8.6.2022

MainsailOS - KLIPPER

Installation und Referenz

Müller, Jörg
AUTOMATISCH GENERIERTE (DEEPL) UND Z.T. MANUELLE ÜBERSETZUNG

Inhalt

Vorwort

Dies ist eine z.T. automatisch übersetzte Dokumentation mit Deepl. Ich habe an einigen Stellen ein paar Korrekturen vorgenommen. Dieses Dokument soll eine Hilfe sein, sich in die Problematik von MainsailOs und Klipper einzuarbeiten und es erhebt nicht den Anspruch auf Fehlerfreiheit und 100-ige fachliche Korrektheit . Aus diesem Grunde ist es immer ratsam das Original zu Rate zu ziehen. Die Hyperlinks in diesem Dokument beziehen sich auch auf die englische Ausgabe.

Der erste Teil stammt aus einer Youtube Reihe von Scott Corn, die kurz und knapp die SW-Installation und Einrichtung des Voron beschreibt.

Der zweite Teil ist dann die übersetzte Klipperdokumentation, in der ich die Deltadrucker nicht mit aufgenommen habe, da ich die gesamte Thematik auf den Voron bezogen habe.

Aber trotzdem wünsche ich allen viel Spaß und neue Erkenntnisse im sich Beschäftigen mit dieser Problematik.

J.Müller

OS für Raspberry Pi (Quelle Scott Corn Youtube)

Vorbereitungen

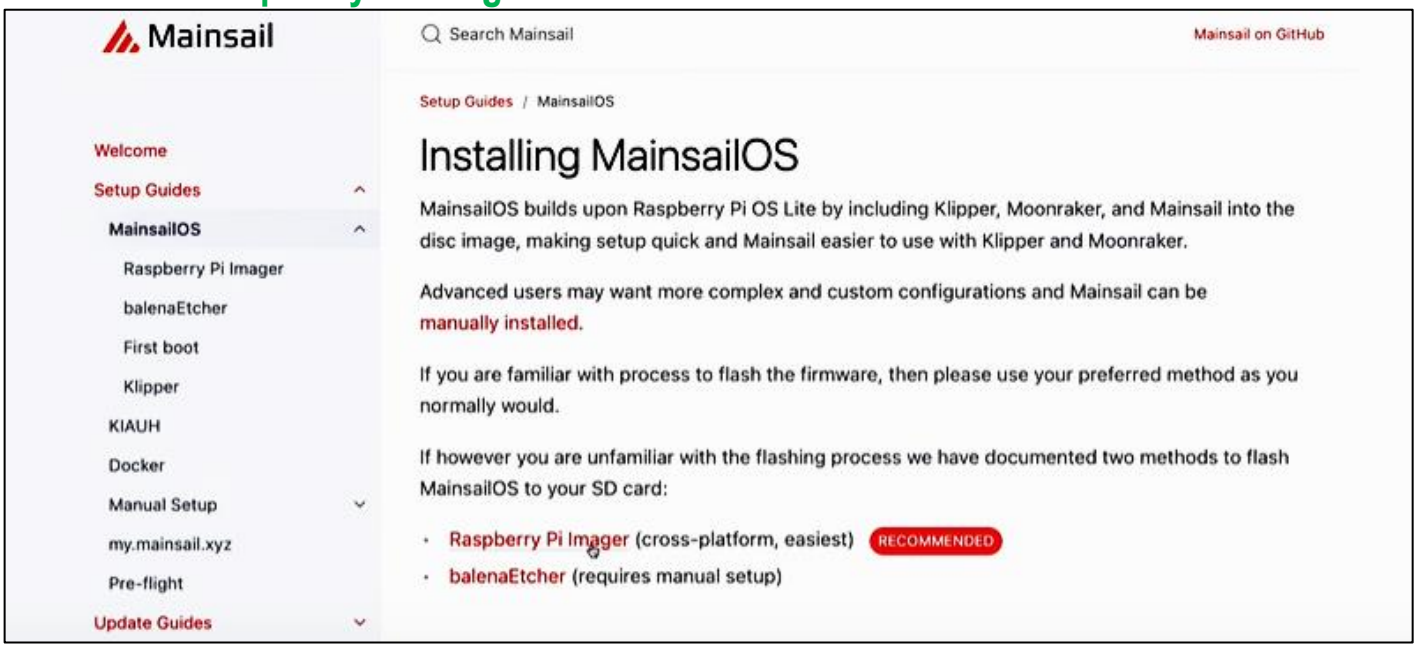
Wir müssen zuerst ein Betriebssystem auf dem Raspberry Pi installieren.
Ich entscheide mich für mainsail.OS. Für mich ist es zu diesem Zeitpunkt 22.3. am einfachsten zu handhaben.

SD-Card-Reader und Mikro-SD-Card

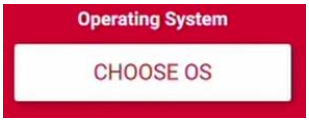
Beginnen Sie mit dem Besuch der MainsailOSweb-Website :

https://docs.mainsail.xyz/setup/mainsail-os
Wir werden den Rasperry Pi Imager verwenden.

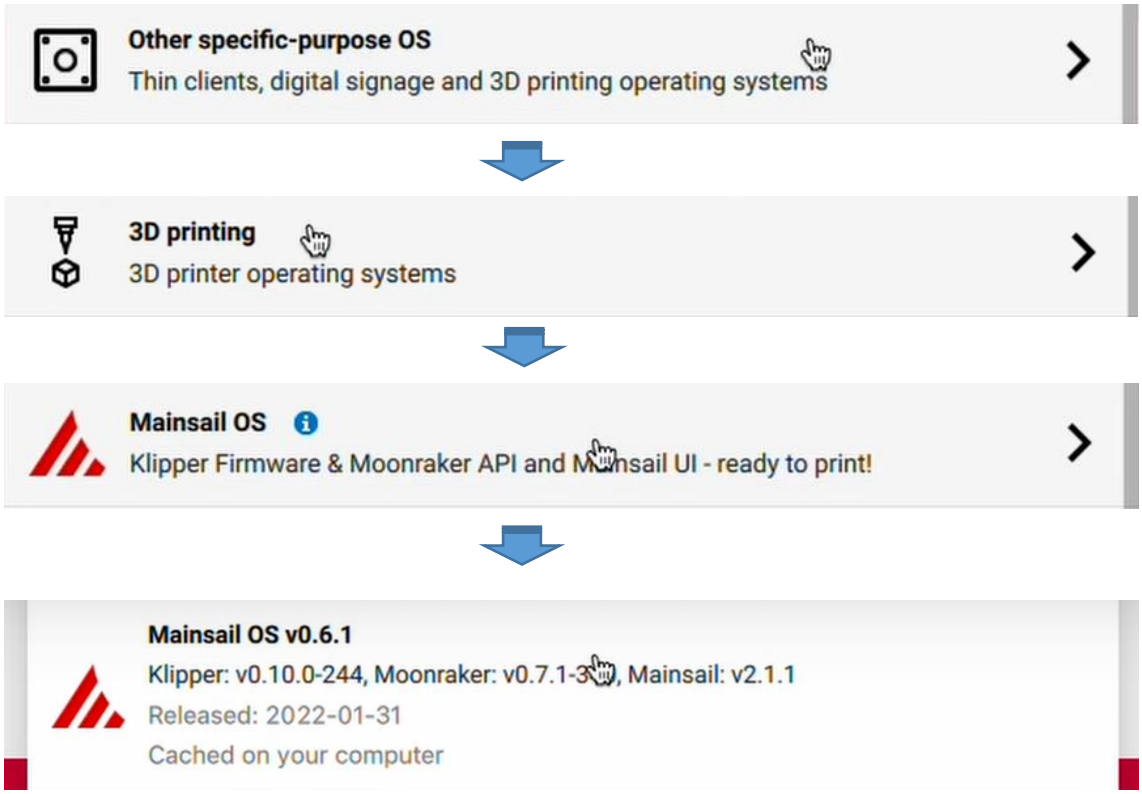
Installation Raspberry Pi Imager

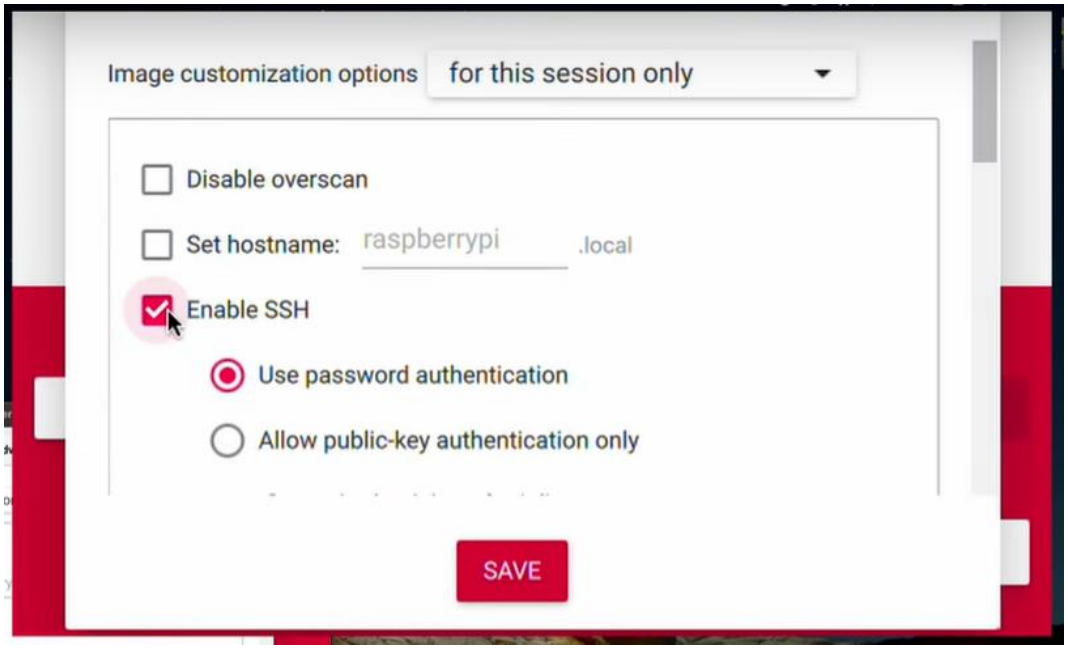


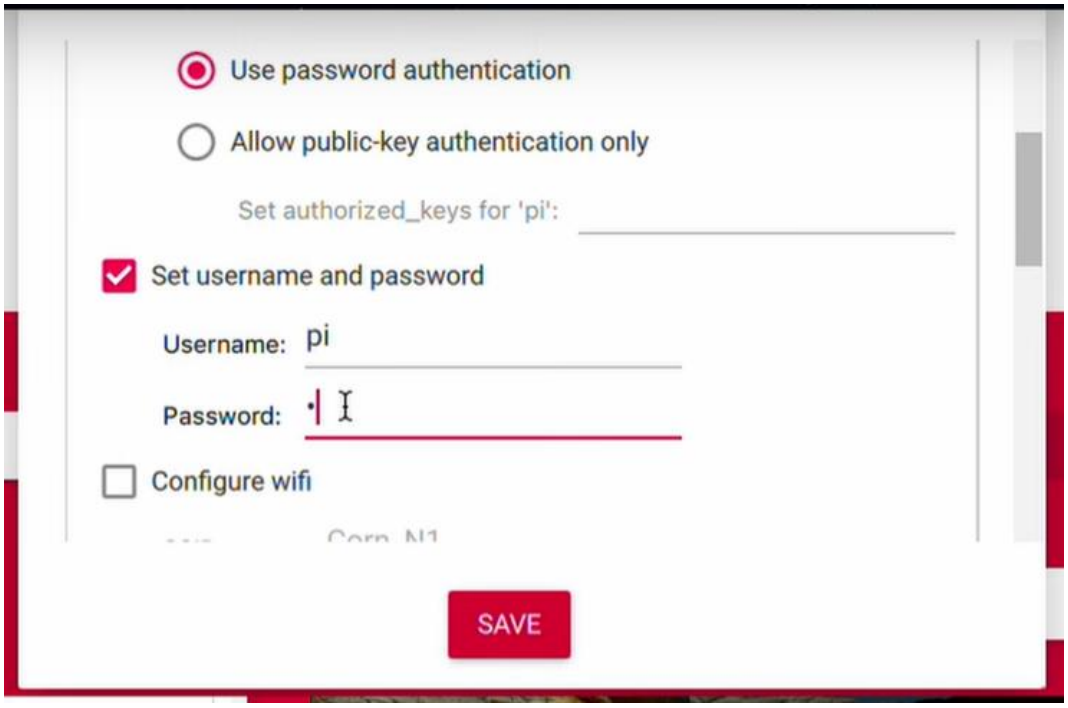
Sobald Sie die Anwendung installiert und ausgeführt haben, sehen Sie Folgendes



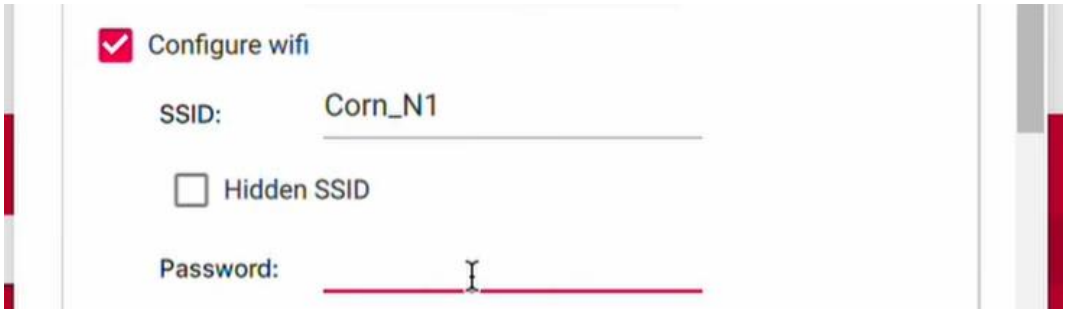
Install Mainsale OS auf der SD-Karte für den Pi



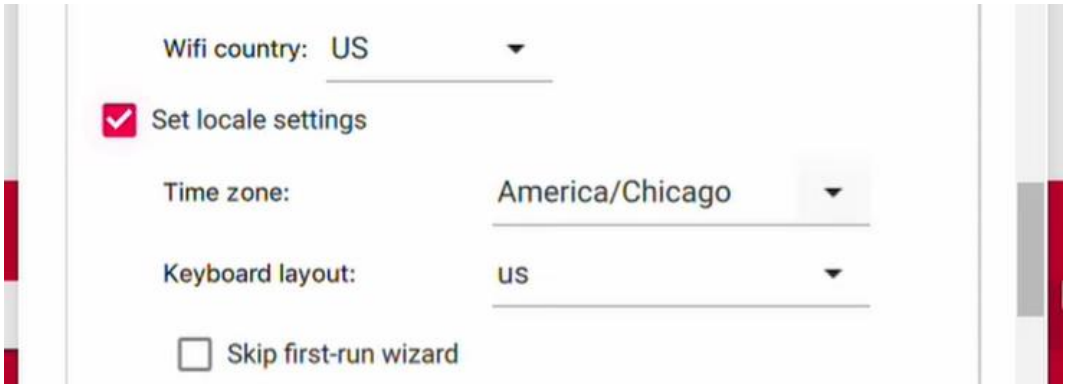




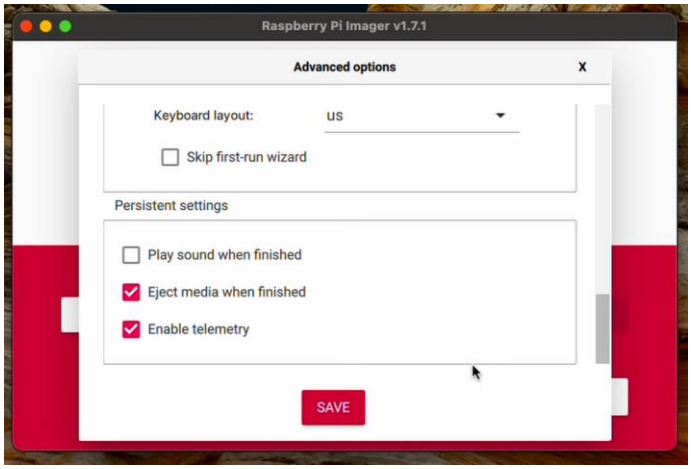
- Benutzernamen belassen

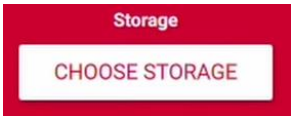


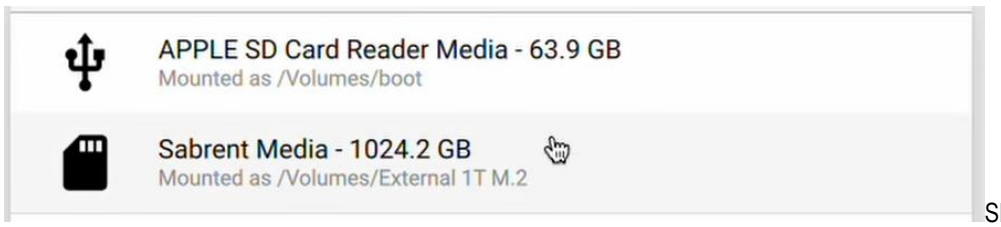
Die Passworteingabe kann manuell auch noch später erfolgen



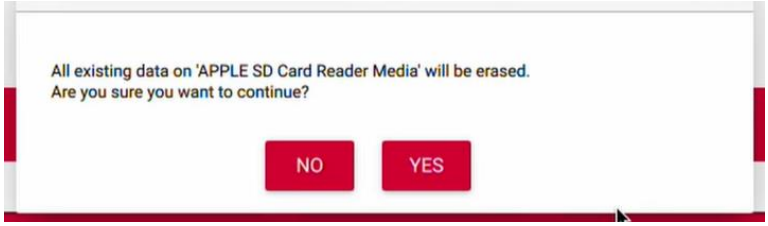
- Eingabe der lokalen Nutzerdaten

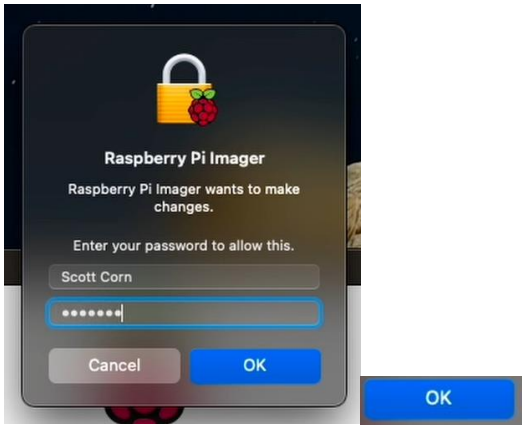




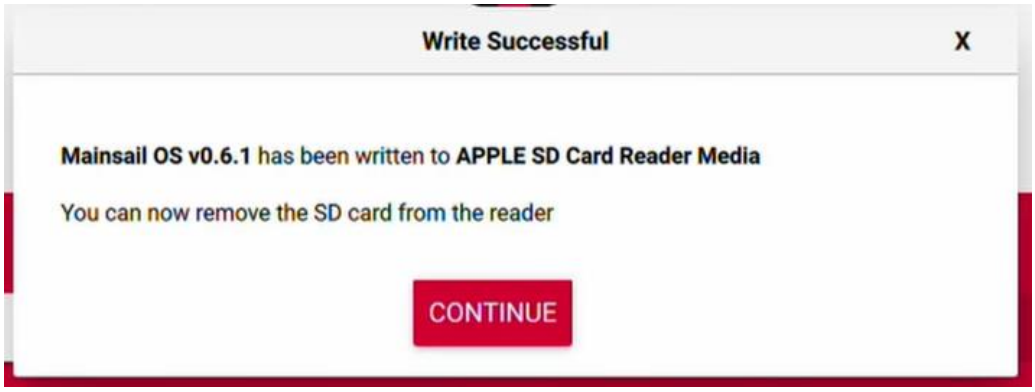


SD-Karte auswählen



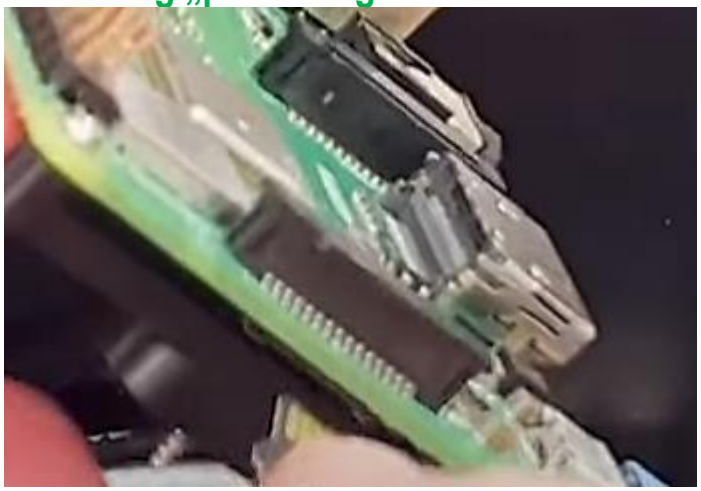


5-10 Minuten

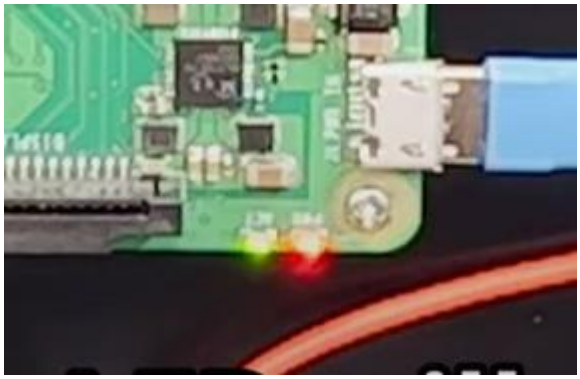


Install Klipper

Erstellung „printer.cfg“

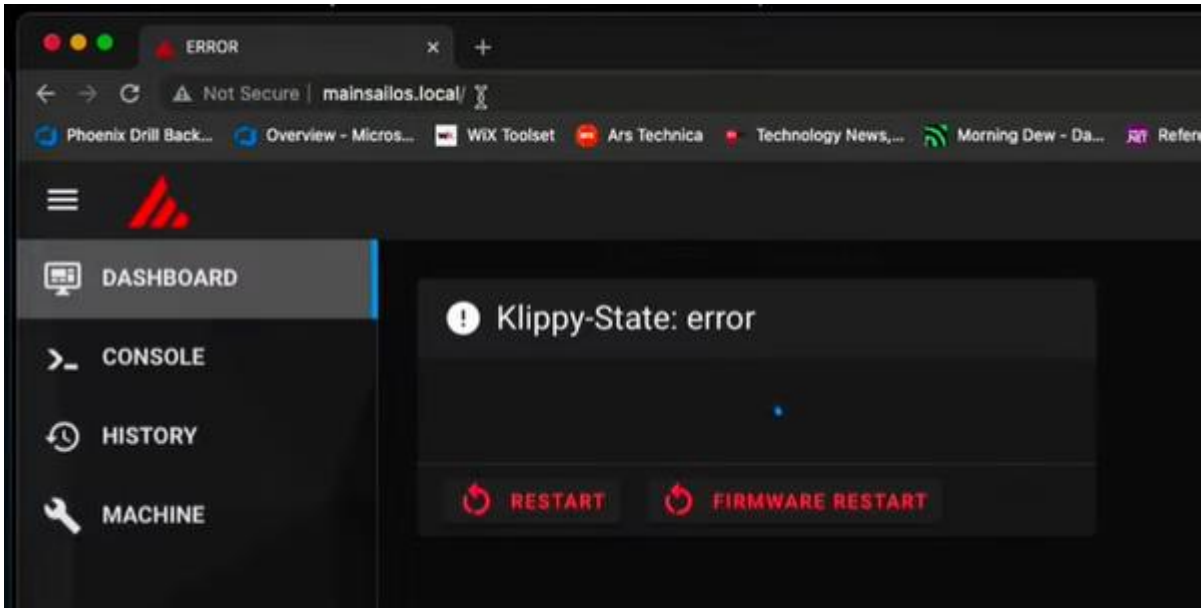


- Einführen der SD-Karte in den Pi
- Drucker einschalten



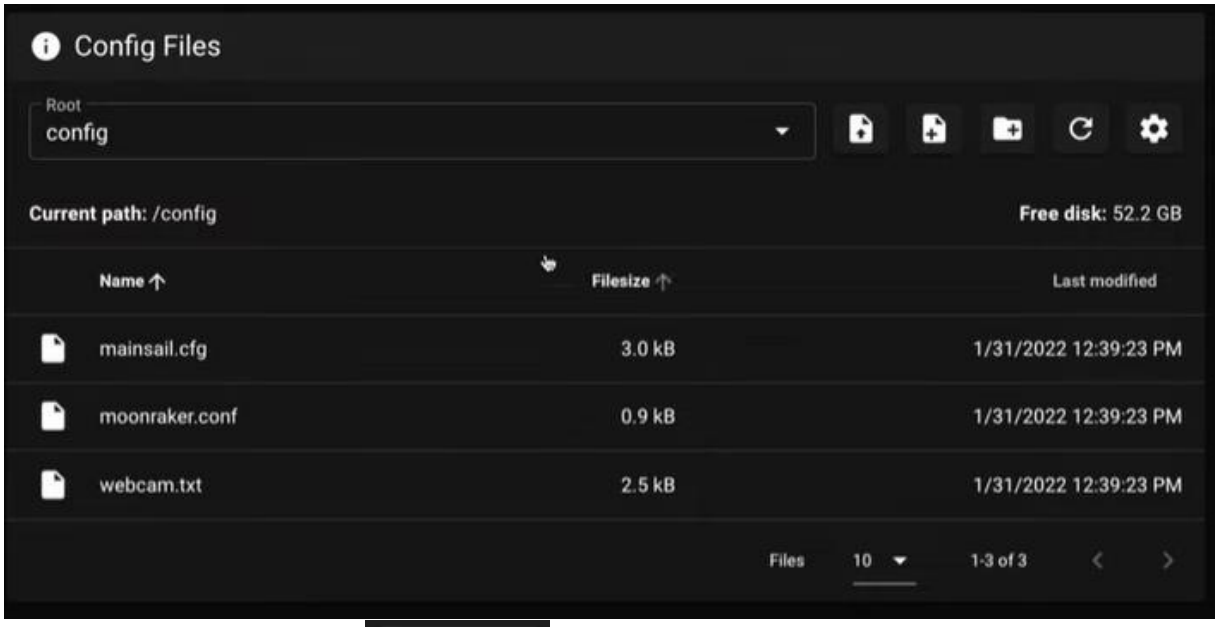
Die grüne LED beginnt zu blinken

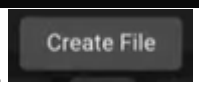
- Warten bis das Blinken langsamer wird oder aufhört
- Öffnen des Browser auf dem PC, der sich im gleichen Netzwerk wie der Drucker befindet
- URL eingeben : http://mainsailos.local

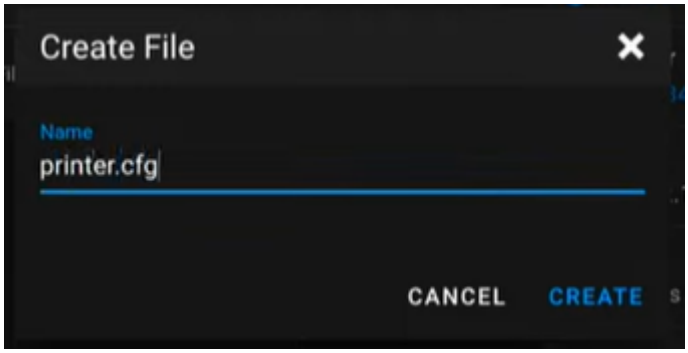


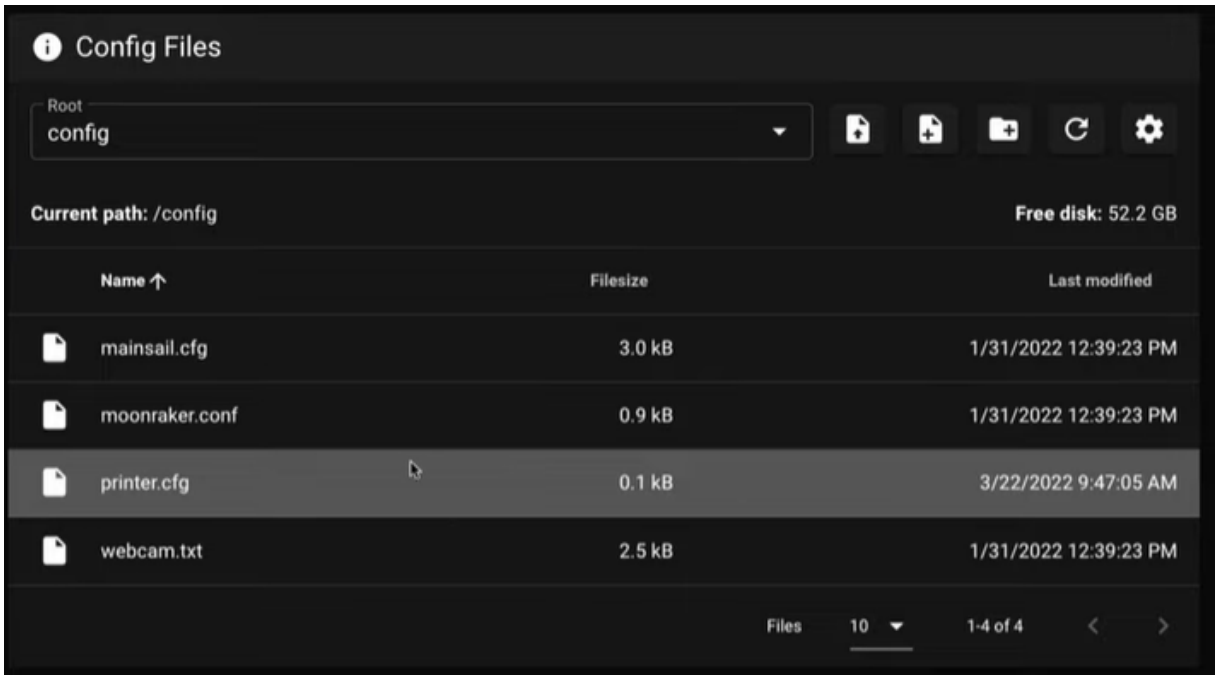
Die Verbindung zum lokalen Netzwerk steht

- Eine erwartete Fehlermeldung taucht auf. Dazu ist eine Start-„printer.cfg“-Datei zu erstellen. 

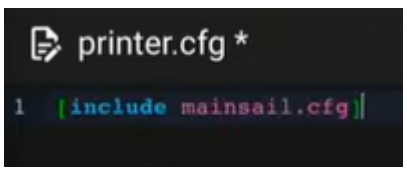


Eine Auflistung der „printer.cfg“ fehlt. 

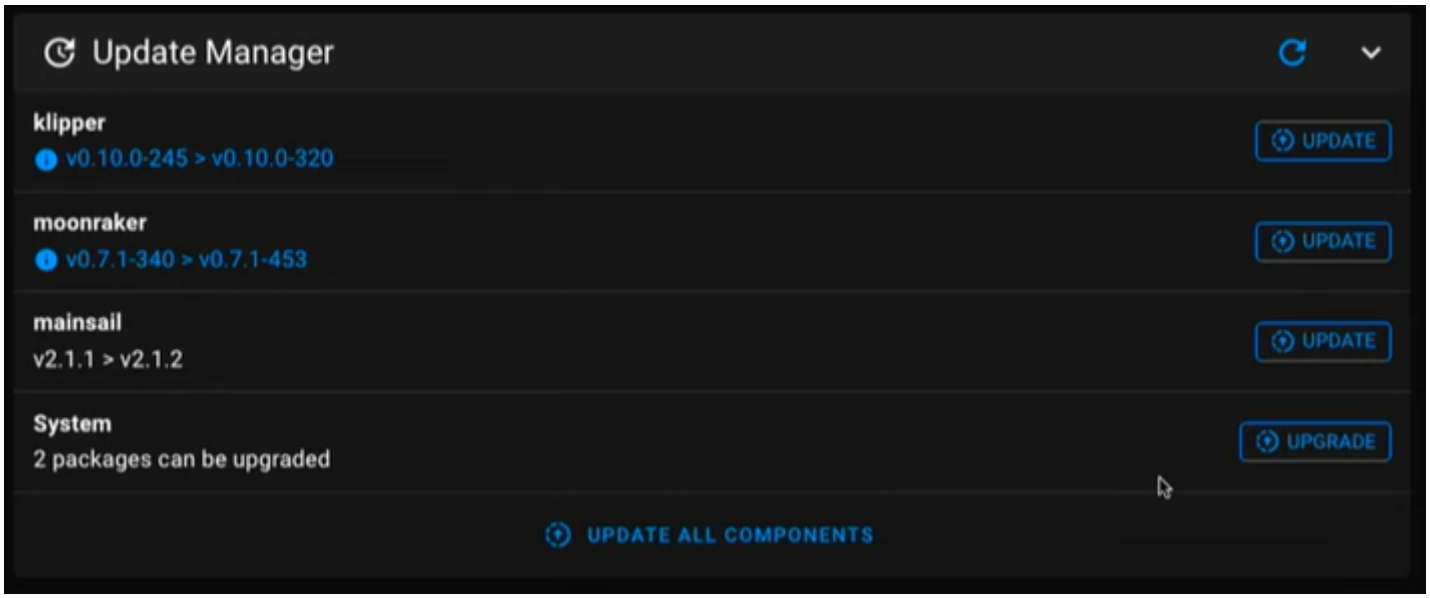


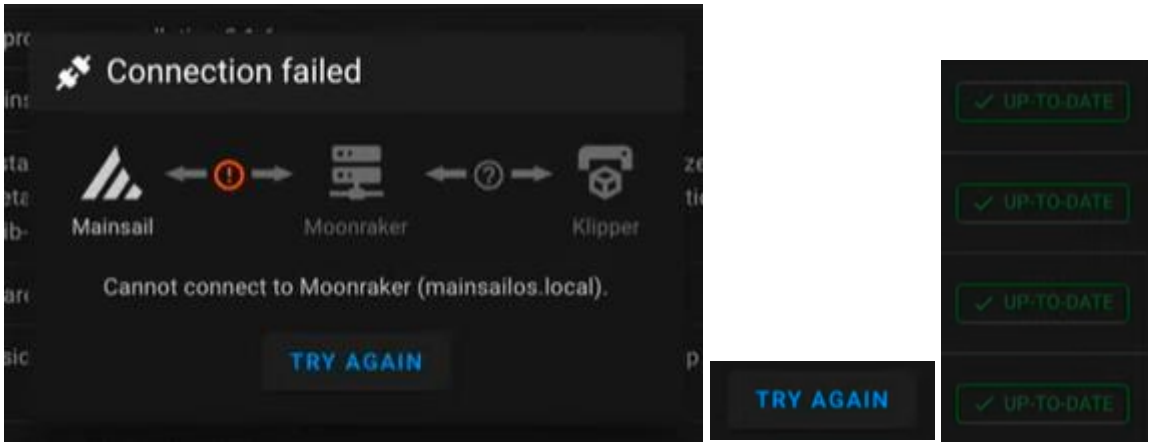


- `include mainsail.cfg`



Der „config-Fehler“ tauch nicht mehr auf, aber eine andere erwartete Fehlermeldung erscheint. Klipper benötigt ein Update. Dazu sind die entsprechenden Update-Buttons nacheinander nach Beendigung des vorangegangenen Updates anzuklicken.





Installation der Firmware

- Öffnen der Textconsole
- `ping mainsailos.local -4` zum Ermitteln der IP-Adresse
- Wenn Windows ein Problem mit dem aufrufen der „.local“-Adresse hat, dann ist eine Direktverbindung notwendig.

```
Command Prompt
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\scott>ping mainsailos.local -4

Pinging mainsailos.local [192.168.86.7] with 32 bytes of data:
Reply from 192.168.86.7: bytes=32 time=71ms TTL=64
Reply from 192.168.86.7: bytes=32 time=2ms TTL=64
Reply from 192.168.86.7: bytes=32 time=2ms TTL=64
Reply from 192.168.86.7: bytes=32 time=2ms TTL=64

Ping statistics for 192.168.86.7:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 71ms, Average = 19ms
```

- `C:\Users\scott>ssh pi@192.168.86.7`
- `pi@192.168.86.7's password:`

Eingeben des Passwortes, das bei der Erstellung der SD-Karte festgelegt wurde

```
pi@192.168.86.7's password:
Linux mainsailos 5.10.103-v7+ #1529 SMP Tue Mar 8 12:21:37 GMT 2022 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Mar 22 19:24:18 2022 from 192.168.86.92
pi@mainsailos:~ $ sudo apt install make
```

- `sudo apt install make`

```
pi@mainsailos:~ $ sudo apt install make

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for pi:
```

- Passwort eingeben

Es erfolgen die Updates der „make utilities“

```
[sudo] password for pi:
Reading package lists... Done
Building dependency tree
Reading state information... Done
make is already the newest version (4.2.1-1.2).
make set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@mainsailos:~ $ cd ~/klipper
```

- `cd ~/klipper`

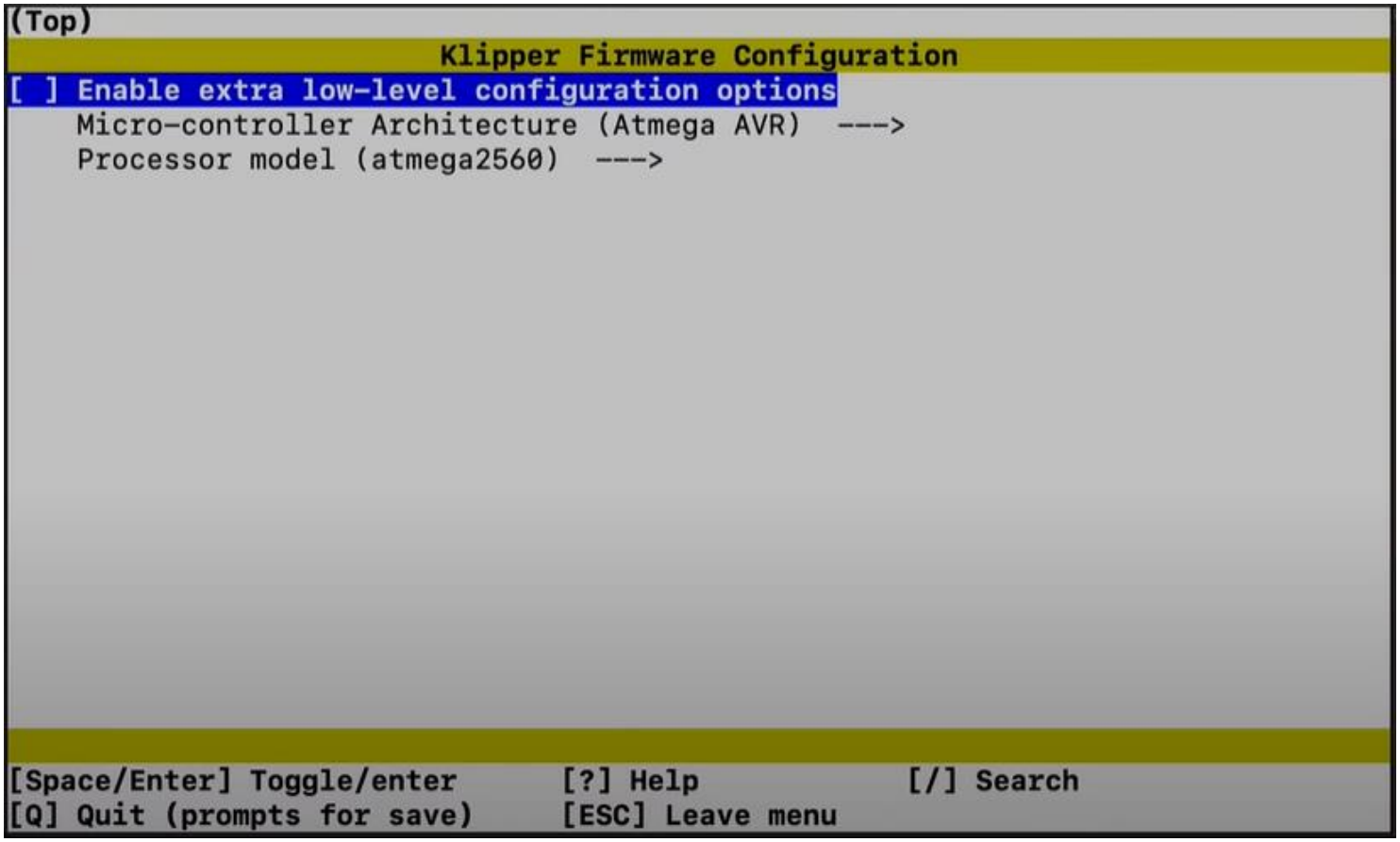
```
pi@mainsailos:~ $ cd ~/klipper
pi@mainsailos:~/klipper $ make clean
```

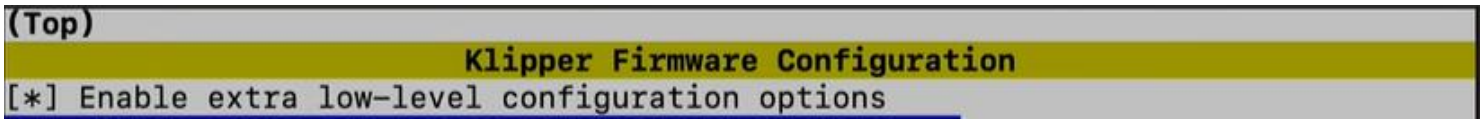
- `make clean`

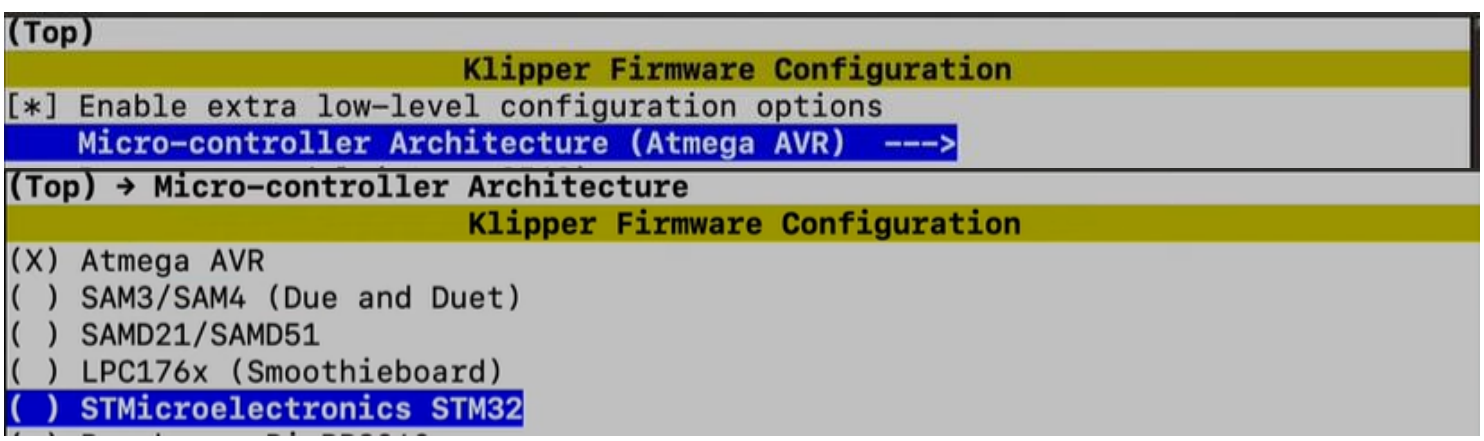
```
pi@mainsailos:~/klipper $ make clean
Using default symbol values (no '/home/pi/klipper/.config')
Configuration saved to '/home/pi/klipper/.config'
  Creating symbolic link out/board
pi@mainsailos:~/klipper $ make menuconfig
```

- `make menuconfig`

Es erscheint ein Fenster mit der Firmwarekonfiguration







```
(Top)
                    Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F103)  --->
```

- Entsprechendes Modell auswählen

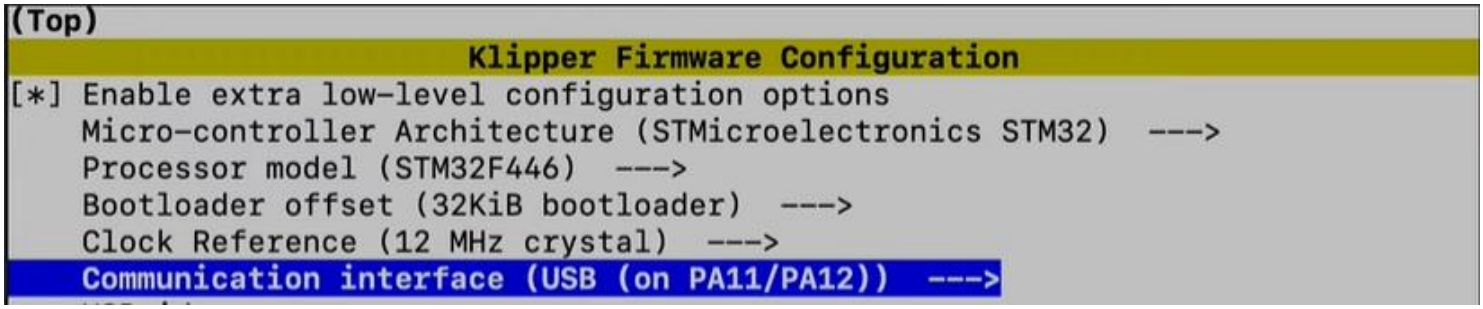
```
(Top) → Processor model
                    Klipper Firmware Configuration
(X) STM32F103
( ) STM32F207
( ) STM32F401
( ) STM32F405
( ) STM32F407
( ) STM32F429
( ) STM32F446
```

```
(Top)
                    Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F446)  --->
    Bootloader offset (32KiB bootloader)  --->
```

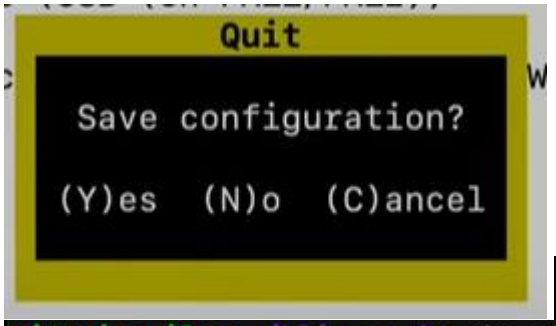
Kann belassen werden.

```
(Top)
                    Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F446)  --->
    Bootloader offset (32KiB bootloader)  --->
    Clock Reference (8 MHz crystal)  --->
```

```
(Top) → Clock Reference
                    Klipper Firmware Configuration
(X) 8 MHz crystal
( ) 12 MHz crystal
```



Kann belassen werden




```
pi@mainsailos:~/klipper $ make menuconfig
  Creating symbolic link out/board
Loaded configuration '/home/pi/klipper/.config'
Configuration saved to '/home/pi/klipper/.config'
pi@mainsailos:~/klipper $ make
```

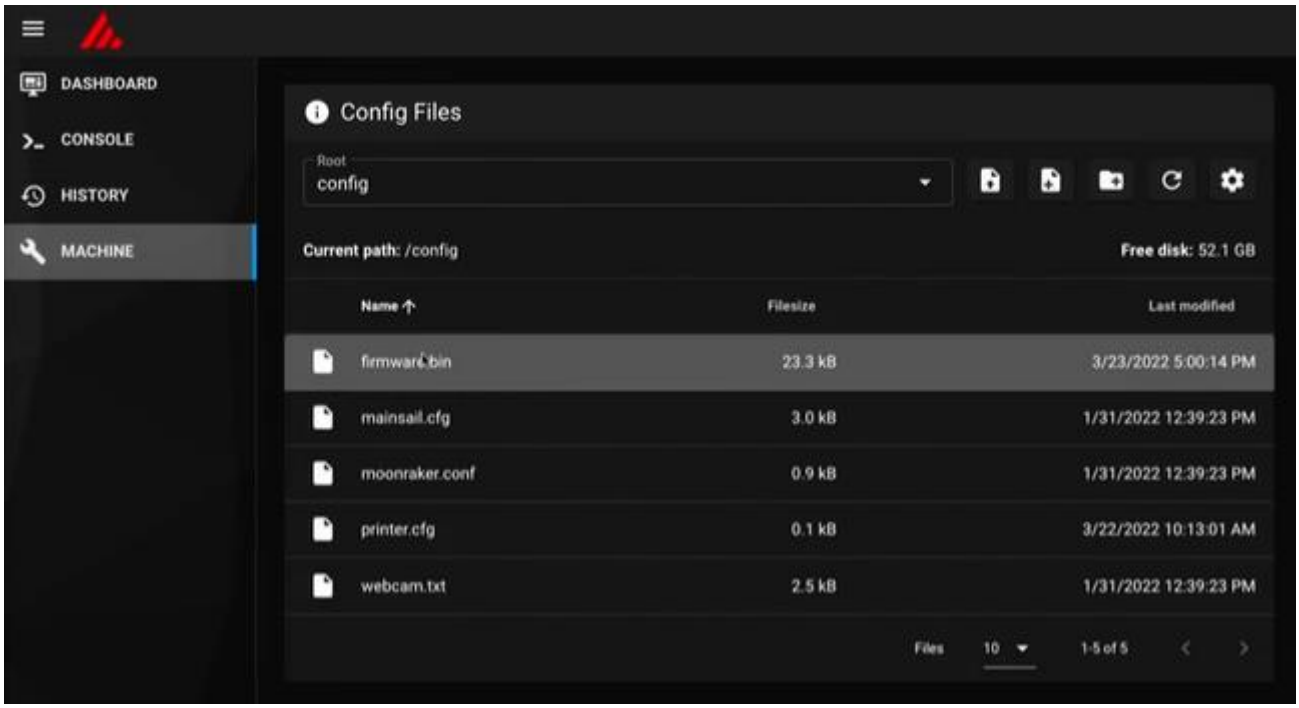
- Make

Die Firmware wird nun erstellt.

