` kompilieren:
  - `sudo apt install gcc build-essential cmake libusb-1.0-0 libusb-1.0-0-dev libgtk-3-dev pandoc -y`
  - `cd ~ && mkdir build -p && cd build`
  - `git clone https://github.com/stlink-org/stlink.git`
  - `cd stlink`
  - `make clean && make release`
  - `sudo make install`
  - `sudo ldconfig`

- Jetzt wird der ST-Link V2 mit dem Board verkabelt. Dazu haben eigentlich alle STM32 Boards einen sogenannten SWD Port. Dieser Port besteht aus mehreren Kontakten und wir benötigen `CLK`, `DIO` und `GND`. Die Verkabelung mit dem ST-Link Adapter <> Druckerboard ist dann wie folgt:
  - SWDIO ↔ DIO
  - SWCLK ↔ CLK
  - GND ↔ GND
- Dann wird das Druckerboard über den USB-Port mit Strom versorgt und der ST-Link Adapter ebenfalls in den SBC gesteckt.
- Jetzt könnt ihr mit folgendem Befehl erstmal den Speicher löschen:
  `st-flash --connect-under-reset erase`
  Achtung: Der Controller muss einen Reset machen damit sich die Software korrekt verbinden kann. Deshalb folgende Reihenfolge einhalten:
  - Reset Taste am Board drücken und halten
  - Das Kommando ausführen als z.B. `st-flash --connect-under-reset erase`
  - Dann die Reset Taste am Board los lassen
  - Macht man das in der Falschen Reihenfolge kommt es zu Fehlermeldungen!
- Anschließend kann man Katapult flashen:
  `st-flash --connect-under-reset write ~/katapult/out/katapult.bin 0x8000000`
  Auch hier wieder die Reihenfolge mit dem Reset Button berücksichtigen!
- Sieht kompliziert aus, ist es aber im Grunde nicht 🙂

Nicht vergessen, die ST-Link Verkabelung wieder zu entfernen!

RP2040

Bei einem RP2040 Board ist die Welt deutlich einfacher 🙂

- Beim `make menuconfig` von Katapult müsst ihr natürlich den RP2040 als Controller einstellen.
- nach dem `make` habt ihr dann ebenfalls im Ordner `out` die Datei `katapult` aber jetzt mit der Endung `uf2`
- Haltet jetzt den Boot Button an eurem Druckerboard gedrückt und verbindet das Board per USB mit dem SBC.
- Startet jetzt im kataput Ordner `make flash`

```
pi@Pi3Test:~/katapult $ make flash
  Flashing out/katapult.uf2
Loaded UF2 image with 21 pages
Found rp2040 device on USB bus 1 address 35
Flashing...
Resetting interface
Locking
Exiting XIP mode
Erasing
Flashing
Rebooting device
```

- Fertig - ja so einfach kanns sein 🙂

Katapult checken

Ihr könnt relativ einfach überprüfen, ob das Flashen von Katapult geklappt hat.

USB

- Am Druckerboard 2x schnell hintereinander den Reset Button drücken.
- Im Terminal `journalctl -kS -5min` aufrufen
- In der Ausgabe solltet ihr dann eine Zeile mit Katapult finden:
  Bsp : `Nov 03 16:33:56 Pi3Test kernel: usb 1-1.4: Manufacturer: katapult`
- Zudem gibt es noch eine Zeile mit “Produkt” wo ihr euren Controller Typ wiederfrinden solltet.

CAN

> **Hinweis**
> Die folgenden Schrotte 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:

```
biqu@BTT-PI12:~/katapult$ ~/klippy-env/bin/python
~/klipper/scripts/canbus_query.py can0
Found canbus_uuid=539892be834d, Application: CanBoot
Total 1 uuids found
```

  - Wird bei diesem Schritt kein Board gefunden, hilft oft ein Reset am Board (entweder über Reset Taster oder 1x Strom weg und wieder dran)
- Klipper Dienst wieder starten
  `sudo systemctl start klipper.service`

Klipper flashen

Klipper kann jetzt über USB bzw. über CAN auf das Druckerboard geflasht werden.

USB

- `cd ~/klipper`

- `make menuconfig`
  - Hier erfolgen die gleichen Einstellungen wie bei Katapult.
  - Der `Bootloader offset` muss so eingestellt werden wie in der Katapult Konfig unter `Application start offset` angegeben. (Beim RP2040 ist der `Bootloader offset` immer 16KiB)
  - Als `Communication interface` muss hier logischerweise die richtige USB Konfig ausgewählt werden (gibt meist eh nur eine)!
- Klipper mittels `make` kompilieren
- Den seriellen Port ermitteln den euer Druckerboard verwendet
  `ls -lR /dev/ | grep -v '\->\s../tty' | grep -v ttyprintk | grep -e 'tty[[:alpha:]]' -e serial`
  Hier bekommt ihr mehrere Zeilen ausgegeben und eine sollte **katapult** oder **klipper** beinhalten (unter `/dev/serial/by-id:`)
  Bsp: `lrwxrwxrwx 1 root root 13 Nov 3 16:44 usb-katapult_rp2040_E66138935F575A28-if00 → ../../ttyACM0`
  Davon interessiert uns der hintere Teil im Moment also `ttyACM0`
- Und jetzt kommt die Magie von Katapult zu Einsatz 🙂
- Flashen geht dann einfach mittels
  `make flash FLASH_DEVICE=/dev/ttyACM0`
  Wobei ihr am Ende halt das ttyACM0 durch das Ergebnis der vorherigen Suche austauschen müsst.
- Das Ergebnis sollte dann so aussehen:

```
Attempting to connect to bootloader
CanBoot Connected
Protocol Version: 1.0.0
Block Size: 64 bytes
Application Start: 0x10004000
MCU type: rp2040
Flashing '/home/pi/klipper/out/klipper.bin'...

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

Write complete: 108 pages
Verifying (block count = 432)...

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

Verification Complete: SHA = AC11C20CD24F671A530404E4BBABBBFE28E69326
CAN Flash Success
```