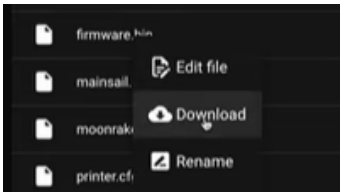
Die entsprechende Datei wird in das entsprechende Verzeichniss kopiert und umbenannt.

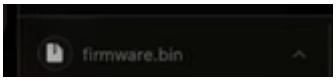
```
Version: v0.10.0-320-g7e654aed
  Preprocessing out/src/generic/armcm_link.ld
  Linking out/klipper.elf
  Creating hex file out/klipper.bin
pi@mainsailos:~/klipper $ cp out/klipper.bin ../klipper_config/firmware.bin
```

- cp out/klipper.bin ../klipper_config/firmware.bin

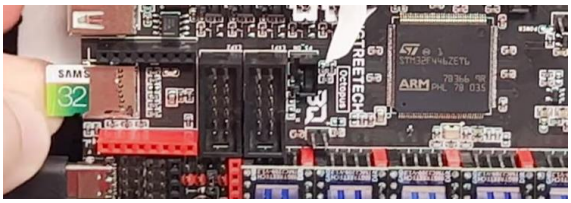


Die Firmwaredatei erscheint in der Auflistung

- Download in Mainsale 



in den persönlichen Downloadordner
- Kopieren der Datei auf eine FAT32-formatierte SD-Karte
- Drucker ausschalten
- Einstecken dieser Karte in das Octopus-Board



- Drucker einschalten

Nach einen kurzen Augenblick sollte das Board geflasht werden

```
pi@mainsailos:~ $ ls /dev/serial/by-id
```

- `ls /dev/serial/by-id`

```
pi@mainsailos:~ $ ls /dev/serial/by-id
usb-Klipper_stm32f446xx_33002F000750534E4E313020-if00
```

Der Flashvorgang war erfolgreich. Wenn keine Seriennummer erscheint, war der Flashvorgang nicht erfolgreich.
usb-STMicroelectronics_MARLIN_BIGTREE_OCTOPUS_V1_CDC_in_FS_Mode_397C38753430-if00
usb-Klipper_stm32f446xx_430049000A51303432383339-if00

- Kopieren der Seriennummer für spätere Verwendung

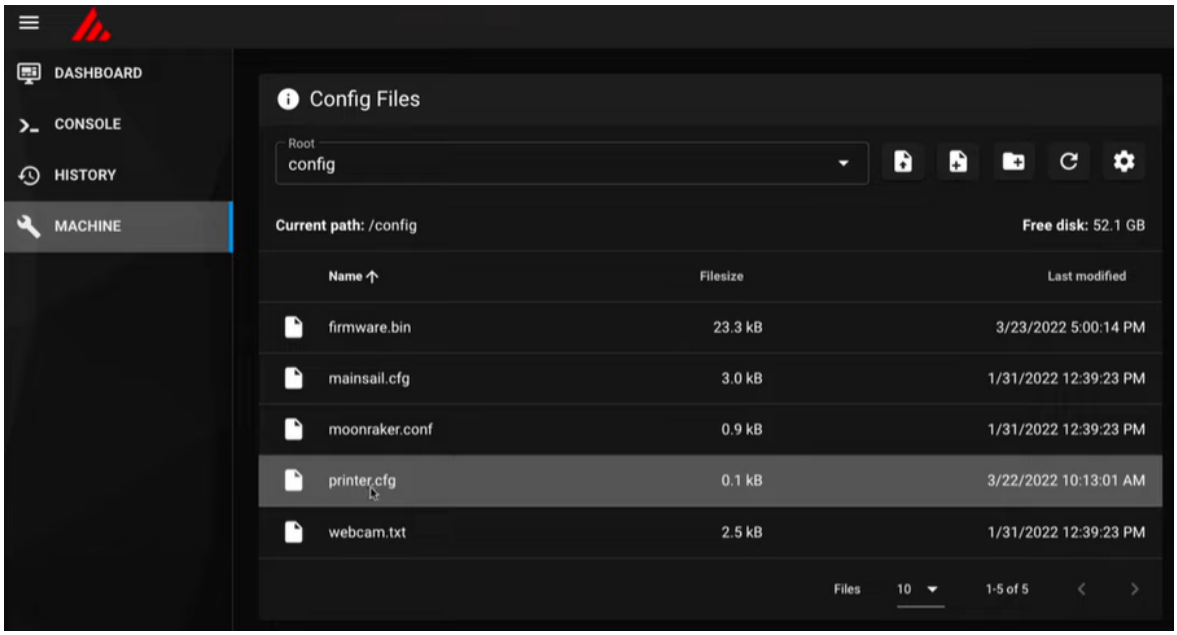
```
pi@mainsailos:~ $ ls /dev/serial/by-id
usb-Klipper_stm32f446xx_33002F000750534E4E313020-if00
pi@mainsailos:~ $ exit
```

- Drucker ausschalten
- Entfernen der SD-Karte

Druckerkonfiguration

Der Drucker ist nun für eine Konfiguration bereit

- Drucker einschalten und mainsailos.local öffnen



Anpassen der printer.cfg

- Printer.cfg öffnen

- Einträge in der „printer.cfg“ löschen
- URL : https://docs.vorondesign.com/build/software/configuration.html
- „V2 Octopus“ auswählen und es erschein die benötigte Konfigurationsdatei
- Alles auswählen und kopieren
- Kopierten Inhalt in Datei einfügen

Einige Einträge müssen geändert werden.

```
25 [mcu]
26 ##  Obtain definition by "ls -l /dev/serial/by-id/" then unplug to verify
27 ##--------------------------------------------------------------------
28 serial: /dev/serial/by-id/{REPLACE WITH YOUR SERIAL}
29 restart_method: command
30 ##--------------------------------------------------------------------
```

- Einfügen der Seriennummer

```
28 serial: /dev/serial/by-id/usb-Klipper_stm32f446xx_33002F000750534E4E313020-if00
```

```
25 [mcu]
26 ##  Obtain definition by "ls -l /dev/serial/by-id/" then unplug to verify
27 ##--------------------------------------------------------------------
28 serial: /dev/serial/by-id/usb-Klipper_stm32f446xx_33002F000750534E4E313020-if00
29 restart_method: command
30 ##--------------------------------------------------------------------
31
32 [include mainsail.cfg]
```

- [Include mainsail.cfg]

```
40 ########################################################################
41 #   X/Y Stepper Settings
42 ########################################################################
```

```
62 ##  Uncomment for 300mm build
63 #position_endstop: 300
64 #position_max: 300
```

- Für die zutreffende Druckergröße „#“ entfernen (Aufheben des Kommentarstatus)

```
75 ##  Make sure to update below for your relevant driver (2208 or 2209)
76 [tmc2209 stepper_x]
```

- Überprüfen,ob der richtige Steppertreiber für X ausgewählt ist.

```
101 ##  Uncomment for 300mm build
102 #position_endstop: 300
103 #position_max: 300
```

- Für die zutreffende Druckergröße „#“ entfernen (Aufheben des Kommentarstatus)

```
114 ##  Make sure to update below for your relevant driver (2208 or 2209)
115 [tmc2209 stepper_y]
```

- Überprüfen,ob der richtige Steppertreiber für Y ausgewählt ist.

```
147 ##  Uncomment below for 300mm build
148 #position_max: 280
```

- Für die zutreffende Druckergröße „#“ entfernen (Aufheben des Kommentarstatus)

```
159 ##  Make sure to update below for your relevant driver (2208 or 2209)
160 [tmc2209 stepper_z]
```

- Überprüfen,ob der richtige Steppertreiber für Z ausgewählt ist.

```
177  ##  Make sure to update below for your relevant driver (2208 or 2209)
178  [tmc2209 stepper_z1]
```

- Überprüfen,ob der richtige Steppertreiber für Z1 ausgewählt ist.

```
195  ##  Make sure to update below for your relevant driver (2208 or 2209)
196  [tmc2209 stepper_z2]
```

- Überprüfen,ob der richtige Steppertreiber für Z2 ausgewählt ist.

```
213  ##  Make sure to update below for your relevant driver (2208 or 2209)
214  [tmc2209 stepper_z3]
```

- Überprüfen,ob der richtige Steppertreiber für Z3 ausgewählt ist.

```
222  #####################################################################
223  #   Extruder
224  #####################################################################
247  ##  Validate the following thermistor type to make sure it is correct
248  ##  See https://www.klipper3d.org/Config_Reference.html#common-thermistors for additional options
249  sensor_type: Generic 3950
```

- „#“ entfernen (Aufheben des Kommentarstatus) und den betreffenden Thermistor eintragen

```
278  [heater_bed]
279  ##  SSR Pin - HE1
280  ##  Thermistor - TB
281  ##  Uncomment the following line if using the default SSR wiring from the docs site
282  heater_pin: PA3
283  ##  Other wiring guides may use BED_OUT to control the SSR. Uncomment the following line for those cases
284  #heater_pin: PA1
285  ##  Validate the following thermistor type to make sure it is correct
286  ##  See https://www.klipper3d.org/Config_Reference.html#common-thermistors for additional options
287  #sensor_type: Generic 3950
```

- „#“ entfernen (Aufheben des Kommentarstatus) bei `heater_pin:PA3` und `sensor_type:` den betreffenden Thermistor eintragen

```
302  [probe]
303  ##  Inductive Probe
304  ##  This probe is not used for Z height, only Quad Gantry Leveling
305
306  ## Select the probe port by type:
307  ## For the PROBE port. Will not work with Diode. May need pull-up resistor from signal to 24V.
308  #pin: ~!PB7
```

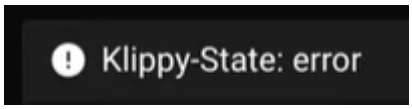
- „#“ entfernen (Aufheben des Kommentarstatus) bei `pin: ~!PB7`

```
355  #[heater_fan exhaust_fan]
356  ##  Exhaust fan - FAN3
357  #pin: PD13
358  #max_power: 1.0
359  #shutdown_speed: 0.0
360  #kick_start_time: 5.0
361  #heater: heater_bed
362  #heater_temp: 60
363  #fan_speed: 1.0
```

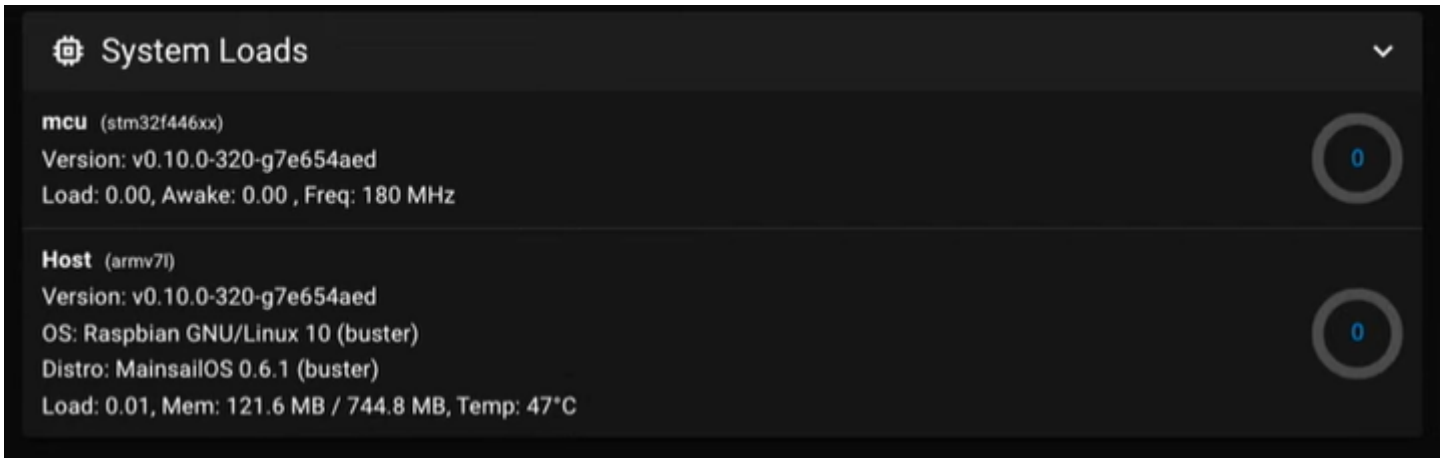
- „#“ entfernen (Aufheben des Kommentarstatus)

```
412  ##  Gantry Corners for 300mm Build
413  ##  Uncomment for 300mm build
414  #gantry_corners:
415  #   -60,-10
416  #   360,370
417  ##  Probe points
418  #points:
419  #   50,25
420  #   50,225
421  #   250,225
422  #   250,25
```

- „#“ entfernen (Aufheben des Kommentarstatus)
- 

Nach dem Restart sollte die Fehlermeldung 

nicht mehr auftauchen

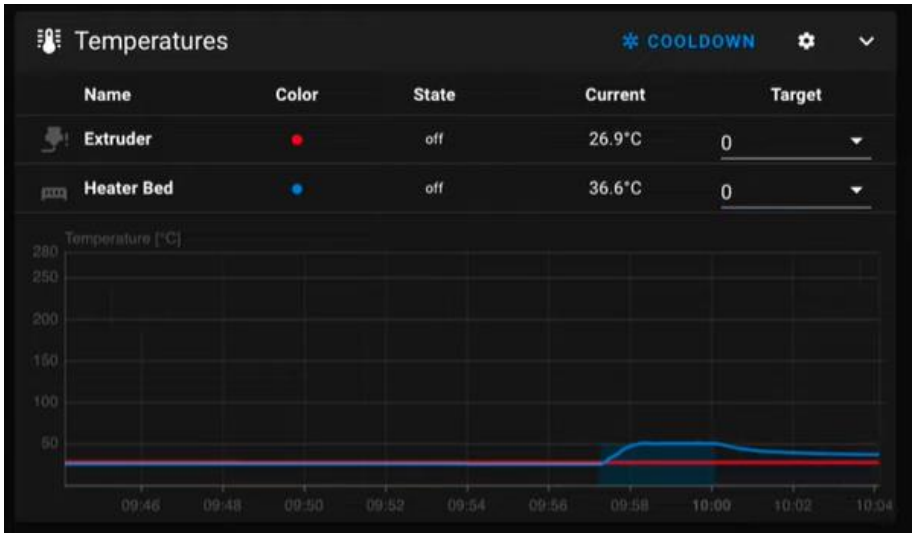


Erste Tests

Dashboard

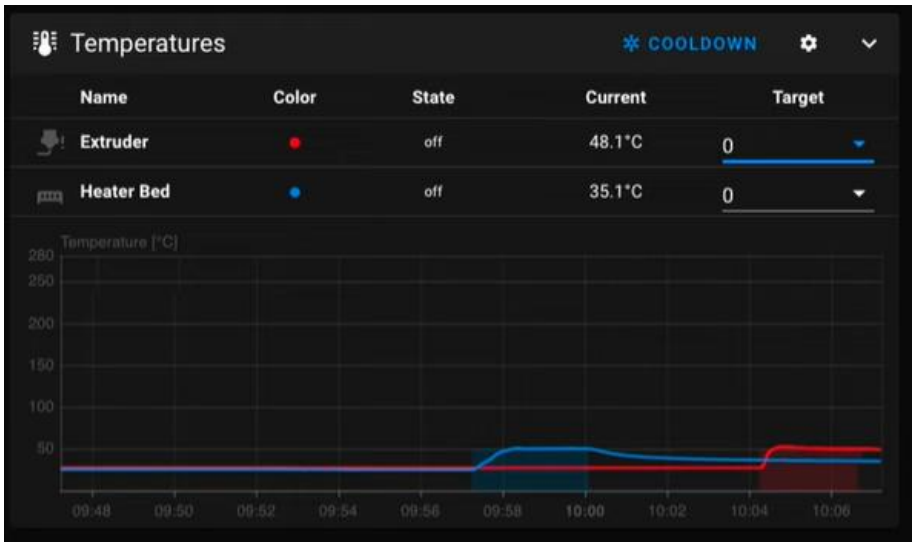
Kontrollieren physisch (Nozzle und Plate) und Dashboard (Tempereaturkurven = 0), dass Hotend und Buildplate sich nicht aufheizen

Heater Bed



- Auf 50°C (Heater Bed) und dann auf 0 stellen

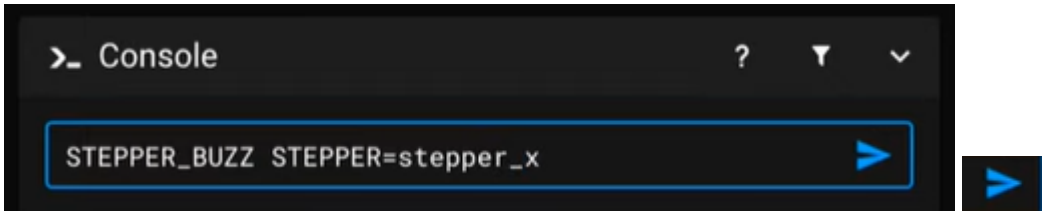
Extruder



- Auf 50°C (Extruder) und dann auf 0 stellen, dabei prüfen, ob der Hotendlüfter startet

Console

Stepperfunktionstest für X, Y, Z

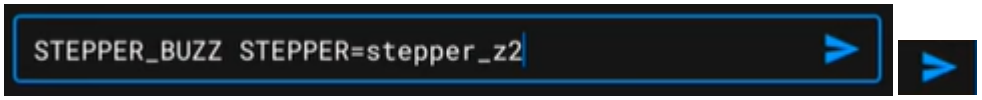


Verfahrweg 1mm 10x hin und her

Für den Z0-Motor vorn lks

Für den Z1-Motor hinten lks

Für den Z2-Motor hinten re



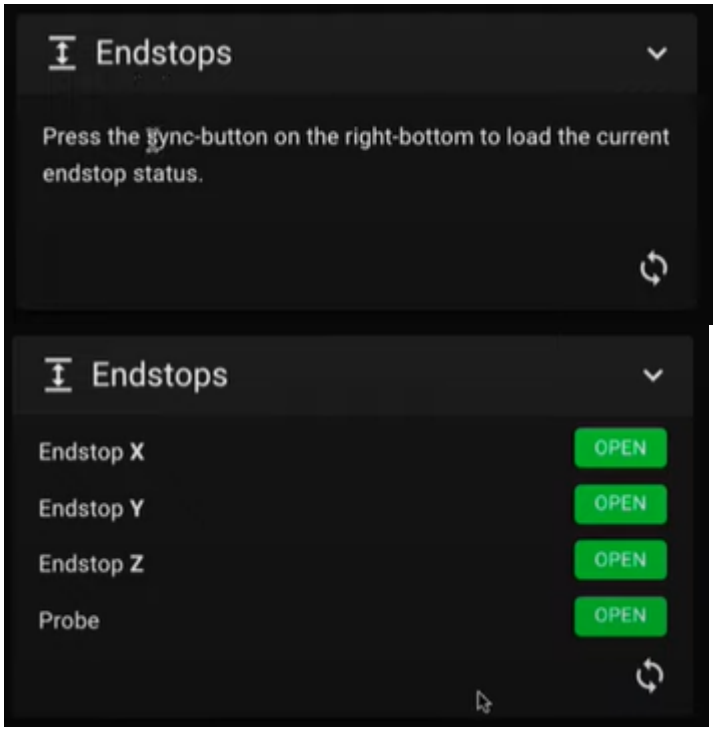
Für den Z-Motor vorn re

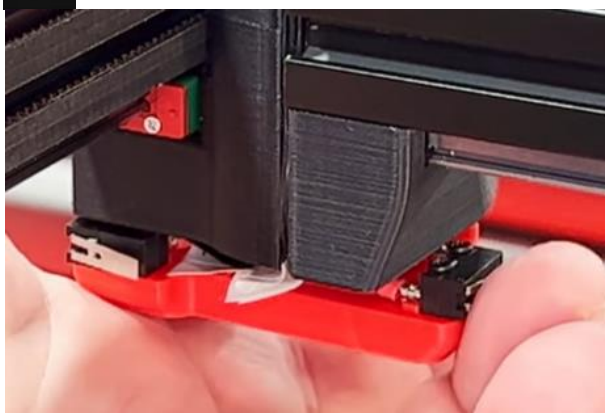
Extruder

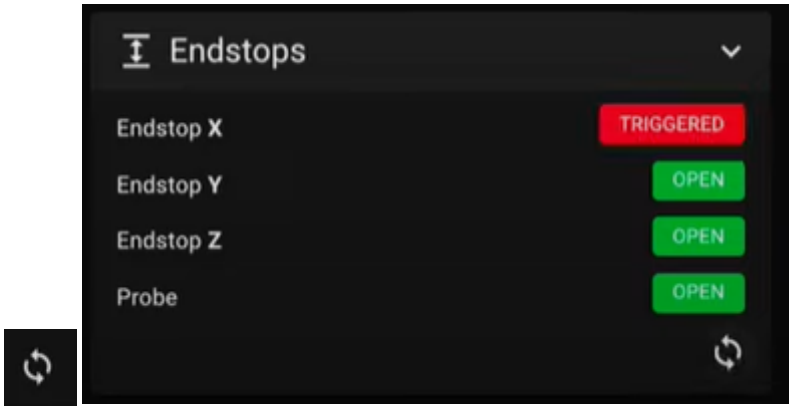
Für den Extruder-Motor
Bei Fehlern, die Pin-Anschlüsse prüfen

X- und Y-Endstop

- Druckkopf ins Zentrum der Buildplate positionieren





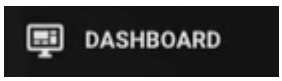


Endstop freigeben

- Für Y-Endstop wiederholen

Bei Fehlern, die Pin-Anschlüsse prüfen

X-Y-Homing

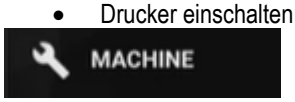


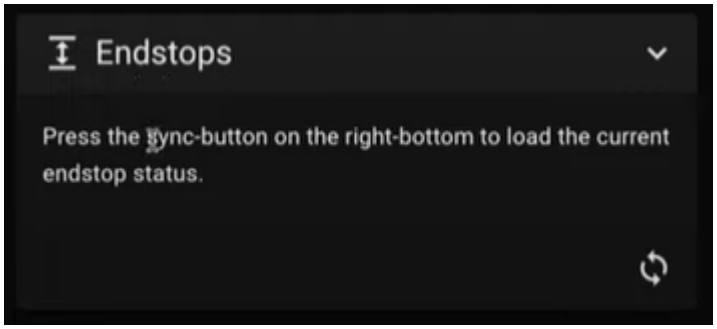
Bei Notfallabschaltung

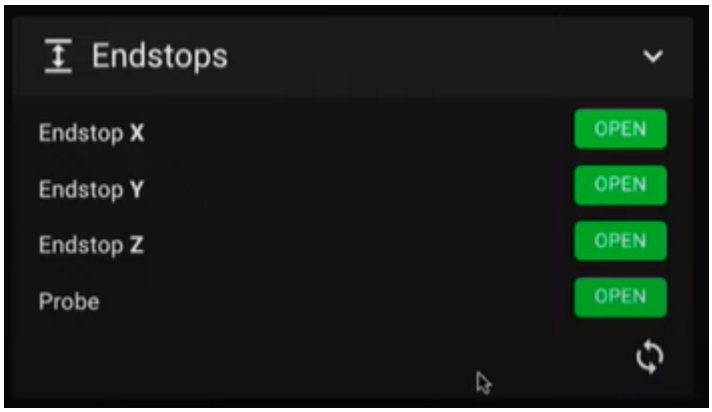
 MainsaiOS beenden und Drucker ausschalten. Druckkopf in Position belassen.

Z-Homing

- Gantry nach unten drücken
- Z-Endstopp und Builtplate so ausrichten, dass Nozzle und Stift zentriert sind.
- Drucker einschalten





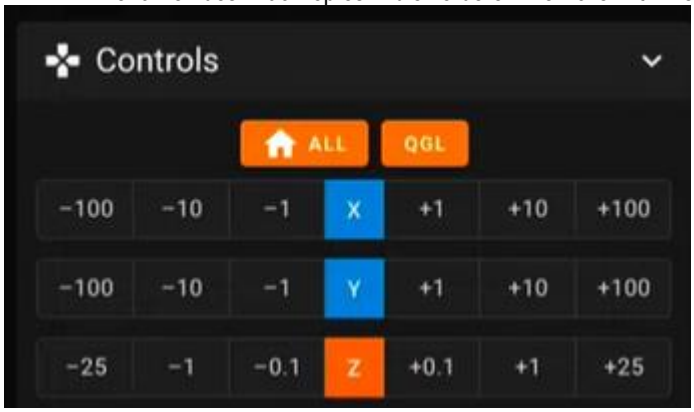


- Drücken des Endschalters ➔ 

anschließend 

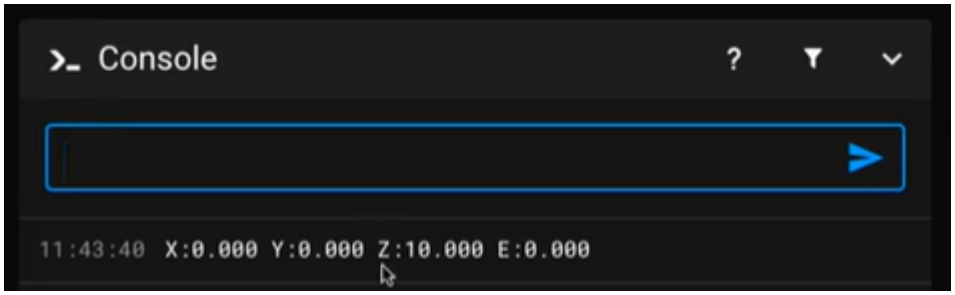
Nullposition einstellen

- Verfahren des Druckkopfes in die vordere linke Ecke mit Hilfe des Control-Panels



Zeigt die aktuelle Position an

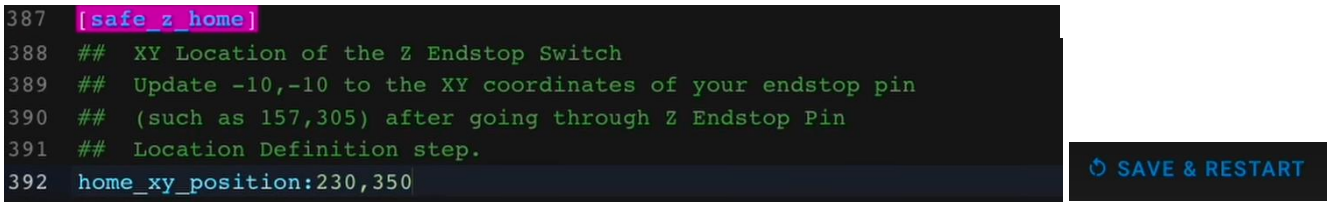
- Verfahren des Druckkopfes in die 0,0-Position



Positionierung genau auf die Ecke, eventuelle geringfügige Änderungen an der Position beim Tuning

X-Y-Position für den Z-Endstop

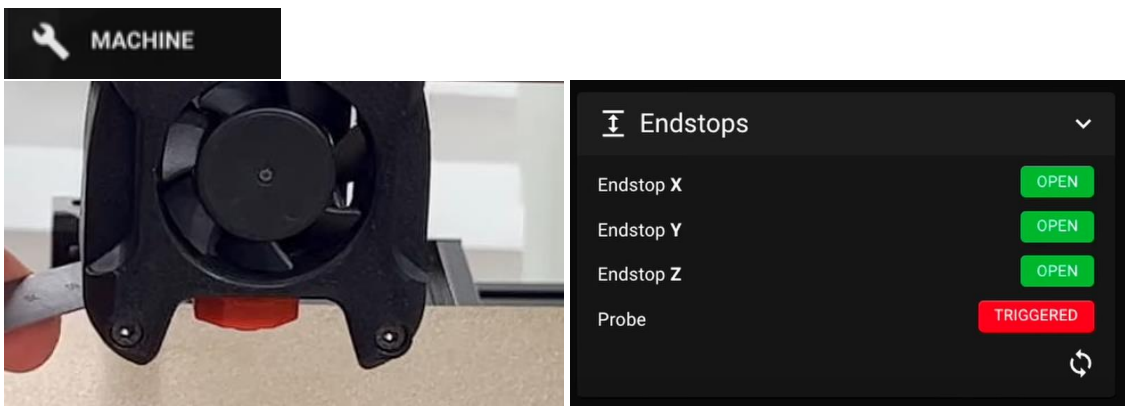
- `G28 x y` in der Console
- Positionieren des Druckkopfes direkt über dem Z-Endschalter im Controlpanel
- `M114` in der Console für die später einzutragende X-Y-Position des Z-Endschalters (Position merken)



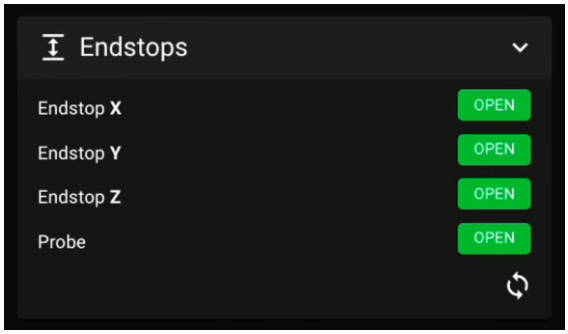
- Eintragen der vermerkten X-Y-Position

- Sichtprüfung am Drucker

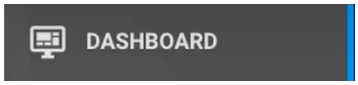
Induktive Probe

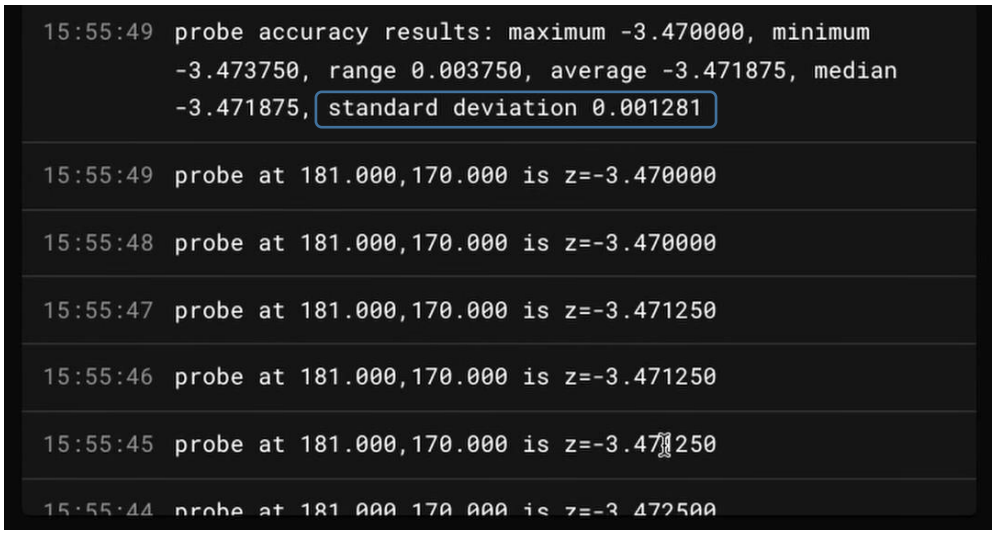


- Metall unter die Sonde platzieren und refresh im Menü durchführen. Hier Sonde Metall erkannt



- Metall entfernen, refresh durchführen – Sonde schaltet





Es werden 10 Messungen durchgeführt, dabei soll die Standardabweichung unter 0,003mm liegen.

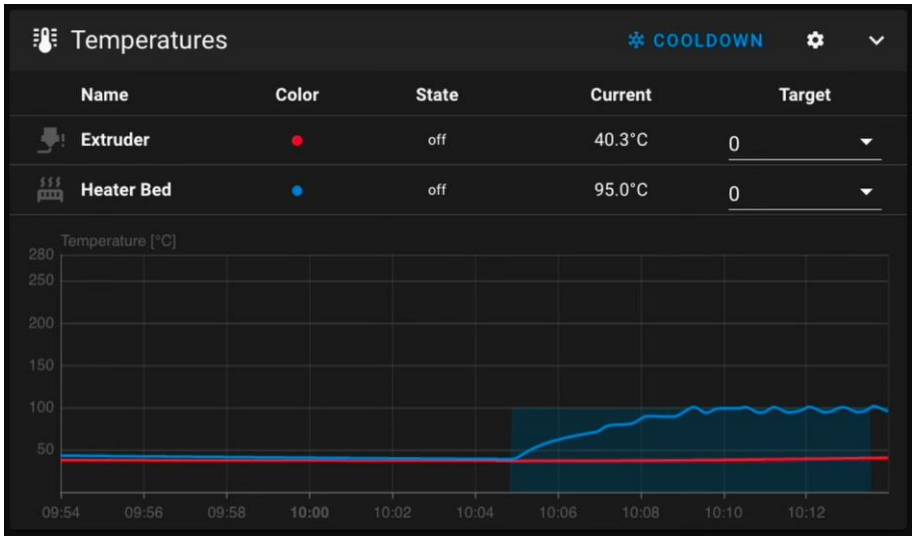
Tuning, Superslicer und der erste Druck

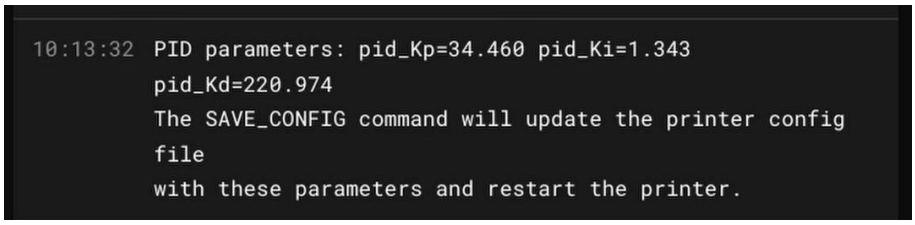
Tuning

PID-Calibration

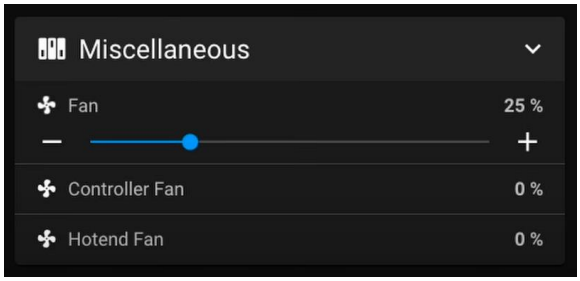
Durckkopf auf die Mitte des Druckbetts zentrieren und 5-10mm darüber positionieren.

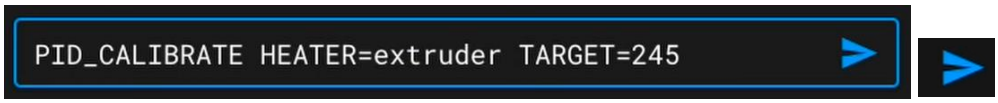
Dauer ca. 10min



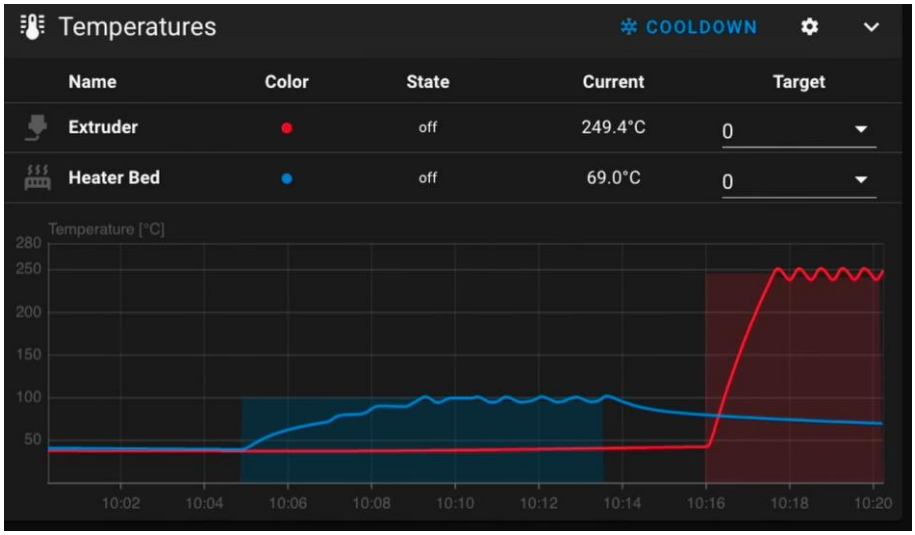


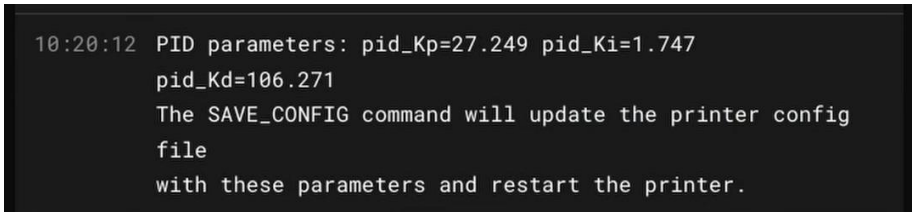
Aktivieren des Bauteilelüfters auf 25%





Dauer ca. 5min





Quad-Gantry-Leveling QGL

- Ausrichten der zum Rahmen im ausgeschaltetem Zustand des Druckers mit einem Distanzstück oder durch Ausmessen, so dass die Grantry parallel zum Bett verläuft.

- Homing aller Achsen
- 

```
16:57:44  Retries: 2/5 Probed points range: 0.006250 tolerance:
          0.007500

16:57:44  Making the following Z adjustments:
          stepper_z = 0.000609
          stepper_z1 = 0.002071
          stepper_z2 = 0.005454
          stepper_z3 = -0.008134

16:57:44  Average: 9.264916

16:57:44  Actuator Positions:
          z: 9.264307 z1: 9.262845 z2: 9.259461 z3: 9.273049
```

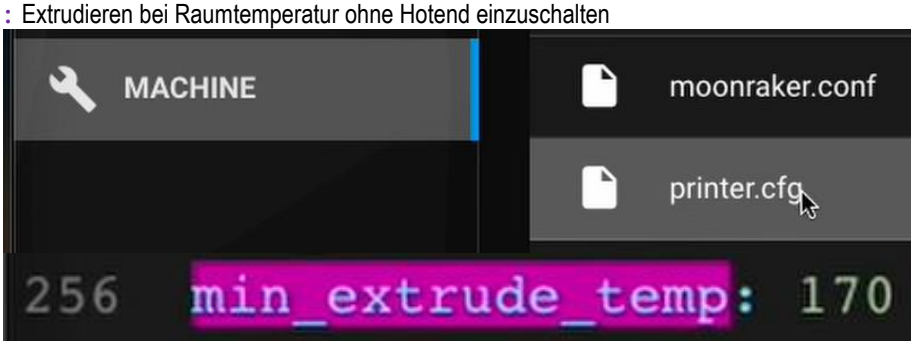
Toleranzbereich ist 0,0075mm

Extruderkalibrierung

Vorbereitung

- Stück Filamentschlauch in Extruder einführen

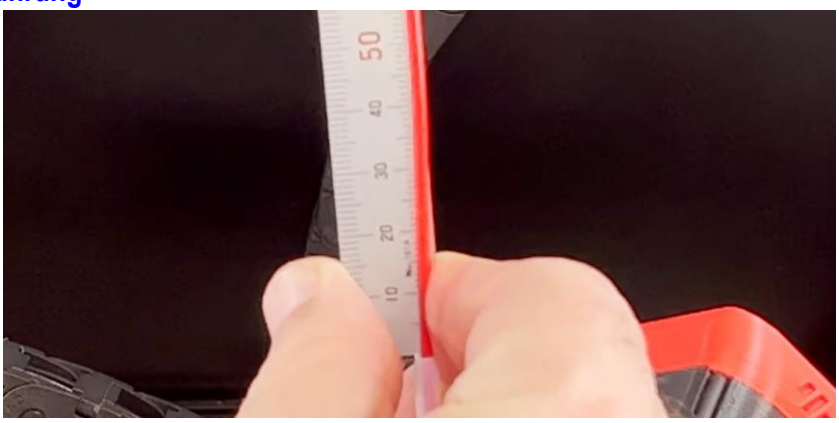
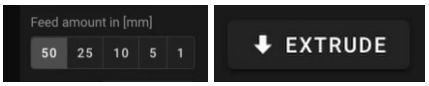
HINWEIS: Extrudieren bei Raumtemperatur ohne Hotend einzuschalten

- 

- Den Wert auf 20 ändern
- 

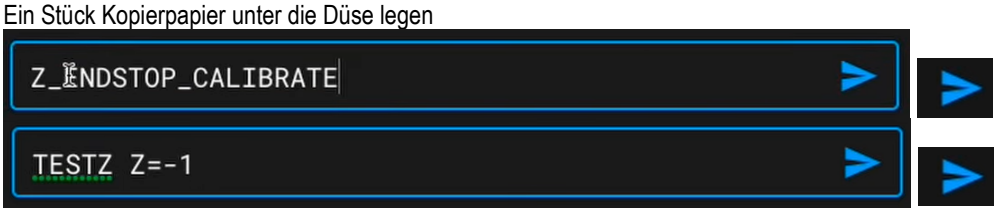
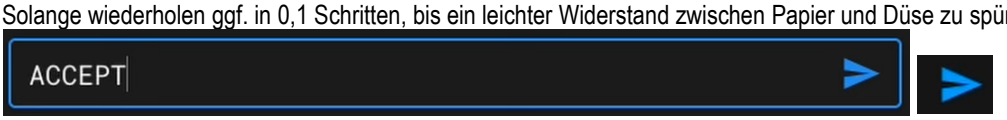
- Demontage des Druckkopfes, um das Filament ohne Hotend durchzuführen. Es soll ja nur der Extruder eingestellt werden.
-  ca. 500mm Filament verwenden und es in den Extruder einführen, bis der Extruder das Filament greift.

Durchführung

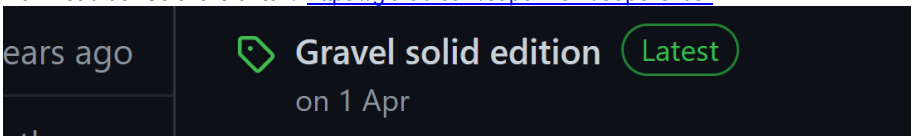
-  120mm ausmessen und markieren.
-  2x50mm extrudieren
- 
  Auf den neuen Wert in der Printer.cfg unter EXTRUDER korrigieren :
  Korrigierter Wert = (Alter Wert in Printer.cfg x Neuer gemessener Wert)/100
- 
- Extruderkalibrierung erneut aufrufen ,kontrollieren und ggf. wiederholen.

Z-Offset einstellen

Vorbereitung

- Einstellen der Temperatur von Hotend auf 240°C und Heizbett auf 100°C
- Temperatur 30min halten
- 
- Zentrieren des Druckkopfes auf die Mitte des Druckbetts
- Ein Stück Kopierpapier unter die Düse legen
- 
- Solange wiederholen ggf. in 0,1 Schritten, bis ein leichter Widerstand zwischen Papier und Düse zu spüren ist.
- 
- 

Superslicer

- Download der Software unter : https://github.com/supermerill/SuperSlicer
- 
- 

```
cd ~/moonraker/scripts./fetch-apikey.sh
```

Klipperdokumentation

Eigenschaften

Klipper hat mehrere überzeugende Eigenschaften:

- Hochpräzise Stepper-Bewegung. Klipper nutzt einen Anwendungsprozessor (z. B. einen kostengünstigen Raspberry Pi) für die Berechnung der Druckerbewegungen. Der Anwendungsprozessor bestimmt, wann die einzelnen Schrittmotoren in Gang gesetzt werden sollen, er komprimiert diese Ereignisse, überträgt sie an den Mikrocontroller, und der Mikrocontroller führt dann jedes Ereignis zum gewünschten Zeitpunkt aus. Jedes Stepper-Ereignis wird mit einer Genauigkeit von 25 Mikrosekunden oder besser geplant. Die Software verwendet keine kinematischen Schätzungen (wie z. B. den Bresenham-Algorithmus), sondern berechnet präzise Schrittzeiten auf der Grundlage der Physik der Beschleunigung und der Physik der Maschinenkinematik. Eine präzisere Schrittbewegung führt zu einem ruhigeren und stabileren Betrieb des Druckers.

- Beste Leistungsparameter. Klipper ist in der Lage, sowohl auf neuen als auch auf alten Mikrocontrollern hohe Schrittraten zu erzielen. Selbst alte 8-Bit-Mikrocontroller können Raten von über 175.000 Schritten pro Sekunde erreichen. Auf neueren Mikrocontrollern sind mehrere Millionen Schritte pro Sekunde möglich. Höhere Stepperraten ermöglichen höhere Druckgeschwindigkeiten. Das Stepper-Event-Timing bleibt auch bei hohen Geschwindigkeiten präzise, was die Gesamtstabilität verbessert.

- Klipper unterstützt Drucker mit mehreren Mikrocontrollern. So kann beispielsweise ein Mikrocontroller zur Steuerung eines Extruders verwendet werden, während ein anderer die Heizungen des Druckers steuert und ein dritter den Rest des Druckers kontrolliert. Die Klipper-Host-Software implementiert eine Taktsynchronisation, um die Taktdrift zwischen den Mikrocontrollern zu berücksichtigen. Es ist kein spezieller Code erforderlich, um mehrere Mikrocontroller zu aktivieren - es sind lediglich ein paar zusätzliche Zeilen in der Konfigurationsdatei erforderlich.

- Konfiguration über einfache Konfigurationsdatei. Es besteht keine Notwendigkeit, den Mikrocontroller neu zu flashen, um eine Einstellung zu ändern. Die gesamte Konfiguration von Klipper wird in einer Standard-Konfigurationsdatei gespeichert, die einfach bearbeitet werden kann. Das macht es einfacher, die Hardware einzurichten und zu warten.

- Klipper unterstützt "Smooth Pressure Advance" - ein Mechanismus, der die Auswirkungen des Drucks in einem Extruder berücksichtigt. Dadurch wird das "Schmieren" des Extruders reduziert und die Qualität der Druckecken verbessert. Die Implementierung von Klipper führt nicht zu sofortigen Änderungen der Extrudergeschwindigkeit, was die Gesamtstabilität und Robustheit verbessert.

- Klipper unterstützt "Input Shaping", um die Auswirkungen von Vibrationen auf die Druckqualität zu reduzieren. Dies kann "Ringing" (auch bekannt als "Ghosting", "Echoing" oder "Rippling") in Drucken reduzieren oder eliminieren. Es kann auch eine schnellere Druckgeschwindigkeit bei gleichbleibend hoher Druckqualität ermöglichen.

- Klipper verwendet einen "iterativen Solver", um präzise Schrittzeiten aus einfachen kinematischen Gleichungen zu berechnen. Das macht die Portierung von Klipper auf neue Robotertypen einfacher und hält die Zeitmessung auch bei komplexer Kinematik präzise (es ist keine "Liniensegmentierung" erforderlich).

- Portierbarer Code. Klipper funktioniert auf ARM-, AVR- und PRU-basierten Mikrocontrollern. Vorhandene Drucker im "Reprap"-Stil können Klipper ohne Hardware-Modifikation betreiben - fügen Sie einfach einen Raspberry Pi hinzu. Das interne Code-Layout von Klipper macht es einfacher, auch andere Mikrocontroller-Architekturen zu unterstützen.

- Einfacherer Code. Klipper verwendet eine sehr hohe Programmiersprache (Python) für den meisten Code. Die Kinematik-Algorithmen, das Parsen des G-Codes, die Heiz- und Thermistor-Algorithmen usw. sind alle in Python geschrieben. Das macht es einfacher, neue Funktionen zu entwickeln.

- Benutzerdefinierte programmierbare Makros. Neue G-Code-Befehle können in der Druckerkonfigurationsdatei definiert werden (es sind keine Codeänderungen erforderlich). Diese Befehle sind programmierbar, so dass sie je nach Zustand des Druckers unterschiedliche Aktionen auslösen können.

- Eingebauter API-Server. Zusätzlich zur Standard-G-Code-Schnittstelle unterstützt Klipper eine umfangreiche JSON-basierte Anwendungsschnittstelle. Dies ermöglicht es Programmierern, externe Anwendungen mit detaillierter Kontrolle über den Drucker zu erstellen.

Zusätzliche Funktionen

- Klipper unterstützt viele Standard-3D-Druckerfunktionen:

- Arbeitet mit Octoprint. Dies ermöglicht die Steuerung des Druckers über einen normalen Webbrowser. Auf demselben Raspberry Pi, auf dem Klipper läuft, kann auch Octoprint laufen.

- Standard G-Code Unterstützung. Gängige G-Code-Befehle, die von typischen "Slicern" (SuperSlicer, Cura, PrusaSlicer, etc.) erzeugt werden, werden unterstützt.

- Unterstützung für mehrere Extruder. Extruder mit gemeinsamen Heizungen und Extruder auf unabhängigen Schlitten (IDEX) werden ebenfalls unterstützt.

- Unterstützung für kartesische, Delta-, Corexy-, Corexz-, Hybrid-Corexy-, Hybrid-Corexz-, Rotations-Delta-, Polar- und Kabelwinden-Drucker.
- Unterstützung der automatischen Bettnivellierung. Klipper kann für eine einfache Erkennung der Bettneigung oder eine komplette Bettnivellierung konfiguriert werden. Wenn das Bett mehrere Z-Stepper verwendet, kann Klipper auch durch unabhängige Manipulation der Z-Stepper nivellieren. Die meisten Z-Höhen-Taster werden unterstützt, einschließlich BL-Touch-Taster und servoaktivierte Taster.
- Unterstützung der automatischen Deltakalibrierung. Das Kalibrierungswerkzeug kann sowohl eine grundlegende Höhenkalibrierung als auch eine erweiterte Kalibrierung der X- und Y-Abmessungen durchführen. Die Kalibrierung kann mit einem Z-Höhenmessfühler oder durch manuelle Abtastung durchgeführt werden.
- Unterstützung für gängige Temperatursensoren (z. B. gängige Thermistoren, AD595, AD597, AD849x, PT100, PT1000, MAX6675, MAX31855, MAX31856, MAX31865, BME280, HTU21D, DS18B20 und LM75). Kundenspezifische Thermistoren und analoge Temperatursensoren können ebenfalls konfiguriert werden. Man kann den internen Mikrocontroller-Temperatursensor und den internen Temperatursensor eines Raspberry Pi überwachen.
- Standardmäßig ist der thermische Schutz der Heizung aktiviert.
- Unterstützung für Standardlüfter, Düsenlüfter und temperaturgesteuerte Lüfter. Es ist nicht notwendig, die Lüfter laufen zu lassen, wenn der Drucker im Leerlauf ist. Die Lüftergeschwindigkeit kann bei Lüftern mit PWM überwacht werden.
- Unterstützung für die Laufzeitkonfiguration von TMC2130-, TMC2208/TMC2224-, TMC2209-, TMC2660- und TMC5160-Schrittmotortreibern. Es gibt auch Unterstützung für die Stromsteuerung herkömmlicher Schrittmotortreiber über AD5206, MCP4451, MCP4728, MCP4018 und PWM-Pins.
- Unterstützung für gängige LCD-Displays, die direkt an den Drucker angeschlossen werden. Ein Standardmenü ist ebenfalls verfügbar. Der Inhalt der Anzeige und des Menüs kann über die Konfigurationsdatei vollständig angepasst werden.
- Konstante Beschleunigung und "Look-ahead"-Unterstützung. Alle Bewegungen des Druckers werden allmählich vom Stillstand auf die Reisegeschwindigkeit beschleunigt und dann wieder auf den Stillstand abgebremst. Der eingehende Strom von G-Code-Bewegungsbefehlen wird in eine Warteschlange gestellt und analysiert - die Beschleunigung zwischen Bewegungen in eine ähnliche Richtung wird optimiert, um den Druckstillstand zu reduzieren und die Gesamtdruckzeit zu verbessern.
- Klipper implementiert einen "Stepper-Phase-Endstop"-Algorithmus, der die Genauigkeit typischer Endstop-Schalter verbessern kann. Wenn er richtig eingestellt ist, kann er die Haftung der ersten Schicht des Druckbetts verbessern.
- Unterstützung für Filament-Präsenzsensoren, Filament-Bewegungssensoren und Filament-Breitensensoren.
- Unterstützung für die Messung und Aufzeichnung der Beschleunigung mit einem adxl345-Beschleunigungsmesser.
- Unterstützung für die Begrenzung der Höchstgeschwindigkeit bei kurzen "Zickzack"-Bewegungen zur Reduzierung von Druckervibrationen und Geräuschen. Weitere Informationen finden Sie im Dokument Kinematik kinematics.
- Beispielkonfigurationsdateien sind für viele gängige Drucker verfügbar. Eine Liste finden Sie im Verzeichnis config config directory.
- Um mit Klipper zu beginnen, lesen Sie die Installationsanleitung installation.

Step-Benchmarks

Nachfolgend finden Sie die Ergebnisse von Stepper-Leistungstests. Die angegebenen Zahlen stellen die Gesamtzahl der Schritte pro Sekunde auf dem Mikrocontroller dar.

Micro-controller	1 stepper active	3 steppers active
16Mhz AVR	157K	99K
20Mhz AVR	196K	123K
SAMD21	686K	471K
STM32F042	814K	578K
Beaglebone PRU	866K	708K
STM32G0B1	1103K	790K
STM32F103	1180K	818K
SAM3X8E	1273K	981K
SAM4S8C	1690K	1385K
LPC1768	1923K	1351K

Micro-controller	1 stepper active	3 steppers active
LPC1769	2353K	1622K
RP2040	2400K	1636K
SAM4E8E	2500K	1674K
SAMD51	3077K	1885K
STM32F407	3652K	2459K
STM32F446	3913K	2634K

Wenn Sie sich nicht sicher sind, welcher Mikrocontroller auf einem bestimmten Board verwendet wird, suchen Sie die entsprechende Konfigurationsdatei config file und achten Sie auf den Namen des Mikrocontrollers in den Kommentaren am Anfang der Datei.

Weitere Einzelheiten zu den Benchmarks sind im Dokument Benchmarks Benchmarks document zu finden.

Häufig gestellte Fragen

1. Wie kann ich für das Projekt spenden?

Danke! Kevin hat eine Patreon-Seite unter: https://www.patreon.com/koconnor

2. Wie berechne ich den Konfigurationsparameter rotation_distance?

Siehe das Rotation Distance Dokument.

3. Wo ist mein serieller Anschluss?

Der allgemeine Weg, einen seriellen USB-Anschluss zu finden, besteht darin, ls /dev/serial/by-id/* von einem ssh-Terminal auf dem Host-Rechner aus auszuführen. Dies wird wahrscheinlich eine Ausgabe ähnlich der folgenden ergeben:

```
/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
```

Der im obigen Befehl gefundene Name ist stabil und es ist möglich, ihn in der Konfigurationsdatei und beim Flashen des Mikrocontroller-Codes zu verwenden. Ein Flash-Befehl könnte zum Beispiel so aussehen:

```
sudo service klipper stop
make flash FLASH_DEVICE=/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
sudo service klipper start
```

und die aktualisierte Konfiguration könnte wie folgt aussehen:

```
[mcu]
serial: /dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
```

Achten Sie darauf, den Namen aus dem "ls"-Befehl, den Sie oben ausgeführt haben, zu kopieren und einzufügen, da der Name für jeden Drucker anders sein wird.

Wenn Sie mehrere Mikrocontroller verwenden und diese keine eindeutigen IDs haben (häufig bei Boards mit einem CH340-USB-Chip), befolgen Sie die obigen Anweisungen und verwenden Sie stattdessen den Befehl `ls /dev/serial/by-path/*`.

4. Wenn der Mikrocontroller neu startet, wechselt das Gerät zu `/dev/ttyUSB1`

Befolgen Sie die Anweisungen im Abschnitt "Wo ist mein serieller Anschluss?" Where's my serial port, um dies zu verhindern.

5. Der Befehl "`make flash`" funktioniert nicht

Der Code versucht, das Gerät mit der für jede Plattform am häufigsten verwendeten Methode zu flashen. Leider gibt es eine große Varianz bei den Flash-Methoden, so dass der "make flash"-Befehl möglicherweise nicht auf allen Boards funktioniert.

Wenn Sie einen intermittierenden Fehler haben oder ein Standard-Setup haben, dann überprüfen Sie, ob Klipper beim Flashen nicht läuft (sudo service klipper stop), stellen Sie sicher, dass OctoPrint nicht versucht, sich direkt mit dem Gerät zu verbinden (öffnen Sie die Registerkarte "Connection" auf der Webseite und klicken Sie auf "Disconnect", wenn der serielle Anschluss auf das Gerät eingestellt ist), und stellen Sie sicher, dass FLASH_DEVICE für Ihr Board richtig eingestellt ist (siehe die Frage oben question above).

Wenn jedoch "make flash" für Ihr Board nicht funktioniert, dann müssen Sie manuell flashen. Sehen Sie nach, ob es eine Konfigurationsdatei im config-Verzeichnis config directory gibt, die spezifische Anweisungen zum Flashen des Geräts enthält. Schauen Sie auch in der Dokumentation des Boardherstellers nach, ob dort beschrieben ist, wie Sie das Gerät flashen können. Schließlich kann es möglich sein, das Gerät mit Tools wie "avrdude" oder "bossac" manuell zu flashen - siehe das Bootloader-Dokument für weitere Informationen.

6. Wie ändere ich die serielle Baudrate?

Die empfohlene Baudrate für Klipper ist 250000. Diese Baudrate funktioniert auf allen Mikrocontroller-Boards, die Klipper unterstützt. Wenn du eine Online-Anleitung gefunden hast, die eine andere Baudrate empfiehlt, dann ignoriere diesen Teil der Anleitung und fahre mit dem Standardwert von 250000 fort.

Wenn Sie die Baudrate trotzdem ändern wollen, dann muss die neue Rate im Mikrocontroller konfiguriert werden (während make menuconfig) und der aktualisierte Code muss kompiliert und in den Mikrocontroller geflasht werden. Die Klipper printer.cfg Datei muss ebenfalls auf die neue Baudrate angepasst werden (siehe die Konfigurationsreferenz für Details). Zum Beispiel:

```
[mcu]
baud: 250000
```

Die auf der OctoPrint-Webseite angezeigte Baudrate hat keinen Einfluss auf die interne Baudrate des Klipper-Mikrocontrollers. Stellen Sie die OctoPrint-Baudrate immer auf 250000 ein, wenn Sie Klipper verwenden.

Die Baudrate des Klipper-Mikrocontrollers steht in keinem Zusammenhang mit der Baudrate des Bootloaders des Mikrocontrollers. Siehe das Bootloader-Dokument bootloader document für zusätzliche Informationen über Bootloader.

7. Kann ich Klipper auch auf einem anderen Gerät als einem Raspberry Pi 3 betreiben?

Die empfohlene Hardware ist ein Raspberry Pi 2, Raspberry Pi 3, oder Raspberry Pi 4.

Klipper läuft auch auf einem Raspberry Pi 1 und dem Raspberry Pi Zero, aber diese Boards haben nicht genug Rechenleistung, um OctoPrint gut auszuführen. Es kommt häufig vor, dass der Druck auf diesen langsameren Geräten stockt, wenn man direkt mit OctoPrint druckt. (Der Drucker bewegt sich möglicherweise schneller, als OctoPrint Bewegungsbefehle senden kann.) Wenn Sie trotzdem auf einem dieser langsameren Boards drucken möchten, sollten Sie die Funktion "virtual_sdcard" verwenden (siehe Konfigurationsreferenz config reference für Details).

Klipper wurde auch auf anderen Rechnern ausgeführt. Die Klipper-Hostsoftware benötigt nur Python auf einem Linux- (oder ähnlichen) Computer. Wenn Sie die Software jedoch auf einem anderen Rechner betreiben möchten, benötigen Sie Linux-Administrationskenntnisse, um die Systemvoraussetzungen für diesen Rechner zu installieren. Im Skript install-octopi.sh install-octopi.sh finden Sie weitere Informationen zu den notwendigen Schritten für Linux-Administratoren.

Wenn Sie die Klipper-Hostsoftware auf einem Low-End-Chip laufen lassen wollen, sollten Sie sich darüber im Klaren sein, dass Sie mindestens eine Maschine mit "double precision floating point" Hardware benötigen.

Wenn Sie die Klipper-Hostsoftware auf einem allgemeinen Desktop- oder Server-Rechner laufen lassen wollen, dann beachten Sie, dass Klipper einige Anforderungen an die Echtzeitplanung stellt. Wenn der Host-Computer während eines Drucks auch eine intensive allgemeine Rechenaufgabe ausführt (z.B. Defragmentierung einer Festplatte, 3D-Rendering, umfangreiches Swapping, usw.), kann dies dazu führen, dass Klipper Druckfehler meldet.

Hinweis: Wenn Sie kein OctoPi-Image verwenden, sollten Sie beachten, dass einige Linux-Distributionen ein "ModemManager"-Paket (oder ein ähnliches) aktivieren, das die serielle Kommunikation unterbrechen kann. (Was dazu führen kann, dass Klipper scheinbar zufällige "Lost communication with MCU"-Fehler meldet.) Wenn du Klipper auf einer dieser Distributionen installierst, musst du dieses Paket eventuell deaktivieren.

8. Kann ich mehrere Instanzen von Klipper auf demselben Rechner laufen lassen?

Es ist möglich, mehrere Instanzen der Klipper-Hostsoftware laufen zu lassen, aber das erfordert Linux-Administrationskenntnisse. Die Klipper-Installationsskripte veranlassen letztlich die Ausführung des folgenden Unix-Befehls:

```
~/klippy-env/bin/python ~/klipper/klippy/klippy.py ~/printer.cfg -l /tmp/klippy.log
```

Man kann mehrere Instanzen des obigen Befehls ausführen, solange jede Instanz ihre eigene Druckerkonfigurationsdatei, ihre eigene Protokolldatei und ihr eigenes Pseudo-Tty hat. Zum Beispiel:

```
~/klippy-env/bin/python ~/klipper/klippy/klippy.py ~/printer2.cfg -l /tmp/klippy2.log -I /tmp/printer2
```

Wenn Sie sich dafür entscheiden, müssen Sie die erforderlichen Start-, Stopp- und Installationsskripte (falls vorhanden) implementieren. Das Skript install-octopi.sh install-octopi.shund das Skript klipper-start.sh klipper-start.sh können als Beispiele dienen.

9. Muss ich OctoPrint verwenden?

Die Klipper-Software ist nicht von OctoPrint abhängig. Es ist möglich, alternative Software zu verwenden, um Befehle an Klipper zu senden, aber dies erfordert Linux-Administrationskenntnisse.

Klipper erstellt eine "virtuelle serielle Schnittstelle" über die Datei "/tmp/printer" und emuliert über diese Datei eine klassische serielle Schnittstelle eines 3D-Druckers. Im Allgemeinen kann alternative Software mit Klipper funktionieren, solange sie so konfiguriert werden kann, dass sie "/tmp/printer" als serielle Schnittstelle für den Drucker verwendet.

10. Warum kann ich den Stepper nicht bewegen, bevor ich den Drucker referenziere

Der Code tut dies, um das Risiko zu verringern, dass der Kopf versehentlich in das Bett oder eine Wand fährt. Sobald der Drucker referenziert ist, versucht die Software zu überprüfen, ob jede Bewegung innerhalb der in der Konfigurationsdatei definierten position_min/max liegt. Wenn die Motoren deaktiviert sind (über einen M84- oder M18-Befehl), müssen die Motoren vor der Bewegung erneut referenziert werden.

Wenn Sie den Kopf nach dem Abbruch eines Druckvorgangs über OctoPrint bewegen möchten, sollten Sie die OctoPrint-Abbruchsequenz so ändern, dass sie dies für Sie erledigt. Sie wird in OctoPrint über einen Webbrowser konfiguriert unter: Einstellungen->GCODE-Skripte

Wenn Sie den Druckkopf nach Beendigung eines Druckvorgangs bewegen möchten, sollten Sie die gewünschte Bewegung in den Abschnitt "Benutzerdefinierter G-Code" Ihres Slicers aufnehmen.

Wenn der Drucker eine zusätzliche Bewegung als Teil des Homing-Prozesses selbst benötigt (oder grundsätzlich keinen Homing-Prozess hat), dann sollten Sie einen safe_z_home oder homing_override Abschnitt in der Konfigurationsdatei verwenden. Wenn Sie einen Stepper zu Diagnose- oder Debugging-Zwecken bewegen müssen, sollten Sie einen force_move-Abschnitt in die Konfigurationsdatei aufnehmen. Siehe Config-Referenz config reference für weitere Details zu diesen Optionen.

11. Warum ist die Z-Position_endstop in den Standardkonfigurationen auf 0,5 eingestellt?

Bei kartesischen Druckern gibt der Z position_endstop an, wie weit die Düse vom Bett entfernt ist, wenn der Endstop ausgelöst wird. Wenn möglich, wird empfohlen, einen Z-max-Endstop zu verwenden und vom Bett wegzufahren (da dies das Potenzial für Bettkollisionen verringert). Wenn jedoch eine Referenzfahrt zum Bett hin erforderlich ist, wird empfohlen, den Endanschlag so zu positionieren, dass er auslöst, wenn die Düse noch einen kleinen Abstand zum Bett hat. Auf diese Weise wird die Achse bei der Referenzfahrt angehalten, bevor die Düse das Bett berührt. Weitere Informationen finden Sie im Dokument zur Betthöhe bed level document.

12. Ich habe meine Konfiguration von Marlin konvertiert und die X/Y-Achsen funktionieren einwandfrei, aber bei der Referenzfahrt der Z-Achse erhalte ich ein kreischendes Geräusch

Kurze Antwort: Stellen Sie zunächst sicher, dass Sie die Stepperkonfiguration wie im Config Check-Dokument config check document beschrieben überprüft haben. Wenn das Problem weiterhin besteht, versuchen Sie, die Einstellung max_z_velocity in der Druckerkonfiguration zu verringern.

Lange Antwort: In der Praxis kann der Marlin in der Regel nur eine Schrittgeschwindigkeit von etwa 10000 Schritten pro Sekunde erreichen. Wenn er sich mit einer Geschwindigkeit bewegen soll, die eine höhere Schrittgeschwindigkeit erfordert, wird Marlin im Allgemeinen nur so schnell wie möglich schreiten. Klipper ist in der Lage, viel höhere Schrittraten zu erreichen, aber der Schrittmotor hat möglicherweise nicht genügend Drehmoment, um eine höhere Geschwindigkeit zu erreichen. Bei einer Z-Achse mit einem hohen Übersetzungsverhältnis oder einer hohen Mikroschritt-Einstellung kann die tatsächlich erreichbare max_z_velocity also kleiner sein als die in Marlin konfigurierte.

13. Mein TMC-Motortreiber schaltet sich mitten in einem Druckvorgang ab

Wenn Sie den TMC2208 (oder TMC2224) Treiber im "Standalone-Modus" verwenden, stellen Sie sicher, dass Sie die neueste Version von Klipper latest version of Klipper verwenden. Ein Workaround für ein TMC2208 "stealthchop" Treiberproblem wurde Mitte März 2020 zu Klipper hinzugefügt.

14. Ich erhalte immer wieder zufällige "Lost communication with MCU"-Fehler

Dies wird in der Regel durch Hardwarefehler in der USB-Verbindung zwischen dem Host-Rechner und dem Mikrocontroller verursacht. Worauf Sie achten sollten:

- Verwenden Sie ein hochwertiges USB-Kabel zwischen dem Host-Rechner und dem Mikrocontroller. Stellen Sie sicher, dass die Stecker fest sitzen.
- Wenn Sie einen Raspberry Pi verwenden, benutzen Sie ein hochwertiges Netzteil für den Raspberry Pi und ein hochwertiges USB-Kabel good quality power supply, um das Netzteil mit dem Pi zu verbinden. Wenn Sie von OctoPrint die Warnung "Unterspannung" erhalten, hängt dies mit der Stromversorgung zusammen und muss behoben werden.
- Stellen Sie sicher, dass die Stromversorgung des Druckers nicht überlastet wird. (Stromschwankungen am USB-Chip des Mikrocontrollers können zu einem Reset des Chips führen).
- Vergewissern Sie sich, dass Stepper-, Heiz- und andere Druckerkabel nicht gequetscht oder ausgefranst sind. (Die Bewegung des Druckers kann ein fehlerhaftes Kabel belasten, so dass es den Kontakt verliert, einen Kurzschluss verursacht oder übermäßiges Rauschen erzeugt).
- Es gibt Berichte über starkes USB-Rauschen, wenn sowohl die Stromversorgung des Druckers als auch die 5-V-Stromversorgung des Hosts gemischt sind. (Wenn Sie feststellen, dass sich der Mikrocontroller einschaltet, wenn entweder die Stromversorgung des Druckers eingeschaltet ist oder das USB-Kabel eingesteckt ist, deutet dies darauf hin, dass die 5-V-Stromversorgungen gemischt werden). Es kann hilfreich sein, den Mikrocontroller so zu konfigurieren, dass er nur von einer Stromquelle gespeist wird. (Wenn die Mikrocontroller-Platine ihre Stromquelle nicht konfigurieren kann, kann man alternativ ein USB-Kabel so modifizieren, dass es keine 5-V-Stromversorgung zwischen Host und Mikrocontroller überträgt).

15. Mein Raspberry Pi startet während des Druckens immer wieder neu

Dies ist höchstwahrscheinlich auf Spannungsschwankungen zurückzuführen. Befolgen Sie die gleichen Schritte zur Fehlerbehebung bei der Fehlermeldung "Lost communication with MCU" "Lost communication with MCU".

16. Wenn ich restart_method=command einstelle, bleibt mein AVR-Gerät bei einem Neustart einfach hängen

Einige alte Versionen des AVR-Bootloaders haben einen bekannten Fehler in der Handhabung von Watchdog-Events. Dieser tritt typischerweise auf, wenn in der Datei printer.cfg restart_method auf "command" gesetzt ist. Wenn der Fehler auftritt, reagiert das AVR-Gerät nicht, bis die Stromversorgung unterbrochen und wiederhergestellt wird (die Strom- oder Status-LEDs können auch wiederholt blinken, bis die Stromversorgung unterbrochen wird).

Der Workaround ist die Verwendung einer anderen restart_method als "command" oder das Flashen eines aktualisierten Bootloaders auf das AVR-Gerät. Das Flashen eines neuen Bootloaders ist ein einmaliger Schritt, der normalerweise ein externes Programmiergerät erfordert - siehe Bootloader Bootloaders für weitere Details.

17. Bleiben die Heizungen an, wenn der Raspberry Pi abstürzt?

Die Software wurde entwickelt, um dies zu verhindern. Sobald der Host eine Heizung aktiviert, muss die Host-Software diese Aktivierung alle 5 Sekunden bestätigen. Wenn der Mikrocontroller nicht alle 5 Sekunden eine Bestätigung erhält, geht er in einen "Shutdown"-Zustand über, der dazu dient, alle Heizungen und Schrittmotoren auszuschalten.

Weitere Einzelheiten sind dem Befehl "config_digital_out" im Dokument MCU-Befehle MCU commands zu entnehmen.

Darüber hinaus wird die Mikrocontroller-Software beim Start mit einem minimalen und maximalen Temperaturbereich für jede Heizung konfiguriert (siehe die Parameter min_temp und max_temp in der Config-Referenz config reference für weitere Einzelheiten). Wenn der Mikrocontroller feststellt, dass die Temperatur außerhalb dieses Bereichs liegt, geht er ebenfalls in einen "Abschalt"-Zustand über.

Unabhängig davon implementiert die Host-Software auch einen Code zur Überprüfung der korrekten Funktion von Heizungen und Temperatursensoren. Siehe die Konfigurationsreferenz config reference für weitere Details.

18. Wie konvertiere ich eine Marlin-Pin-Nummer in einen Klipper-Pin-Namen?

Kurze Antwort: Eine Zuordnung ist in der Datei sample-aliases.cfg sample-aliases.cfg verfügbar. Verwenden Sie diese Datei als Leitfaden, um die tatsächlichen Mikrocontroller-Pinnamen zu finden. (Es ist auch möglich, den entsprechenden board_pins-Konfigurationsabschnitt board_pins in Ihre Konfigurationsdatei zu kopieren und die Aliase in Ihrer Konfiguration zu verwenden, aber es ist vorzuziehen, die tatsächlichen Mikrocontroller-Pinnamen zu übersetzen und zu verwenden). Beachte, dass die sample-aliases.cfg-Datei Pin-Namen verwendet, die mit dem Präfix "ar" anstelle von "D" beginnen (z.B. Arduino Pin D23 ist Klipper alias ar23) und das Präfix "analog" anstelle von "A" (z.B. Arduino Pin A14 ist Klipper alias analog14).

Lange Antwort: Klipper verwendet die vom Mikrocontroller definierten Standard-Pinnamen. Auf den Atmega-Chips haben diese Hardware-Pins Namen wie PA4, PC7, oder PD2.

Vor langer Zeit beschloss das Arduino-Projekt, die Standard-Hardware-Namen zu vermeiden und stattdessen eigene Pin-Namen zu verwenden, die auf inkrementellen Zahlen basieren - diese Arduino-Namen sehen im Allgemeinen aus wie D23 oder A14. Dies war eine unglückliche Entscheidung, die zu einer großen Verwirrung geführt hat. Vor allem die Arduino-Pin-Nummern entsprechen häufig nicht den gleichen Hardware-Namen. Zum Beispiel ist D21 auf einem gängigen Arduino-Board PD0, aber auf einem anderen gängigen Arduino-Board ist es PC7.

Um diese Verwirrung zu vermeiden, verwendet der Klipper-Kerncode die vom Mikrocontroller definierten Standard-Pinnamen.

19. Muss ich mein Gerät mit einem bestimmten Typ von Mikrocontroller-Pin verdrahten?

Das hängt von der Art des Gerätes und der Art des Pins ab:

ADC-Pins (oder Analog-Pins): Bei Thermistoren und ähnlichen "analogen" Sensoren muss das Gerät an einen "analogen" oder "ADC"-fähigen Pin des Mikrocontrollers angeschlossen werden. Wenn Sie Klipper so konfigurieren, dass ein Pin verwendet wird, der nicht analogfähig ist, meldet Klipper den Fehler "Not a valid ADC pin".

PWM-Pins (oder Timer-Pins): Klipper verwendet standardmäßig keine Hardware-PWM für irgendein Gerät. Im Allgemeinen kann man also Heizungen, Lüfter und ähnliche Geräte mit jedem beliebigen IO-Pin verbinden. Lüfter und output_pin-Geräte können jedoch optional so konfiguriert werden, dass sie hardware_pwm verwenden: True, in diesem Fall muss der Mikrocontroller Hardware-PWM auf dem Pin unterstützen (andernfalls meldet Klipper einen "Not a valid PWM pin"-Fehler).

IRQ-Pins (oder Interrupt-Pins): Klipper verwendet keine Hardware-Interrupts auf IO-Pins, daher ist es nie notwendig, ein Gerät mit einem dieser Mikrocontroller-Pins zu verbinden.

SPI-Pins: Bei Verwendung von Hardware-SPI ist es notwendig, die Pins mit den SPI-fähigen Pins des Mikrocontrollers zu verbinden. Die meisten Geräte können jedoch so konfiguriert werden, dass sie "Software-SPI" verwenden; in diesem Fall können alle Allzweck-IO-Pins verwendet werden.

I2C-Pins: Bei Verwendung von I2C müssen die Pins mit den I2C-fähigen Pins des Mikrocontrollers verbunden werden.

Andere Geräte können mit jedem beliebigen Allzweck-IO-Pin verbunden werden. Zum Beispiel können Stepper, Heizungen, Lüfter, Z-Sonden, Servos, LEDs, gewöhnliche hd44780/st7920-LCD-Displays und die Trinamic-UART-Steuerleitung mit jedem beliebigen Allzweck-IO-Pin verbunden werden.

20. Wie breche ich eine M109/M190 "Warte auf Temperatur"-Anforderung ab?

Navigieren Sie zur OctoPrint-Terminal-Registerkarte und geben Sie einen M112-Befehl in das Terminalfeld ein. Der M112-Befehl bewirkt, dass Klipper in einen "Shutdown"-Status übergeht und OctoPrint die Verbindung zu Klipper trennt. Navigieren Sie zum OctoPrint-Verbindungsbereich und klicken Sie auf "Verbinden", damit OctoPrint die Verbindung wieder aufnimmt. Navigieren Sie zurück zur Registerkarte "Terminal" und geben Sie einen FIRMWARE_RESTART-Befehl aus, um den Fehlerstatus von Klipper zu löschen. Nach Abschluss dieser Sequenz wird die vorherige Heizanforderung abgebrochen und ein neuer Druck kann gestartet werden.

21. Kann ich herausfinden, ob der Drucker Schritte verloren hat?

In gewisser Weise, ja. Fahren Sie den Drucker in die Homeposition, geben Sie einen GET_POSITION-Befehl aus, führen Sie Ihren Druck aus, fahren Sie erneut die Homeposition an und geben Sie erneut `GET_POSITION` aus. Vergleichen Sie dann die Werte in der Zeile `mcu:`.

Dies kann hilfreich sein, um Einstellungen wie Schrittmotorströme, Beschleunigungen und Geschwindigkeiten einzustellen, ohne tatsächlich etwas zu drucken und Filament zu verschwenden: Führen Sie einfach einige Hochgeschwindigkeitsbewegungen zwischen den GET_POSITION-Befehlen aus.

Beachten Sie, dass die Endschalter selbst dazu neigen, bei leicht unterschiedlichen Positionen auszulösen, so dass ein Unterschied von ein paar Mikroschritten wahrscheinlich das Ergebnis von Ungenauigkeiten der Endschalter ist. Ein Schrittmotor selbst kann nur Schritte in Schritten von 4 Vollschritten verlieren. (Wenn man also 16 Mikroschritte verwendet, würde ein verlorener Schritt des Schrittmotors dazu führen, dass der "mcu:"-Schrittzähler um ein Vielfaches von 64 Mikroschritten daneben liegt).

22. Warum meldet Klipper Fehler? Ich habe meinen Druck verloren!

Kurze Antwort: Wir wollen wissen, ob unsere Drucker ein Problem erkennen, damit das zugrundeliegende Problem behoben werden kann und wir qualitativ hochwertige Ausdrucke erhalten können. Wir wollen auf keinen Fall, dass unsere Drucker unbemerkt minderwertige Ausdrucke produzieren.

Lange Antwort: Klipper wurde entwickelt, um viele vorübergehende Probleme automatisch zu umgehen. So werden z. B. Kommunikationsfehler automatisch erkannt und erneut übertragen; es werden Aktionen im Voraus geplant und Befehle auf mehreren Ebenen gepuffert, um auch bei intermittierenden Störungen ein präzises Timing zu ermöglichen. Stellt die Software jedoch einen Fehler fest, von dem sie sich nicht erholen kann, wird ihr eine ungültige Aktion befohlen, oder stellt sie fest, dass sie hoffnungslos unfähig ist, die ihr befohlene Aufgabe zu erfüllen, meldet Klipper einen Fehler. In diesen Situationen besteht ein hohes Risiko, dass ein minderwertiger Druck (oder Schlimmeres) erzeugt wird. Man hofft, dass der Benutzer durch die Warnung in die Lage versetzt wird, das zugrundeliegende Problem zu beheben und die Gesamtqualität seiner Ausdrucke zu verbessern.

Es gibt einige damit verbundene Fragen: Warum hält Klipper den Druck nicht einfach an? Eine Warnung ausgeben? Vor dem Druck auf Fehler prüfen? Fehler in vom Benutzer eingegebenen Befehlen ignorieren? usw.? Gegenwärtig liest Klipper Befehle über das G-Code-Protokoll, und leider ist das G-Code-Befehlsprotokoll nicht flexibel genug, um diese Alternativen heute praktikabel zu machen. Es gibt ein Interesse der Entwickler, die Benutzererfahrung bei abnormalen Ereignissen zu verbessern, aber es wird erwartet, dass dies erhebliche Infrastrukturarbeit erfordert (einschließlich einer Abkehr von G-Code).

23. Wie kann ich ein Upgrade auf die neueste Software durchführen?

Der erste Schritt zur Aktualisierung der Software besteht darin, das Dokument mit den letzten Konfigurationsänderungen config changes zu lesen. Gelegentlich werden Änderungen an der Software vorgenommen, die es erforderlich machen, dass die Benutzer ihre Einstellungen im Rahmen eines Software-Upgrades aktualisieren. Es ist ratsam, dieses Dokument vor dem Upgrade zu lesen.

Wenn Sie bereit für ein Upgrade sind, ist die allgemeine Methode, sich per ssh in den Raspberry Pi einzuloggen und folgendes auszuführen:

```
cd ~/klipper
git pull
~/klipper/scripts/install-octopi.sh
```

Dann kann man den Code des Mikrocontrollers neu kompilieren und flashen. Zum Beispiel:

```
make menuconfig
make clean
make

sudo service klipper stop
make flash FLASH_DEVICE=/dev/ttyACM0
sudo dienst klipper start
```

Es kommt jedoch häufig vor, dass sich nur die Hostsoftware ändert. In diesem Fall kann man nur die Hostsoftware mit aktualisieren und neu starten:

```
cd ~/klipper
git pull
sudo service klipper restart
```

Wenn nach der Verwendung dieser Abkürzung die Software eine Warnung ausgibt, dass der Mikrocontroller neu geflasht werden muss, oder ein anderer ungewöhnlicher Fehler auftritt, dann befolgen Sie die oben beschriebenen Schritte zum vollständigen Upgrade.

Wenn weiterhin Fehler auftreten, überprüfen Sie bitte das Dokument config changes, da Sie möglicherweise die Druckerkonfiguration ändern müssen.

Beachten Sie, dass die g-code Befehle RESTART und FIRMWARE_RESTART keine neue Software laden - die oben genannten Befehle "sudo service klipper restart" und "make flash" werden benötigt, damit eine Softwareänderung wirksam wird.

24. Wie kann ich Klipper deinstallieren?

Auf der Seite der Firmware muss nichts Besonderes geschehen. Folge einfach den Anweisungen zum Flashen der neuen Firmware.

Auf der Raspberry Pi Seite ist ein Deinstallationsskript in scripts/klipper-uninstall.sh verfügbar. Zum Beispiel:

```
sudo ~/klipper/scripts/klipper-uninstall.sh
rm -rf ~/klippy-env ~/klipper
```

Installation

Diese Anleitung geht davon aus, dass die Software auf einem Raspberry Pi-Computer in Verbindung mit OctoPrint läuft. Es wird empfohlen, einen Raspberry Pi 2, 3 oder 4 als Host-Computer zu verwenden (siehe die FAQ FAQ für andere Computer).

Klipper unterstützt derzeit eine Reihe von Atmel ATmega-basierten Mikrocontrollern, ARM-basierten Mikrocontrollern und Beaglebone PRU-basierten Druckern.

Vorbereiten eines OS-Images

Beginne mit der Installation von OctoPi auf dem Raspberry Pi Computer. Verwenden Sie OctoPi v0.17.0 oder höher - siehe die OctoPi releases für Informationen zu den Versionen. Überprüfen Sie, ob OctoPi bootet und ob der OctoPrint-Webserver funktioniert. Nachdem Sie sich mit der OctoPrint-Webseite verbunden haben, folgen Sie der Aufforderung, OctoPrint auf v1.4.2 oder höher zu aktualisieren.

Nach der Installation von OctoPi und dem Upgrade von OctoPrint müssen Sie sich per ssh in den Zielcomputer einloggen, um eine Handvoll Systembefehle auszuführen. Wenn Sie einen Linux- oder MacOS-Desktop verwenden, sollte die "ssh"-Software bereits auf dem Desktop installiert sein. Für andere Desktops gibt es kostenlose ssh-Clients (z. B. PuTTY). Verwenden Sie das ssh-Dienstprogramm, um sich mit dem Raspberry Pi zu verbinden (ssh pi@octopi -- Passwort ist "raspberry") und führen Sie die folgenden Befehle aus:

```
git clone https://github.com/Klipper3d/klipper
./klipper/scripts/install-octopi.sh
```

Die obigen Befehle laden Klipper herunter, installieren einige Systemabhängigkeiten, richten Klipper so ein, dass es beim Systemstart ausgeführt wird, und starten die Klipper-Hostsoftware. Dieser Vor^xgang erfordert eine Internetverbindung und kann ein paar Minuten in Anspruch nehmen.

Bau und Flashen des Mikrocontrollers

Um den Code des Mikrocontrollers zu kompilieren, führe zunächst diese Befehle auf dem Raspberry Pi aus:

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

Wähle den passenden Mikrocontroller aus und überprüfe alle anderen angebotenen Optionen. Sobald die Konfiguration abgeschlossen ist, führen Sie aus:

```
make
```

Es ist notwendig, die serielle Schnittstelle zu bestimmen, die mit dem Mikrocontroller verbunden ist. Bei Mikrocontrollern, die über USB angeschlossen sind, führen Sie Folgendes aus:

```
ls /dev/serial/by-id/*
```

Es sollte etwas Ähnliches wie das Folgende angezeigt werden:

```
/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
```

Es ist üblich, dass jeder Drucker einen eigenen, eindeutigen Namen für den seriellen Anschluss hat. Dieser eindeutige Name wird beim Flashen des Mikrocontrollers verwendet. Es ist möglich, dass es mehrere Zeilen in der obigen Ausgabe gibt - wenn dies der Fall ist, wählen Sie die Zeile, die dem Mikrocontroller entspricht (weitere Informationen finden Sie in den FAQ).