> **Hinweis**
> In der Original Anleitung soll das Klipper flashen so funktionieren:
> `/usr/bin/python3 ~/katapult/scripts/flashtool.py -d /dev/ttyACM0`
> Das klappt aber nicht, weil der Bootloader nicht aktiviert wird. Also muss man entweder den Katapult Bootloader über 2x schnelles Reset drücken manuell aktivieren oder über dieses Kommando:
> `cd ~/klipper/scripts && /usr/bin/python3 -c 'import flash_usb as u; u.enter_bootloader("/dev/ttyACM1")' ''`

> `Deshalb lieber gleich die ''make flash …` Variante verwenden 😉

CAN

- `cd ~/klipper`
- `make menuconfig`
  - Hier erfolgen die gleichen Einstellungen wie bei Katapult.
  - Der `Bootloader offset` muss so eingestellt werden wie in der Katapult Konfig unter `Application start offset` angegeben. (Beim RP2040 ist der `Bootloader offset` immer 16KiB)
  - Als `Communication interface` muss hier logischerweise die richtige CAN Konfig ausgewählt werden!
- Klipper mittels `make` kompilieren
- 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:

```
biqu@BTT-PI12:~/katapult$ ~/klippy-env/bin/python
~/klipper/scripts/canbus_query.py can0
Found canbus_uuid=539892be834d, Application: CanBoot
Total 1 uuids found
```

  - Wird bei diesem Schritt kein Board gefunden, hilft oft ein Reset am Board (entweder über Reset Taster oder 1x Strom weg und wieder dran)
  - Nicht wundern das hier ggf. `CanBoot` als Applikation gemeldet wird. Das ist ok!
  - Es kann auch sein, dass hier als Applikation `Klipper` angezeigt wird. Das ist ebenfalls ok!
  - Wir notieren und so oder so die UUID. In diesem Fall **539892be834d**
- Klipper flashen
  `~/klippy-env/bin/python3 ~/katapult/scripts/flash_can.py -i can0 -f ~/katapult/out/klipper.bin -u <uuid>`
  In diesem Beispiel wäre das dann
  `~/klippy-env/bin/python3 ~/katapult/scripts/flash_can.py -i can0 -f ~/klipper/out/klipper.bin -u 539892be834d`
- Das Ergebnis:

```
biqu@BTT-PI12:~/katapult/scripts$ ~/klippy-env/bin/python3 ~/katapult/scripts/flash_can.py -i can0 -f ~/klipper/out/klipper.bin -u 539892be834d
Sending bootloader jump command...
Resetting all bootloader node IDs...
Checking for Katapult nodes...
Detected UUID: 539892be834d, Application: Katapult
Attempting to connect to bootloader
Katapult Connected
Protocol Version: 1.0.0
Block Size: 64 bytes
Application Start: 0x8002000
MCU type: stm32f072xb
Verifying canbus connection
Flashing '/home/biqu/klipper/out/klipper.bin'...

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

Write complete: 14 pages
Verifying (block count = 420)...

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

Verification Complete: SHA = 3C6BC41CFD4FB0999C52590C2483F5E0741B96FD
Flash Success
```

- Kleine Überprüfung ob alles geklappt hat
  `~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0`
  liefert als Ergebnis danach dann
  `Found canbus_uuid=539892be834d, Application: **Klipper**`
- Klipper Dienst wieder starten
  `sudo systemctl start klipper.service`

nützliche ST-Link Kommandos

Die folgenden Kommandos dienen nur als Referenz

- Chip erkennen & ST-Link testen
  `st-info --connect-under-reset --probe`
- Firmware flashen
  `st-flash --connect-under-reset write out/katapult.bin 0x8000000`
- Flash löschen
  `st-flash --connect-under-reset erase`
- Flash auslesen
  `st-flash --connect-under-reset read mem.bin 0x8000000 64k`

Links

- Voron CAN Infos
  https://github.com/Esoterical/voron_canbus/tree/main
- Bootloader Entry Infos
  https://github.com/Klipper3d/klipper/blob/master/docs/Bootloader_Entry.md

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

Permanent link:
https://drklipper.de/doku.php?id=klipper_faq:can:66_-_katapult-bootloader&rev=1699176849

Last update: **2023/11/05 10:34**