Für gängige Mikrocontroller kann der Code mit etwas Ähnlichem wie folgt geflasht werden:

```
sudo service klipper stop
make flash FLASH_DEVICE=/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
sudo service klipper start
```

Stellen Sie sicher, dass Sie `FLASH_DEVICE` mit dem eindeutigen Namen des seriellen Anschlusses des Druckers aktualisieren.

Stellen Sie beim ersten Flashen sicher, dass OctoPrint nicht direkt mit dem Drucker verbunden ist (klicken Sie auf der OctoPrint-Webseite unter dem Abschnitt "Verbindung" auf "Trennen").

OctoPrint für die Verwendung von Klipper konfigurieren

Der OctoPrint-Webserver muss für die Kommunikation mit der Klipper-Hostsoftware konfiguriert werden. Melden Sie sich mit einem Webbrowser auf der OctoPrint-Webseite an und konfigurieren Sie dann die folgenden Punkte:

Gehen Sie auf die Registerkarte "Einstellungen" (das Schraubenschlüssel-Symbol oben auf der Seite). Fügen Sie unter "Serielle Verbindung" in "Zusätzliche serielle Anschlüsse" "/tmp/printer" hinzu. Klicken Sie dann auf "Speichern".

Öffnen Sie erneut die Registerkarte "Einstellungen" und ändern Sie unter "Serielle Verbindung" die Einstellung "Serieller Anschluss" in "/tmp/printer".

Navigieren Sie auf der Registerkarte "Einstellungen" zur Unterregisterkarte "Verhalten" und wählen Sie die Option "Laufende Druckvorgänge abbrechen, aber mit dem Drucker verbunden bleiben". Klicken Sie auf "Speichern".

Vergewissern Sie sich auf der Hauptseite unter dem Abschnitt "Verbindung" (oben links auf der Seite), dass der "Serielle Anschluss" auf "/tmp/printer" eingestellt ist, und klicken Sie auf "Verbinden". (Wenn "/tmp/printer" nicht zur Auswahl steht, versuchen Sie, die Seite neu zu laden).

Sobald die Verbindung hergestellt ist, navigieren Sie zur Registerkarte "Terminal", geben "status" (ohne Anführungszeichen) in das Befehlseingabefeld ein und klicken auf "Senden". Das Terminalfenster wird wahrscheinlich melden, dass ein Fehler beim Öffnen der Konfigurationsdatei aufgetreten ist - das bedeutet, dass OctoPrint erfolgreich mit Klipper kommuniziert. Fahren Sie mit dem nächsten Abschnitt fort.

Klipper konfigurieren

Die Klipper-Konfiguration wird in einer Textdatei auf dem Raspberry Pi gespeichert. Schauen Sie sich die Beispielkonfigurationsdateien im config-Verzeichnis config directoryan. Die Config Reference Config Reference enthält eine Dokumentation über die Konfigurationsparameter.

Der wohl einfachste Weg, die Klipper-Konfigurationsdatei zu aktualisieren, ist die Verwendung eines Desktop-Editors, der das Editieren von Dateien über das "scp"- und/oder "sftp"-Protokoll unterstützt. Es gibt frei verfügbare Tools, die dies unterstützen (z.B. Notepad++, WinSCP und Cyberduck). Verwenden Sie eine der Beispielkonfigurationsdateien als Ausgangspunkt und speichern Sie sie als Datei mit dem Namen "printer.cfg" im Home-Verzeichnis des Pi-Benutzers (z. B. /home/pi/printer.cfg).

Alternativ kann man die Datei auch direkt auf den Raspberry Pi über ssh kopieren und bearbeiten - zum Beispiel:

```
cp ~/klipper/config/example-cartesian.cfg ~/printer.cfg
nano ~/drucker.cfg
```

Stellen Sie sicher, dass Sie jede Einstellung, die für die Hardware geeignet ist, überprüfen und aktualisieren.

Es ist üblich, dass jeder Drucker seinen eigenen eindeutigen Namen für den Mikrocontroller hat. Der Name kann sich nach dem Flashen von Klipper ändern, also führen Sie den Befehl ls /dev/serial/by-id/* erneut aus und aktualisieren Sie die Konfigurationsdatei mit dem eindeutigen Namen. Aktualisiere zum Beispiel den Abschnitt [mcu] so, dass er ungefähr so aussieht:

```
[mcu]
serial: /dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
```

Nachdem Sie die Datei erstellt und bearbeitet haben, müssen Sie im OctoPrint-Webterminal einen "Neustart"-Befehl eingeben, um die Konfiguration zu laden. Ein "Status"-Befehl meldet, dass der Drucker bereit ist, wenn die Klipper-Konfigurationsdatei erfolgreich gelesen und der Mikrocontroller erfolgreich gefunden und konfiguriert wurde. Es ist nicht ungewöhnlich, dass während der Ersteinrichtung Konfigurationsfehler auftreten - aktualisieren Sie die Konfigurationsdatei des Druckers und geben Sie den Befehl "restart", bis "status" meldet, dass der Drucker bereit ist.

Klipper meldet Fehlermeldungen über die OctoPrint-Terminal-Registerkarte. Der Befehl "status" kann verwendet werden, um Fehlermeldungen erneut zu melden. Das Standard-Startskript von Klipper legt außerdem ein Protokoll in /tmp/klippy.log an, das detailliertere Informationen enthält.

Zusätzlich zu den üblichen g-code Befehlen unterstützt Klipper einige erweiterte Befehle - "status" und "restart" sind Beispiele für diese Befehle. Verwenden Sie den Befehl "help", um eine Liste der anderen erweiterten Befehle zu erhalten.

Nachdem Klipper meldet, dass der Drucker bereit ist, gehst du zum Dokument config check document, um einige grundlegende Überprüfungen der Pin-Definitionen in der Konfigurationsdatei durchzuführen.

Konfigurations-Referenz

Dieses Dokument ist eine Referenz für die in der Klipper-Konfigurationsdatei verfügbaren Optionen.

Die Beschreibungen in diesem Dokument sind so formatiert, dass es möglich ist, sie auszuschneiden und in eine Druckerkonfigurationsdatei einzufügen. Informationen über die Einrichtung von Klipper und die Auswahl einer ersten Konfigurationsdatei finden Sie im Installationsdokument.

Mikrocontroller-Konfiguration

Format der Mikrocontroller-Pinnamen

Viele Konfigurationsoptionen erfordern den Namen eines Mikrocontroller-Pins. Klipper verwendet die Hardware-Namen für diese Pins - zum Beispiel PA4.

Den Pin-Namen kann ein ! vorangestellt werden, um anzuzeigen, dass eine umgekehrte Polarität verwendet werden soll (z.B. Trigger auf low statt high).

Eingangspins kann ein ^ vorangestellt werden, um anzuzeigen, dass ein Hardware-Pull-up-Widerstand für den Stift aktiviert werden soll. Wenn der Mikrocontroller Pull-Down-Widerstände unterstützt, kann einem Eingangspin alternativ ~ vorangestellt werden.

Beachten Sie, dass einige Konfigurationsabschnitte zusätzliche Pins "erzeugen" können. In diesem Fall muss der Konfigurationsabschnitt, der die Pins definiert, in der Konfigurationsdatei vor allen Abschnitten aufgeführt werden, die diese Pins verwenden.

[mcu]

Konfiguration des primären Mikrocontrollers.

```
[mcu]
serial:
# Die serielle Schnittstelle, die mit der MCU verbunden werden soll. Wenn Sie sich nicht sicher sind
```

```
# (oder wenn sie sich ändert), siehe Abschnitt "Wo ist meine serielle Schnittstelle?" der FAQ.
# Dieser Parameter muss angegeben werden, wenn eine serielle Schnittstelle verwendet wird.
#baud: 250000
# Die zu verwendende Baudrate. Die Voreinstellung ist 250000.
#canbus_uuid:
# Wenn Sie ein Gerät verwenden, das an einen CAN-Bus angeschlossen ist, wird hier der eindeutige
# Chip-Identifikator, mit dem verbunden werden soll. Dieser Wert muss angegeben werden, wenn der
# CAN-Bus für die Kommunikation.
#canbus_interface:
# Wenn ein an einen CAN-Bus angeschlossenes Gerät verwendet wird, wird hier die CAN
# Netzwerkschnittstelle zu verwenden. Der Standardwert ist 'can0'.
#restart_method:
# Dies steuert den Mechanismus, den der Host zum Zurücksetzen des
# Mikrocontroller zurückzusetzen. Die Auswahlmöglichkeiten sind 'arduino', 'cheetah', 'rpi_usb',
# und 'Befehl'. Die 'arduino'-Methode (DTR umschalten) ist üblich bei
# Arduino-Boards und Klonen üblich. Die 'Fyseth'-Methode ist eine spezielle Methode, die für einige
# Fysetc Cheetah-Boards benötigt wird. Die 'rpi_usb'-Methode
# ist nützlich für Raspberry Pi Boards mit Mikrocontrollern, die über USB versorgt werden - sie
# schaltet kurzzeitig die Stromversorgung aller USB-Ports ab, um einen Mikrocontroller-Reset
# durchzuführen.
# Die 'command'-Methode beinhaltet einen Klipper-Befehl an den Mikrocontroller zu senden, damit dieser
# selbst zurücksetzen kann. Die Vorgabe ist 'arduino', wenn der Mikrocontroller über eine serielle
# Schnittstelle kommuniziert, andernfalls 'command'.
```

[mcu my_extra_mcu]

Zusätzliche Mikrocontroller (man kann eine beliebige Anzahl von Abschnitten mit einem "mcu"-Präfix definieren). Mit zusätzlichen Mikrocontrollern werden zusätzliche Pins eingeführt, die als Heizungen, Stepper, Lüfter usw. konfiguriert werden können. Wenn zum Beispiel ein "[mcu extra_mcu]"-Abschnitt eingeführt wird, dann können Pins wie "extra_mcu:ar9" an anderer Stelle in der Konfiguration verwendet werden (wobei "ar9" ein Hardware-Pin-Name oder ein Alias-Name auf dem gegebenen mcu ist).

[mcu my_extra_mcu]

```
# Siehe den Abschnitt "mcu" für Konfigurationsparameter.
```

Allgemeine kinematische Einstellungen

[printer]

Der Abschnitt "printer" steuert übergeordnete Druckereinstellungen.

```
[printer]
Kinematik:
# Der Typ des verwendeten Druckers. Diese Option kann eine der folgenden sein: cartesian,
# corexy, corexz, hybrid_corexy, hybrid_corexz, rotary_delta, delta, polar, winch, oder keine. Dieser
# Parameter muss angegeben werden.
max_velocity:
# Maximale Geschwindigkeit (in mm/s) des Werkzeugkopfs (relativ zum Druck). Dieser Parameter muss
# angegeben werden.
max_accel:
# Maximale Beschleunigung (in mm/s^2) des Werkzeugkopfes (relativ zum Druck). Dieser Parameter muss
# angegeben werden.
#max_accel_to_decel:
# Eine Pseudobeschleunigung (in mm/s^2), die steuert, wie schnell der
# Werkzeugkopf von der Beschleunigung zur Verzögerung wechseln darf. Sie wird verwendet, um
# die Höchstgeschwindigkeit von kurzen Zick-Zack-Bewegungen zu reduzieren (und damit die
# Druckervibration durch diese Bewegungen). Die Vorgabe ist die Hälfte von max_accel.
#square_corner_velocity: 5.0
# Die maximale Geschwindigkeit (in mm/s), mit der der Werkzeugkopf eine 90
# Grad-Ecke fahren darf. Ein Wert ungleich Null kann Änderungen der Extruder-Durchflussmengen
# reduzieren, indem sie unmittelbare Geschwindigkeitsänderungen der Geschwindigkeit des Werkzeugkopfs
# während der Kurvenfahrt. Dieser Wert konfiguriert die interne Algorithmus für die Kurvenfahrt mit
# zentripetaler Geschwindigkeit; Ecken mit Winkeln größer als 90 Grad haben eine höhere
# Kurvengeschwindigkeit, während Kurven mit Winkeln kleiner als 90 Grad haben eine niedrigere
# Kurvengeschwindigkeit. Wenn dies auf Null gesetzt ist, wird der Werkzeugkopf an jeder Ecke auf Null
# abbremsen. Die Voreinstellung ist 5mm/s.
```

[stepper]

Schrittmotor-Definitionen. Verschiedene Druckertypen (wie durch die Option "kinematics" im Abschnitt [printer] config angegeben) erfordern unterschiedliche Namen für den Stepper (z.B. `stepper_x` vs. `stepper_a`). Nachfolgend finden Sie gängige Stepper-Definitionen.

Informationen zur Berechnung des Parameters rotation_distance finden Sie im Dokument rotation distance. Informationen über die Referenzfahrt mit mehreren Mikrocontrollern finden Sie im Dokument Multi-MCU-Homing.

```
[stepper_x]
step_pin:
# Schritt-GPIO-Pin (hoch getriggert). Dieser Parameter muss angegeben werden.
dir_pin:
# Richtungs-GPIO-Pin (High bedeutet positive Richtung). Dieser
# Parameter muss angegeben werden.
enable_pin:
# Enable-Pin (Standard ist enable high; verwenden Sie ! um enable
# low). Wenn dieser Parameter nicht angegeben wird, muss der Schrittmotor
# Treiber immer aktiviert sein.
rotation_distance:
# Abstand (in mm), den die Achse bei einer vollen Umdrehung des Schrittmotors (oder des letzten
# Getriebes, wenn gear_ratio angegeben ist).Dieser Parameter muss angegeben werden.
microsteps:
# Die Anzahl der Mikroschritte, die der Schrittmotortreiber verwendet. Dieser Parameter muss
# angegeben werden.
#fulll_steps_per_rotation: 200
# Die Anzahl der Vollschritte für eine Umdrehung des Schrittmotors. Setzen Sie diesen Wert auf 200 für
# einen 1,8-Grad-Schrittmotor oder auf 400 für einen 0,9-Grad-Motor. Die Voreinstellung ist 200.
#gear_ratio:
# Das Übersetzungsverhältnis, wenn der Schrittmotor über ein Getriebe mit der Achse verbunden ist
# Zum Beispiel kann man "5:1" angeben, wenn ein 5:1-Getriebe verwendet wird. Wenn die Achse mehrere
# Getriebe hat, kann man eine durch Komma getrennte Aufzählung von Getriebeübersetzungen angeben (zum #
Beispiel "57:11, 2:1").
# Wenn ein gear_ratio angegeben ist, gibt rotation_distance die den Weg, den die Achse für eine volle
# Umdrehung des letzten Gangs zurücklegt. Die Vorgabe ist, kein Übersetzungsverhältnis zu verwenden.
#step_pulse_duration:
# Die Mindestzeit zwischen der Schrittimpuls-Signalflanke und der folgenden "unstep"-Signalflanke.
# Dies wird auch verwendet, um die Mindestzeit zwischen einem Schrittimpuls und einem
# Richtungsänderungssignal festzulegen. Der Standardwert ist 0.000000100 (100ns) für TMC-Stepper,
# die im UART- oder SPI-Modus konfiguriert sind, und die Voreinstellung ist 0,000002 (das
# 2us beträgt) für alle anderen Stepper.
endstop_pin:
# Endstop-Schalter-Erkennungspin. Wenn dieser Endstop-Pin auf einer anderen CPU als der Schrittmotor,
# dann wird die "Multi-MCU Referenzfahrt". Dieser Parameter muss für die X-, Y- und Z-Stepper
# angegeben werden. Schrittmotoren auf kartesischen Druckern angegeben werden.
Position_min: 0
# Minimaler gültiger Abstand (in mm), zu dem der Benutzer den Stepper anweisen kann
# zu bewegen.  Die Vorgabe ist 0 mm.
position_endstop:
# Position des Endanschlags (in mm). Dieser Parameter muss angegeben werden
# für die X-, Y- und Z-Stepper bei kartesischen Druckern.
position_max:
# Maximal gültige Entfernung (in mm), die der Benutzer dem Stepper befehlen kann sich
# zu bewegen. Dieser Parameter muss für die X-, Y- und Z-Stepper auf kartesischen Druckern angegeben
# werden.
# Stepper auf kartesischen Druckern angegeben werden.
#homing_speed: 5.0
# Maximale Geschwindigkeit (in mm/s) des Steppers bei der Referenzfahrt. Die Vorgabe
# ist 5mm/s.
#homing_retract_dist: 5.0
# Abstand zum Rücklauf (in mm) vor der zweiten Referenzfahrt während der # Referenzfahrt.
# Referenzfahrt. Setzen Sie diesen Wert auf Null, um die zweite Referenzfahrt zu deaktivieren. Die
# Vorgabe ist 5mm.
#homing_retract_speed:
# Geschwindigkeit für die Rückzugsbewegung nach der # Referenzfahrt, falls diese Geschwindigkeit von
# der Referenzfahrtgeschwindigkeit abweicht, die die Vorgabe für diesen Parameter ist.
#second_homing_speed:
# Geschwindigkeit (in mm/s) des Steppers beim Ausführen der zweiten Referenzfahrt.
# Die Vorgabe ist homing_speed/2.
#homing_positive_dir:
# Wenn true, führt die Referenzfahrt dazu, dass sich der Stepper in eine positive
# Richtung (weg von Null); wenn false, wird der Stepper in Richtung Null referenziert. Es ist besser,
# die Vorgabe zu verwenden, als diesen Parameter anzugeben. Die Vorgabe ist true, wenn
# Position_endstop nahe position_max ist und false wenn sie in der Nähe von position_min liegt.
```

Kartesische Kinematik

Siehe example-cartesian.cfg für ein Beispiel einer kartesischen Kinematik-Konfigurationsdatei.

Hier werden nur Parameter beschrieben, die für kartesische Drucker spezifisch sind - siehe allgemeine Kinematikeinstellungen common kinematic settings für verfügbare Parameter.

```
[Drucker]
Kinematik: kartesisch
max_z_velocity:
# Hier wird die maximale Geschwindigkeit (in mm/s) der Bewegung entlang der z
# Achse. Diese Einstellung kann verwendet werden, um die maximale Geschwindigkeit des
# z-Schrittmotor zu begrenzen. Die Vorgabe ist, max_velocity für die z-Achse zu verwenden.
# max_z_velocity.
max_z_accel:
# Hier wird die maximale Beschleunigung (in mm/s^2) der Bewegung entlang der z-Achse. Sie begrenzt die
# Beschleunigung des z-Schrittmotors. Die Standard ist die Verwendung von max_accel für max_z_accel.
# Der Abschnitt stepper_x wird verwendet, um den Schrittmotor zu beschreiben, der
# die X-Achse in einem kartesischen Roboter.
[stepper_x]
# Der Abschnitt stepper_y wird verwendet, um den Stepper zu beschreiben, der
# die Y-Achse in einem kartesischen Roboter.
[stepper_y]
# Der Abschnitt stepper_z wird verwendet, um den Stepper zu beschreiben, der
# Z-Achse in einem kartesischen Roboter.
[stepper_z]
```

CoreXY Kinematik

Siehe example-corexz.cfg für eine Beispiel-Kinematikdatei für Corexy (und h-bot).
Hier werden nur Parameter beschrieben, die für Corexy-Drucker spezifisch sind - siehe common kinematic settings für verfügbare Parameter.

```
[Drucker]
Kinematik: corexy
max_z_velocity:
# Hier wird die maximale Geschwindigkeit (in mm/s) der Bewegung entlang der z
# Achse. Diese Einstellung kann verwendet werden, um die maximale Geschwindigkeit des
# z-Schrittmotor zu begrenzen. Die Vorgabe ist, max_velocity für die z-Achse zu verwenden.
# max_z_velocity.
max_z_accel:
# Hier wird die maximale Beschleunigung (in mm/s^2) der Bewegung entlang
# der z-Achse. Sie begrenzt die Beschleunigung des z-Schrittmotors. Die
# Standard ist die Verwendung von max_accel für max_z_accel.
# Der Abschnitt stepper_x wird verwendet, um die X-Achse sowie den
# Stepper, der die X+Y-Bewegung steuert.
[stepper_x]
# Der Abschnitt stepper_y dient zur Beschreibung der Y-Achse sowie des
# Stepper, der die X-Y-Bewegung steuert.
[stepper_y]
# Der Abschnitt stepper_z wird verwendet, um den Stepper zu beschreiben, der
# die Z-Achse.
[stepper_z]
```

Gemeinsamer Extruder und Heizbettträger

[extruder]

Der Abschnitt Extruder wird verwendet, um die Heizungsparameter für das Düsenheizgerät zusammen mit dem Stepper zu beschreiben, der den Extruder steuert. Siehe die command reference für zusätzliche Informationen. Informationen zur Einstellung des Druckvorschubs pressure advance guide finden Sie in der Anleitung zum Druckvorschub.

```
[extruder]
step_pin:
dir_pin:
enable_pin:
microsteps:
rotation_distance:
#full_steps_per_rotation:
#gear_ratio:
# Siehe den Abschnitt "stepper" für eine Beschreibung der obigen Parameter. Wenn keiner der obigen
# Parameter angegeben ist, kein Stepper mit der Düse verbunden (obwohl ein SYNC_EXTRUDER_MOTION-Befehl
# kann zur Laufzeit einen Schrittmacher zuordnen).
nozzle_diameter:
# Durchmesser der Düsenöffnung (in mm). Dieser Parameter muss angegeben werden.
filament_diameter:
# Der Nenndurchmesser des Rohfilaments (in mm), wenn es in den Extruder eintritt. Dieser Parameter
# muss angegeben werden.
#max_extrude_cross_section:
# Maximale Fläche (in mm^2) eines Extrusionsquerschnitts (z.B., Extrusionsbreite multipliziert mit
# Schichthöhe). Diese Einstellung verhindert übermäßige Extrusion bei relativ kleinen XY-Bewegungen.
# Wenn eine Bewegung eine Extrusionsrate anfordert, die diesen Wert überschreitet wird ein Fehler
```

```
# zurückgegeben. Die Vorgabe ist: 4.0 * nozzle_diameter^2
#instantaneous_corner_velocity: 1.000
# Die maximale momentane Geschwindigkeitsänderung (in mm/s) des
Extruders während der Verbindung von zwei Bewegungen. Die Vorgabe ist 1mm/s.
#max_extrude_only_distance: 50.0
# Maximale Länge (in mm des Rohfilaments), die eine Rückzugs- oder Nur-Extrudieren-Bewegung haben
# darf. Wenn eine Rückzugs- oder Nur-Extrudier-Bewegung eine größere Distanz als diesen Wert
# anfordert, wird ein Fehler Fehler zurückgegeben werden. Der Standardwert ist 50mm.
#max_extrude_only_velocity:
#max_extrude_only_accel:
# Maximale Geschwindigkeit (in mm/s) und Beschleunigung (in mm/s^2) des Extrudermotors für Rückzüge
# und reine Extrudierbewegungen. Diese Einstellungen haben keinen Einfluss auf normale
# Druckbewegungen. Wenn sie nicht angegeben, werden sie so berechnet, dass sie dem Limit einer XY
# Druckbewegung mit einem Querschnitt von 4.0*Düsen_Durchmesser^2
# haben.
#pressure_advance: 0.0
Die Menge des Rohfilaments, die während der Beschleunigung des # Extruders in den Extruder gedrückt
# wird. Extruderbeschleunigung in den Extruder gedrückt wird. Die gleiche Menge an Filament wird
# während der Verzögerung zurückgezogen. Sie wird in Millimetern pro Millimeter/Sekunde. Der
# Standardwert ist 0, was den Druckvorschub. deaktiviert.
#pressure_advance_smooth_time: 0.040
# Ein Zeitbereich (in Sekunden), der bei der Berechnung der durchschnittlichen Extrudergeschwindigkeit
# für den Druckvorschub. Ein größerer Wert führt zu glattere Extruderbewegungen. Dieser Parameter darf
# 200ms nicht überschreiten. Diese Einstellung gilt nur, wenn pressure_advance ungleich Null ist. Die
# Voreinstellung ist 0,040 (40 Millisekunden). Die übrigen Variablen beschreiben die Extruderheizung.
heater_pin:
# PWM-Ausgangspin zur Steuerung der Heizung. Dieser Parameter muss bereitgestellt werden.
#max_power: 1.0
# Die maximale Leistung (ausgedrückt als Wert von 0,0 bis 1,0), auf die der heater_pin eingestellt
# werden kann. Der Wert 1.0 erlaubt es, den Pin für längere Zeit voll aktiviert werden, während ein
# Wert von 0,5 der Pin nicht mehr als die Hälfte der Zeit aktiviert werden kann. Diese Einstellung
# kann verwendet werden, um die Gesamtleistungsabgabe (über längere Zeiträume) an die Heizung zu
# begrenzen. Der Standardwert ist 1.0.
sensor_type:
# Art des Sensors - gängige Thermistoren sind "EPCOS 100K B57560G104F",
# "ATC Semitec 104GT-2", "ATC Semitec 104NT-4-R025H42G", "Generic
# 3950", "Honeywell 100K 135-104LAG-J01", "NTC 100K MGB18-104F39050L32",
# "SliceEngineering 450" und "TDK NTCG104LH104JT1". Siehe den Abschnitt  "Temperatursensoren" für
# andere Sensoren. Dieser Parameter muss angegeben werden.
sensor_pin:
# Analogeingangspin, der mit dem Sensor verbunden ist. Dieser Parameter muss bereitgestellt werden.
#pullup_resistor: 4700
# Der Widerstand (in Ohm) des Pullups, der an den Thermistor angeschlossen ist. Dieser Parameter ist
# nur gültig, wenn der Sensor ein Thermistor ist. Die Voreinstellung ist 4700 Ohm.
#smooth_time: 1.0
# Ein Zeitwert (in Sekunden), über den die Temperaturmessungen geglättet werden, um die Auswirkungen
# von Messrauschen zu reduzieren. Die Vorgabe ist 1 Sekunde.
control:
# Steuerungsalgorithmus (entweder pid oder Wasserzeichen). Dieser Parameter muss angegeben werden.
pid_Kp:
pid_Ki:
pid_Kd:
# Die Proportional- (pid_Kp), Integral- (pid_Ki) und Ableitungs (pid_Kd) für das PID-Regelsystem. Klipper
bewertet die PID-Einstellungen mit der folgenden allgemeinen Formel:
# heater_pwm = (Kp*Fehler + Ki*Integral(Fehler) - Kd*Ableitung(Fehler)) / 255
# wobei "error" "angeforderte_Temperatur - gemessene_Temperatur" ist und "heater_pwm" die angeforderte
# Heizrate ist, wobei 0,0 für voll 0,0 voll ausgeschaltet und 1,0 voll eingeschaltet ist. Erwägen Sie
# die Verwendung des PID_CALIBRATE Befehl zu verwenden, um diese Parameter zu erhalten. Die Parameter
# pid_Kp, pid_Ki, und pid_Kd Parameter müssen für PID-Heizungen angegeben werden.
#max_delta: 2.0
# Bei 'watermark'-gesteuerten Heizgeräten ist dies die Anzahl der Grad in Celsius über der
# Solltemperatur, bevor die Heizung abgeschaltet wird. sowie die Anzahl der Grad unter der
# Zieltemperatur, bevor das Heizgerät wieder aktiviert wird. Der Standardwert ist 2 Grad Celsius.
#pwm_cycle_time: 0.100
# Zeit in Sekunden für jeden Software-PWM-Zyklus der Heizung. Es wird nicht empfohlen, diesen Wert zu setzen,
es sei denn, es gibt eine elektrische Anforderung, die Heizung schneller als 10 Mal pro Sekunde zu schalten.
Der Standardwert ist 0.100 Sekunden.
#min_extrude_temp: 170
# Die minimale Temperatur (in Celsius), bei der Extruder-Bewegungen
# Befehle ausgegeben werden dürfen. Die Vorgabe ist 170 Celsius.
min_temp:
max_temp:
# Der maximale Bereich gültiger Temperaturen (in Celsius), in dem das Heizgerät innerhalb dieses
```

```
# Bereichs bleiben muss. Dies steuert eine Sicherheitsfunktion im Code des Mikrocontrollers
# implementiert - sollte die gemessene Temperatur jemals außerhalb dieses Bereichs liegen, wird der
# Mikrocontroller in einen Abschaltzustand übergehen. Diese Prüfung kann helfen, einige Heizungs- und
# Sensor-Hardware-Fehler zu erkennen. Setzen Sie diesen Bereich gerade weit genug so, dass vernünftige
# Temperaturen nicht zu einem Fehler führen. Diese Parameter müssen angegeben werden.
```

[heater_bed]

Der Abschnitt "heater_bed" beschreibt ein beheiztes Bett. Es werden die gleichen Heizungseinstellungen verwendet, die im Abschnitt "Extruder" beschrieben sind.

```
[heater_bed]
heater_pin:
sensor_type:
sensor_pin:
Steuerung:
min_temp:
max_temp:
# Siehe den Abschnitt "Extruder" für eine Beschreibung der oben genannten Parameter.
```

Bed level support

[bed_mesh]

Mesh Bed Leveling. Man kann einen bed_mesh-Konfigurationsabschnitt definieren, um Bewegungstransformationen zu aktivieren, die die z-Achse auf der Grundlage eines Netzes versetzen, das aus angetasteten Punkten erzeugt wurde. Wenn eine Sonde zum Ausrichten der z-Achse verwendet wird, empfiehlt es sich, in der Datei printer.cfg einen Abschnitt safe_z_home zu definieren, um das Bett in der Mitte des Druckbereichs auszurichten.

Weitere Informationen finden Sie in der Anleitung zum Bettnetz bed mesh guide und in der Befehlsreferenz command reference.

Visuelle Beispiele:

```
 Rechteckiges Bett, probe_count = 3, 3:
             x---x---x (max_point)
             |
             x---x---x
                     |
 (min_punkt) x---x---x

 round bed, round_probe_count = 5, bed_radius = r:
                  x (0, r) end
                /
             x---x---x
                       \
 (-r, 0) x---x---x---x---x (r, 0)
           \
             x---x---x
                   /
                  x (0, -r) start
```

```
[bed_mesh]
#speed: 50
# Die Geschwindigkeit (in mm/s) der nicht-probierenden Bewegungen während der Kalibrierung.
# Die Voreinstellung ist 50.
#horizontal_move_z: 5
# Die Höhe (in mm), auf die sich der Kopf bewegen soll kurz vor dem Start eines Antastvorgangs. Die
# Voreinstellung ist 5.
#mesh_radius:
# Legt den Radius des Netzes fest, das für runde Betten sondiert werden soll. Beachten Sie, dass
# der Radius relativ zu der Koordinate ist, die durch die Option mesh_origin. Dieser Parameter muss für runde
Betten angegeben werden und für rechteckige Betten weggelassen werden.
#mesh_origin:
# Definiert die mittlere X- und Y-Koordinate des Netzes für runde Betten. Diese
# Koordinate ist relativ zum Standort der Sonde. Es kann sinnvoll sein
# den Maschenursprung anzupassen, um die Größe des Maschenradius zu maximieren
# Maschenradius zu maximieren. Die Vorgabe ist 0, 0. Dieser Parameter muss bei # rechteckigen Betten
# weggelassen werden.
#mesh_min:
# Definiert die minimale X-, Y-Koordinate des Netzes für rechteckige Betten. Diese Koordinate ist
# relativ zur Position der Sonde. Diese wird der erste Punkt sein, der dem Ursprung am nächsten liegt.
# Dieser Parameter muss für rechteckige Betten angegeben werden.
#mesh_max:
```

```
# Definiert die maximale X-, Y-Koordinate des Netzes für rechteckige Betten. Es gilt dasselbe Prinzip
# wie bei mesh_min, allerdings wird dies der am weitesten vom Ursprung des Bettes entfernte Punkt
# sein. Dieser Parameter muss für rechteckige Betten angegeben werden.
#probe_count: 3, 3
# Für rechteckige Betten ist dies ein durch Komma getrenntes Paar von Ganzzahlwerten Werte X, Y, die
# die Anzahl der zu untersuchenden Punkte entlang jeder Achse. Ein einzelner Wert ist auch gültig, in
# diesem Fall wird dieser Wert auf beide Achsen angewandt wird. Voreinstellung ist 3, 3.
#round_probe_count: 5
# Für runde Betten definiert dieser Integer-Wert die maximale Anzahl von Punkte, die entlang jeder
# Achse zu sondieren sind. Dieser Wert muss eine ungerade Zahl sein. Voreinstellung ist 5.
#fade_start: 1.0
# Die Gcode-Z-Position, an der das Ausblenden der Z-Anpassung beginnen soll, wenn Fade aktiviert ist.
# Die Vorgabe ist 1.0.
#fade_end: 0.0
# Die gcode z-Position, an der das Phasing Out abgeschlossen wird. Wenn sie auf einen Wert
# unter fade_start gesetzt wird, ist die Ausblendung deaktiviert. Es ist zu beachten, dass
# Fade zu einer unerwünschten Skalierung entlang der z-Achse eines Drucks führen kann. Wenn ein
# Benutzer die Überblendung aktivieren möchte, wird ein Wert von 10.0 empfohlen.
# Die Voreinstellung ist 0.0, das die Überblendung deaktiviert.
#fade_target:
# Die z-Position, in der die Überblendung konvergieren soll. Wenn dieser Wert auf einen Wert ungleich
# Null gesetzt wird, muss er innerhalb des Bereichs der z-Werte in dem Netz liegen. Benutzer, die auf
# die z-Homing-Position konvergieren wollen Standard ist der durchschnittliche z-Wert des Netzes.
#split_delta_z: .025
# Der Betrag der Z-Differenz (in mm) entlang einer Bewegung, der eine eine Teilung auslöst.
# Voreinstellung ist .025.
#move_check_distance: 5.0
# Der Abstand (in mm) entlang eines Zuges, der auf split_delta_z geprüft wird.
# Dies ist auch die Mindestlänge, die ein Zug geteilt werden kann. Voreinstellung ist 5.0.
#mesh_pps: 2, 2
# Ein durch Kommata getrenntes Paar von ganzen Zahlen X, Y, die die Anzahl der Punkte pro Segment, die
# im Netz entlang jeder Achse interpoliert werden sollen. Ein "Segment" kann als der Raum zwischen
# den einzelnen Messpunkten definiert werden. Der Benutzer kann einen einzigen Wert eingeben, der auf
# beide Achsen angewendet wird. Achsen angewendet wird. Die Voreinstellung ist 2, 2.
#Algorithmus: Lagrange
# Der zu verwendende Interpolationsalgorithmus. Kann entweder "lagrange" oder "bikubisch" sein. Diese
# Option wirkt sich nicht auf 3x3-Gitter aus, die gezwungen sind Lagrange-Sampling zu verwenden.
# Voreinstellung ist lagrange.
#bicubic_tension: .2
# Bei Verwendung des bikubischen Algorithmus kann der obige Parameter tension der obige Parameter
# tension angewendet werden, um den Anteil der interpolierten Steigung zu ändern. Größere Zahlen
# erhöhen die Neigung, was zu einer stärkeren Krümmung im Netz führt. Die Voreinstellung ist .2.
#relativer_referenz_index:
# Ein Punktindex im Netz, auf den sich alle z-Werte beziehen. Aktivieren von dieses Parameters wird
# ein Netz relativ zur gemessenen z-Position am angegebenen Index.
#faulty_region_1_min:
#faulty_region_1_max:
# Optionale Punkte, die eine fehlerhafte Region definieren.  Siehe docs/Bed_Mesh.md
# für Details zu fehlerhaften Regionen.  Es können bis zu 99 fehlerhafte Regionen hinzugefügt werden.
# Standardmäßig sind keine fehlerhaften Regionen gesetzt.
```

[bed_tilt]

Kompensation der Bettneigung. Man kann einen bed_tilt-Konfigurationsabschnitt definieren, um Bewegungstransformationen zu ermöglichen, die ein geneigtes Bett berücksichtigen. Beachten Sie, dass bed_mesh und bed_tilt nicht kompatibel sind; beide können nicht definiert werden.

Siehe die Befehlsreferenz command reference für weitere Informationen.

```
[bed_tilt]
#x_adjust: 0
# Der Betrag, der zur Z-Höhe jeder Bewegung für jeden mm auf der X-Achse addiert wird. Achse. Die
# Vorgabe ist 0.
#y_adjust: 0
# Der Betrag, der für jeden mm auf der Y-Achse zur Z-Höhe jeder Bewegung addiert wird. Die Vorgabe ist 0.
#z_adjust: 0
# Der Betrag, der zur Z-Höhe addiert wird, wenn die Düse nominell auf 0, 0 steht. 0, 0. Die Vorgabe
# ist 0. Die übrigen Parameter steuern einen BED_TILT_CALIBRATE extended g-code-Befehl, der zur
# Kalibrierung der entsprechenden x- und y-Parameter verwendet werden kann. Anpassungsparameter zu
# kalibrieren.
#Punkte:
# Eine Aufzählung von X-, Y-Koordinaten (eine pro Zeile; nachfolgende Zeilen eingerückt), die während
# eines BED_TILT_CALIBRATE-Befehls abgetastet werden sollen. Befehls abgetastet werden sollen. Geben
# Sie die Koordinaten der Düse an und stellen Sie sicher, dass die Sonde über dem Bett an den
```

```
# angegebenen Düsenkoordinaten befindet. Die Vorgabe ist den Befehl nicht zu aktivieren.
#Geschwindigkeit: 50
# Die Geschwindigkeit (in mm/s) der Nicht-Sondierbewegungen während der Kalibrierung. Die Vorgabe ist # 50.
#horizontal_move_z: 5
# Die Höhe (in mm), auf die sich der Kopf bewegen soll kurz vor dem Start eines Antastvorgangs. Die
# Voreinstellung ist 5.
```

[bed_screws]

Werkzeug zum Einstellen der Bettnivellierschrauben. Man kann einen [bed_screws]-Konfigurationsabschnitt definieren, um einen BED_SCREWS_ADJUST g-code Befehl zu aktivieren.

Weitere Informationen finden Sie in der Nivellierungsanleitung leveling guide und der Befehlsreferenz command reference.

```
[bed_screws]
#screw1:
# Die X-, Y-Koordinate der ersten Bettnivellierschraube. Dies ist eine Position für die Düse, die sich direkt
über der Bettschraube befindet.
# Schraube (oder so nah wie möglich, aber immer noch über dem Bett). Dieser Parameter muss angegeben
# werden.
#Schnecke1_name:
# Ein beliebiger Name für die angegebene Schnecke. Dieser Name wird angezeigt, wenn das Hilfsskript läuft. Die
Vorgabe ist, einen Namen zu verwenden, der auf der XY-Position der Schraube.
#Schraube1_fein_einstellen:
# Eine X- und Y-Koordinate, die die Düse ansteuern soll, damit man die Feinabstimmung der
# Bettnivellierschraube vorzunehmen. Die Vorgabe ist, keine Feineinstellung Feineinstellungen an der
# Bettschraube vorzunehmen.
#Schraube2:
#screw2_name:
#screw2_fine_adjust:
#...
# Zusätzliche Bettnivellierschrauben. Es müssen mindestens drei Schrauben
# definiert sein.
#horizontal_move_z: 5
# Die Höhe (in mm), auf die sich der Kopf bewegen soll wenn er sich von einer Schraubenposition zur
# nächsten bewegt. Die Vorgabe ist 5.
#probe_height: 0
# Die Höhe der Sonde (in mm) nach Anpassung an die thermische Ausdehnung von Bett und Düse. Die
# Voreinstellung ist Null.
#speed: 50
# Die Geschwindigkeit (in mm/s) der Nicht-Sondierbewegungen während der Kalibrierung. Die
# Voreinstellung ist 50.
#probe_speed: 5
# Die Geschwindigkeit (in mm/s) bei der Bewegung von einer horizontal_move_z Position zu einer
# probe_height-Position. Die Voreinstellung ist 5.
```

[screws_tilt_adjust].

Werkzeug zur Einstellung der Neigung der Bettschrauben mit dem Z-Taster. Man kann einen screws_tilt_adjust Konfigurationsabschnitt definieren, um einen SCREWS_TILT_CALCULATE g-code Befehl zu aktivieren.

Weitere Informationen finden Sie in der Nivellierungsanleitung leveling guide und der Befehlsreferenz command reference.

```
[screws_tilt_adjust]
#screw1:
# Die (X, Y) Koordinate der ersten Bettnivellierschraube. Dies ist eine Position, die die Düse
# ansteuern soll, so dass sich die Sonde direkt
# über der Bettschraube befindet (oder so nah wie möglich, aber immer noch über dem Bett befindet). Dies ist
die für die Berechnungen verwendete Basisschraube. Dieser
# Parameter muss angegeben werden.
#Schraube1_name:
# Ein beliebiger Name für die angegebene Schraube. Dieser Name wird angezeigt, wenn das Hilfsskript läuft. Die
Vorgabe ist, einen Namen zu verwenden, der auf der XY-Position der Schraube.
#Schraube2:
#Schraube2_name:
#...
# Zusätzliche Bettnivellierschrauben. Es müssen mindestens zwei Schrauben definiert sein.
#speed: 50
# Die Geschwindigkeit (in mm/s) der Nicht-Probing-Bewegungen während der Kalibrierung.
# Die Voreinstellung ist 50.
#horizontal_move_z: 5
```

```
# Die Höhe (in mm), auf die sich der Kopf bewegen soll kurz vor dem Start eines Antastvorgangs. Die
# Voreinstellung ist 5.
#Schraubengewinde: CW-M3
# Die Art der Schraube, die für die Betthöhe verwendet wird, M3, M4 oder M5 und die Richtung des Knopfes, mit
dem das Bett nivelliert wird, im Uhrzeigersinn verringern gegen den Uhrzeigersinn abnehmen. Akzeptierte Werte:
CW-M3, CCW-M3, CW-M4, CCW-M4, CW-M5, CCW-M5.
# Standardwert ist CW-M3, die meisten Drucker verwenden eine M3-Schraube und Drehen des Knopfes im
Uhrzeigersinn verringert den Abstand.
```

[z_tilt]

Einstellung der Neigung mehrerer Z-Stepper. Diese Funktion ermöglicht die unabhängige Einstellung mehrerer Z-Stepper (siehe Abschnitt "stepper_z1") zur Anpassung der Neigung. Wenn dieser Abschnitt vorhanden ist, wird ein erweiterter G-Code-Befehl Z_TILT_ADJUST verfügbar.

```
[z_tilt]
#z_positions:
# Eine Aufzählung von X-, Y-Koordinaten (eine pro Zeile; nachfolgende Zeilen eingerückt), die die
# Position jedes Bett-"Drehpunkts" beschreiben. Der "Pivot-Punkt" ist der Punkt, an dem das Bett mit
# dem angegebenen # Z-Stepper verbunden ist. Stepper. Er wird mit Düsenkoordinaten beschrieben
# (die X-,Y-Position der Düse, wenn sie sich direkt über dem Punkt bewegen könnte). Der erste Eintrag
# entspricht stepper_z, der zweite stepper_z1, der dritte für stepper_z2, usw. Dieser Parameter muss
#angegeben werden.
#Punkte:
# Eine Aufzählung von X-, Y-Koordinaten (eine pro Zeile; nachfolgende Zeilen eingerückt), die bei
# einem Z_TILT_ADJUST-Befehl abgetastet werden sollen. Geben Sie die Koordinaten der Düse an und
# stellen Sie sicher, dass sich die Sonde über dem Bett an den angegebenen Düsenkoordinaten befindet.
#  Dieser Parameter muss angegeben werden.
#Geschwindigkeit: 50
# Die Geschwindigkeit (in mm/s) der Nicht-Sondierbewegungen während der Kalibrierung.
# Die Vorgabe ist 50.
#horizontal_move_z: 5
# Die Höhe (in mm), auf die sich der Kopf bewegen soll kurz vor dem Start eines Antastvorgangs. Die
# Voreinstellung ist 5.
#Wiederholungen: 0
# Anzahl der Wiederholungen, wenn die angetasteten Punkte nicht innerhalb der Toleranz liegen.
#retry_tolerance: 0
# Wenn Wiederholungen aktiviert sind, wird der Versuch wiederholt, wenn der größte und der kleinste
# ermittelte Punkte mehr als die retry_tolerance abweichen. Beachten Sie, dass die kleinste Einheit
# der Änderung wäre hier ein einzelner Schritt. Wenn Sie jedoch mehr Punkte als Schritte, dann haben
# Sie wahrscheinlich einen festen Mindestwert für den Bereich der getesteten Punkte, den Sie durch
# Beobachtung der Befehlsausgabe.
```

[quad_gantry_level]

Nivellierung eines beweglichen Portals mit 4 unabhängig voneinander gesteuerten Z-Motoren. Korrigiert hyperbolische Parabel-Effekte (Kartoffelchips) auf einem beweglichen Portal, das flexibler ist. WARNUNG: Die Verwendung dieser Funktion auf einem beweglichen Bett kann zu unerwünschten Ergebnissen führen. Wenn dieser Abschnitt vorhanden ist, wird ein erweiterter G-Code-Befehl QUAD_GANTRY_LEVEL verfügbar. Diese Routine geht von der folgenden Z-Motor-Konfiguration aus:

```
----------------
|Z1            Z2|
|   ---------   |
|   |       |   |
|   |       |   |
|   x--------   |
|Z           Z3 |
----------------
```

Dabei ist x der Punkt 0, 0 auf dem Bett

```
[quad_gantry_level]
#gantry_corners:
# Eine durch Zeilenumbruch getrennte Aufzählung von X- und Y-Koordinaten, die die beiden
# gegenüberliegenden Ecken des Gantrys beschreiben. Der erste Eintrag entspricht Z, der zweite Z2.
# Dieser Parameter muss angegeben werden.
#Punkte:
# Eine durch Zeilenumbrüche getrennte Aufzählung von vier X-, Y-Punkten, die während eines
# QUAD_GANTRY_LEVEL-Befehls. Die Reihenfolge der Punkte ist wichtig und sollte den Positionen Z, Z1,
# Z2 und Z3 in Reihenfolge entsprechen. Dieser Parameter muss angegeben werden. Für maximale
# Genauigkeit, stellen Sie sicher, dass Ihre Sondenversätze konfiguriert sind.
#Geschwindigkeit: 50
# Die Geschwindigkeit (in mm/s) der Nicht-Antastbewegungen während der Kalibrierung.
```

```
# Der Standardwert ist 50.
#Horizontale_Verschiebung_z: 5
# Die Höhe (in mm), auf die sich der Kopf bewegen soll
# kurz vor dem Start eines Antastvorgangs. Die Voreinstellung ist 5.
#max_adjust: 4
# Sicherheitsgrenze, wenn eine Justierung größer als dieser Wert angefordert wird.
# quad_gantry_level wird abgebrochen.
#wiederholungen: 0
# Anzahl der Wiederholungen, wenn die angefragten Punkte nicht innerhalb der Toleranz liegen.
#retry_tolerance: 0
# Wenn Wiederholungen aktiviert sind, wird der Versuch wiederholt, wenn der größte und der kleinste
# ermittelte Punkte um mehr als die retry_tolerance abweichen.
```

[skew_correction]

Drucker-Schräglage-Korrektur. Es ist möglich, die Schräglage des Druckers über 3 Ebenen (xy, xz, yz) per Software zu korrigieren. Dazu wird ein Kalibrierungsmodell entlang einer Ebene gedruckt und drei Längen gemessen. Aufgrund der Art der Schräglagenkorrektur werden diese Längen über den G-Code eingestellt. Siehe Schräglagenkorrektur und Befehlsreferenz für Details.

```
[skew_correction]
```

Benutzerdefinierte Referenzfahrt

[safe_z_home]

Sichere Z-Referenzierung. Mit diesem Mechanismus kann man die Z-Achse an einer bestimmten X- und Y-Koordinate ausrichten. Dies ist nützlich, wenn der Werkzeugkopf z.B. in die Mitte des Bettes fahren muss, bevor die Z-Achse referenziert werden kann.

```
[safe_z_home]
home_xy_position:
# Eine X-, Y-Koordinate (z.B. 100, 100), an der die Z-Homing-Funktion ausgeführt werden soll.
# durchgeführt werden soll. Dieser Parameter muss angegeben werden.
#Geschwindigkeit: 50.0
# Geschwindigkeit, mit der der Werkzeugkopf zur sicheren Z-Home-Position bewegt wird.
# Koordinate bewegt wird. Der Standardwert ist 50 mm/s.
#z_hop:
# Abstand (in mm), um die Z-Achse vor der Referenzfahrt anzuheben. Dies wird auf jeden
# Referenzfahrtbefehl angewendet, auch wenn er die Z-Achse nicht referenziert. Wenn die Z-Achse
# bereits referenziert ist und die aktuelle Z-Position kleiner ist z_hop, dann wird der Kopf auf eine
# Höhe von z_hop angehoben. Wenn die Z-Achse nicht bereits referenziert ist, wird der Kopf um z_hop
# angehoben. Die Voreinstellung ist, dass kein Z-Hop implementiert wird.
#z_hop_speed: 20.0
# Geschwindigkeit (in mm/s), mit der die Z-Achse vor der Referenzfahrt angehoben wird. Die
# Voreinstellung ist 20mm/s.
#move_to_previous: False
# Wenn auf True gesetzt, werden die X- und Y-Achsen nach der Referenzfahrt der Z-Achse auf ihre vorherigen
Positionen zurückgesetzt.
# Positionen nach der Referenzfahrt der Z-Achse zurückgesetzt. Die Vorgabe ist False.
```

[homing_override]

Homing override. Dieser Mechanismus kann verwendet werden, um eine Reihe von G-Code-Befehlen anstelle eines G28 in der normalen G-Code-Eingabe auszuführen. Dies kann bei Druckern nützlich sein, die eine bestimmte Prozedur zum Referenzieren des Geräts erfordern.

```
[homing_override]
gcode:
# Eine Aufzählung von G-Code-Befehlen, die anstelle von G28-Befehlen ausgeführt werden
# die in der normalen G-Code-Eingabe zu finden sind. Siehe docs/Command_Templates.md
# für das G-Code-Format. Wenn ein G28 in dieser Aufzählung von Befehlen enthalten ist
# dann wird das normale Referenzfahrtverfahren für den Drucker aufgerufen.
# Die hier aufgeführten Befehle müssen alle Achsen referenzieren. Dieser Parameter muss
# angegeben werden.
#Achsen: xyz
# Die Achsen, die außer Kraft gesetzt werden sollen. Wenn dieser Parameter zum Beispiel auf "z"
# gesetzt ist, wird das Skript nur ausgeführt, wenn die z-Achse referenziert wird (z.B. durch Befehl
#  "G28" oder "G28 Z0"). Beachten Sie, dass das Override-Skript immer noch alle Achsen referenzieren.
# Die Vorgabe ist "xyz", wodurch das Überschreibungsskript anstelle aller G28-Befehle ausgeführt wird.
#set_position_x:
#set_position_y:
#set_position_z:
# Wenn angegeben, nimmt der Drucker an, dass sich die Achse an der angegebenen Position, bevor er die
# oben genannten G-Code-Befehle ausführt. Das Setzen dieser wird die Überprüfung der Referenzfahrt für
# diese Achse deaktiviert. Dies kann nützlich sein, wenn der Kopf bewegt werden muss, bevor der
# normale G28-Mechanismus für eine Achse aufgerufen wird. Die Voreinstellung ist, dass keine Position
```

```
# für eine Achse erzwungen wird.
```

[endstop_phase]

Endstopps für Schrittmotorphasen. Um diese Funktion zu verwenden, definieren Sie eine Konfigurationssektion mit einem "endstop_phase"-Präfix, gefolgt vom Namen der entsprechenden Stepper-Konfigurationssektion (zum Beispiel "[endstop_phase stepper_z]"). Diese Funktion kann die Genauigkeit der Endstopp-Schalter verbessern. Fügen Sie eine reine "[endstop_phase]"-Deklaration hinzu, um den Befehl ENDSTOP_PHASE_CALIBRATE zu aktivieren.

Weitere Informationen finden Sie in der Anleitung für Endstopp-Phasen endstop phases guide und in der Befehlsreferenz command reference.

```
[endstop_phase stepper_z]
#endstop_accuracy:
# Legt die erwartete Genauigkeit (in mm) des Endstopps fest. Dies stellt die maximale Fehlerdistanz,
# die der Endstopp auslösen kann (z.B. wenn ein (z.B. wenn ein Endstopp gelegentlich 100um zu früh
# oder bis zu 100um zu spät auslöst dann setzen Sie dies auf 0.200 für 200um). Der Standardwert ist
# 4*rotation_distance/full_steps_per_rotation.
#trigger_phase:
# Dies gibt die Phase des Schrittmotortreibers an, die erwartet wird wenn der Endanschlag erreicht
# wird. Sie besteht aus zwei Zahlen, die durch einen Schrägstrich getrennt - die Phase und die
# Gesamtzahl der Phasen (z.B. "7/64"). Setzen Sie diesen Wert nur, wenn Sie sicher sind, dass der
# Schrittmotortreiber jedes Mal zurückgesetzt wird, wenn die # MCU zurückgesetzt wird. Wenn dieser
# nicht eingestellt ist, wird die Schrittmotorphase bei der ersten Phase erkannt und diese Phase wird
# bei allen folgenden Referenzfahrten verwendet.
#endstop_align_zero: False
# Wenn true, dann wird der Positionsendanschlag der Achse effektiv
# so geändert, dass die Nullposition für die Achse bei einem vollen Schritt
# Schritt des Schrittmotors erfolgt. (Wenn auf der Z-Achse verwendet und die Druckhöhe
# (Wenn auf der Z-Achse verwendet und die Höhe der Druckschicht ein Vielfaches eines Vollschritts ist, # wird
jede Ebene bei einem Vollschritt auf.) Die Voreinstellung ist False.
```

G-Code-Makros und Ereignisse

[gcode_macro]

G-Code-Makros (man kann eine beliebige Anzahl von Abschnitten mit dem Präfix "gcode_macro" definieren). Weitere Informationen finden Sie in der Anleitung zur Befehlsvorlage command template guide.

```
[gcode_macro my_cmd]
#gcode:
# Eine Aufzählung von G-Code-Befehlen, die anstelle von "my_cmd" ausgeführt werden sollen. Siehe
# docs/Command_Templates.md für das G-Code-Format. Dieser Parameter muss angegeben werden.
#variable_<name>:
# Man kann eine beliebige Anzahl von Optionen mit einem "variable_"-Präfix angeben. Der angegebene
# Variablenname wird mit dem angegebenen Wert belegt (geparst als Python-Literal geparst) und ist
# während der Makroexpansion verfügbar. Eine Konfiguration mit "variable_fan_speed = 75" könnte zum
# Beispiel gcode-Befehle enthalten "M106 S{ fan_speed * 255 }". Variablen kann zur Laufzeit mit dem
# Befehl SET_GCODE_VARIABLE geändert werden. (siehe docs/Command_Templates.md für Details).
# Variablennamen dürfen keine Großbuchstaben verwenden.
#rename_existing:
# Diese Option veranlasst das Makro, einen bestehenden G-Code-Befehl zu überschreiben G-Code-Befehl
# überschreibt und die vorherige Definition des Befehls über den hier angegebenen Namen. Dies kann
# verwendet werden, um eingebaute G-Code # Befehle zu überschreiben. Befehle zu überschreiben. Beim
# Überschreiben von Befehlen ist Vorsicht geboten, denn es kann zu komplexen und unerwarteten
# Ergebnissen führen kann. Die Vorgabe ist, dass kein einen bestehenden G-Code-Befehl zu
# überschreiben.
#Beschreibung: G-Code-Makro
# Dies fügt eine kurze Beschreibung hinzu, die beim HELP-Befehl oder bei der der Autovervollständigung
# verwendet wird. Standard "G-Code-Makro"
```

[delayed_gcode]

Führt einen G-Code mit einer bestimmten Verzögerung aus. Weitere Informationen finden Sie in der Anleitung zur Befehlsvorlage command template guide und in der Befehlsreferenz command reference.

```
[delayed_gcode my_delayed_gcode]
gcode:
# Eine Aufzählung von G-Code-Befehlen, die ausgeführt werden, wenn die Verzögerungsdauer verstrichen
# ist. G-Code-Vorlagen werden unterstützt. Dieser Parameter muss angegeben werden.
#initial_duration: 0.0
# Die Dauer der anfänglichen Verzögerung (in Sekunden). Wenn er auf einen Wert Nicht-Null-Wert
# gesetzt, wird der delayed_gcode die angegebene Anzahl Anzahl von Sekunden ausgeführt, nachdem der
# Drucker in den Zustand "bereit" eingetreten ist. Dies kann für Initialisierungsprozeduren oder einen
# sich wiederholenden delayed_gcode # nützlich sein. Ist der Wert 0, wird der delayed_gcode beim Start
```

```
# nicht ausgeführt. Voreinstellung ist 0.
```

[save_variables]

Unterstützt das Speichern von Variablen auf der Festplatte, so dass sie über Neustarts hinweg erhalten bleiben. Siehe Befehlsvorlagen command templates und in der Befehlsreferenz G-Code reference für weitere Informationen.

```
[save_variables]
filename:
# Erforderlich - geben Sie einen Dateinamen an, der verwendet wird, um die Variablen auf der
# Festplatte zu speichern, z. B. ~/variables.cfg
```

[idle_timeout]

Idle timeout (Leerlaufzeit). Eine Leerlaufzeitüberschreitung ist automatisch aktiviert - fügen Sie einen expliziten Konfigurationsabschnitt idle_timeout hinzu, um die Standardeinstellungen zu ändern.

```
[idle_timeout]
#gcode:
# Eine Aufzählung von G-Code-Befehlen, die bei einer Leerlaufzeitüberschreitung ausgeführt werden
# sollen. Siehe docs/Command_Templates.md für das G-Code-Format. Die Vorgabe ist die Ausführung von
# "TURN_OFF_HEATERS" und "M84".
#Timeout: 600
# Leerlaufzeit (in Sekunden), die vor der Ausführung der oben genannten G-Code Befehle ausgeführt
# werden. Der Standardwert ist 600 Sekunden.
```

Optionale G-Code-Funktionen

[virtual_sdcard]

Eine virtuelle SD-Karte kann nützlich sein, wenn der Host-Rechner nicht schnell genug ist, um OctoPrint gut auszuführen. Sie ermöglicht es der Klipper-Hostsoftware, G-Code-Dateien, die in einem Verzeichnis auf dem Host gespeichert sind, mit Standard-Sdcard-G-Code-Befehlen (z. B. M24) direkt zu drucken.

```
[virtual_sdcard]
Pfad:
# Der Pfad des lokalen Verzeichnisses auf dem Hostrechner, in dem nach G-Code-Dateien zu suchen. Dies
# ist ein schreibgeschütztes Verzeichnis (sdcard file writes werden nicht unterstützt). Man kann hier
# auf das Upload-Verzeichnis von OctoPrint verweisen Verzeichnis von OctoPrint verweisen (im
# Allgemeinen ~/.octoprint/uploads/ ). Dieser Parameter muss angegeben werden.
```

[sdcard_loop]

Bei einigen Druckern mit Stage-Clearing-Funktionen, wie z. B. einem Teileauswerfer oder einem Banddrucker, kann es sinnvoll sein, Abschnitte der sdcard-Datei in einer Schleife zu drucken. (Zum Beispiel, um dasselbe Teil immer wieder zu drucken, oder um einen Abschnitt eines Teils für eine Kette oder ein anderes wiederholtes Muster zu wiederholen).

Siehe die Befehlsreferenz command reference für unterstützte Befehle. In der Datei sample-macros.cfg sample-macros.cfg finden Sie ein Marlin-kompatibles M808-G-Code-Makro.

```
[sdcard_loop]
```

[force_move]

Unterstützt das manuelle Bewegen von Schrittmotoren zu Diagnosezwecken. Beachten Sie, dass die Verwendung dieser Funktion den Drucker in einen ungültigen Zustand versetzen kann - siehe die Befehlsreferenz command reference für wichtige Details.

```
[force_move]
#enable_force_move: False
# Auf true setzen, um FORCE_MOVE und SET_KINEMATIC_POSITION zu aktivieren.
# erweiterte G-Code-Befehle. Die Voreinstellung ist false.
```

[pause_resume]

Pause/Resume-Funktionalität mit Unterstützung von Positionserfassung und -wiederherstellung. Siehe die Befehlsreferenz command reference für weitere Informationen.

```
pause_resume: [pause_resume]
#recover_velocity: 50.
# Wenn Capture/Restore aktiviert ist, die Geschwindigkeit, mit der zur
# der erfassten Position zurückkehren soll (in mm/s). Die Voreinstellung ist 50,0 mm/s.
```

[firmware_retraction]

Firmware-Filament-Rückzug. Dies ermöglicht die GCODE-Befehle G10 (Einfahren) und G11 (Ausfahren), die von vielen Slicern ausgegeben werden. Die nachstehenden Parameter sind Standardwerte für den Start, können aber mit dem Befehl SET_RETRACTION command angepasst werden, so dass Einstellungen für jedes einzelne Filament und die Laufzeit vorgenommen werden können.

```
[firmware_retraction]
#retract_length: 0
# Die Länge des Filaments (in mm), das bei Aktivierung von G10 zurückgezogen wird, und nicht
# zurückgezogen werden soll, wenn G11 aktiviert ist (siehe aber unretract_extra_length unten). Die
# Voreinstellung ist 0 mm.
#retract_speed: 20
# Die Geschwindigkeit des Einzugs in mm/s. Die Vorgabe ist 20 mm/s.
#unretract_extra_length: 0
# Die Länge (in mm) des *zusätzlichen* Filaments, beim Vorschub des Filaments.
#unretract_speed: 10
# Die Geschwindigkeit des Vorschubs in mm/s. Die Vorgabe ist 10 mm/s.
```

[gcode_arcs]

Unterstützung für gcode arc (G2/G3) Befehle.

```
[gcode_arcs]
#Auflösung: 1.0
# Ein Bogen wird in Segmente aufgeteilt. Die Länge eines jeden Segments wird entspricht der oben
# eingestellten Auflösung in mm. Niedrigere Werte erzeugen einen feineren Bogen, aber auch mehr Arbeit
# für Ihre Maschine. Bögen, die kleiner sind als als der eingestellte Wert, werden zu geraden Linien.
# Der Standardwert ist 1mm.
```

[respond]

Aktivieren Sie die erweiterten Befehle "M118" und "RESPOND" commands.

```
[respond]
#default_type: echo
# Setzt das Standardpräfix der "M118"- und "RESPOND"-Ausgabe auf eine
# der folgenden:
# echo: "echo: " (Dies ist die Voreinstellung)
# Befehl: "// "
# error: "!! "
#default_prefix: echo:
# Setzt direkt das Standardpräfix. Falls vorhanden, überschreibt dieser Wert
# den "default_type" außer Kraft.
```

[exclude_object]

Ermöglicht das Ausschließen oder Abbrechen einzelner Objekte während des Druckvorgangs.

Weitere Informationen finden Sie in der Anleitung und Befehlsreferenz zum Ausschließen von Objekten. Ein mit Marlin/RepRapFirmware kompatibles M486 G-Code-Makro finden Sie in der Datei sample-macros.cfg.

```
[exclude_object]
```

Resonanzkompensation.

[input_shaper]

Aktiviert die Resonanzkompensation resonance compensation. Siehe auch die Befehlsreferenz command reference..

```
[input_shaper]
#shaper_freq_x: 0
# Eine Frequenz (in Hz) des Eingangs-Shapers für die X-Achse. Dies ist normalerweise eine
# Resonanzfrequenz der X-Achse, die der Input Shaper unterdrücken soll. Für komplexere Shaper, wie 2-
# und 3-Hump EI Eingangs-Shaper, kann dieser Parameter aus verschiedenen Überlegungen eingestellt
# werden. Der Standardwert ist 0, was die Eingangsformung Shaping für die X-Achse.
#shaper_freq_y: 0
# Die Frequenz (in Hz) des Eingangs-Shapers für die Y-Achse. Dies ist normalerweise eine
# Resonanzfrequenz der Y-Achse, die der Eingangs-Shaper unterdrücken soll. Für komplexere Shaper, wie
# 2- und 3-Buckel-EI Eingangs-Shaper, kann dieser Parameter unter verschiedenen Gesichtspunkten
# festgelegt werden. Überlegungen eingestellt werden. Der Standardwert ist 0, was die Eingangsformung
# Shaping für die Y-Achse.
#shaper_type: mzv
# Ein Typ des Input Shapers, der für die X- und Y-Achse verwendet werden soll. Unterstützt
# Shaper sind zv, mzv, zvd, ei, 2hump_ei und 3hump_ei. Die Vorgabe ist mzv input shaper.
```

```
#shaper_type_x:
#shaper_type_y:
# Wenn shaper_type nicht gesetzt ist, können diese beiden Parameter verwendet werden, um
# unterschiedliche Input Shaper für die X- und Y-Achse konfigurieren. Die gleichen Werte werden
# unterstützt wie für den Parameter shaper_type.
#damping_ratio_x: 0.1
#damping_ratio_y: 0.1
# Dämpfungsverhältnisse der Schwingungen der X- und Y-Achse, die von den Eingangs-Shapern verwendet
# werden zur Verbesserung der Vibrationsunterdrückung. Der Standardwert ist 0.1, was ein guter
# Allround-Wert für die meisten Drucker ist. Unter den meisten Umständen erfordert dieser Parameter in
# den meisten Fällen keine Einstellung und sollte nicht geändert werden.
```

[adxl345]

Unterstützung für ADXL345-Beschleunigungssensoren. Diese Unterstützung ermöglicht die Abfrage von Beschleunigungsmesser-Messungen vom Sensor. Dies ermöglicht einen ACCELEROMETER_MEASURE-Befehl (siehe G-Codes für weitere Informationen G-Codes). Der Standard-Chipname ist "default", aber man kann einen expliziten Namen angeben (z.B. [adxl345 my_chip_name]).

```
[adxl345]
cs_pin:
# Der SPI-Enable-Pin für den Sensor. Dieser Parameter muss angegeben werden.
#spi_speed: 5000000
# Die SPI-Geschwindigkeit (in hz), die bei der Kommunikation mit dem Chip verwendet werden soll.
# Der Standardwert ist 5000000.
#spi_bus:
#spi_software_sclk_pin:
#spi_software_mosi_pin:
#spi_software_miso_pin:
# Siehe den Abschnitt "Allgemeine SPI-Einstellungen" für eine Beschreibung der obigen Parameter.
#axes_map: x, y, z
# Die Beschleunigungsmesserachse für jede der X-, Y- und Z-Achsen des Druckers. Dies kann nützlich
# sein, wenn der Beschleunigungssensor in einer Ausrichtung montiert ist, die nicht mit der
# Ausrichtung des Druckers übereinstimmt. Zum Beispiel kann man beispielsweise "y, x, z" einstellen,
# um die X- und Y-Achse zu vertauschen. Es ist auch möglich, eine Achse zu negieren, wenn der
# Beschleunigungssensor Richtung umgekehrt ist (z.B. "x, z, -y"). Die Voreinstellung ist "x, y, z".
#Rate: 3200
# Ausgangsdatenrate für ADXL345. ADXL345 unterstützt die folgenden Datenraten rates: 3200, 1600, 800,
# 400, 200, 100, 50 und 25. Beachten Sie, dass es dass es nicht empfohlen wird, diese Rate von der
# Standardeinstellung 3200 zu ändern, und Raten unter 800 beeinträchtigen die Qualität der
# Resonanzmessungen erheblich.
```

[resonance_tester]

Unterstützung für Resonanztests und die automatische Kalibrierung von Input Shapern. Um die meisten Funktionen dieses Moduls nutzen zu können, müssen zusätzliche Software-Abhängigkeiten installiert werden; weitere Informationen finden Sie unter Resonanzmessung Measuring Resonances und in der Befehlsreferenz command reference. Weitere Informationen zum Parameter max_smoothing und seiner Verwendung finden Sie im Abschnitt Max smoothing Max smoothing des Leitfadens Resonanzen messen.

```
[resonance_tester]
#probe_points:
# Eine Aufzählung von X-, Y- und Z-Koordinaten von Punkten (ein Punkt pro Zeile), die getestet
# werden sollen. Resonanzen zu testen. Mindestens ein Punkt ist erforderlich. Stellen Sie sicher, dass
# alle Punkte mit einer gewissen Sicherheitsmarge in der XY-Ebene (~ einige Zentimeter) für den
# Werkzeugkopf erreichbar sind.
#accel_chip:
# Ein Name des Beschleunigungssensor-Chips, der für die Messungen verwendet werden soll. Wenn adxl345
# Chip ohne expliziten Namen definiert wurde, kann dieser Parameter kann dieser Parameter einfach als
# "accel_chip: adxl345" referenziert werden, andernfalls muss ein expliziter Name angegeben werden,
# z.B. "accel_chip: adxl345 mein_chip_name". Entweder dieser oder die nächsten beiden Parameter müssen
# gesetzt werden.
#accel_chip_x:
#accel_chip_y:
# Namen der Beschleunigungsmesserchips, die für die Messungen für jede der Achsen zu verwenden. Dies
# kann z.B. bei Bettschleuderdruckern nützlich sein, wenn zwei separate Beschleunigungsaufnehmer auf
# dem Bett (für die Y-Achse) und am Werkzeugkopf (für die X-Achse) montiert sind. Diese Parameter
# haben das gleiche Format wie der Parameter 'accel_chip'. Nur 'accel_chip' oder diese beiden
# Parameter müssen angegeben werden.
#max_smoothing:
# Maximale Shaper-Glättung, die für jede Achse während des Shapers zugelassen wird. Auto-Kalibrierung
# (mit dem Befehl 'SHAPER_CALIBRATE'). Standardmäßig wird keine maximale Glättung angegeben. Siehe
# Measuring_Resonances guide für weitere Details zur Verwendung dieser Funktion.
#min_freq: 5
# Minimale Frequenz zum Testen auf Resonanzen. Die Vorgabe ist 5 Hz.
```

```
#max_freq: 133.33
# Maximale Frequenz zum Testen auf Resonanzen. Die Vorgabe ist 133,33 Hz.
#accel_per_hz: 75
# Mit diesem Parameter wird bestimmt, welche Beschleunigung verwendet wird, um eine bestimmte Frequenz
#  zu testen: accel = accel_per_hz * freq. Je höher der Wert, desto höher ist die Energie der
# Schwingungen. Kann auf einen Wert gesetzt werden auf einen niedrigeren Wert als den Standardwert
# gesetzt werden, wenn die Resonanzen auf dem Drucker zu stark werden. Niedrigere Werte machen jedoch
# die Messungen von Hochfrequenzresonanzen ungenauer. Der Standardwert ist 75(mm/sec).
#hz_per_sec: 1
# Bestimmt die Geschwindigkeit des Tests. Beim Testen aller Frequenzen im Bereich [min_freq,
# max_freq], erhöht sich die Frequenz jede Sekunde um hz_per_sec. Kleine Werte machen den Test
# langsam, und die großen Werte Werte verringern die Genauigkeit des Tests. Der Standardwert ist 1.0
# (Hz/sec == sec^-2).
```

Config file helpers

[board_pins]

Board-Pin-Aliase (man kann eine beliebige Anzahl von Abschnitten mit einem "board_pins"-Präfix definieren). Verwenden Sie dies, um Aliase für die Pins auf einem Mikrocontroller zu definieren.

```
[board_pins my_aliases]
mcu: mcu
# Eine durch Komma getrennte Aufzählung von Mikrocontrollern, die die Aliase verwenden können. Aliasen
# verwenden können. Standardmäßig werden die Aliase auf den Hauptcontroller "mcu" angewendet.
aliases:
aliases_<name>:
# Eine durch Komma getrennte Aufzählung von "name=wert" Aliasen, die für den Mikrocontroller zu
# erstellen. Zum Beispiel, "EXP1_1=PE6" würde einen "EXP1_1"-Alias für den "PE6"-Pin erstellen. Wenn
# jedoch "Wert" in "<>" eingeschlossen ist in "<>" eingeschlossen ist, wird "name" als reservierter
# Pin erstellt (z. B., "EXP1_9=<GND>" würde "EXP1_9" reservieren). Beliebige Anzahl von Optionen die
# mit "aliases_" beginnen, können angegeben werden.
```

[include]

Unterstützung von Include-Dateien. Man kann zusätzliche Konfigurationsdateien aus der Hauptkonfigurationsdatei des Druckers einbinden. Es können auch Platzhalter verwendet werden (z. B. "configs/*.cfg").

```
[include my_other_config.cfg]
```

[duplicate_pin_override]

Mit diesem Tool kann ein einzelner Mikrocontroller-Pin ohne normale Fehlerprüfung mehrfach in einer Konfigurationsdatei definiert werden. Dies ist für Diagnose- und Debugging-Zwecke gedacht. Dieser Abschnitt wird nicht benötigt, wenn Klipper die mehrfache Verwendung desselben Pins unterstützt, und die Verwendung dieses Override kann zu verwirrenden und unerwarteten Ergebnissen führen.

```
[duplicate_pin_override]
Pins:
# Eine durch Komma getrennte Aufzählung von Pins, kann in einer Konfigurationsdatei ohne normale
# Fehlerprüfung mehrfach verwendet werden.Dieser Parameter muss bereitgestellt werden.
```

Bed probing hardware

[probe]

Z-Höhen-Sonde. Dieser Abschnitt kann definiert werden, um die Hardware für die Z-Höhen-Sondierung zu aktivieren. Wenn dieser Abschnitt aktiviert ist, werden die erweiterten g-code Befehle g-code commands PROBE und QUERY_PROBE verfügbar. Siehe auch die Anleitung zum Kalibrieren der Sonde probe calibrate guide. Der Probe-Abschnitt erzeugt auch einen virtuellen "probe:z_virtual_endstop"-Pin. Bei kartesischen Druckern, die den Taster anstelle eines z-Endanschlags verwenden, kann der stepper_z endstop_pin auf diesen virtuellen Pin gesetzt werden. Wenn Sie "probe:z_virtual_endstop" verwenden, definieren Sie keinen position_endstop im stepper_z Konfigurationsabschnitt.

```
[probe]
pin:
# Pin für die Sondenerkennung. Wenn sich der Pin auf einem anderen Mikrocontroller
# als die Z-Stepper, dann aktiviert er "multi-mcu homing". Dieser Parameter muss angegeben werden.
#deactivate_on_each_sample: True
# Dies legt fest, ob Klipper den Deaktivierungs-Gcode Deaktivierungscode zwischen den einzelnen
# Probenversuchen ausführen soll, wenn eine Sequenz ausführen soll. Die Voreinstellung ist True.
#x_offset: 0.0
# Der Abstand (in mm) zwischen der Sonde und der Düse entlang der x-Achse. Die Voreinstellung ist 0.
#y_offset: 0.0
# Der Abstand (in mm) zwischen der Sonde und der Düse entlang der y-Achse. Die Vorgabe ist 0.
z_offset:
```

```
# Der Abstand (in mm) zwischen dem Bett und der Düse, wenn die Sonde auslöst. Dieser Parameter muss
# angegeben werden.
#speed: 5.0
# Geschwindigkeit (in mm/s) der Z-Achse beim Antasten. Der Standardwert ist 5mm/s.
#Samples: 1
# Die Anzahl der Abtastungen für jeden Punkt. Die abgetasteten Z-Werte werden gemittelt. Der Standardwert der
Anzahl ist 1.
# sample_retract_dist: 2.0
# Der Abstand (in mm), um den Werkzeugkopf zwischen jeder Probe anzuheben (wenn
# mehr als eine Probe genommen wird). Die Voreinstellung ist 2 mm.
#lift_speed:
# Geschwindigkeit (in mm/s) der Z-Achse beim Anheben des Tastkopfes zwischen
# Proben. Standardmäßig wird derselbe Wert wie für den Parameter 'speed' verwendet.
# Parameter.
#samples_result: Durchschnitt
# Die Berechnungsmethode bei mehr als einer Stichprobe - entweder
# "Median" oder "Durchschnitt". Die Vorgabe ist "average".
#samples_tolerance: 0.100
# Der maximale Z-Abstand (in mm), den eine Probe von anderen Proben abweichen darf. Proben abweichen
#  darf. Wenn diese Toleranz überschritten wird, wird entweder ein Fehler Fehler gemeldet oder der
# Versuch wird neu gestartet (siehe samples_tolerance_retries). Der Standardwert ist 0,100 mm.
#samples_tolerance_retries: 0
# Die Anzahl der Wiederholungen, wenn eine Probe gefunden wird, die die samples_tolerance
# überschreitet. Bei einem erneuten Versuch werden alle aktuellen Proben verworfen und der Versuch
# wird neu gestartet. Wenn ein gültiger Satz von Proben gefunden wird, wird Fehler gemeldet. Der
# Standardwert ist Null, was dazu führt, dass ein Fehler gemeldet wird bei der ersten Probe, die die
# samples_tolerance überschreitet.
#activate_gcode:
# Eine Aufzählung von G-Code-Befehlen, die vor jedem Probeversuch auszuführen sind. Siehe
# docs/Command_Templates.md für das G-Code-Format. Dies kann nützlich sein, wenn die Sonde auf
# irgendeine Weise aktiviert werden muss. Geben Sie keine Befehle, die den Werkzeugkopf bewegen (z.B.
# G1). Der Standard ist, dass bei der Aktivierung keine speziellen G-Code-Befehle ausgeführt werden.
#deactivate_gcode:
# Eine Aufzählung von G-Code-Befehlen, die nach jedem Probeversuch ausgeführt werden sollen, dann
# komplett ist. Siehe docs/Command_Templates.md für das G-Code-Format. Geben sie hier keine Befehle
# aus, die den Werkzeugkopf bewegen. Die Vorgabe ist, dass keine speziellen G-Code-Befehle bei der
# Deaktivierung auszuführen sind.
```

[bltouch]

BLTouch-Taster. Man kann diesen Abschnitt (anstelle eines Tasterabschnitts) definieren, um einen BLTouch-Taster zu aktivieren. Weitere Informationen finden Sie im BL-Touch-Handbuch BL-Touch guide und in der Befehlsreferenz command reference. Ein virtueller "probe:z_virtual_endstop"-Pin wird ebenfalls erstellt (siehe den Abschnitt "probe" für weitere Informationen).

```
[bltouch]
sensor_pin:
# Pin, der mit dem BLTouch-Sensor-Pin verbunden ist. Die meisten BLTouch-Geräte benötigen einen Pullup am
Sensor-Pin (dem Pin-Namen wird ein "^" vorangestellt). Dieser Parameter muss angegeben werden.
control_pin:
# Pin, der mit dem BLTouch-Steuerpin verbunden ist. Dieser Parameter muss vorhanden sein.
#pin_move_time: 0.680
# Die Zeit (in Sekunden), die gewartet wird, bis der BLTouch-Pin nach oben oder unten bewegt. Der
# Standardwert ist 0,680 Sekunden.
#stow_on_each_sample: True
# Legt fest, ob Klipper dem Pin befehlen soll, sich nach oben zu bewegen zwischen den einzelnen
# Probeversuchen bei der Durchführung einer Sequenz. Lies die Anweisungen in docs/BLTouch.md, bevor du
#  diese Einstellung False zu setzen. Die Voreinstellung ist True.
#probe_with_touch_mode: False
# Wenn dies auf True gesetzt ist, ist Klipper mit dem Gerät im
# "touch_mode". Der Standardwert ist False (Sondierung im "pin_down" Modus).
#pin_up_reports_not_triggered: True
# Legt fest, ob der BLTouch die Sonde nach einem erfolgreichen "not Zustand "nicht getriggert" nach
# einem erfolgreichen "pin_up"-Befehl meldet. Dies sollte True für alle echten BLTouch-Geräte sein.
# Lesen Sie die Anweisungen in docs/BLTouch.md, bevor Sie dies auf False setzen. Die Voreinstellung
# ist True.
#pin_up_touch_mode_reports_triggered: True
# Legt fest, ob das BLTouch konsequent einen "getriggerten" Zustand nach den Befehlen "pin_up" gefolgt
# von "touch_mode", einnimmt. Dies sollte True für alle echten BLTouch-Geräte sein. Lesen Sie die
# Anweisungen in docs/BLTouch.md, bevor Sie dies auf False setzen. Die Voreinstellung ist True.
#set_output_mode:
# Fordert einen bestimmten Sensor-Pin-Ausgangsmodus auf dem BLTouch V3.0 (und später). Diese
# Einstellung sollte nicht für andere Fühlertypen verwendet werden. Setzen Sie auf "5V", um einen
# Sensor-Pin-Ausgang von 5 Volt anzufordern (nur verwenden, wenn (nur verwenden, wenn die
```

```
# Steuerplatine den 5V-Modus benötigt und auf ihrer Eingangsleitung 5V tolerant ist). Signalleitung
# toleriert). Setzen Sie auf "OD", um den Sensor-Pin-Ausgang im Open-Drain-Modus zu verwenden.
# Standardmäßig wird kein Ausgangsmodus angefordert.
#x_offset:
#y_offset:
#z_offset:
#speed:
#lift_speed:
#samples:
#sample_retract_dist:
#samples_result:
#samples_tolerance:
#samples_tolerance_retries:
# Informationen zu diesen Parametern finden Sie im Abschnitt "probe".
```

[smart_effector]

Der "Smart Effector" von Duet3d implementiert einen Z-Taster unter Verwendung eines Kraftsensors. Man kann diesen Abschnitt anstelle von [probe] definieren, um die Smart-Effector-spezifischen Funktionen zu aktivieren. Dies ermöglicht auch Laufzeitbefehle runtime commands, um die Parameter des Smart-Effektors zur Laufzeit anzupassen.

```
[smart_effector]
Pin:
# Pin, der mit dem Ausgangspin des Smart-Effektors Z-Probe (Pin 5) verbunden ist. Beachten Sie, dass
# der Pullup-Widerstand auf der Platine im Allgemeinen nicht erforderlich ist. Wenn jedoch der
# Ausgangspin über einen Pullup-Widerstand mit dem Board-Pin verbunden ist, muss dieser Widerstand
# einen hohen Wert haben (z.B. 10K Ohm oder mehr). Einige Boards haben einen Pullup-Widerstand am Z-
# Tastereingang, was wahrscheinlich zu einem immer getriggerten Sondenzustand führt. In diesem Fall
# schließen Sie den Smart-Effektor an einen anderen Pin auf der Platine an. Dieser Parameter ist
# erforderlich.
#control_pin:
# Pin, der mit dem Steuereingangspin des Smart-Effektors (Pin 7) verbunden ist. Falls vorhanden,
# werden Befehle zur Programmierung der Empfindlichkeit des Smart-Effektors verfügbar.
#probe_accel:
# Begrenzt die Beschleunigung der Antastbewegungen (in mm/sec^2), falls gesetzt. Eine plötzliche große
# Beschleunigung zu Beginn der Antastbewegung kann kann zu einer fehlerhaften Auslösung der Sonde
# führen, insbesondere wenn das Hotend schwer ist. Um dies zu verhindern, kann es notwendig sein, die
#  Beschleunigung der Beschleunigung der Antastbewegungen über diesen Parameter zu reduzieren.
#Wiederherstellungszeit: 0.4
# Eine Verzögerung zwischen den Fahrbewegungen und den Antastbewegungen in Sekunden. Eine schnelle
# Verfahrbewegung vor der Sondierung kann zu einer falschen Sondierungsauslösung führen. Dies kann zu
# 'Probe triggered prior to movement' Fehlern führen, wenn keine Verzögerung eingestellt ist. Der Wert
# 0 deaktiviert die Erholungsverzögerung. Der Standardwert ist 0,4.
#x_offset:
#y_offset:
# Sollte nicht eingestellt werden (oder auf 0 gesetzt werden).
z_offset:
# Triggerhöhe der Sonde. Beginnen Sie mit -0.1 (mm), und passen Sie später mit dem Befehl
# "PROBE_CALIBRATE". Dieser Parameter muss angegeben werden.
#Geschwindigkeit:
# Geschwindigkeit (in mm/s) der Z-Achse beim Antasten. Es wird empfohlen, mit einer Antast-
# geschwindigkeit von 20 mm/s zu beginnen und sie nach Bedarf anzupassen, um die Genauigkeit und
# Wiederholbarkeit der Messtasterauslösung zu verbessern.
#Samples:
#sample_retract_dist:
#samples_result:
#samples_tolerance:
#samples_tolerance_retries:
#aktivieren_gcode:
#deaktivieren_gcode:
#deactivate_on_each_sample:
# Weitere Informationen zu den obigen Parametern finden Sie im Abschnitt "Sonde".
```

Zusätzliche Schrittmotoren und Extruder

[stepper_z1]

Multi-Stepper-Achsen. Bei einem kartesischen Drucker kann der Stepper, der eine bestimmte Achse steuert, zusätzliche Konfigurationsblöcke haben, die Stepper definieren, die zusammen mit dem primären Stepper geschaltet werden sollen. Es können beliebig viele Abschnitte mit einem numerischen Suffix beginnend bei 1 definiert werden (z. B. "stepper_z1", "stepper_z2" usw.).

```
[stepper_z1]
#step_pin:
```

```
#dir_pin:
#enable_pin:
#microsteps:
#rotation_distance:
# Siehe den Abschnitt "stepper" für die Definition der oben genannten Parameter.
#endstop_pin:
# Wenn ein endstop_pin für den zusätzlichen Stepper definiert ist, wird der Stepper solange aus, bis
# der Endstopp ausgelöst wird. Andernfalls wird der Stepper so lange aus, bis der Endanschlag des
# primären Steppers für die Achse ausgelöst wird.
```

[extruder1].

In einem Multi-Extruder-Drucker fügen Sie für jeden zusätzlichen Extruder einen weiteren Extruder-Abschnitt hinzu. Die zusätzlichen Extruderabschnitte sollten "extruder1", "extruder2", "extruder3" usw. genannt werden. Im Abschnitt "extruder" finden Sie eine Beschreibung der verfügbaren Parameter. Siehe sample-multi-extruder.cfgfür eine Beispielkonfiguration.

```
[extruder1]
#step_pin:
#dir_pin:
#...
# Siehe den Abschnitt "Extruder" für verfügbare Stepper- und Heizungs# Parameter.
#shared_heater:
# Diese Option ist veraltet und sollte nicht mehr angegeben werden.
```

[dual_carriage]

Unterstützung für kartesische Drucker mit zwei Schlitten auf einer Achse. Der aktive Schlitten wird über den erweiterten g-code Befehl SET_DUAL_CARRIAGE gesetzt. Der Befehl "SET_DUAL_CARRIAGE CARRIAGE=1" aktiviert den in diesem Abschnitt definierten Schlitten (CARRIAGE=0 führt zur Aktivierung des primären Schlittens zurück). Die Unterstützung von zwei Schlitten wird in der Regel mit zusätzlichen Extrudern kombiniert - der Befehl SET_DUAL_CARRIAGE wird oft gleichzeitig mit dem Befehl ACTIVATE_EXTRUDER aufgerufen. Stellen Sie sicher, dass die Schlitten während der Deaktivierung geparkt werden.

Siehe sample-idex.cfg für eine Beispielkonfiguration.

```
[dual_carriage]
Achse:
# Die Achse, auf der sich dieser zusätzliche Wagen befindet (entweder x oder y). Dieser Parameter
# muss angegeben werden.
#step_pin:
#dir_pin:
#enable_pin:
#microsteps:
#rotation_distance:
#endstop_pin:
#position_endstop:
#position_min:
#Position_max:
# Siehe den Abschnitt "stepper" für die Definition der obigen Parameter.
```

[extruder_stepper]

Unterstützung für zusätzliche Stepper, die mit der Bewegung eines Extruders synchronisiert sind (man kann eine beliebige Anzahl von Abschnitten mit einem "extruder_stepper"-Präfix definieren).

Siehe die command reference für weitere Informationen.

```
[extruder_stepper my_extra_stepper]
extruder:
# Der Extruder, mit dem dieser Stepper synchronisiert ist. Wenn dieser Wert auf einen Leerstring
# gesetzt wird, wird der Stepper nicht mit einem Extruder synchronisiert. Dieser Parameter muss
# angegeben werden.
#step_pin:
#dir_pin:
#enable_pin:
#microsteps:
#rotation_distance:
# Siehe den Abschnitt "stepper" für die Definition der obigen Parameter.
```

[manual_stepper]

Manuelle Stepper (man kann eine beliebige Anzahl von Abschnitten mit einem "manual_stepper"-Präfix definieren). Dies sind Stepper, die mit dem g-code Befehl MANUAL_STEPPER gesteuert werden. Zum Beispiel: "MANUAL_STEPPER STEPPER=mein_stepper MOVE=10 SPEED=5". Eine Beschreibung des Befehls MANUAL_STEPPER finden Sie in der Datei G-Codes. Die Stepper sind nicht mit der normalen Druckerkinematik verbunden.

```
[manual_stepper my_stepper]
#step_pin:
#dir_pin:
#enable_pin:
#microsteps:
#rotation_distance:
# Siehe Abschnitt "Stepper" für eine Beschreibung dieser Parameter.
#velocity:
# Legt die Standard-Geschwindigkeit (in mm/s) für den Stepper fest. Dieser Wert wird verwendet, wenn
# ein MANUAL_STEPPER-Befehl keinen SPEED-Parameter angibt. Parameter angibt. Der Standardwert ist 5mm/s.
#accel:
# Legt die Standardbeschleunigung (in mm/s^2) für den Stepper fest. Eine Beschleunigung von Null führt
# zu keiner Beschleunigung. Dieser Wert wird verwendet, wenn ein MANUAL_STEPPER-Befehl keinen ACCEL
# Parameter angibt. Der Standardwert ist Null.
#endstop_pin:
# Endstoppschalter-Erkennungspin. Wenn angegeben, können "Referenzfahrten" durchgeführt werden, indem man einen
STOP_ON_ENDSTOP-Parameter zu MANUAL_STEPPER Bewegungsbefehlen hinzufügt.
```

Benutzerdefinierte Heizungen und Sensoren

[verify_heater]

Überprüfung von Heizungen und Temperatursensoren. Die Überprüfung von Heizungen wird automatisch für jede Heizung aktiviert, die auf dem Drucker konfiguriert ist. Verwenden Sie die Abschnitte verify_heater, um die Standardeinstellungen zu ändern.

```
[verify_heater heater_config_name]
#max_error: 120
# Der maximale "kumulative Temperaturfehler", bevor ein Fehler ausgelöst wird. Kleinere Werte führen
# zu einer strengeren Prüfung und größere Werte erlauben mehr Zeit, bevor ein Fehler gemeldet wird.
# Konkret wird die Temperatur einmal pro Sekunde überprüft, und wenn sie Wenn sie nahe an der
# Zieltemperatur liegt, wird ein interner "Fehlerzähler" zurückgesetzt. ein interner "Fehlerzähler"
#  zurückgesetzt; andernfalls, wenn die Temperatur unter dem Zielbereich liegt, wird der Zähler um den
#  Betrag erhöht, um den diedie gemeldete Temperatur von  diesem Bereich abweicht. Sollte der Zähler
# Zähler diesen "max_error" überschreiten, wird ein Fehler ausgelöst. Der Standardwert ist 120.
#check_gain_time:
# Dies steuert die Überprüfung der Heizung während des ersten Aufheizens. Kleinere Werte führen zu einer
strengeren Prüfung und größere Werte erlauben eine mehr Zeit, bevor ein Fehler gemeldet wird. Konkret bedeutet
dies, dass während während des ersten Aufheizens, solange die Temperatur der Heizung
# (angegeben in Sekunden) ansteigt, wird der interne "Fehlerzähler" zurückgesetzt. Der Standardwert ist 20
Sekunden für Extruder und 60 Sekunden für heater_bed.
#Hysterese: 5
# Die maximale Temperaturdifferenz (in Celsius) zu einer Zieltemperatur. Die maximale
# Temperaturdifferenz (in Celsius) zu einer Zieltemperatur, die als im Bereich des Ziels betrachtet
# wird. Diese steuert die max_error range Prüfung. Es ist selten, dass dieser Wert angepasst wird.
# Wert anzupassen. Der Standardwert ist 5.
#heating_gain: 2
# Die Mindesttemperatur (in Celsius), um die die Heizung während der check_gain_time-Prüfung ansteigen
# muss, während der check_gain_time-Prüfung. Es ist selten, dass dieser Wert angepasst wird.
# Der Standardwert ist 2.
```

[homing_heaters]

Werkzeug zum Deaktivieren von Heizungen beim Referenzieren oder Antasten einer Achse.

```
[homing_heaters]
#steppers:
# Eine durch Kommata getrennte Aufzählung von Steppern, die bewirken sollen, dass die Heizungen
# deaktiviert werden sollen. Standardmäßig werden die Heizungen bei jeder Referenzfahrt/Tastfahrt-
# Bewegung deaktiviert. Typisches Beispiel: stepper_z
#Heizungen:
# Eine durch Kommata getrennte Aufzählung von Heizungen, die während des Referenzfahrt/Tastfahrt-
# Bewegung deaktiviert wird. Standardmäßig werden alle Heizelemente deaktiviert.
# Typisches Beispiel: extruder, heater_bed
```

[thermistor]

Benutzerdefinierte Thermistoren (man kann eine beliebige Anzahl von Abschnitten mit einem "Thermistor"-Präfix definieren). Ein benutzerdefinierter Thermistor kann im sensor_type Feld eines heater config Abschnitts verwendet werden. (Wenn man zum Beispiel einen Abschnitt "[thermistor my_thermistor]" definiert, kann man "sensor_type : my_thermistor" verwenden, wenn man ein Heizgerät definiert). Stellen Sie sicher, dass Sie den Abschnitt thermistor in der Konfigurationsdatei vor seiner ersten Verwendung in einem Heizungsabschnitt platzieren.

```
[thermistor my_thermistor]
```

```
#temperature1:
#resistance1:
#temperature2:
#resistance2:
#temperature3:
#resistance3:
# Drei Widerstandsmessungen (in Ohm) bei den angegebenen Temperaturen (in Celsius). Die drei Messungen werden
verwendet, um die Steinhart-Hart-Koeffizienten für den Thermistor zu berechnen. Diese Parameter müssen
angegeben werden, wenn Steinhart-Hart zur Definition des Thermistor verwendet werden soll.
#beta:
# Alternativ kann man auch Temperatur1, Widerstand1 und Beta um die Thermistor-Parameter zu
# definieren. Dieser Parameter muss angegeben werden, wenn "beta" zur Definition des Thermistors
# verwendet wird.
```

[adc_temperature]

Benutzerdefinierte ADC-Temperatursensoren (man kann eine beliebige Anzahl von Abschnitten mit einem "adc_temperature"-Präfix definieren). Dies ermöglicht die Definition eines benutzerdefinierten Temperatursensors, der eine Spannung an einem Analog-Digital-Wandler (ADC)-Pin misst und eine lineare Interpolation zwischen einer Reihe von konfigurierten Temperatur-/Spannungsmessungen (oder Temperatur-/Widerstandsmessungen) zur Bestimmung der Temperatur verwendet. Der resultierende Sensor kann als sensor_type in einem Heizungsabschnitt verwendet werden. (Wenn man zum Beispiel einen "[adc_temperature my_sensor]"-Abschnitt definiert, kann man bei der Definition einer Heizung einen "sensor_type : my_sensor" verwenden). Stellen Sie sicher, dass Sie den Sensorabschnitt in der Konfigurationsdatei vor seiner ersten Verwendung in einem Heizungsabschnitt platzieren.

```
[adc_temperature my_sensor]
#temperature1:
#voltage1:
#temperature2:
#voltage2:
#...
# Ein Satz von Temperaturen (in Celsius) und Spannungen (in Volt) zur Verwendung als Referenz bei der
# Umwandlung einer Temperatur. Ein Heizabschnitt, der diesen Sensor verwendet, kann auch die Parameter
# adc_voltage und voltage_offset Parameter angeben, um die ADC-Spannung zu definieren (siehe Abschnitt
# "Allgemeine Temperatur Verstärker" für Details). Mindestens zwei Messungen müssen angegeben werden.
#temperature1:
#resistance1:
#temperature2:
#resistance2:
#...
# Alternativ kann man auch einen Satz von Temperaturen (in Celsius) und Widerstand (in Ohm) angeben,
# die als Referenz bei der Konvertierung einer Temperatur zu verwenden. Ein Heizabschnitt, der diesen
# Sensor verwendet, kann auch einen Pullup_resistor-Parameter angeben (siehe Abschnitt "Extruder" für
# Details). Es müssen mindestens zwei Messwerte angegeben werden.
```

[heater_generic

Generische Heizelemente (man kann eine beliebige Anzahl von Abschnitten mit dem Präfix "heater_generic" definieren). Diese Heizelemente verhalten sich ähnlich wie Standard-Heizelemente (Extruder, Heizbetten). Verwenden Sie den Befehl SET_HEATER_TEMPERATURE (siehe G-Codes für Details), um die Zieltemperatur einzustellen.

```
[heater_generic my_generic_heater]
#gcode_id:
# Die ID, die bei der Meldung der Temperatur im Befehl M105 zu verwenden ist.
# Dieser Parameter muss angegeben werden.
#heater_pin:
#max_power:
#sensor_type:
#sensor_pin:
#smooth_time:
#control:
#pid_Kp:
#pid_Ki:
#pid_Kd:
#pwm_cycle_time:
#min_temp:
#max_temp:
# Siehe den Abschnitt "Extruder" für die Definition der oben genannten Parameter.
```

[temperature_sensor]

Allgemeine Temperatursensoren. Es können beliebig viele zusätzliche Temperatursensoren definiert werden, die über den Befehl M105 gemeldet werden.

```
[temperature_sensor my_sensor]
#sensor_type:
#sensor_pin:
#min_temp:
#max_temp:
# Siehe den Abschnitt "Extruder" für die Definition der obigen Parameter.
#gcode_id:
# Siehe den Abschnitt "heater_generic" für die Definition dieses Parameters.
```

Temperatursensoren.

Klipper enthält Definitionen für viele Arten von Temperatursensoren. Diese Sensoren können in jedem Konfigurationsabschnitt verwendet werden, der einen Temperatursensor benötigt (wie z.B. ein [extruder] oder [heated_bed] Abschnitt).

Allgemeine Thermistoren

Übliche Thermistoren. Die folgenden Parameter sind in Heizungsabschnitten verfügbar, die einen dieser Sensoren verwenden.

```
sensor_type:
# Einer von "EPCOS 100K B57560G104F", "ATC Semitec 104GT-2", ATC Semitec 104NT-4-R025H42G", "Generic
# 3950", Honeywell 100K 135-104LAG-J01", "NTC 100K MGB18-104F39050L32", "SliceEngineering 450" oder
# "TDK NTCG104LH104JT1".
sensor_pin:
# Analogeingangspin, der mit dem Thermistor verbunden ist. Dieser Parameter muss bereitgestellt
# werden.
#pullup_resistor: 4700
# Der Widerstand (in Ohm) des Pullups, der an den Thermistor angeschlossen ist. Die Vorgabe ist 4700
# Ohm.
#inline_resistor: 0
# Der Widerstand (in Ohm) eines zusätzlichen (nicht wärmeveränderlichen) Widerstands der inline mit
# dem Thermistor platziert wird. Es ist selten, dies einzustellen. Die Voreinstellung ist 0 Ohm.
```

Gemeinsame Temperaturverstärker

Die folgenden Parameter sind in Heizungsabschnitten verfügbar, die einen dieser Sensoren verwenden.

```
sensor_type:
# Einer von "PT100 INA826", "AD595", "AD597", "AD8494", "AD8495", "AD8496" oder "AD8497".
sensor_pin:
# Analogeingangspin, der mit dem Sensor verbunden ist. Dieser Parameter muss bereitgestellt werden.
#adc_voltage: 5.0
# Die ADC-Vergleichsspannung (in Volt). Der Standardwert ist 5 Volt.
#voltage_offset: 0
# Der ADC-Spannungsoffset (in Volt). Die Voreinstellung ist 0.
```

Direkt angeschlossener PT1000-Sensor

Die folgenden Parameter sind in Heizungsabschnitten verfügbar, die einen dieser Sensoren verwenden.

```
sensor_type: PT1000
sensor_pin:
# Analogeingangspin, der mit dem Sensor verbunden ist. Dieser Parameter muss bereitgestellt werden.
#pullup_resistor: 4700
# Der Widerstand (in Ohm) des an den Sensor angeschlossenen Pullups. Standardwert ist 4700 Ohm.
```

MAXxxxxx-Temperatursensoren

MAXxxxxx Temperatursensoren mit serieller Peripherieschnittstelle (SPI). Die folgenden Parameter sind in Heizungsabschnitten verfügbar, die einen dieser Sensortypen verwenden.

```
sensor_type:
# Einer von "MAX6675", "MAX31855", "MAX31856" oder "MAX31865".
sensor_pin:
# Die Chip-Select-Leitung für den Sensorchip. Dieser Parameter muss bereitgestellt werden.
#spi_speed: 4000000
# Die SPI-Geschwindigkeit (in hz), die bei der Kommunikation mit dem Chip verwendet werden soll.
# Der Standardwert ist 4000000.
#spi_bus:
#spi_software_sclk_pin:
#spi_software_mosi_pin:
#spi_software_miso_pin:
# Siehe den Abschnitt "Allgemeine SPI-Einstellungen" für eine Beschreibung der obigen Parameter.
#tc_type: K
#tc_use_50Hz_filter: False
```

```
#tc_averaging_count: 1
# Die obigen Parameter steuern die Sensorparameter des MAX31856 Chips. Die Standardwerte für jeden
# Parameter stehen neben dem Parameternamen Namen in der obigen Aufzählung.
#rtd_nominal_r: 100
#rtd_reference_r: 430
#rtd_num_of_wires: 2
#rtd_use_50Hz_filter: False
# Die obigen Parameter steuern die Sensorparameter des MAX31865 Chips. Die Standardwerte für jeden
# Parameter stehen neben dem Parameternamen Namen in der obigen Aufzählung.
```

BMP280/BME280/BME680 Temperatursensor

BMP280/BME280/BME680 Umweltsensoren mit Zweidrahtschnittstelle (I2C). Beachten Sie, dass diese Sensoren nicht für die Verwendung mit Extrudern und Heizbetten vorgesehen sind, sondern für die Überwachung der Umgebungstemperatur (C), des Drucks (hPa), der relativen Luftfeuchtigkeit und im Falle des BME680 des Gasstands. Siehe sample-macros.cfg für ein gcode_macro, das zusätzlich zur Temperatur auch Druck und Luftfeuchtigkeit melden kann.

```
sensor_type: BME280
#i2c_address:
# Standard ist 118 (0x76). Einige BME280-Sensoren haben eine Adresse von 119 (0x77).
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
```

HTU21D-Sensor

Zweidrahtschnittstelle (I2C) des Umweltsensors der HTU21D-Familie. Beachten Sie, dass dieser Sensor nicht für die Verwendung mit Extrudern und Heizbetten gedacht ist, sondern für die Überwachung der Umgebungstemperatur (C) und der relativen Luftfeuchtigkeit. In sample-macros.cfg finden Sie ein gcode_macro, das zusätzlich zur Temperatur auch die Luftfeuchtigkeit melden kann.

```
sensor_type:
# Muss "HTU21D", "SI7013", "SI7020", "SI7021" oder "SHT21" sein.
#i2c_address:
# Standard ist 64 (0x40).
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
#htu21d_hold_master:
# Wenn der Sensor den I2C-Buf während des Lesens halten kann. Wenn True, kann keine andere
# Buskommunikation durchgeführt werden, während das Lesen läuft. Standard ist False.
#htu21d_resolution:
# Die Auflösung der Temperatur- und Luftfeuchtigkeitsmessung.
# Gültige Werte sind:
# 'TEMP14_HUM12' -> 14bit für Temp und 12bit für Feuchte
# 'TEMP13_HUM10' -> 13bit für Temp und 10bit für Feuchte
# 'TEMP12_HUM08' -> 12bit für Temp und 08bit für Feuchte
# 'TEMP11_HUM11' -> 11bit für Temp und 11bit für Feuchte
# Standard ist: "TEMP11_HUM11"
#htu21d_report_time:
# Intervall in Sekunden zwischen den Messwerten. Voreinstellung ist 30
```

LM75-Temperatursensor

LM75/LM75A Temperatursensoren, die über zwei Leitungen (I2C) angeschlossen sind. Diese Sensoren haben einen Bereich von -55~125 C, so dass sie z.B. für die Überwachung der Kammertemperatur geeignet sind. Sie können auch als einfache Lüfter-/Heizungssteuerungen fungieren.

```
sensor_type: LM75
#i2c_address:
# Voreinstellung ist 72 (0x48). Der normale Bereich ist 72-79 (0x48-0x4F) und die 3 Low-Bits der
# Adresse werden über Pins auf dem Chip konfiguriert (normalerweise mit Jumpern oder fest verdrahtet).
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
#lm75_report_time:
# Intervall in Sekunden zwischen den Messwerten. Standard ist 0.8, mit Minimum 0.5.
```

Eingebauter Mikrocontroller-Temperatursensor

Die Mikrocontroller atsam, atsamd und stm32 enthalten einen internen Temperatursensor. Man kann den Sensor "temperature_mcu" verwenden, um diese Temperaturen zu überwachen.

```
sensor_type: temperatur_mcu
#sensor_mcu: mcu
# Der Mikrocontroller, von dem gelesen werden soll. Die Vorgabe ist "mcu".
#sensor_temperature1:
#sensor_adc1:
# Geben Sie die beiden obigen Parameter an (eine Temperatur in Celsius und einen ADC-Wert als
# Fließkommazahl zwischen 0,0 und 1,0) zur Kalibrierung der Mikrocontroller-Temperatur zu kalibrieren.
# Dies kann die gemeldete Temperaturgenauigkeit auf einigen Chips verbessern. Ein typischer Weg, um
# diese Kalibrierungsinformationen zu erhalten, ist es, den Drucker  tromversorgung des Druckers für
# einige Stunden vollständig zu unterbrechen (um sicherzustellen, dass er die Umgebungstemperatur
# hat).(um  sicherzustellen, dass er die Umgebungstemperatur hat), ihn dann einzuschalten und den
# Befehl QUERY_ADC zu verwenden eine ADC-Messung zu erhalten. Verwenden Sie einen anderen
# Temperatursensor am des Druckers, um die entsprechende Umgebungstemperatur zu ermitteln. Die
# Standardeinstellung ist die Verwendung der Werkskalibrierungsdaten auf dem Mikrocontroller (falls
# zutreffend) oder die Nennwerte aus der Mikrocontroller-Spezifikation.
#sensor_temperature2:
#sensor_adc2:
# Wenn sensor_temperature1/sensor_adc1 angegeben ist, kann man auch Kalibrierungsdaten für
# sensor_temperature2/sensor_adc2 angegeben werden. Auf diese Weise kann man kalibrierte Informationen
# über die "Temperatursteigung" erhalten. Die Standard ist die Verwendung der Werkskalibrierungsdaten
# auf dem Mikrocontroller (falls zutreffend) oder die Nennwerte aus der Mikrocontroller-Spezifikation.
```

Host-Temperatursensor

Temperatur vom Gerät (z.B. Raspberry Pi), auf dem die Host-Software läuft.

```
sensor_type: temperature_host
#sensor_path:
# Der Pfad zur Temperatursystemdatei. Der Standard ist "/sys/class/thermal/thermal_zone0/temp", das
# ist die Temperatur Systemdatei auf einem Raspberry Pi Computer ist.
```

DS18B20 Temperatursensor

Der DS18B20 ist ein digitaler 1-Draht (w1) Temperatursensor. Beachten Sie, dass dieser Sensor nicht für die Verwendung mit Extrudern und Heizbetten gedacht ist, sondern eher für die Überwachung der Umgebungstemperatur (C). Diese Sensoren haben einen Messbereich von bis zu 125 C, so dass sie z.B. für die Überwachung der Kammertemperatur geeignet sind. Sie können auch als einfache Lüfter-/Heizungsregler eingesetzt werden. DS18B20-Sensoren werden nur von der "Host-MCU", z. B. dem Raspberry Pi, unterstützt. Das w1-gpio Linux Kernel Modul muss installiert sein.

```
sensor_type: DS18B20
serial_no:
# Jedes 1-Wire-Gerät hat eine eindeutige Seriennummer, die zur Identifizierung des Geräts dient,
# normalerweise im Format 28-031674b175ff. Dieser Parameter muss angegeben werden. Angeschlossene 1-
# Wire-Geräte können mit dem folgenden Linux-Befehl aufgelistet werden: ls /sys/bus/w1/devices/
#ds18_report_time:
# Intervall in Sekunden zwischen den Messwerten. Standard ist 3.0, mit einem Minimum von 1.0
#sensor_mcu:
# Der Mikrocontroller, von dem gelesen werden soll. Muss der host_mcu sein.
```

Fans

Drucker Lüfter

```
[fan]
Pin:
# Ausgangspin zur Steuerung des Lüfters. Dieser Parameter muss angegeben werden.
#max_power: 1.0
# Die maximale Leistung (ausgedrückt als Wert von 0,0 bis 1,0), auf die der Pin eingestellt werden
# kann. Der Wert 1.0 erlaubt es, den Pin für längere Zeit vollständig längere Zeit zu aktivieren,
# während ein Wert von 0,5 es erlaubt, den Pin nicht mehr als die Hälfte der Zeit zu aktivieren.
# Diese Einstellung kann verwendet werden, um die Gesamtausgangsleistung (über längere
# Zeiträume) für den Lüfter zu begrenzen. Wenn dieser Wert kleiner als 1.0 ist, wird die angeforderte
# Lüftergeschwindigkeit zwischen Null und max_power skaliert, (wenn z.B. max_power
# 0,9 ist und eine Lüftergeschwindigkeit von 80% angefordert wird, wird die Lüfter Leistung auf 72%
# gesetzt). Der Standardwert ist 1.0.
#shutdown_speed: 0
# Die gewünschte Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), wenn
# die Mikrocontroller-Software in einen Fehlerzustand gerät. Die Vorgabe ist 0.
#cycle_time: 0.010
# Die Zeitspanne (in Sekunden) für jeden PWM-Stromzyklus für den Lüfter. Es wird empfohlen, dass
# dieser Wert 10 Millisekunden oder mehr beträgt, wenn Software-basierte PWM verwendet wird. Der
# Standardwert ist 0,010 Sekunden.
#hardware_pwm: False
```

```
# Aktivieren Sie dies, um Hardware-PWM anstelle von Software-PWM zu verwenden.
# Die meisten Lüfter arbeiten nicht gut mit Hardware-PWM, daher ist es nicht empfehlenswert
# zu aktivieren, es sei denn, es gibt eine elektrische Anforderung, bei sehr hohen Geschwindigkeiten
# zu schalten.
# Bei Verwendung von Hardware-PWM ist die tatsächliche Zykluszeit durch die Implementierung
# eingeschränkt und kann erheblich deutlich von der angeforderten cycle_time abweichen. Die Vorgabe
# ist False.
#kick_start_time: 0.100
# Zeit (in Sekunden), um den Lüfter bei voller Geschwindigkeit zu starten, wenn er entweder zum ersten # Mal
aktiviert wird oder um mehr als 50% zu erhöhen (hilft, den Lüfter zum Drehen zu
# bringen). Der Standardwert ist 0,100 Sekunden.
#off_below: 0.0
# Die minimale Eingangsgeschwindigkeit, die den Lüfter antreibt (ausgedrückt als ein
# Wert von 0.0 bis 1.0). Wenn eine niedrigere Geschwindigkeit als off_below verwendet wird,
# wird der Lüfter stattdessen ausgeschaltet. Diese Einstellung kann verwendet werden, um ein
# Stillstand des Lüfters zu verhindern und um sicherzustellen, dass dieser effektiv arbeiten kann. Die #
Voreinstellung ist 0.0.
# Diese Einstellung sollte jedes Mal neu kalibriert werden, wenn max_power angepasst wird.
# Um diese Einstellung zu kalibrieren, beginnen Sie mit off_below auf 0,0 und dem
# Lüfter läuft. Verringern Sie schrittweise die Lüfterdrehzahl, um die niedrigste Drehzahl zu
# ermitteln, die den Lüfter zuverlässig antreibt, ohne ihn zu blockieren. Stellen Sie
# off_below auf das diesem Wert entsprechende Tastverhältnis (z. B.12% -> 0,12) oder etwas höher.
#tachometer_pin:
# Tachometer-Eingangspin zur Überwachung der Lüfterdrehzahl. Ein Pullup ist im Allgemeinen
# erforderlich. Dieser Parameter ist optional.
#tachometer_ppr: 2
# Wenn tachometer_pin angegeben ist, ist dies die Anzahl der Pulse pro
# Umdrehung des Tachometersignals. Bei einem BLDC-Lüfter ist dies
# normalerweise die Hälfte der Anzahl der Pole. Die Voreinstellung ist 2.
#tachometer_poll_interval: 0.0015
# Wenn tachometer_pin angegeben ist, ist dies die Abfrageperiode des
# Tachometer-Pin, in Sekunden. Der Standardwert ist 0,0015, was schnell genug ist
# schnell genug für Lüfter unter 10000 RPM bei 2 PPR. Dieser Wert muss kleiner sein als
# 30/(tachometer_ppr*rpm), mit einem gewissen Spielraum, wobei rpm die
# Höchstgeschwindigkeit (in RPM) des Gebläses ist.
```

[heater_fan]

Heizungslüfter (es können beliebig viele Abschnitte mit dem Präfix "heater_fan" definiert werden). Ein "heater_fan" ist ein Lüfter, der immer dann aktiviert wird, wenn die zugehörige Heizung aktiv ist. Standardmäßig hat ein "heater_fan" eine shutdown_speed, die gleich der max_power ist.

```
[heater_fan my_nozzle_fan]
#pin:
#max_power:
#shutdown_speed:
#cycle_time:
#hardware_pwm:
#kick_start_time:
#off_below:
#tachometer_pin:
#tachometer_ppr:
#tachometer_poll_interval:
# Siehe Abschnitt "Lüfter" für eine Beschreibung der oben genannten Parameter.
#heater: extruder
# Name des Konfigurationsabschnitts, der das Heizgerät definiert, dem dieser Lüfter
# verbunden ist. Wenn hier eine kommagetrennte Aufzählung von Heizungsnamen angegeben wird, dann wird der
Lüfter aktiviert, wenn einer der angegebenen Heizungen aktiviert sind. Die Vorgabe ist "Extruder".
#heater_temp: 50.0
# Eine Temperatur (in Celsius), unter die das Heizgerät fallen muss, bevor der Lüfter deaktiviert
# wird. Die Vorgabe ist 50 Celsius.
#fan_speed: 1.0
# Die Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), auf die der Lüfter
# eingestellt wird, wenn die zugehörige Heizung aktiviert ist. Die Vorgabe
# ist 1.0
```

[controller_fan]

Reglerlüfter (es können beliebig viele Abschnitte mit dem Präfix "controller_fan" definiert werden). Ein "controller fan" ist ein Lüfter, der immer dann aktiviert wird, wenn die zugehörige Heizung oder der zugehörige Stepper-Treiber aktiv ist. Der Lüfter wird gestoppt, wenn ein idle_timeout erreicht wird, um sicherzustellen, dass nach der Deaktivierung einer überwachten Komponente keine Überhitzung auftritt.

```
[controller_fan my_controller_fan]
```

```
#pin:
#max_power:
#shutdown_speed:
#cycle_time:
#hardware_pwm:
#kick_start_time:
#off_below:
#tachometer_pin:
#tachometer_ppr:
#tachometer_poll_interval:
# Siehe Abschnitt "Lüfter" für eine Beschreibung der oben genannten Parameter.
#fan_speed: 1.0
# Die Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), auf die der Lüfter
# auf die der Lüfter eingestellt wird, wenn eine Heizung oder ein Stepper-Treiber aktiv ist.
# Der Standardwert ist 1.0
#idle_timeout:
# Die Zeitspanne (in Sekunden), nachdem ein Steppertreiber oder eine Heizung
# aktiv war und der Lüfter weiterlaufen soll. Die Vorgabe ist 30 Sekunden.
#idle_speed:
# Die Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), auf die der Lüfter eingestellt
# wird, wenn eine Heizung oder ein Steppertreiber aktiv war und bevor der idle_timeout erreicht wird. # Die
Vorgabe ist fan_speed.
#heater:
#stepper:
# Name des Konfigurationsabschnitts, definiert den Heater/Stepper,  mit dem dieser Lüfter
# bezeichnet sind. Wenn diese durch Komma getrennte Aufzählung von Heizungs-Steppernamen
# angegeben wird, dann wird der Lüfter aktiviert, wenn einer der angegebenen
# Heizungen/Steuerungen aktiviert sind. Die Standard-Heizung ist "extruder", der Standard-Stepper ist # "all of
them".
```

[temperature_fan]

Temperaturgesteuerte Kühllüfter (man kann eine beliebige Anzahl von Abschnitten mit dem Präfix "temperature_fan" definieren). Ein "Temperaturlüfter" ist ein Lüfter, der immer dann aktiviert wird, wenn sein zugehöriger Sensor über einer bestimmten Temperatur liegt. Standardmäßig hat ein temperature_fan eine shutdown_speed, die gleich der max_power ist.

Siehe die command reference für weitere Informationen.

```
[temperature_fan my_temp_fan]
#pin:
#max_power:
#shutdown_speed:
#cycle_time:
#hardware_pwm:
#kick_start_time:
#off_below:
#tachometer_pin:
#tachometer_ppr:
#tachometer_poll_interval:
# Siehe Abschnitt "Lüfter" für eine Beschreibung der oben genannten Parameter.
#sensor_type:
#sensor_pin:
#Steuerung:
#max_delta:
#min_temp:
#max_temp:
# Siehe Abschnitt "Extruder" für eine Beschreibung der oben genannten Parameter.
#pid_Kp:
#pid_Ki:
#pid_Kd:
# Die Proportional- (pid_Kp), Integral- (pid_Ki) und Ableitungswerte (pid_Kd) für das PID-Regelsystem.
# Klipper bewertet die PID-Einstellungen mit der folgenden allgemeinen Formel:
# fan_pwm = max_power - (Kp*e + Ki*integral(e) - Kd*derivative(e)) / 255
# wobei "e" "Zieltemperatur - gemessene Temperatur" ist und "fan_pwm" die angeforderte Lüfterrate ist,
# wobei 0,0 für "voll aus" und 1,0 voll ein ist. Die Parameter pid_Kp, pid_Ki und pid_Kd müssen
# bereitgestellt werden, wenn der PID-Regelalgorithmus aktiviert ist.
#pid_deriv_time: 2.0
# Ein Zeitwert (in Sekunden), über den die Temperaturmessungen geglättet werden, wenn der PID-
# Regelalgorithmus verwendet wird. Dies kann die Auswirkung Auswirkungen des Messrauschens. Die
# Vorgabe ist 2 Sekunden.
#target_temp: 40.0
# Eine Temperatur (in Celsius), die die Zieltemperatur sein wird. Die Vorgabe ist 40 Grad.
#max_speed: 1.0
```

```
# Die Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), auf die der Lüfterausgedrückt als
# Wert von 0,0 bis 1,0), auf die der Lüfter eingestellt # wird, wenn die Sensortemperatur den
# eingestellten Wert überschreitet. Der Standardwert ist 1.0.
#min_speed: 0.3
# Die minimale Lüftergeschwindigkeit (ausgedrückt als Wert von 0,0 bis 1,0), auf die
# der Lüfter bei PID-Temperaturlüftern eingestellt wird. Der Standardwert ist 0.3.
#gcode_id:
# Wenn gesetzt, wird die Temperatur in M105-Abfragen unter Verwendung der
# gegebenen id. Die Standardeinstellung ist, dass die Temperatur nicht über M105 gemeldet wird.
```

[fan_generic]

Manuell gesteuerter Lüfter (es können beliebig viele Abschnitte mit einem "fan_generic"-Präfix definiert werden). Die Drehzahl eines manuell gesteuerten Lüfters wird mit dem gcode-Befehl SET_FAN_SPEED eingestellt.

```
[fan_generic extruder_partfan]
#pin:
#max_power:
#shutdown_speed:
#cycle_time:
#hardware_pwm:
#kick_start_time:
#off_below:
#tachometer_pin:
#tachometer_ppr:
#tachometer_poll_interval:
# Eine Beschreibung der oben genannten Parameter finden Sie im Abschnitt "Lüfter".
```

LEDs.

[led]

Unterstützung für LEDs (und LED-Streifen), die über PWM-Pins des Mikrocontrollers gesteuert werden (man kann eine beliebige Anzahl von Abschnitten mit dem Präfix "led" definieren). Siehe die command reference für weitere Informationen.

```
[led my_led]
#red_pin:
#green_pin:
#blau_pin:
#white_pin:
# Der Pin, der die angegebene LED-Farbe steuert. Mindestens einer der oben genannten
# Parameter muss angegeben werden.
#cycle_time: 0.010
# Die Zeitspanne (in Sekunden) pro PWM-Zyklus. Es wird empfohlen 10 Millisekunden oder mehr zu sein,
# wenn softwarebasierte PWM verwendet wird. Der Standardwert ist 0,010 Sekunden.
#hardware_pwm: False
# Aktivieren Sie dies, um Hardware-PWM anstelle von Software-PWM zu verwenden. Wenn Hardware-PWM wird
# die tatsächliche Zykluszeit durch die Implementierung begrenzt und kann sich erheblich von der
# angeforderten cycle_time. Die Voreinstellung ist False.
#initial_RED: 0.0
#initial_GREEN: 0.0
#initial_BLUE: 0.0
#initial_WHITE: 0.0
# Legt die anfängliche LED-Farbe fest. Jeder Wert sollte zwischen 0.0 und 1.0. Der Standardwert für
# jede Farbe ist 0.
```

[neopixel]

Neopixel (aka WS2812) LED Unterstützung (man kann eine beliebige Anzahl von Abschnitten mit einem "neopixel" Präfix definieren). Siehe die command reference für weitere Informationen.

Beachten Sie, dass die linux mcu -Implementierung derzeit keine direkt angeschlossenen Neopixel unterstützt. Das aktuelle Design, das die Linux-Kernel-Schnittstelle verwendet, erlaubt dieses Szenario nicht, da die Kernel-GPIO-Schnittstelle nicht schnell genug ist, um die erforderlichen Pulsraten zu liefern.

```
[neopixel my_neopixel]
pin:
# Der Pin, der mit dem Neopixel verbunden ist. Dieser Parameter muss angegeben werden.
#chain_count:
# Die Anzahl der Neopixel-Chips, die mit dem angegebenen Pin verbunden sind. Der Standardwert ist 1
# (was bedeutet, dass nur ein einziger Neopixel mit dem Pin verbunden ist).
#Farbreihenfolge: GRB
# Legt die von der LED-Hardware geforderte Pixelreihenfolge fest (unter Verwendung eines Strings
```

```
# die die Buchstaben R, G, B, W enthält, wobei W optional ist). Alternativ dazu, kann dies eine durch
# Kommata getrennte Aufzählung von Pixelreihenfolgen sein - eine für jede LED in der Kette. Die
# Vorgabe ist GRB.
#initial_RED: 0.0
#initial_GREEN: 0.0
#initial_BLUE: 0.0
#initial_WHITE: 0.0
# Informationen zu diesen Parametern finden Sie im Abschnitt "led".
```

[dotstar]

Dotstar (auch bekannt als APA102) LED-Unterstützung (man kann eine beliebige Anzahl von Abschnitten mit einem "dotstar"-Präfix definieren). Siehe die command reference für weitere Informationen.

```
[dotstar my_dotstar]
data_pin:
# Der Pin, der mit der Datenleitung des Dotstars verbunden ist. Dieser Parameter muss angegeben
# werden.
clock_pin:
# Der Pin, der mit der Clock-Leitung des Dotstar verbunden ist. Dieser Parameter muss angegeben
# werden.
#chain_count:
# Siehe Abschnitt "Neopixel" für Informationen zu diesem Parameter.
#initial_RED: 0.0
#initial_GREEN: 0.0
#initial_BLUE: 0.0
# Informationen zu diesen Parametern finden Sie im Abschnitt "led".
```

[pca9533]

PCA9533 LED-Unterstützung. Der PCA9533 wird auf dem mightyboard verwendet.

```
[pca9533 my_pca9533]
#i2c_address: 98
# Die i2c-Adresse, die der Chip auf dem i2c-Bus verwendet. Verwenden Sie 98 für
# den PCA9533/1, 99 für den PCA9533/2. Die Voreinstellung ist 98.
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
#initial_RED: 0.0
#initial_GREEN: 0.0
#initial_BLUE: 0.0
#initial_WHITE: 0.0
# Informationen zu diesen Parametern finden Sie im Abschnitt "led".
```

[pca9632]

PCA9632-LED-Unterstützung. Der PCA9632 wird auf dem FlashForge Dreamer verwendet.

```
[pca9632 my_pca9632]
#i2c_address: 98
# Die i2c-Adresse, die der Chip auf dem i2c-Bus verwendet. Dies kann sein 96, 97, 98 oder 99 sein.
# Die Voreinstellung ist 98.
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
#scl_pin:
#sda_pin:
# Alternativ, wenn der pca9632 nicht an einen Hardware I2C Bus angeschlossen ist, kann man alternativ
# die "clock" (scl_pin) und "data" (sda_pin) Pins angeben. Die Voreinstellung ist die Verwendung von
# Hardware I2C.
#color_order: RGBW
# Legt die Pixelreihenfolge der LED fest (mit einem String, der die Buchstaben R, G, B, W). Die
# Voreinstellung ist RGBW.
#initial_RED: 0.0
#initial_GREEN: 0.0
#initial_BLUE: 0.0
#initial_WHITE: 0.0
# Informationen zu diesen Parametern finden Sie im Abschnitt "led".
```

Zusätzliche Servos, Tasten und andere Pins

[servo]

Servos (man kann eine beliebige Anzahl von Sektionen mit dem Präfix "servo" definieren). Die Servos können mit dem g-code Befehl SET_SERVO gesteuert werden. Zum Beispiel: SET_SERVO SERVO=my_servo ANGLE=180

```
[servo my_servo]
Pin:
# PWM-Ausgangspin zur Steuerung des Servos. Dieser Parameter muss bereitgestellt werden.
#maximum_servo_angle: 180
# Der maximale Winkel (in Grad), auf den dieses Servo eingestellt werden kann. Die
# Voreinstellung ist 180 Grad.
#minimum_pulse_width: 0.001
# Die minimale Pulsbreite (in Sekunden). Dies sollte mit einem Winkel von 0 Grad entsprechen. Die
# Vorgabe ist 0,001 Sekunden.
#maximale_Impulsbreite: 0.002
# Die maximale Impulsbreite (in Sekunden). Dies sollte mit einem Winkel von maximum_servo_angle. Die
# Vorgabe ist 0.002 Sekunden.
#Initialer_Winkel:
# Anfangswinkel (in Grad), auf den das Servo eingestellt werden soll. Die Vorgabe ist, dass
# kein Signal beim Starten gesendet wird.
#initial_pulse_width:
# Anfängliche Pulsbreite (in Sekunden), auf die das Servo eingestellt werden soll. (Diese
# ist nur gültig, wenn initial_angle nicht gesetzt ist). Die Vorgabe ist, kein Signal beim Starten
# gesendet wird.
```

[gcode_button]

Führt gcode aus, wenn eine Taste gedrückt oder losgelassen wird (oder wenn ein Pin seinen Zustand ändert). Du kannst den Zustand des Knopfes mit QUERY_BUTTON button=my_gcode_button überprüfen.

```
[gcode_button my_gcode_button]
pin:
# Der Pin, an dem der Button angeschlossen ist. Dieser Parameter muss angegeben werden.
#analog_range:
# Zwei durch Komma getrennte Widerstände (in Ohm), die den minimalen und maximalen Widerstandsbereich
#  für den Taster. Wenn analog_range angegeben wird angegeben ist, muss der Pin ein analoger Pin sein.
# Der Standard ist die Verwendung eines digitalen gpio für die Taste.
#analog_pullup_resistor:
# Der Pullup-Widerstand (in Ohm), wenn analog_range angegeben ist. Die Vorgabe ist 4700 Ohm.
#press_gcode:
# Eine Aufzählung von G-Code-Befehlen, die ausgeführt werden sollen, wenn die Taste gedrückt wird.
# G-Code-Vorlagen werden unterstützt. Dieser Parameter muss angegeben werden.
#release_gcode:
# Eine Aufzählung von G-Code-Befehlen, die ausgeführt werden sollen, wenn die Taste losgelassen wird.
# G-Code-Vorlagen werden unterstützt. Standardmäßig werden keine Befehle beim Loslassen einer Taste
# ausgeführt.
```

[output_pin]

Zur Laufzeit konfigurierbare Ausgangspins (man kann eine beliebige Anzahl von Abschnitten mit einem "output_pin"-Präfix definieren). Die hier konfigurierten Pins werden als Ausgangspins eingerichtet und können zur Laufzeit mit erweiterten g-code-Befehlen vom Typ "SET_PIN PIN=my_pin VALUE=.1" geändert werden.

```
[output_pin my_pin]
Pin:
# Der Pin, der als Ausgang konfiguriert werden soll. Dieser Parameter muss
# angegeben werden.
#pwm: False
# Legt fest, ob der Ausgangspin für die Pulsweitenmodulation geeignet sein soll. Wenn dies wahr ist,
#  sollten die Wertefelder zwischen 0 und 1 liegen; wenn es false sind, sollten die Wertefelder
# entweder 0 oder 1 sein. Die Voreinstellung ist Falsch.
#static_value:
# Wenn dies gesetzt ist, dann wird der Pin beim Start auf diesen Wert gesetzt und der Pin kann während
# der Laufzeit nicht geändert werden. Ein statischer Pin verbraucht etwas weniger Speicherplatz im
# Mikrocontroller. Die Vorgabe ist, die Laufzeitkonfiguration der Pins.
#Wert:
# Der Wert, auf den der Pin während der MCU-Konfiguration anfänglich gesetzt werden soll.
# Der Standardwert ist 0 (für niedrige Spannung).
#shutdown_value:
# Der Wert, auf den der Pin bei einem MCU-Shutdown-Ereignis gesetzt werden soll. Die Vorgabe
# ist 0 (für niedrige Spannung).
#maximum_mcu_duration:
```

```
# Die maximale Dauer, die ein Nicht-Shutdown-Wert von der MCU gesteuert werden darf
# ohne Bestätigung durch den Host. Wenn der Host nicht mit einer Aktualisierung Schritt halten kann,
# schaltet die MCU ab und setzt alle Pins auf ihre jeweiligen Shutdown-Werte. Standard: 0
# (deaktiviert) Übliche Werte sind etwa 5 Sekunden.
#cycle_time: 0.100
# Die Zeitspanne (in Sekunden) pro PWM-Zyklus. Es wird empfohlen 10 Millisekunden oder mehr zu sein,
# wenn softwarebasierte PWM verwendet wird. Die Vorgabe ist 0,100 Sekunden für pwm-Pins.
#hardware_pwm: False
# Aktivieren Sie dies, um Hardware-PWM anstelle von Software-PWM zu verwenden. Wenn Hardware-PWM wird
# die tatsächliche Zykluszeit durch die Implementierung begrenzt und kann sich erheblich von der
# angeforderten cycle_time. Die Voreinstellung ist False.
#Skalierung:
# Dieser Parameter kann verwendet werden, um zu ändern, wie die Parameter 'value' und
# 'shutdown_value' Parameter für pwm Pins interpretiert werden. Wenn angegeben, dann sollte der
# 'value' Parameter zwischen 0.0 und 'scale' liegen. Dies kann nützlich sein, wenn ein PWM-Pin
# konfiguriert wird, der eine Schrittmotor-Spannungsreferenz steuert. Die 'scale' kann eingestellt
# werden auf die äquivalente Stepper-Stromstärke, wenn die PWM voll aktiviert wäre, und der Parameter
# 'Wert' kann dann mit der gewünschten Amperezahl für den Stepper angegeben werden. Standardmäßig wird
#  der Parameter 'value' nicht  skaliert.
```

[static_digital_output]

Statisch konfigurierte digitale Ausgangspins (man kann eine beliebige Anzahl von Abschnitten mit einem "static_digital_output"-Präfix definieren). Die hier konfigurierten Pins werden während der MCU-Konfiguration als GPIO-Ausgang eingerichtet. Sie können während der Laufzeit nicht geändert werden.

```
[static_digital_output my_output_pins]
Pins:
# Eine durch Komma getrennte Aufzählung von Pins, die als GPIO-Ausgangspins festgelegt werden sollen. # Der Pin
wird auf einen High-Pegel gesetzt, es sei denn, dem Pin-Namen wird mit "!" vorangestellt
# ist. Dieser Parameter muss angegeben werden.
```

[multi_pin]

Ausgänge mit mehreren Pins (man kann eine beliebige Anzahl von Abschnitten mit einem "multi_pin"-Präfix definieren). Ein multi_pin-Ausgang erzeugt einen internen Pin-Alias, der jedes Mal, wenn der Alias-Pin gesetzt wird, mehrere Ausgangspins ändern kann. Beispielsweise könnte man ein "[multi_pin my_fan]"-Objekt definieren, das zwei Pins enthält, und dann "pin=multi_pin:my_fan" im "[fan]"-Abschnitt setzen - bei jeder Änderung des Lüfters würden dann beide Ausgangspins aktualisiert werden. Diese Aliasnamen dürfen nicht mit Schrittmotor-Pins verwendet werden.

```
[multi_pin my_multi_pin]
Pins:
# Eine durch Kommata getrennte Aufzählung von Pins, die mit diesem Alias verknüpft sind. Dieser
# Parameter muss angegeben werden.
```

TMC-Schrittmotortreiber-Konfiguration

Konfiguration der Trinamic-Schrittmotortreiber im UART/SPI-Modus. Weitere Informationen finden Sie in der Anleitung für TMC Drivers guide und in der command reference.

[tmc2209

Konfigurieren Sie einen TMC2209-Schrittmotortreiber über Single-Wire-UART. Um diese Funktion zu nutzen, definieren Sie einen Konfigurationsabschnitt mit einem "tmc2209"-Präfix, gefolgt vom Namen des entsprechenden Schrittmotor-Konfigurationsabschnitts (z. B. "[tmc2209 stepper_x]").

```
[tmc2209 stepper_x]
uart_pin:
#tx_pin:
#select_pins:
#interpolate: True
run_current:
#hold_current:
#sense_resistor: 0.110
#stealthchop_threshold: 0
# Siehe Abschnitt "tmc2208" für die Definition dieser Parameter.
#uart_address:
# Die Adresse des TMC2209-Chips für UART-Nachrichten (eine ganze Zahl zwischen 0 und 3). Dies wird
# typischerweise verwendet, wenn mehrere TMC2209 Chips an denselben UART-Pin angeschlossen sind. Der
# Standardwert ist Null.
#driver_IHOLDDELAY: 8
#treiber_TPOWERDOWN: 20
#driver_TBL: 2
#driver_TOFF: 3
#driver_HEND: 0
```

```
#treiber_HSTRT: 5
#driver_PWM_AUTOGRAD: Wahr
#driver_PWM_AUTOSCALE: Wahr
#treiber_PWM_LIM: 12
#Treiber_PWM_REG: 8
#treiber_PWM_FREQ: 1
#treiber_PWM_GRAD: 14
#treiber_PWM_OFS: 36
#driver_SGTHRS: 0
# Setzen Sie das angegebene Register während der Konfiguration des TMC2209 Chips. Dies kann verwendet
#,werden, um eigene Motorparameter zu setzen. Die Standardwerte für jeden Parameter stehen neben dem
# Parameternamen in der obigen Aufzählung.
#diag_pin:
# Der Mikrocontroller-Pin, der an die DIAG-Leitung des TMC2209 Chip angeschlossen ist. Dem Pin wird
# normalerweise ein "^" vorangestellt, um einen Pullup zu aktivieren.
# Das Setzen dieses Pins erzeugt einen virtuellen "tmc2209_stepper_x:virtual_endstop"Pin, der als
# Endstop-Pin des Steppers verwendet werden kann. Diese Einstellung ermöglicht eine "sensorlose
# Referenzfahrt". (Stellen Sie sicher, dass Sie auch driver_SGTHRS auf einen angemessenen Wert für die
# Empfindlichkeit setzen.) Die Standard ist, dass die
# sensorlose Referenzfahrt deaktiviert ist.
```

Display-Unterstützung

[display]

Unterstützung für ein an den Mikrocontroller angeschlossenes Displays

```
# Diese Datei enthält eine Beispielkonfiguration für gängige "RepRap"-Style
# LCD-Displays, die EXP1/EXP2-Stecker verwenden.

# Um ein Display aus dieser Datei zu konfigurieren, stellen Sie sicher, dass die Hauptdatei
# Datei printer.cfg einen Abschnitt [board_pins] config enthält, der die Pin
# Aliasen für die EXP1/EXP2-Stecker definiert, suchen Sie den entsprechenden LCD-Typ in
# dieser Datei, und fügen Sie diesen Abschnitt in printer.cfg ein.

# Siehe docs/Config_Reference.md für eine Beschreibung der Parameter.

##################################################################
# MKS Mini 12864 LCD
##################################################################

# Stellen Sie sicher, dass EXP1 und EXP2 auf der Anzeigetafel richtig gedreht sind.
# Display-Platine gedreht sind. Die Aussparungen an den Anschlüssen sollten zur
# Mitte der Platine zeigen. Siehe:
# https://reprap.org/wiki/MKS_MINI_12864#Physical_Interface
# Wenn sie falsch sind, kann das Steckergehäuse mit einem kleinen Schraubenzieher abhebeln und die
# Ausrichtung korrigieren.

[display]
lcd_type: uc1701
cs_pin: EXP1_6
a0_pin: EXP1_7
kontrast: 40
encoder_pins: ^EXP2_3, ^EXP2_5
click_pin: ^!EXP1_2
## Für einige Mikrocontroller-Boards muss möglicherweise ein Spi-Bus angegeben werden:
#spi_bus: spi
## Alternativ dazu können einige Mikrocontroller-Boards mit Software-Spi arbeiten:
#spi_software_miso_pin: EXP2_1
#spi_software_mosi_pin: EXP2_6
#spi_software_sclk_pin: EXP2_2

[output_pin beeper]
Pin: EXP1_1

##################################################################
# Fysetc Mini 12864Panel v2.1 (mit Neopixel-LEDs als Hintergrundbeleuchtung)
##################################################################

[display]
lcd_type: uc1701
cs_pin: EXP1_3
a0_pin: EXP1_4
```

```
rst_pin: EXP1_5
Kontrast: 63
kodierer_pins: ^EXP2_5, ^EXP2_3
click_pin: ^!EXP1_2
## Für einige Mikrocontroller-Boards muss möglicherweise ein Spi-Bus angegeben werden:
#spi_bus: spi
## Alternativ dazu können einige Mikrocontroller-Boards mit Software-Spi arbeiten:
#spi_software_miso_pin: EXP2_1
#spi_software_mosi_pin: EXP2_6
#spi_software_sclk_pin: EXP2_2

[output_pin beeper]
Pin: EXP1_1

[neopixel fysetc_mini12864]
pin: EXP1_6
chain_count: 3
farbe_ordnung: RGB
initial_RED: 0.4
anfänglich_GRÜN: 0.4
initial_BLUE: 0.4


#####################################################################
# Positionen der Steckerstifte
#####################################################################

# Einige Mikrocontroller-Platinen und Displays verwenden inkonsistente Beschriftungen
# für die EXP1- und EXP2-Header. Das folgende Diagramm zeigt die
# korrekte Position von Pin 1 zusammen mit den Masse- und Stromversorgungspins auf den
# EXP1- und EXP2-Steckern:
#
#      EXP1:                EXP2:
# +-----------------+ +-----------------+
# |  o  o  o  o  5V | |  o  o  o  o  o  |
# |  1  o  o  o GND | |  1  o  o  o GND |
# +------     ------+ +------     ------+
#
# Bei einigen Platinen kann sich der Ausschnitt an der falschen Stelle befinden. Wenn dies der Fall
# ist, kann es kann es möglich sein, die Kunststoffleiste mit einem kleinen Schraubendreher
# vorsichtig von den Stiften zu lösen, dann die Ausrichtung zu korrigieren und die
# Plastikkopfleiste wieder anbringen.
```

Display-Klipper-Konfiguration

Die meisten Voron-Konfigurationsdateien haben bereits entsprechende Konfigurationen für diese Anzeige eingebaut, die einfach unkommentiert werden müssen. Diese Konfigurationen sind für die spezifischen Builds angepasst und sollten von Ihnen verwendet werden. Hinweis: Es sind mehrere Konfigurationsabschnitte erforderlich, um das mini12864 voll funktionsfähig zu machen: [display], [neopixel fysetc_mini12864] und [delayed_gcode setdisplayneopixel]_

Manche Benutzer möchten die Drehrichtung des Menürads umkehren. Sie können die Funktion einfach umkehren, indem Sie die Reihenfolge der beiden Pins in der Zeile [display] encoder_pins vertauschen. Wenn Ihre Standardkonfiguration zum Beispiel die folgende Zeile enthält
`encoder_pins: ^PC7,^PC6`
änderst du sie in
`encoder_pins: ^PC6,^PC7`

Mini12864 Checkliste zur Fehlersuche

Das Mini12864-Display kann ein wenig knifflig sein, wenn man es richtig zum Laufen bringen will. Hier ist eine kurze Checkliste, die helfen soll, einige der häufigsten Probleme zu überprüfen.

- Haben Sie die Header auf der Rückseite des Displays gedreht? Siehe Hardware oben
- Ist EXP1 mit EXP1 und EXP2 mit EXP2 verbunden?
- Haben Sie alle Konfigurationsabschnitte aktiviert?

Sie müssen [display], [output_pin beeper], [neopixel fysetc_mini12864], UND [delayed_gcode setdisplayneopixel] haben, um alle Funktionen Ihres Displays vollständig zu aktivieren. (viele Benutzer werden es nicht für nötig halten, [output_pin beeper] zu aktivieren)

[display_data]

Unterstützung für die Anzeige von benutzerdefinierten Daten auf einem LCD-Bildschirm. Man kann eine beliebige Anzahl von Anzeigegruppen und eine beliebige Anzahl von Datenelementen unter diesen Gruppen erstellen. Die Anzeige zeigt alle Daten einer bestimmten Gruppe an, wenn die Option display_group im Abschnitt [display] auf den angegebenen Gruppennamen gesetzt ist.

Ein Standardsatz von Anzeigegruppen wird automatisch erstellt. Sie können diese display_data-Elemente ersetzen oder erweitern, indem Sie die Standardeinstellungen in der Hauptkonfigurationsdatei printer.cfg außer Kraft setzen.

```
[display_data my_group_name my_data_name]
Position:
# Durch Kommata getrennte Zeile und Spalte der Anzeigeposition, die
# die für die Anzeige der Informationen verwendet werden soll. Dieser Parameter muss
# angegeben werden.
text:
# Der Text, der an der angegebenen Position angezeigt werden soll. Dieses Feld wird ausgewertet
# Dieses Feld wird mit Hilfe von Befehlsvorlagen ausgewertet (siehe docs/Command_Templates.md). Dieser
# Parameter muss angegeben werden.
```

[display_template]

Anzeige von Datentext-"Makros" (man kann eine beliebige Anzahl von Abschnitten mit einem display_template-Präfix definieren). Informationen zur Auswertung von Vorlagen finden Sie im Dokument zu den Befehlsvorlagen.

Diese Funktion ermöglicht es, sich wiederholende Definitionen in display_data-Abschnitten zu reduzieren. Man kann die eingebaute render()-Funktion in display_data-Abschnitten verwenden, um eine Vorlage auszuwerten. Wenn man zum Beispiel [display_template my_template] definiert, kann man { render('my_template') } in einem display_data-Abschnitt verwenden.

Diese Funktion kann auch für kontinuierliche LED-Aktualisierungen mit dem Befehl SET_LED_TEMPLATE verwendet werden.

```
[display_template my_template_name]
#param_<name>:
# Man kann eine beliebige Anzahl von Optionen mit einem "param_"-Präfix angeben. Der
# angegebene Name wird mit dem angegebenen Wert belegt (geparst als Python
# Literal) und ist während der Makroexpansion verfügbar. Wenn der
# Parameter im Aufruf von render() übergeben wird, wird dieser Wert
# während der Makroexpansion verwendet. Zum Beispiel, eine Konfiguration mit
# "param_speed = 75" könnte einen Aufrufer haben mit
# "render('mein_template_name', param_speed=80)". Parameternamen dürfen
# keine Großbuchstaben verwenden.
text:
# Der Text, der zurückgegeben werden soll, wenn die Vorlage gerendert wird. Dieses Feld
# wird mit Befehlsvorlagen ausgewertet (siehe
# docs/Command_Templates.md). Dieser Parameter muss angegeben werden.
```

[display_glyph]

Zeigt eine benutzerdefinierte Glyphe auf Displays an, die sie unterstützen. Der angegebene Name wird den angegebenen Anzeigedaten zugewiesen, die dann in den Anzeigevorlagen durch ihren Namen, umgeben von zwei "Tilde"-Symbolen, referenziert werden können, z.B. ~my_display_glyph~

Siehe sample-glyphs.cfg für einige Beispiele.

```
[display_glyph my_display_glyph]
#Daten:
# Die Anzeigedaten, gespeichert als 16 Zeilen, bestehend aus 16 Bits (1 pro
# Pixel), wobei '.' für ein leeres Pixel und '*' für ein eingeschaltetes Pixel steht (z.B.,
# ****************", um eine durchgehende horizontale Linie anzuzeigen).
# Alternativ kann man auch '0' für ein leeres Pixel und '1' für ein eingeschaltetes Pixel verwenden.
# Pixel verwenden. Legen Sie jede Anzeigezeile in eine separate Konfigurationszeile. Die
# Glyphe muss aus genau 16 Zeilen mit je 16 Bit bestehen. Dieser
# Parameter ist optional.
#hd44780_data:
# Glyphe zur Verwendung auf 20x4 hd44780-Displays. Die Glyphe muss bestehen aus
# genau 8 Zeilen mit je 5 Bits bestehen. Dieser Parameter ist optional.
# hd44780_slot:
# Der hd44780-Hardware-Index (0..7), unter dem die Glyphe gespeichert werden soll. Wenn
# mehrere unterschiedliche Bilder denselben Slot verwenden, dann stellen Sie sicher, dass nur
# nur eines dieser Bilder in einem bestimmten Bildschirm zu verwenden. Dieser Parameter ist
# erforderlich, wenn hd44780_data angegeben ist.
```

[display my_extra_display]

Wenn ein primärer [display]-Abschnitt in printer.cfg wie oben gezeigt definiert wurde, ist es möglich, mehrere Hilfsanzeigen zu definieren. Beachten Sie, dass Hilfsdisplays derzeit keine Menüfunktionalität unterstützen, d. h. sie unterstützen weder die "Menü"-Optionen noch die Konfiguration der Schaltflächen.

```
[display my_extra_display]
# Siehe Abschnitt "display" für die verfügbaren Parameter.
```

[menu]

Anpassbare Menüs auf dem LCD-Display.

Ein default set of menus wird automatisch erstellt. Sie können das Menü ersetzen oder erweitern, indem Sie die Standardeinstellungen in der Konfigurationsdatei printer.cfg überschreiben.

Informationen zu den Menüattributen, die beim Rendern der Vorlage verfügbar sind, finden Sie im Dokument zur Befehlsvorlage command template document.

```
# Gemeinsame Parameter, die für alle Menükonfigurationsabschnitte verfügbar sind.
#[menu __some_list __some_name]
#type: deaktiviert
# Dauerhaft deaktiviertes Menüelement, einziges erforderliches Attribut ist 'type'.
# Ermöglicht das einfache Deaktivieren/Ausblenden vorhandener Menüpunkte.

#[menu some_name]
#type:
# Eines von command, input, list, text:
# command - grundlegendes Menüelement mit verschiedenen Script-Triggern
# input - wie 'command', aber mit der Möglichkeit, Werte zu ändern.
# Drücken startet/stoppt den Bearbeitungsmodus.
# Liste - ermöglicht die Gruppierung von Menüpunkten in einer
# scrollbaren Liste.  Hinzufügen zur Liste durch Erstellen von Menü
# Konfigurationen mit "some_list" als Präfix - zum
# Beispiel: [menu some_list some_item_in_the_list]
# vsdlist - wie 'list', aber es werden Dateien von der virtuellen # SD-Karte angehängt.
# (wird in Zukunft entfernt)
#name:
# Name des Menüpunkts - wird als Vorlage ausgewertet.
#enable:
# Schablone, die auf True oder False ausgewertet wird.
#index:
# Position, an der ein Eintrag in die Liste eingefügt werden soll. Standardmäßig
# wird der Eintrag am Ende eingefügt.

#[menu some_list]
#Typ: Liste
#name:
#enable:
# Siehe oben für eine Beschreibung dieser Parameter.

#[menu some_list some_command]
#type: Befehl
#name:
#enable:
# Siehe oben für eine Beschreibung dieser Parameter.
#gcode:
# Skript, das beim Klicken einer Schaltfläche oder eines langen Klicks ausgeführt wird. Ausgewertet
# als Vorlage.

#[menu some_list some_input]
#Typ: Eingabe
#name:
#enable:
# Siehe oben für eine Beschreibung dieser Parameter.
#input:
# Anfangswert, der bei der Bearbeitung verwendet wird - wird als Vorlage ausgewertet. Ergebnis muss
# Float sein.
#input_min:
# Mindestwert des Bereichs - wird als Vorlage ausgewertet. Voreinstellung -99999.
#input_max:
# Maximalwert des Bereichs - wird als Schablone ausgewertet. Voreinstellung 99999.
#eingabe_schritt:
# Editierschritt - Muss ein positiver Integer- oder Float-Wert sein. Es hat
# internen schnellen Schritt. Wenn "(input_max - input_min) /
# (input_max - input_min) / # input_step > 100", dann ist der schnelle Ratenschritt 10 * input_step,
# sonst ist der schnelle
# Ratenschritt ist gleich input_step.
#Echtzeit:
# Dieses Attribut akzeptiert einen statischen booleschen Wert. Wenn aktiviert, dann
# wird das gcode-Skript nach jeder Wertänderung ausgeführt. Der Standardwert ist False.
```

```
#gcode:
# Skript, das beim Klick auf die Schaltfläche, beim langen Klick oder bei einer Wertänderung
# ausgeführt wird.
# Wird als Vorlage ausgewertet. Der Schaltflächenklick löst den Start oder das Ende des
# Bearbeitungsmodus aus.
# Modus starten oder beenden.
```

Filament-Sensoren

[filament_switch_sensor]

Filament-Schalter-Sensor. Unterstützung für die Erkennung von Filamenteinlagen und -ausläufen mithilfe eines Schaltersensors, z. B. eines Endstoppschalters.

Siehe die Befehlsreferenz für weitere Informationen.

```
[filament_switch_sensor my_sensor]
#pause_on_runout: True
# Wenn diese Option auf True gesetzt ist, wird eine PAUSE unmittelbar nach der Erkennung eines
# Auslaufs ausgeführt. erkannt wird. Beachten Sie, dass, wenn pause_on_runout auf False steht und der
# runout_gcode weggelassen wird, ist die Auslauferkennung deaktiviert. Voreinstellung ist True.
#runout_gcode:
# Eine Liste von G-Code-Befehlen, die ausgeführt werden sollen, nachdem ein Filament-Runout erkannt
# wurde. Siehe docs/Command_Templates.md für das G-Code-Format. Wenn pause_on_runout auf True gesetzt
#  ist, wird dieser G-Code ausgeführt, nachdem die PAUSE beendet ist. In der Voreinstellung werden
# keine G-Code-Befehle ausgeführt.
#insert_gcode:
# Eine Liste von G-Code-Befehlen, die ausgeführt werden, nachdem ein Filamenteinsatz erkannt wird.
# Siehe docs/Command_Templates.md für das G-Code-Format. Die Voreinstellung ist, keine G-Code-Befehle
# auszuführen, welches die Filamenterkennung deaktiviert.
#event_delay: 3.0
# Die minimale Zeitspanne in Sekunden, die zwischen den Ereignissen vergehen soll. Ereignisse, die
# während dieser Zeitspanne ausgelöst werden,  werden stillschweigendignoriert. Der Standardwert ist 3
# Sekunden.
#pause_delay: 0.5
# Die Zeitspanne in Sekunden, die zwischen dem Senden des Pause-Befehls und der Ausführung des
# runout_gcode. Es kann sinnvoll sein, diese diese Verzögerung zu erhöhen, wenn OctoPrint ein
# seltsames Pausenverhalten zeigt. Standard ist 0,5 Sekunden.
#switch_pin:
# Der Pin, an dem der Schalter angeschlossen ist. Dieser Parameter muss angegeben werden.
```

[filament_motion_sensor]

Filament Motion Sensor. Unterstützung für die Erkennung von Filamentein- und -auslauf mithilfe eines Encoders, der den Ausgangspin während der Filamentbewegung durch den Sensor umschaltet.

Siehe die command referencefür weitere Informationen.

```
[filament_motion_sensor my_sensor]
detection_length: 7.0
# Die minimale Länge des Fadens, der durch den Sensor gezogen wird, um eine eine Zustandsänderung am
# switch_pin auszulösen. Standard ist 7 mm.
extruder:
# Der Name des Extruderabschnitts, dem dieser Sensor zugeordnet ist. Dieser Parameter muss angegeben
# werden.
switch_pin:
#pause_on_runout:
#runout_gcode:
#insert_gcode:
#event_delay:
#pause_delay:
# Siehe den Abschnitt "filament_switch_sensor" für eine Beschreibung der obigen Parameter.
```

[tsl1401cl_filament_width_sensor]

TSLI401CL-basierter Filamentbreitensensor. Weitere Informationen finden Sie in der Anleitung.

```
[tsl1401cl_filament_width_sensor]
#pin:
#default_nominal_filament_diameter: 1.75 # (mm)
# Maximal zulässiger Unterschied im Filamentdurchmesser in mm.
#max_difference: 0.2
# Der Abstand vom Sensor zur Schmelzkammer in mm.
```

```
#measurement_delay: 100
```

[hall_filament_width_sensor]

Hall-Filamentbreitensensor (siehe Hall-Filamentbreitensensor).

```
[hall_filament_width_sensor]
adc1:
adc2:
# Analoge Eingangsstifte, die mit dem Sensor verbunden sind. Diese Parameter müssen
# bereitgestellt werden.
#cal_dia1: 1.50
#cal_dia2: 2.00
# Die Kalibrierungswerte (in mm) für die Sensoren. Der Standardwert ist
# 1.50 für cal_dia1 und 2.00 für cal_dia2.
#raw_dia1: 9500
#raw_dia2: 10500
# Die Rohkalibrierungswerte für die Sensoren. Der Standardwert ist 9500
# für raw_dia1 und 10500 für raw_dia2.
#default_nominal_filament_diameter: 1.75
# Der nominale Filamentdurchmesser. Dieser Parameter muss angegeben werden.
#max_difference: 0.200
# Maximal zulässiger Unterschied im Filamentdurchmesser in Millimetern (mm).
# Wenn die Differenz zwischen dem nominalen Filamentdurchmesser und dem Sensorausgang
# mehr als +- max_difference ist, wird der Extrusionsmultiplikator
# auf %100 zurückgesetzt. Der Standardwert ist 0,200.
# Messung_Verzögerung: 70
# Der Abstand vom Sensor zur Schmelzkammer/zum heißen Ende in
# Millimetern (mm). Der Faden zwischen dem Sensor und dem heißen Ende
# wird als default_nominal_filament_diameter behandelt. Host
# Modul arbeitet mit FIFO-Logik. Es speichert jeden Sensorwert und jede Position in einem Array und
# setzt sie in die richtige Position zurück. Dieser
# Parameter muss angegeben werden.
#Aktivieren: False
# Sensor aktiviert oder deaktiviert nach dem Einschalten. Die Voreinstellung ist deaktivieren.
#measurement_interval: 10
# Der ungefähre Abstand (in mm) zwischen den Sensormessungen. Die Voreinstellung ist 10 mm.
#logging: False
# Out diameter to terminal and klipper.log can be turn on|of by Befehl.
#min_diameter: 1.0
# Minimaler Durchmesser für Trigger virtueller filament_switch_sensor.
#use_current_dia_while_delay: False
# Use the current diameter instead of the nominal diameter while die Messverzögerung noch nicht
# durchgelaufen ist.
#pause_on_runout:
#runout_gcode:
#insert_gcode:
#event_delay:
#pause_delay:
# Siehe den Abschnitt "filament_switch_sensor" für eine Beschreibung der obigen Parameter.
```

Board-spezifische Hardware-Unterstützung

[sx1509]

Konfigurieren Sie einen SX1509 I2C zu GPIO Expander. Aufgrund der Verzögerung, die durch die I2C-Kommunikation entsteht, sollten Sie die SX1509-Pins NICHT als Stepper-Enable-, Step- oder Dir-Pins oder andere Pins verwenden, die schnelles Bit-Banging erfordern. Sie werden am besten als statische oder gcode-gesteuerte digitale Ausgänge oder Hardware-Pwm-Pins für z.B. Lüfter verwendet. Man kann eine beliebige Anzahl von Sektionen mit einem "sx1509"-Präfix definieren. Jeder Expander bietet einen Satz von 16 Pins (sx1509_my_sx1509 : PIN_0 bis sx1509_my_sx1509 : PIN_15), die in der Druckerkonfiguration verwendet werden können.

Siehe die Datei generic-duet2-duex.cfg für ein Beispiel.

```
[sx1509 my_sx1509]
i2c_address:
# I2C-Adresse, die von diesem Expander verwendet wird. Abhängig von der Hardware Jumper ist dies eine
# der folgenden Adressen: 62 63 112 113. Dieser Parameter muss angegeben werden.
#i2c_mcu:
#i2c_bus:
#i2c_speed:
# Siehe den Abschnitt "Allgemeine I2C-Einstellungen" für eine Beschreibung der obigen Parameter.
#i2c_bus:
# Wenn die I2C-Implementierung Ihres Mikrocontrollers mehrere I2C-Busse unterstützt, können Sie hier
```

```
# den Busnamen angeben. Standard ist die Verwendung des Standard-i2c-Busses des Mikrocontrollers.
```

[samd_sercom]

SAMD SERCOM-Konfiguration, um festzulegen, welche Pins auf einer bestimmten SERCOM verwendet werden sollen. Man kann eine beliebige Anzahl von Abschnitten mit einem "samd_sercom"-Präfix definieren. Jede SERCOM muss konfiguriert werden, bevor sie als SPI- oder I2C-Peripheriegerät verwendet werden kann. Platzieren Sie diesen Konfigurationsabschnitt über jedem anderen Abschnitt, der den SPI- oder I2C-Bus verwendet.

```
[samd_sercom my_sercom]
sercom:
# Der Name des sercom-Busses, der im Mikrocontroller konfiguriert werden soll.
# Verfügbare Namen sind "sercom0", "sercom1", usw.. Dieser Parameter
# muss angegeben werden.
tx_pin:
# MOSI-Pin für SPI-Kommunikation oder SDA (Daten)-Pin für I2C
# Kommunikation. Der Pin muss eine gültige Pinmux-Konfiguration haben
# für das gegebene SERCOM-Peripheriegerät haben. Dieser Parameter muss angegeben werden.
#rx_pin:
# MISO-Pin für die SPI-Kommunikation. Dieser Pin wird nicht für die I2C
# Kommunikation verwendet (I2C verwendet tx_pin sowohl zum Senden als auch zum Empfangen).
# Der Pin muss eine gültige Pinmux-Konfiguration für die angegebene
# SERCOM-Peripheriegerät haben. Dieser Parameter ist optional.
clk_pin:
# CLK-Pin für SPI-Kommunikation oder SCL (Takt) Pin für I2C
# Kommunikation. Der Pin muss eine gültige Pinmux-Konfiguration haben
# für das angegebene SERCOM-Peripheriegerät haben. Dieser Parameter muss angegeben werden.
```

[adc_scaled]

Duet2 Maestro analoge Skalierung durch vref und vssa Messwerte. Die Definition einer adc_scaled-Sektion ermöglicht virtuelle adc-Pins (wie z.B. "my_name:PB0"), die automatisch durch die vref- und vssa-Überwachungspins der Karte angepasst werden. Stellen Sie sicher, dass Sie diesen Konfigurationsabschnitt über allen Konfigurationsabschnitten definieren, die einen dieser virtuellen Pins verwenden.

Siehe die Datei generic-duet2-maestro.cfg für ein Beispiel.

```
[adc_scaled my_name]
vref_pin:
# Der ADC-Pin, der für die VREF-Überwachung verwendet werden soll. Dieser Parameter muss
# angegeben werden.
vssa_pin:
# Der ADC-Pin, der für die VSSA-Überwachung verwendet werden soll. Dieser Parameter muss
# bereitgestellt werden.
#smooth_time: 2.0
# Ein Zeitwert (in Sekunden), über den die vref- und vssa Messungen geglättet werden, um die
# Auswirkungen des Messrauschens zu reduzieren. Rauschen zu reduzieren. Der Standardwert ist 2
# Sekunden.
```

[replicape]

Replicape-Unterstützung - siehe den Beaglebone-Leitfaden und die Datei generic-replicape.cfg für ein Beispiel.

```
# Der "replicape" Konfigurationsabschnitt fügt "replicape:stepper_x_enable" hinzu.
# virtuelle Stepper-Enable-Pins (für die Stepper X, Y, Z, E und H) und "replicape:power_x" PWM-
# Ausgangspins (für hotbed, e, h, fan0, fan1, fan2, and fan3), die dann an anderer Stelle in der
#  Konfigurationsdatei verwendet werden können.
[replicape]
Revision:
# Die Replicape-Hardware-Revision. Derzeit wird nur die Revision "B3" unterstützt. Dieser Parameter
# muss angegeben werden.
#enable_pin: !gpio0_20
# Der globale Enable-Pin des Replikators. Der Standardwert ist !gpio0_20 (auch bekannt als P9_41).
host_mcu:
# Der Name der mcu-Konfigurationssektion, die mit dem Klipper "linux process" mcu Instanz
# kommuniziert. Dieser Parameter muss angegeben werden.
#standstill_power_down: False
# Dieser Parameter steuert die CFG6_ENN-Zeile auf allen Schrittmotoren Motoren. True setzt die
Freigabeleitungen auf "offen". Die Voreinstellung ist False.
#stepper_x_microstep_mode:
#stepper_y_microstep_mode:
#stepper_z_microstep_mode: #stepper_z_microstep_mode:
#stepper_e_microstep_mode:
#stepper_h_microstep_mode:
# Dieser Parameter steuert die CFG1 und CFG2 Pins des angegebenen Schrittmotortreibers. Verfügbare
```

```
# Optionen sind: disable, 1, 2, spread2, 4, 16, spread4, spread16, stealth4, und stealth16. Die
# Voreinstellung ist disable.
#stepper_x_current:
#stepper_y_current:
#stepper_z_current:
#stepper_e_current:
#stepper_h_current:
# Der konfigurierte maximale Strom (in Ampere) des Schrittmotors Treibers. Dieser Parameter muss angegeben
werden, wenn sich der Stepper nicht in einem Deaktivierungsmodus befindet.
#stepper_x_chopper_off_time_high:
#stepper_y_chopper_off_time_high:
#stepper_z_chopper_off_time_high:
#stepper_e_chopper_off_time_high: #stepper_e_chopper_off_time_high:
#stepper_h_chopper_off_time_high: #stepper_h_chopper_off_time_high:
# Dieser Parameter steuert den CFG0-Pin des Schrittmotortreibers. (True setzt CFG0 hoch, False setzt
# ihn niedrig). Die Voreinstellung ist False.
#stepper_x_chopper_hysteresis_high:
#stepper_y_chopper_hysteresis_high:
#stepper_z_chopper_hysteresis_high: #stepper_z_chopper_hysteresis_high:
#stepper_e_chopper_hysteresis_high: #stepper_e_chopper_hysteresis_high:
#stepper_h_chopper_hysteresis_high: #stepper_h_chopper_hysteresis_high:
# Dieser Parameter steuert den CFG4-Pin des Schrittmotortreibers.
# (True setzt CFG4 hoch, False setzt ihn niedrig). Die Voreinstellung ist False.
#stepper_x_chopper_blank_time_high:
#stepper_y_chopper_blank_time_high: #stepper_z_chopper_blank_time_high:
#stepper_z_chopper_blank_time_high: #stepper_z_chopper_blank_time_high:
#stepper_e_chopper_blank_time_high: #stepper_e_chopper_blank_time_high:
#stepper_h_chopper_blank_time_high: #stepper_h_chopper_blank_time_high:
# Dieser Parameter steuert den CFG5-Pin des Schrittmotortreibers
# (True setzt CFG5 hoch, False setzt ihn niedrig). Die Voreinstellung ist True.
```

Andere benutzerdefinierte Module

[palette2]

Palette 2 Multimaterial-Unterstützung - bietet eine engere Integration mit Unterstützung von Palette 2-Geräten im angeschlossenen Modus.

Dieses Modul benötigt auch [virtual_sdcard] und [pause_resume] für die volle Funktionalität.

Wenn Sie dieses Modul verwenden, benutzen Sie nicht das Palette 2 Plugin für Octoprint, da es zu Konflikten kommt und 1 nicht richtig initialisiert werden kann, was zu einem Abbruch des Druckvorgangs führen kann.

Wenn Sie Octoprint verwenden und gcode über die serielle Schnittstelle streamen, anstatt von virtual_sd zu drucken, dann entfernen Sie M1 und M0 aus den Pausenbefehlen in Einstellungen > Serielle Verbindung > Firmware & Protokoll, um zu verhindern, dass der Druck auf der Palette 2 gestartet und die Pause in Octoprint aufgehoben werden muss, damit der Druck beginnt.

```
[palette2]
seriell:
# Die serielle Schnittstelle für die Verbindung mit der Palette 2.
#Baud: 115200
# Die zu verwendende Baudrate. Die Standardeinstellung ist 115200.
#feedrate_splice: 0.8
# Die Vorschubrate, die beim Spleißen verwendet werden soll, Standard ist 0.8.
#feedrate_normal: 1.0
# Die Vorschubgeschwindigkeit, die nach dem Spleißen verwendet werden soll, Standard ist 1.0.
#auto_load_speed: 2
# Extrudiervorschub beim automatischen Laden, Voreinstellung ist 2 (mm/s)
#auto_cancel_variation: 0.1
# Automatischer Druckabbruch, wenn die Ping-Variation über diesem Schwellenwert liegt
```

[winkel]

Unterstützung für magnetische Hall-Winkelsensoren zum Auslesen von Schrittmotor-Winkelwellenmessungen mit a1333, as5047d oder tle5012b SPI-Chips. Die Messungen sind über den API-Server und das Bewegungsanalysetool verfügbar. Siehe die G-Code-Referenz für die verfügbaren Befehle.

```
[angle my_angle_sensor]
sensor_type:
# Der Typ des magnetischen Hall-Sensor-Chips. Verfügbare Auswahlmöglichkeiten sind
# "a1333", "as5047d" und "tle5012b". Dieser Parameter muss
# angegeben werden.
#sample_period: 0.000400
# Der Abfragezeitraum (in Sekunden), der bei den Messungen verwendet werden soll. Standardwert ist
# 0.000400 (das sind 2500 Proben pro Sekunde).
```

```
# stepper:
# Der Name des Steppers, an den der Winkelsensor angeschlossen ist (z.B., "stepper_x"). Das Setzen
# dieses Wertes aktiviert eine Winkelkalibrierung. Werkzeug. Um diese Funktion zu nutzen, muss das
# Python-Paket "numpy" installiert sein. installiert sein. In der Voreinstellung ist die
# Winkelkalibrierung für den Winkelsensor nicht aktiviert.
cs_pin:
# Der SPI-Freigabe-Pin für den Sensor. Dieser Parameter muss angegeben werden.
#spi_speed:
#spi_bus:
#spi_software_sclk_pin:
#spi_software_mosi_pin:
#spi_software_miso_pin:
# Siehe den Abschnitt "Allgemeine SPI-Einstellungen" für eine Beschreibung der
# obigen Parameter.
```

Gemeinsame Bus-Parameter

Gemeinsame SPI-Einstellungen

Die folgenden Parameter sind für Geräte, die einen SPI-Bus verwenden, allgemein verfügbar.

```
#spi_speed:
# Die SPI-Geschwindigkeit (in hz), die bei der Kommunikation mit dem Gerät verwendet werden soll.
# Die Voreinstellung hängt vom Gerätetyp ab.
#spi_bus:
# Wenn der Mikrocontroller mehrere SPI-Busse unterstützt, kann man den Namen des Mikrocontroller-
# Busses hier angeben. Die Vorgabe hängt ab von dem Typ des Mikrocontrollers ab.
#spi_software_sclk_pin:
#spi_software_mosi_pin:
#spi_software_miso_pin:
# Geben Sie die obigen Parameter an, um "softwarebasiertes SPI" zu verwenden. Dieser Modus erfordert
# keine Mikrocontroller-Hardwareunterstützung (typischerweise können beliebige Allzweck-Pins verwendet
# werden). Die Voreinstellung ist, kein "software spi".
```

Allgemeine I2C-Einstellungen

Die folgenden Parameter sind im Allgemeinen für Geräte verfügbar, die einen I2C-Bus verwenden.

```
#i2c_address:
# Die i2c-Adresse des Geräts. Diese muss als Dezimalzahl angegeben werden
# Zahl angegeben werden (nicht in Hex). Die Voreinstellung hängt von der Art des Geräts ab.
#i2c_mcu:
# Der Name des Mikrocontrollers, an den der Chip angeschlossen ist. Die Vorgabe ist "mcu".
#i2c_bus:
# Wenn der Mikrocontroller mehrere I2C-Busse unterstützt, kann man hier den Namen des Mikrocontrollers
# angeben. Die Vorgabe hängt ab von dem Typ des Mikrocontrollers ab.
#i2c_speed:
# Die I2C-Geschwindigkeit (in Hz), die bei der Kommunikation mit dem Gerät verwendet werden soll.
# Bei einigen Mikrocontrollern hat das Ändern dieses Wertes keine Auswirkungen. Die
# Voreinstellung ist 100000.
```

Rotation distance

Schrittmotortreiber auf Klipper benötigen einen rotation_distance-Parameter in jeder Stepper-Konfigurationssektion stepper config section.. Die rotation_distance ist die Strecke, die die Achse bei einer vollen Umdrehung des Schrittmotors zurücklegt. Dieses Dokument beschreibt, wie man diesen Wert konfigurieren kann.

Ermitteln von rotation_distance aus steps_per_mm (oder step_distance)

Die Konstrukteure Ihres 3D-Druckers haben steps_per_mm ursprünglich aus einem Rotationsabstand berechnet. Wenn Sie die Steps_per_mm kennen, können Sie diese allgemeine Formel verwenden, um den ursprünglichen Rotationsabstand zu erhalten:

```
rotation_distance = <full_steps_per_rotation> * <microsteps> / <steps_per_mm>
```

Oder, wenn Sie eine ältere Klipper-Konfiguration haben und den Parameter step_distance kennen, können Sie diese Formel verwenden:

```
rotation_distance = <full_steps_per_rotation> * <microsteps> * <step_distance>
```

Die Einstellung <volle_Schritte_pro_Umdrehung> wird durch den Typ des Schrittmotors bestimmt. Die meisten Schrittmotoren sind "1,8-Grad-Schrittmotoren" und haben daher 200 Vollschritte pro Umdrehung (360 geteilt durch 1,8 ist 200). Einige Schrittmotoren sind "0,9-Grad-Schrittmotoren" und haben daher 400 Vollschritte pro Umdrehung. Andere Schrittmotoren sind selten. Wenn Sie unsicher sind, setzen Sie full_steps_per_rotation in der Konfigurationsdatei nicht ein und verwenden Sie 200 in der obigen Formel.

Die Einstellung <microsteps> wird durch den Schrittmotortreiber bestimmt. Die meisten Treiber verwenden 16 Mikroschritte. Wenn Sie unsicher sind, setzen Sie microsteps: 16 in der Konfiguration ein und setzen Sie 16 in die obige Formel ein.

Fast alle Drucker sollten eine ganze Zahl für `rotation_distance` auf den X-, Y- und Z-Achsen haben. Wenn die obige Formel einen Rotationsabstand ergibt, der innerhalb von 0,01 einer ganzen Zahl liegt, runden Sie den Endwert auf diese ganze Zahl.

Kalibrierung des Rotationsabstands bei Extrudern

Bei einem Extruder ist der Rotationsabstand die Strecke, die das Filament bei einer vollen Umdrehung des Schrittmotors zurücklegt. Der beste Weg, um einen genauen Wert für diese Einstellung zu erhalten, ist die Verwendung eines "Mess- und Trimmverfahrens".

Beginnen Sie zunächst mit einer ersten Schätzung für den Rotationsabstand. Dieser kann anhand von steps_per_mm oder durch eine inspecting the hardware ermittelt werden.

Dann verwenden Sie das folgende Verfahren zum "Messen und Trimmen":

1. Vergewissern Sie sich, dass der Extruder mit Filament gefüllt ist, das Hotend auf eine angemessene Temperatur aufgeheizt ist und der Drucker bereit zum Extrudieren ist.
2. Verwenden Sie einen Marker, um eine Markierung auf dem Filament etwa 70 mm vom Einlass des Extrudergehäuses entfernt anzubringen. Verwenden Sie dann einen digitalen Messschieber, um den tatsächlichen Abstand von dieser Markierung so genau wie möglich zu messen. Notieren Sie dies als `<Initial_mark_distance>.`
3. Extrudieren Sie 50 mm Filament mit der folgenden Befehlsfolge: `G91` gefolgt von `G1 E50 F60`. Notieren Sie 50 mm als `<requested_extrude_distance>.` Warten Sie, bis der Extruder die Bewegung beendet hat (dies dauert etwa 50 Sekunden). Es ist wichtig, die langsame Extrusionsrate für diesen Test zu verwenden, da eine schnellere Rate einen hohen Druck im Extruder verursachen kann, der die Ergebnisse verfälscht. (Verwenden Sie für diesen Test nicht die Schaltfläche "Extrudieren" auf grafischen Front-Ends, da diese mit einer hohen Geschwindigkeit extrudieren).
4. Verwenden Sie den digitalen Messschieber, um den neuen Abstand zwischen dem Extruderkörper und der Markierung auf dem Filament zu messen. Notieren Sie dies als `<subsequent_mark_distance>`. Berechnen Sie dann: `actual_extrude_distance = <initial_mark_distance> - <subsequent_mark_distance>`
5. Berechnen Sie den Rotationsabstand wie folgt: `rotation_distance = <previous_rotation_distance> * <actual_extrude_distance> / <requested_extrude_distance>` Runden Sie den neuen Rotationsabstand auf drei Dezimalstellen.

Wenn der tatsächliche_Extrudenabstand um mehr als 2 mm vom angeforderten_Extrudenabstand abweicht, ist es ratsam, die obigen Schritte ein zweites Mal durchzuführen.

Hinweis: Verwenden Sie zur Kalibrierung von x-, y- oder z-Achsen keine "Messen und Trimmen"-Methode. Die "Messen und Trimmen"-Methode ist für diese Achsen nicht genau genug und wird wahrscheinlich zu einer schlechteren Konfiguration führen. Stattdessen können diese Achsen bei Bedarf durch Messen der Riemen, Riemenscheiben und Leitspindeln bestimmt werden measuring the belts, pulleys, and lead screw hardware.

Ermittlung der Rotationsdistanz durch Inspektion der Hardware

Es ist möglich, den Rotationsabstand zu berechnen, wenn man die Schrittmotoren und die Kinematik des Druckers kennt. Dies kann nützlich sein, wenn die Schritte_pro_mm nicht bekannt sind oder wenn ein neuer Drucker entworfen wird.

Riemengetriebene Achsen

Es ist einfach, den Rotationsabstand für eine lineare Achse zu berechnen, die einen Riemen und eine Riemenscheibe verwendet.

Bestimmen Sie zunächst die Art des Riemens. Die meisten Drucker verwenden eine Riementeilung von 2 mm (d. h. jeder Zahn auf dem Riemen hat einen Abstand von 2 mm). Zählen Sie dann die Anzahl der Zähne auf der Schrittmotor-Riemenscheibe. Die Rotation_distance wird dann wie folgt berechnet:

`Rotation_distance = <Riemenabstand> * <Anzahl_der_Zähne_auf_der_Riemenscheibe>`

Wenn ein Drucker beispielsweise einen 2 mm breiten Riemen hat und eine Riemenscheibe mit 20 Zähnen verwendet, beträgt der Rotationsabstand 40.

Achsen mit einer Leitspindel

Der Rotationsabstand für gängige Gewindespindeln lässt sich leicht mit der folgenden Formel berechnen:

`rotation_distance = <Spindelsteigung> * <Anzahl_der_getrennten_Gewinde>`

Zum Beispiel hat die übliche "T8-Gewindespindel" einen Drehabstand von 8 (sie hat eine Steigung von 2 mm und 4 separate Gewinde).

Ältere Drucker mit "Gewindestangen" haben nur ein "Gewinde" auf der Gewindespindel, so dass der Drehabstand der Steigung der Spindel entspricht. (Die Spindelsteigung ist der Abstand zwischen den einzelnen Rillen der Spindel.) So hat zum Beispiel eine metrische M6-Stange einen Drehabstand von 1 und eine M8-Stange einen Drehabstand von 1,25.

Extruder

Der anfängliche Rotationsabstand für Extruder kann durch Messen des Durchmessers des "Wälzbolzens", der das Filament vorantreibt, und durch Anwendung der folgenden Formel ermittelt werden: rotation_distance = <Durchmesser> * 3.14

Wenn der Extruder über ein Getriebe verfügt, muss auch das Übersetzungsverhältnis des Extruders bestimmt und eingestellt werden determine and set the gear_ratio.

Die tatsächliche Rotationsdistanz eines Extruders variiert von Drucker zu Drucker, da der Griff des "Abwälzbolzens", der das Filament festhält, unterschiedlich sein kann. Er kann sogar zwischen verschiedenen Filamentspulen variieren. Nachdem Sie einen anfänglichen Rotationsabstand ermittelt haben, verwenden Sie das Mess- und Trimmverfahren measure and trim procedure , um eine genauere Einstellung zu erhalten.

Verwendung einer gear_ratio

Die Einstellung einer `gear_ratio` kann die Konfiguration der `rotation_distance` bei Steppern, die mit einem Getriebe (oder ähnlichem) ausgestattet sind, erleichtern. Die meisten Stepper haben kein Getriebe - wenn du dir nicht sicher bist, setze `gear_ratio` nicht in der Konfiguration.

Wenn `gear_ratio` gesetzt ist, repräsentiert die rotation_distance den Weg, den die Achse bei einer vollen Umdrehung des letzten Zahnrads am Getriebe zurücklegt. Wenn man z.B. ein Getriebe mit einer "5:1"-Übersetzung verwendet, könnte man die Rotation_distance mit Kenntnis der Hardware knowledge of the hardware berechnen und dann `gear_ratio: 5:1` zur config hinzufügen.

Bei Getrieben, die mit Riemen und Riemenscheiben realisiert sind, ist es möglich, die gear_ratio durch Zählen der Zähne auf den Riemenscheiben zu bestimmen. Wenn zum Beispiel ein Stepper mit einer Riemenscheibe mit 16 Zähnen die nächste Riemenscheibe mit 80 Zähnen antreibt, würde man `gear_ratio: 80:16` verwenden. In der Tat könnte man ein handelsübliches "Getriebe" öffnen und die Zähne zählen, um das Übersetzungsverhältnis zu ermitteln.

Beachten Sie, dass ein Getriebe manchmal ein etwas anderes Übersetzungsverhältnis hat als das, als das es beworben wird. Die üblichen BMG-Extrudermotor-Getriebe sind ein Beispiel dafür - sie werden als "3:1" beworben, haben aber tatsächlich ein "50:17"-Getriebe. (Die Verwendung von Zahnzahlen ohne gemeinsamen Nenner kann den Gesamtverschleiß des Getriebes erhöhen, da die Zähne nicht bei jeder Umdrehung auf die gleiche Weise ineinandergreifen.) Das übliche "5,18:1 Planetengetriebe" ist genauer mit `gear_ratio: 57:11` konfiguriert.

Wenn mehrere Zahnräder auf einer Achse verwendet werden, ist es möglich, gear_ratio mit einer kommagetrennten Liste zu versehen. Ein "5:1"-Getriebe, das eine Riemenscheibe mit 16 und 80 Zähnen antreibt, könnte zum Beispiel `gear_ratio: 5:1, 80:16` verwenden.

In den meisten Fällen sollte gear_ratio mit ganzen Zahlen definiert werden, da übliche Zahnräder und Riemenscheiben eine ganze Anzahl von Zähnen haben. In Fällen, in denen ein Riemen eine Riemenscheibe über Reibung statt über Zähne antreibt, kann es jedoch sinnvoll sein, eine Fließkommazahl für das Übersetzungsverhältnis zu verwenden (z. B. `gear_ratio: 107.237:16`).

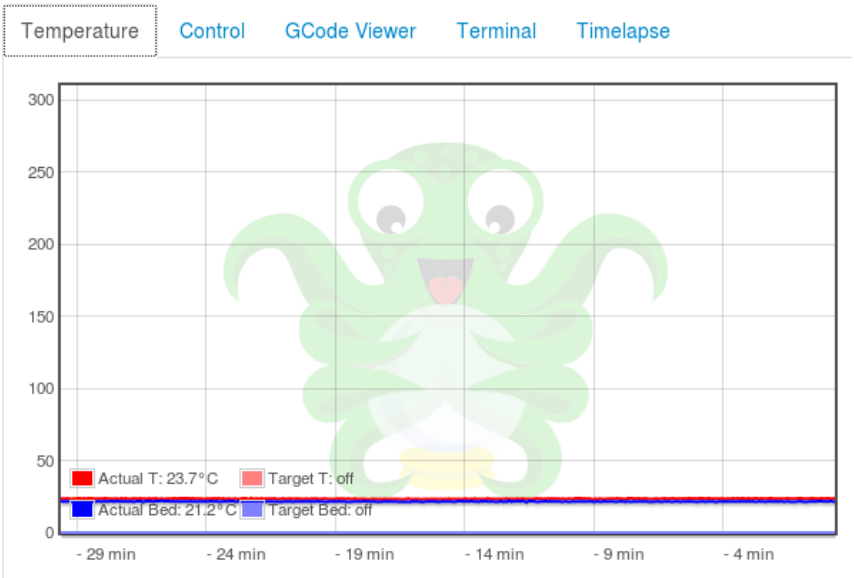
Konfigurationsprüfungen

Dieses Dokument enthält eine Liste von Schritten, die helfen sollen, die Pin-Einstellungen in der Datei Klipper printer.cfg zu überprüfen. Es ist eine gute Idee, diese Schritte durchzugehen, nachdem Sie die Schritte im Installationsdokument installation document befolgt haben.

Während dieser Anleitung kann es notwendig sein, Änderungen an der Klipper-Konfigurationsdatei vorzunehmen. Vergewissern Sie sich, dass Sie nach jeder Änderung der Konfigurationsdatei einen RESTART-Befehl eingeben, um sicherzustellen, dass die Änderung wirksam wird (geben Sie "restart" in die Registerkarte des Octoprint-Terminals ein und klicken Sie dann auf "Senden"). Es ist auch eine gute Idee, nach jedem RESTART einen STATUS-Befehl zu geben, um zu überprüfen, ob die Konfigurationsdatei erfolgreich geladen wurde.

Überprüfen der Temperatur

Überprüfen Sie zunächst, ob die Temperaturen korrekt gemeldet werden. Wechseln Sie auf die Registerkarte Octoprint-Temperatur.



Überprüfen Sie, ob die Temperatur der Düse und des Bettes (falls zutreffend) vorhanden ist und nicht ansteigt. Wenn sie ansteigt, schalten Sie den Drucker aus. Wenn die Temperaturen nicht korrekt sind, überprüfen Sie die Einstellungen "sensor_type" und "sensor_pin" für die Düse und/oder das Bett.

Überprüfen Sie M112

Navigieren Sie zur Octoprint-Terminal-Registerkarte und geben Sie einen M112-Befehl in das Terminalfeld ein. Dieser Befehl fordert Klipper auf, in einen "Shutdown"-Zustand zu gehen. Dies führt dazu, dass Octoprint die Verbindung zu Klipper trennt - navigieren Sie zum Bereich "Verbindung" und klicken Sie auf "Verbinden", damit Octoprint die Verbindung wieder aufnimmt. Navigieren Sie dann zur Registerkarte "Octoprint-Temperatur" und überprüfen Sie, ob die Temperaturen weiterhin aktualisiert werden und nicht ansteigen. Wenn die Temperaturen ansteigen, schalten Sie den Drucker aus.

Der M112-Befehl veranlasst Klipper, in einen "Shutdown"-Status zu gehen. Um diesen Zustand zu löschen, geben Sie den Befehl FIRMWARE_RESTART auf der Registerkarte Octoprint Terminal ein.

Heizungen überprüfen

Navigieren Sie zur Registerkarte Octoprint-Temperatur und geben Sie im Feld "Tool"-Temperatur 50 ein, gefolgt von der Eingabetaste. Die Extrudertemperatur in der Grafik sollte nun ansteigen (innerhalb von ca. 30 Sekunden). Gehen Sie dann zum Dropdown-Feld "Werkzeugtemperatur" und wählen Sie "Aus". Nach einigen Minuten sollte die Temperatur wieder auf den ursprünglichen Wert der Raumtemperatur zurückgehen. Wenn die Temperatur nicht ansteigt, überprüfen Sie die "heater_pin"-Einstellung in der Konfiguration.

Wenn der Drucker über ein beheiztes Bett verfügt, führen Sie den obigen Test erneut mit dem Bett durch.

Prüfen Sie die Freigabe der Schrittmotor-Pins

Überprüfen Sie, ob sich alle Achsen des Druckers manuell frei bewegen können (die Schrittmotoren sind deaktiviert). Ist dies nicht der Fall, geben Sie einen M84-Befehl zum Deaktivieren der Motoren aus. Wenn sich eine der Achsen immer noch nicht frei bewegen kann, überprüfen Sie die "enable_pin"-Konfiguration des Schrittmotors für die jeweilige Achse. Bei den meisten handelsüblichen Schrittmotortreibern ist der Motorfreigabe-Pin "active low" und daher sollte dem Freigabe-Pin ein "!" vorangestellt werden (zum Beispiel "enable_pin: !ar38").

Überprüfen der Endstops
Bewegen Sie alle Achsen des Druckers manuell, so dass keine von ihnen mit einem Endanschlag in Berührung kommt. Senden Sie einen QUERY_ENDSTOPS-Befehl über das Octoprint-Terminalregister. In der Antwort sollte der aktuelle Zustand aller konfigurierten Endstopps angegeben werden und alle sollten den Zustand "offen" melden. Führen Sie für jeden der Endstops den Befehl QUERY_ENDSTOPS erneut aus und lösen Sie dabei den Endstop manuell aus. Der QUERY_ENDSTOPS-Befehl sollte den Endstopp als "TRIGGERED" melden.

Wenn die Endsperre invertiert erscheint (sie meldet "offen", wenn sie ausgelöst wird, und umgekehrt), fügen Sie ein "!" zur Pin-Definition hinzu (z. B. "endstop_pin: ^!ar3"), oder entfernen Sie das "!", wenn bereits eines vorhanden ist.

Wenn sich der Endstopp überhaupt nicht ändert, bedeutet dies im Allgemeinen, dass der Endstopp mit einem anderen Pin verbunden ist. Es kann jedoch auch eine Änderung der Pullup-Einstellung des Pins erforderlich sein (das '^' am Anfang des endstop_pin-Namens - die meisten Drucker verwenden einen Pullup-Widerstand und das '^' sollte vorhanden sein).

Überprüfen der Schrittmotoren

Verwenden Sie den Befehl STEPPER_BUZZ, um die Konnektivität der einzelnen Schrittmotoren zu überprüfen. Beginnen Sie mit der manuellen Positionierung der gegebenen Achse auf einen mittleren Punkt und führen Sie dann STEPPER_BUZZ STEPPER=stepper_x aus. Der Befehl STEPPER_BUZZ veranlasst den angegebenen Schrittmotor, sich einen Millimeter in positiver Richtung zu bewegen und dann in seine Ausgangsposition zurückzukehren. (Wenn der Endanschlag mit position_endstop=0 definiert ist, bewegt sich der Stepper zu Beginn jeder Bewegung vom Endanschlag weg). Diese Oszillation wird zehnmal durchgeführt.

Wenn sich der Schrittmotor überhaupt nicht bewegt, überprüfen Sie die Einstellungen "enable_pin" und "step_pin" für den Schrittmotor. Wenn sich der Schrittmotor bewegt, aber nicht in seine ursprüngliche Position zurückkehrt, überprüfen Sie die Einstellung "dir_pin". Wenn der Schrittmotor in eine falsche Richtung schwingt, deutet dies im Allgemeinen darauf hin, dass der "dir_pin" für die Achse invertiert werden muss. Fügen Sie dazu in der Druckerkonfigurationsdatei ein '!' an den "dir_pin" an (oder entfernen Sie es, falls bereits eines vorhanden ist). Wenn sich der Motor deutlich mehr oder weniger als einen Millimeter bewegt, überprüfen Sie die Einstellung "rotation_distance".

Führen Sie den obigen Test für jeden in der Konfigurationsdatei definierten Schrittmotor durch. (Setzen Sie den STEPPER-Parameter des STEPPER_BUZZ-Befehls auf den Namen des Konfigurationsabschnitts, der getestet werden soll). Wenn sich kein Filament im Extruder befindet, kann man STEPPER_BUZZ verwenden, um die Konnektivität des Extrudermotors zu überprüfen (STEPPER=extruder verwenden). Andernfalls ist es am besten, den Extrudermotor separat zu testen (siehe nächster Abschnitt).

Nach der Überprüfung aller Endanschläge und der Überprüfung aller Schrittmotoren sollte der Referenzfahrtmechanismus getestet werden. Geben Sie einen G28-Befehl aus, um alle Achsen zu referenzieren. Trennen Sie den Drucker von der Stromversorgung, wenn er nicht ordnungsgemäß referenziert. Wiederholen Sie die Schritte zur Überprüfung der Endanschläge und Schrittmotoren, falls erforderlich.

Prüfen des Extrudermotors

Um den Extrudermotor zu testen, müssen Sie den Extruder auf Drucktemperatur aufheizen. Navigieren Sie zur Registerkarte Octoprint-Temperatur und wählen Sie eine Zieltemperatur aus dem Temperatur-Dropdown-Feld aus (oder geben Sie manuell eine geeignete Temperatur ein). Warten Sie, bis der Drucker die gewünschte Temperatur erreicht hat. Wechseln Sie dann zur Registerkarte "Octoprint-Steuerung" und klicken Sie auf die Schaltfläche "Extrudieren". Überprüfen Sie, ob sich der Extrudermotor in die richtige Richtung dreht. Wenn dies nicht der Fall ist, lesen Sie die Tipps zur Fehlerbehebung im vorherigen Abschnitt, um die Einstellungen "enable_pin", "step_pin" und "dir_pin" für den Extruder zu überprüfen.

PID-Einstellungen kalibrieren

Klipper unterstützt die PID-Regelung für den Extruder und die Bettheizungen. Um diesen Regelungsmechanismus zu nutzen, ist es notwendig, die PID-Einstellungen auf jedem Drucker zu kalibrieren (PID-Einstellungen, die in anderen Programmen oder in den Beispielkonfigurationsdateien gefunden wurden, funktionieren oft schlecht).

Um den Extruder zu kalibrieren, navigieren Sie zur Registerkarte OctoPrint-Terminal und führen Sie den Befehl PID_CALIBRATE aus. Zum Beispiel:
`PID_CALIBRATE HEATER=extruder TARGET=170`

Nach Abschluss des Einstellungstests führen Sie `SAVE_CONFIG` aus, um die Datei printer.cfg mit den neuen PID-Einstellungen zu aktualisieren.

Wenn der Drucker über ein beheiztes Bett verfügt und dieses mit PWM (Pulsweitenmodulation) angesteuert werden kann, wird empfohlen, eine PID-Regelung für das Bett zu verwenden (wenn die Bettheizung mit dem PID-Algorithmus gesteuert wird, kann sie zehnmal pro Sekunde ein- und ausgeschaltet werden, was für Heizungen mit einem mechanischen Schalter möglicherweise nicht geeignet ist). Ein typischer PID-Kalibrierungsbefehl für das Bett lautet: `PID_CALIBRATE HEATER=heater_bed TARGET=60`

Nächste Schritte

Diese Anleitung soll bei der grundlegenden Überprüfung der Pin-Einstellungen in der Klipper-Konfigurationsdatei helfen. Lesen Sie unbedingt die Anleitung zum Nivellieren des Bettes bed leveling. Siehe auch das Dokument Slicers für Informationen zur Konfiguration eines Slicers mit Klipper.

Nachdem Sie überprüft haben, dass der Grunddruck funktioniert, sollten Sie die Kalibrierung des Druckvorschubs in Betracht ziehen pressure advance.

Es kann notwendig sein, andere Arten von detaillierter Druckerkalibrierung durchzuführen - eine Reihe von Anleitungen sind online verfügbar, um dabei zu helfen (z.B. eine Websuche nach "3d printer calibration"). Wenn Sie z. B. den Effekt des Klingelns feststellen, können Sie die folgende Anleitung zur Resonanzkompensation resonance compensation ausprobieren.

Bettnivellierung

Die Nivellierung des Druckbetts (manchmal auch als "Bettnivellierung" bezeichnet) ist entscheidend für qualitativ hochwertige Drucke. Wenn ein Bett nicht richtig "nivelliert" ist, kann dies zu schlechter Haftung des Bettes, "Verziehen" und subtilen Problemen im gesamten Druck führen. Dieses Dokument dient als Leitfaden für die Durchführung der Bettnivellierung in Klipper.

Es ist wichtig, das Ziel der Bettnivellierung zu verstehen. Wenn der Drucker während eines Druckvorgangs auf eine Position `X0 Y0 Z10` befohlen wird, dann ist das Ziel, dass die Düse des Druckers genau 10 mm vom Druckbett entfernt ist. Wird der Drucker dann in eine Position `X50 Z10` befohlen, soll die Düse während der gesamten horizontalen Bewegung einen exakten Abstand von 10 mm zum Bett einhalten.

Um qualitativ hochwertige Drucke zu erhalten, sollte der Drucker so kalibriert werden, dass die Z-Abstände mit einer Genauigkeit von etwa 25 Mikrometern (0,025 mm) eingehalten werden. Dies ist ein kleiner Abstand - deutlich kleiner als die Breite eines typischen menschlichen Haares. Dieser Maßstab kann nicht "mit dem Auge" gemessen werden. Subtile Effekte (wie z. B. Wärmeausdehnung) beeinflussen die Messungen in diesem Bereich. Das Geheimnis für eine hohe Genauigkeit ist ein wiederholbarer Prozess und eine Nivellierungsmethode, die die hohe Genauigkeit des druckereigenen Bewegungssystems ausnutzt.

Wählen Sie den geeigneten Kalibrierungsmechanismus

Die verschiedenen Druckertypen verwenden unterschiedliche Methoden zur Nivellierung des Druckbetts. Sie alle hängen letztlich vom "Papiertest" ab (siehe unten). Der eigentliche Prozess für einen bestimmten Druckertyp wird jedoch in anderen Dokumenten beschrieben.

Bevor Sie eines dieser Kalibrierungswerkzeuge ausführen, sollten Sie die im Dokument config check document beschriebenen Prüfungen durchführen. Vor der Durchführung der Bettnivellierung muss die grundlegende Bewegung des Druckers überprüft werden.

Bei Druckern mit einer "automatischen Z-Sonde" müssen Sie die Sonde gemäß den Anweisungen im Dokument Probe Calibrate kalibrieren. Für Delta-Drucker siehe das Dokument Delta Calibrate (Deltakalibrierung). Für Drucker mit Bettschrauben und herkömmlichen Z-Anschlägen siehe das Dokument Manual Level.

Während der Kalibrierung kann es erforderlich sein, die Z-Position_min des Druckers auf eine negative Zahl zu setzen (z. B. position_min = -2). Der Drucker führt auch während der Kalibrierungsroutinen Grenzwertprüfungen durch. Die Einstellung einer negativen Zahl ermöglicht es dem Drucker, sich unter die Nennposition des Bettes zu bewegen, was bei dem Versuch, die tatsächliche Bettposition zu bestimmen, hilfreich sein kann.

Der "Papiertest"

Der wichtigste Mechanismus zur Bettkalibrierung ist der "Papiertest". Dabei wird ein normales Stück "Kopierpapier" zwischen Bett und Düse des Druckers gelegt und die Düse auf verschiedene Z-Höhen eingestellt, bis man beim Hin- und Herschieben des Papiers eine geringe Reibung spürt.

Es ist wichtig, den "Papiertest" zu verstehen, auch wenn man eine "automatische Z-Sonde" hat. Die Sonde selbst muss oft kalibriert werden, um gute Ergebnisse zu erzielen. Die Kalibrierung der Sonde erfolgt mit Hilfe dieses "Papiertests".

Um den Papiertest durchzuführen, schneiden Sie mit einer Schere ein kleines rechteckiges Stück Papier aus (z. B. 5x3 cm). Das Papier hat im Allgemeinen eine Dicke von etwa 100 Mikron (0,100 mm). (Die genaue Dicke des Papiers ist nicht entscheidend.)

Der erste Schritt des Papiertests besteht darin, die Düse und das Bett des Druckers zu überprüfen. Vergewissern Sie sich, dass sich kein Plastik (oder andere Verunreinigungen) auf der Düse oder dem Bett befindet.

Untersuchen Sie die Düse und das Bett, um sicherzustellen, dass kein Plastik vorhanden ist!

Wenn Sie immer auf einem bestimmten Klebeband oder einer bestimmten Druckoberfläche drucken, können Sie den Papiertest mit diesem Klebeband bzw. dieser Oberfläche durchführen. Beachten Sie jedoch, dass das Klebeband selbst eine bestimmte Dicke hat und unterschiedliche Klebebänder (oder andere Druckoberflächen) die Z-Messungen beeinflussen. Führen Sie den Papiertest unbedingt erneut durch, um jede Art von Oberfläche zu messen.

Wenn sich Plastik auf der Düse befindet, erhitzen Sie den Extruder und verwenden Sie eine Metallpinzette, um das Plastik zu entfernen. Warten Sie, bis der Extruder vollständig auf Raumtemperatur abgekühlt ist, bevor Sie mit dem Papiertest fortfahren. Während die Düse abkühlt, verwenden Sie die Metallpinzette, um eventuell austretendes Plastik zu entfernen.

Führen Sie den Papiertest immer durch, wenn sowohl die Düse als auch das Bett Raumtemperatur haben!

Wenn die Düse erhitzt wird, verändert sich ihre Position (relativ zum Bett) aufgrund der Wärmeausdehnung. Diese thermische Ausdehnung beträgt in der Regel etwa 100 Mikrometer, was in etwa der Dicke eines typischen Stücks Druckerpapier entspricht. Der genaue Betrag der thermischen Ausdehnung ist nicht entscheidend, ebenso wenig wie die genaue Dicke des Papiers. Gehen Sie von der Annahme aus, dass beide gleich groß sind (eine Methode zur Bestimmung der Differenz zwischen den beiden Abständen finden Sie weiter unten).

Es mag seltsam erscheinen, den Abstand bei Raumtemperatur zu kalibrieren, wenn das Ziel darin besteht, bei Erwärmung einen gleichmäßigen Abstand zu erzielen. Wenn man jedoch kalibriert, wenn die Düse erwärmt ist, neigt sie dazu, kleine Mengen geschmolzenen Kunststoffs auf das Papier zu übertragen, was die gefühlte Reibung verändert. Das macht es schwieriger, eine gute Kalibrierung zu erhalten. Wenn Sie kalibrieren, während das Bett/die Düse heiß ist, erhöht sich auch das Risiko, sich zu verbrennen. Der Betrag der thermischen Ausdehnung ist stabil, so dass er später im Kalibrierungsprozess leicht berücksichtigt werden kann.

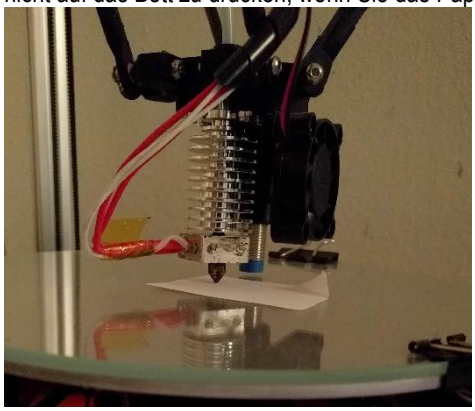
Verwenden Sie ein automatisiertes Werkzeug, um präzise Z-Höhen zu bestimmen!

Klipper verfügt über mehrere Hilfsskripte (z.B. MANUAL_PROBE, Z_ENDSTOP_CALIBRATE, PROBE_CALIBRATE, DELTA_CALIBRATE). Siehe die oben beschriebenen Dokumente described above , um eines davon auszuwählen.

Führen Sie den entsprechenden Befehl im OctoPrint-Terminalfenster aus. Das Skript fordert den Benutzer zur Interaktion in der OctoPrint-Terminalausgabe auf. Sie wird etwa so aussehen:

```
Recv: // Starting manual Z probe. Use TESTZ to adjust position.
Recv: // Finish with ACCEPT or ABORT command.
Recv: // Z position: ?????? --> 5.000 <-- ??????
```

Die aktuelle Höhe der Düse (so wie der Drucker sie derzeit versteht) wird zwischen den Zeichen "--> <--" angezeigt. Die Zahl rechts ist die Höhe des letzten Antastversuchs, der gerade größer als die aktuelle Höhe war, und links ist der letzte Antastversuch, der kleiner als die aktuelle Höhe war (oder ??????, wenn kein Antastversuch stattgefunden hat).
Legen Sie das Papier zwischen Düse und Bett. Es kann sinnvoll sein, eine Ecke des Papiers zu falten, damit es leichter zu greifen ist. (Versuchen Sie, nicht auf das Bett zu drücken, wenn Sie das Papier hin und her bewegen).



papier-test

Verwenden Sie den Befehl TESTZ, um die Düse aufzufordern, sich dem Papier zu nähern. Zum Beispiel:

```
TESTZ Z=-.1
```

Mit dem Befehl TESTZ wird die Düse um eine relative Entfernung von der aktuellen Position der Düse bewegt. (Z=-.1 bedeutet also, dass die Düse um 0,1 mm näher an das Papierbett heranfahren soll.) Nachdem die Düse angehalten hat, schieben Sie das Papier hin und her, um zu prüfen, ob die Düse das Papier berührt und um die Reibung zu spüren. Geben Sie so lange TESTZ-Befehle ein, bis Sie beim Testen mit dem Papier eine geringe Reibung spüren.

Wenn zu viel Reibung festgestellt wird, kann man einen positiven Z-Wert verwenden, um die Düse nach oben zu bewegen. Es ist auch möglich, TESTZ `Z=+` oder `TESTZ Z=-` zu verwenden, um die letzte Position zu "halbieren", d. h. eine Position auf halbem Weg zwischen zwei Positionen anzufahren. Wenn man zum Beispiel die folgende Eingabeaufforderung von einem TESTZ-Befehl erhält:

```
Recv: // Z-Position: 0.130 --> 0.230 <-- 0.280
```

Dann würde ein TESTZ Z=- die Düse auf eine Z-Position von 0,180 (auf halbem Weg zwischen 0,130 und 0,230) bewegen. Mit dieser Funktion kann man schnell eine konsistente Reibung finden. Es ist auch möglich, mit Z=++ und Z=-- direkt zu einer früheren Messung zurückzukehren - zum Beispiel würde nach der obigen Aufforderung ein TESTZ-Befehl Z=-- die Düse in eine Z-Position von 0,130 bringen.

Nachdem eine geringe Reibung festgestellt wurde, führen Sie den Befehl ACCEPT aus: `ACCEPT`

Damit wird die angegebene Z-Höhe akzeptiert und mit dem angegebenen Kalibrierungswerkzeug fortgefahren.

Das genaue Ausmaß der Reibung ist nicht entscheidend, ebenso wenig wie das Ausmaß der thermischen Ausdehnung und die genaue Breite des Papiers. Versuchen Sie einfach, jedes Mal, wenn Sie den Test durchführen, den gleichen Reibungswert zu erhalten.

Wenn während des Tests etwas schief geht, kann man das Kalibrierungswerkzeug mit dem Befehl ABORT verlassen.

Bestimmung der thermischen Ausdehnung

Nach erfolgreicher Nivellierung des Bettes kann man einen genaueren Wert für die kombinierte Auswirkung von "thermischer Ausdehnung", "Dicke des Papiers" und "gefühlter Reibung während des Papiertests" berechnen.

Diese Art der Berechnung ist in der Regel nicht erforderlich, da die meisten Benutzer der Meinung sind, dass der einfache "Papiertest" gute Ergebnisse liefert.

Am einfachsten lässt sich diese Berechnung durchführen, indem man ein Testobjekt ausdruckt, das an allen Seiten gerade Wände hat. Das große hohle Quadrat in docs/prints/square.stl kann hierfür verwendet werden. Stellen Sie beim Schneiden des Objekts sicher, dass der Slicer für die erste Ebene dieselbe Schichthöhe und Extrusionsbreite verwendet wie für alle nachfolgenden Schichten. Verwenden Sie eine grobe Schichthöhe (die Schichthöhe sollte etwa 75 % des Düsendurchmessers betragen) und verwenden Sie kein Brim oder ein Raft.

Drucken Sie das Testobjekt, warten Sie, bis es abgekühlt ist, und nehmen Sie es vom dem Bett. Prüfen Sie die unterste Schicht des Objekts. (Es kann auch hilfreich sein, mit einem Finger oder Nagel an der Unterkante entlangzufahren.) Wenn sich die unterste Schicht an allen Seiten des Objekts leicht auswölbt, deutet dies darauf hin, dass die Düse etwas näher am Bett war, als sie sein sollte. Mit dem Befehl SET_GCODE_OFFSET Z=+.010 kann man die Höhe erhöhen. Bei nachfolgenden Drucken kann man dieses Verhalten überprüfen und bei Bedarf weitere Anpassungen vornehmen. Anpassungen dieser Art erfolgen in der Regel in 10er-Mikrometern (.010mm).

Wenn die unterste Schicht durchgängig schmaler erscheint als die nachfolgenden Schichten, kann man mit dem Befehl SET_GCODE_OFFSET eine negative Z-Einstellung vornehmen. Wenn man sich nicht sicher ist, kann man die Z-Anpassung verringern, bis die unterste Ebene der Drucke eine kleine Ausbuchtung aufweist, und dann zurückgehen, bis sie verschwindet.

Der einfachste Weg, die gewünschte Z-Anpassung vorzunehmen, besteht darin, ein g-code-Makro START_PRINT zu erstellen, dieses Makro zu Beginn jedes Druckvorgangs vom Slicer aufrufen zu lassen und diesem Makro den Befehl SET_GCODE_OFFSET hinzuzufügen. Weitere Einzelheiten finden Sie im Slicer-Dokument.

Taster-Kalibrierung

Dieses Dokument beschreibt die Methode zur Kalibrierung der X-, Y- und Z-Offsets eines "automatischen z-Tasters" in Klipper. Dies ist nützlich für Benutzer, die einen [probe] oder [bltouch] Abschnitt in ihrer Konfigurationsdatei haben.

Kalibrierung der X- und Y-Offsets der Sonde

Um den X- und Y-Offset zu kalibrieren, wechseln Sie zur OctoPrint-Registerkarte "Steuerung", fahren Sie den Drucker in die Ausgangsposition und verwenden Sie dann die OctoPrint-Tasten, um den Kopf in eine Position nahe der Mitte des Druckbetts zu bringen.

Legen Sie ein Stück blaues Malerband (oder ähnliches) auf das Bett unter der Sonde. Navigieren Sie zur OctoPrint-Registerkarte "Terminal" und geben Sie einen PROBE-Befehl ein:

```
PROBE
```

Platzieren Sie eine Markierung auf dem Klebeband direkt unter der Sonde (oder verwenden Sie eine ähnliche Methode, um die Position auf dem Bett zu notieren).

Geben Sie einen `GET_POSITION` Befehl aus und notieren Sie die von diesem Befehl gemeldete XY-Position des Werkzeugkopfs. Zum Beispiel, wenn man sieht:

```
Recv: // toolhead: X:46.500000 Y:27.000000 Z:15.000000 E:0.000000
```

dann würde man eine X-Position des Messtasters von 46,5 und eine Y-Position des Messtasters von 27 aufzeichnen.

Nach der Aufzeichnung der Tasterposition geben Sie eine Reihe von G1-Befehlen aus, bis sich die Düse direkt über der Markierung auf dem Bett befindet. Zum Beispiel könnte man eingeben:

```
G1 F300 X57 Y30 Z15
```

um die Düse auf eine X-Position von 57 und eine Y-Position von 30 zu bewegen. Sobald man die Position direkt über der Markierung gefunden hat, verwendet man den Befehl GET_POSITION, um diese Position zu melden. Dies ist die Position der Düse.

Der x_offset ist dann die `nozzle_x_position - probe_x_position` und der y_offset ist in gleicher Weise die `nozzle_y_position - probe_y_position`. Aktualisieren Sie die Datei printer.cfg mit den angegebenen Werten, entfernen Sie das Band/die Markierungen vom Bett und geben Sie dann den Befehl `RESTART`, damit die neuen Werte wirksam werden.

Kalibrierung des Sonden-Z-Offsets

Die Bereitstellung eines genauen Sonden-Z-Offsets ist entscheidend für die Erzielung hochwertiger Ausdrucke. Der z_offset ist der Abstand zwischen der Düse und dem Bett, wenn die Sonde auslöst. Das Klipper `PROBE_CALIBRATE` Werkzeug kann verwendet werden, um diesen Wert zu erhalten - es lässt einen automatischen Taster laufen, um die Z-Auslöseposition des Tasters zu messen und startet dann einen manuellen Taster, um die Z-Höhe der Düse zu erhalten. Der Taster z_offset wird dann aus diesen Messungen berechnet.

Beginnen Sie mit der Referenzierung des Druckers und bringen Sie den Kopf in eine Position nahe der Mitte des Bettes. Navigieren Sie zur Registerkarte OctoPrint-Terminal und führen Sie den Befehl `PROBE_CALIBRATE` aus, um das Werkzeug zu starten.

Dieses Werkzeug führt eine automatische Abtastung durch, hebt dann den Kopf an, bewegt die Düse über die Position des Abtastpunkts und startet das manuelle Abtastwerkzeug. Wenn sich die Düse nicht über den automatischen Messpunkt bewegt, brechen Sie den manuellen Messvorgang ab `ABORT` und führen Sie die oben beschriebene Kalibrierung des XY-Messkopf-Offsets durch.

Sobald das Handsondenwerkzeug startet, führen Sie die unter "the paper test" beschriebenen Schritte durch, um den tatsächlichen Abstand zwischen Düse und Bett an der gegebenen Stelle zu bestimmen. Sobald diese Schritte abgeschlossen sind, kann man die Position AKZEPTIEREN `ACCEPT` und die Ergebnisse in der Konfigurationsdatei speichern mit:

```
SAVE_CONFIG
```

Beachten Sie, dass die Ergebnisse von PROBE_CALIBRATE ungültig werden, wenn das Bewegungssystem des Druckers, die Position des Hotends oder die Position der Sonde geändert wird.

Wenn die Sonde einen X- oder Y-Versatz hat und die Bettneigung geändert wird (z. B. durch Verstellen der Bettschrauben, Ausführen von DELTA_CALIBRATE, Ausführen von Z_TILT_ADJUST, Ausführen von QUAD_GANTRY_LEVEL o. ä.), werden die Ergebnisse von PROBE_CALIBRATE ungültig. Nach Durchführung einer der oben genannten Anpassungen muss PROBE_CALIBRATE erneut ausgeführt werden.

Wenn die Ergebnisse von PROBE_CALIBRATE ungültig sind, sind auch alle früheren Bettnetz-Ergebnisse bed mesh, die mit der Sonde erzielt wurden, ungültig - es ist dann erforderlich, BED_MESH_CALIBRATE nach der Neukalibrierung der Sonde erneut auszuführen.

Überprüfung der Wiederholbarkeit

Nach der Kalibrierung der X-, Y- und Z-Offsets der Sonde ist es ratsam, zu überprüfen, ob die Sonde wiederholbare Ergebnisse liefert. Beginnen Sie mit der Referenzfahrt des Druckers und bringen Sie dann den Kopf in eine Position nahe der Mitte des Bettes. Wechseln Sie zur Registerkarte OctoPrint-Terminal und führen Sie den Befehl `PROBE_ACCURACY` aus.

Dieser Befehl lässt die Sonde zehnmal laufen und erzeugt eine Ausgabe ähnlich der folgenden:

```
Recv: // Sondengenauigkeit: bei X:0.000 Y:0.000 Z:10.000
Recv: // und 10 Mal mit einer Geschwindigkeit von 5 mm/s ablesen
Recv: // Tastkopf bei -0,003,0,005 ist z=2,506948
Recv: // Sonde bei -0,003,0,005 ist z=2,519448
Empfangen: // Sonde bei -0,003,0,005 ist z=2,519448
Empfangen: // Sonde bei -0,003,0,005 ist z=2,506948
```

```
Empfangen: // Sonde bei -0,003,0,005 ist z=2,519448
Empfangen: // Sonde bei -0,003,0,005 ist z=2,519448
Empfangen: // Sonde bei -0,003,0,005 ist z=2,506948
Empfangen: // Sonde bei -0,003,0,005 ist z=2,506948
Empfangen: // Sonde bei -0,003,0,005 ist z=2,519448
Empfangen: // Sonde bei -0,003,0,005 ist z=2,506948
Empfangen: // Ergebnisse der Sondengenauigkeit: Maximum 2,519448, Minimum 2,506948, Bereich
0,012500, Durchschnitt 2,513198, Median 2,513198, Standardabweichung 0,006250
```

Im Idealfall meldet das Gerät einen identischen Höchst- und Mindestwert. (Das heißt, dass der Messtaster im Idealfall bei allen zehn Messtastern ein identisches Ergebnis liefert.) Es ist jedoch normal, dass die Minimal- und Maximalwerte um einen Z-"Schrittabstand" oder bis zu 5 Mikrometer (.005 mm) voneinander abweichen. Ein "Schrittabstand" ist `rotation_distance/(full_steps_per_rotation*microsteps)`. Der Abstand zwischen dem Mindest- und dem Höchstwert wird als Bereich bezeichnet. Da der Drucker im obigen Beispiel einen Z-Schritt-Abstand von 0,0125 verwendet, würde ein Bereich von 0,012500 als normal angesehen werden.

Wenn die Testergebnisse einen Bereich von mehr als 25 Mikrometer (0,025 mm) ergeben, ist die Genauigkeit der Sonde für typische Bettnivellierverfahren nicht ausreichend. Es kann möglich sein, die Sondengeschwindigkeit und/oder die Sondenstarthöhe einzustellen, um die Wiederholbarkeit der Sonde zu verbessern. Mit dem Befehl `PROBE_ACCURACY` können Sie Tests mit verschiedenen Parametern durchführen, um deren Auswirkungen zu ermitteln - weitere Einzelheiten finden Sie im Dokument G-Codes document. Wenn die Sonde im Allgemeinen wiederholbare Ergebnisse liefert, aber gelegentlich einen Ausreißer aufweist, dann kann es möglich sein, dies durch die Verwendung mehrerer Proben an jeder Sonde zu berücksichtigen - lesen Sie die Beschreibung der Konfigurationsparameter der Sondenproben in der Konfigurationsreferenz für weitere Details.

Wenn neue Sondengeschwindigkeit, Probenanzahl oder andere Einstellungen erforderlich sind, aktualisieren Sie die Datei printer.cfg und geben Sie den Befehl RESTART. In diesem Fall ist es ratsam, den z_offset erneut zu kalibrieren. Wenn keine wiederholbaren Ergebnisse erzielt werden können, sollten Sie die Sonde nicht für die Bettnivellierung verwenden. Klipper verfügt über mehrere manuelle Sonden, die stattdessen verwendet werden können - siehe das Dokument Bed Level für weitere Details.

Prüfung der Lageabhängigkeit

Einige Messtaster können eine systembedingte Abweichungen aufweisen, die die Ergebnisse des Messtasters an bestimmten Werkzeugkopfpositionen verfälscht. Wenn zum Beispiel die Sondenhalterung bei der Bewegung entlang der Y-Achse leicht geneigt ist, kann dies dazu führen, dass die Sonde an verschiedenen Y-Positionen ungenaue Ergebnisse anzeigt.

Dies ist ein häufiges Problem bei Messtastern auf Delta-Druckern, kann aber bei allen Druckern auftreten.

Mit dem Befehl PROBE_CALIBRATE können Sie den z_offset der Sonde an verschiedenen X- und Y-Positionen messen, um zu prüfen, ob eine Abweichung vorliegt. Im Idealfall wäre der Sonden-Z_Offset an jeder Druckerposition ein konstanter Wert.

Versuchen Sie bei Deltadruckern, den z_offset an einer Position in der Nähe des A-Turms, an einer Position in der Nähe des B-Turms und an einer Position in der Nähe des C-Turms zu messen. Bei kartesischen, Corexy- und ähnlichen Druckern versuchen Sie, den z_offset an Positionen in der Nähe der vier Ecken des Bettes zu messen.

Bevor Sie mit diesem Test beginnen, kalibrieren Sie zunächst die X-, Y- und Z-Offsets der Sonde wie am Anfang dieses Dokuments beschrieben. Fahren Sie dann den Drucker hoch und navigieren Sie zur ersten XY-Position. Folgen Sie den Schritten bei der Kalibrierung des Z-Offsets der Sonde calibrating probe Z offset , um den Befehl `PROBE_CALIBRATE`, die Befehle `TESTZ` und `ACCEPT` auszuführen, aber führen Sie nicht `SAVE_CONFIG` aus. Notieren Sie sich den gefundenen z_offset. Navigieren Sie dann zu den anderen XY-Positionen, wiederholen Sie diese `PROBE_CALIBRATE`-Schritte und notieren Sie den gemeldeten z_offset.

Wenn die Differenz zwischen dem minimalen gemeldeten z_offset und dem maximalen gemeldeten z_offset größer als 25 Mikrometer (.025mm) ist, ist die Sonde nicht für typische Bettnivellierverfahren geeignet. Siehe das Dokument Bettniveau Bed Level document für manuelle Sondenalternativen.

Temperaturabweichung

Viele Sonden haben eine systembedingte Abweichungen, wenn sie bei unterschiedlichen Temperaturen messen. Zum Beispiel kann die Sonde bei einer höheren Temperatur durchweg auf einer niedrigeren Höhe auslösen.

Es wird empfohlen, die Bettnivelliergeräte bei einer konstanten Temperatur zu betreiben, um diese Abweichung auszugleichen. Lassen Sie die Werkzeuge zum Beispiel immer laufen, wenn der Drucker Raumtemperatur hat, oder lassen Sie die Werkzeuge immer laufen, nachdem der Drucker eine konstante Drucktemperatur erreicht hat. In beiden Fällen ist es ratsam, einige Minuten zu warten, nachdem die gewünschte Temperatur erreicht wurde, damit der Drucker die gewünschte Temperatur konstant hält.

Um zu prüfen, ob eine Temperaturabweichung vorliegt, starten Sie den Drucker bei Raumtemperatur, fahren Sie ihn dann in die Ausgangsposition zurück, bringen Sie den Druckkopf in eine Position nahe der Mitte des Druckbetts und führen Sie den Befehl `PROBE_ACCURACY` aus. Notieren Sie die Ergebnisse. Heizen Sie dann die Düse und das Bett des Druckers auf Drucktemperatur auf und führen Sie den `PROBE_ACCURACY`-Befehl erneut aus, ohne die Referenzfahrt durchzuführen oder die Schrittmotoren zu deaktivieren. Im Idealfall meldet der Befehl identische Ergebnisse. Wenn die Sonde eine Temperaturabweichung aufweist, achten Sie darauf, dass Sie die Sonde immer bei einer konstanten Temperatur verwenden.

BL-Touch

Anschließen von BL-Touch

Eine Warnung, bevor Sie beginnen: Vermeiden Sie es, den BL-Touch-Stift mit bloßen Fingern zu berühren, da er sehr empfindlich auf Fingerfett reagiert. Und wenn Sie ihn doch berühren, seien Sie sehr vorsichtig, um nichts zu verbiegen oder zu drücken.

Verbinden Sie den BL-Touch "Servo"-Anschluss mit einem `control_pin` gemäß der BL-Touch-Dokumentation oder Ihrer MCU-Dokumentation. Bei der Originalverdrahtung ist der gelbe Draht des Tripels der `control_pin` und der weiße Draht des Paares ist der sensor_pin. Sie müssen diese Pins entsprechend Ihrer Verdrahtung konfigurieren. Die meisten BL-Touch-Geräte erfordern einen Pullup am Sensor-Pin (stellen Sie dem Pin-Namen ein "^" voran). Zum Beispiel:

```
[bltouch]
sensor_pin: ^P1.24
control_pin: P1.26
```

Wenn der BL-Touch zum Referenzieren der Z-Achse verwendet wird, setzen Sie endstop_pin: probe:z_virtual_endstop und entfernen position_endstop im [stepper_z] Konfigurationsabschnitt, dann fügen Sie einen [safe_z_home] Konfigurationsabschnitt hinzu, um die Z-Achse anzuheben, die xy-Achsen zu referenzieren, zur Mitte des Bettes zu fahren und die Z-Achse zu referenzieren. Zum Beispiel:

```
[safe_z_home]
home_xy_position: 100, 100      # Koordinaten auf die Mitte des Druckbetts ändern
Geschwindigkeit: 50
z_hop: 10                       # 10mm nach oben bewegen
z_hop_speed: 5
```

Es ist wichtig, dass die z_hop-Bewegung in safe_z_home hoch genug ist, damit der Messtaster nichts trifft, selbst wenn der Messtasterstift in seinem niedrigsten Zustand ist.

Erste Tests

Bevor Sie fortfahren, überprüfen Sie, ob der BL-Touch in der richtigen Höhe montiert ist; der Stift sollte sich im eingefahrenen Zustand etwa 2 mm über der Düse befinden.

Wenn Sie den Drucker einschalten, sollte die BL-Touch-Sonde einen Selbsttest durchführen und den Stift ein paar Mal auf und ab bewegen. Sobald der Selbsttest abgeschlossen ist, sollte der Stift zurückgezogen werden und die rote LED auf der Sonde sollte leuchten. Sollten Fehler auftreten, z. B. wenn die Sonde rot blinkt oder der Stift nach unten statt nach oben zeigt, schalten Sie den Drucker aus und überprüfen Sie die Verkabelung und Konfiguration.

Wenn alles in Ordnung ist, ist es an der Zeit zu testen, ob der Kontrollstift richtig funktioniert. Führen Sie zunächst `BLTOUCH_DEBUG COMMAND=pin_down` in Ihrem Druckerterminal aus. Vergewissern Sie sich, dass sich der Stift nach unten bewegt und dass die rote LED auf der Sonde erlischt. Ist dies nicht der Fall, überprüfen Sie Ihre Verkabelung und Konfiguration erneut. Geben Sie als Nächstes `einen BLTOUCH_DEBUG COMMAND=pin_up` ein und überprüfen Sie, ob sich der Stift nach oben bewegt und die rote Lampe wieder aufleuchtet. Wenn es blinkt, gibt es ein Problem.

Der nächste Schritt ist die Bestätigung, dass der Sensorstift richtig funktioniert. Führen Sie `BLTOUCH_DEBUG COMMAND=pin_down` aus und überprüfen Sie, ob sich der Stift nach unten bewegt, führen Sie `BLTOUCH_DEBUG COMMAND=touch_mode` aus, führen Sie `QUERY_PROBE` aus und überprüfen Sie, ob der Befehl "probe: open" meldet. Drücken Sie dann den Stift mit dem Fingernagel leicht nach oben und führen Sie erneut `QUERY_PROBE` aus. Vergewissern Sie sich, dass der Befehl die Meldung "probe: `TRIGGERED`". Wenn eine der beiden Abfragen nicht die richtige Meldung meldet, deutet dies in der Regel auf eine falsche Verdrahtung oder Konfiguration hin (obwohl einige Klone eine besondere Behandlung erfordern können). Nach Abschluss dieses Tests führen Sie `BLTOUCH_DEBUG COMMAND=pin_up` aus und überprüfen Sie, ob sich der Pin nach oben bewegt.

Nach Abschluss der Tests des BL-Touch-Steuerpins und des Sensorpins ist es nun an der Zeit, die Abtastung zu testen, allerdings mit einer anderen Vorgehensweise. Anstatt den Sensorstift das Druckbett berühren zu lassen, lassen Sie ihn den Nagel Ihres Fingers berühren. Positionieren Sie den Werkzeugkopf weit vom Druckbett entfernt, geben Sie ein `G28` aus (oder `PROBE`, wenn Sie nicht probe:z_virtual_endstop verwenden), warten Sie, bis der Werkzeugkopf beginnt, sich nach unten zu bewegen, und stoppen Sie die Bewegung, indem Sie den Stift ganz sanft mit Ihrem Nagel berühren. Möglicherweise müssen Sie dies zweimal tun, da die Standardkonfiguration für die Referenzpunktfahrt zweimal testet. Stellen Sie sich darauf ein, den Drucker auszuschalten, wenn er nicht anhält, wenn Sie den Stift berühren.

Wenn dies erfolgreich war, machen Sie einen weiteren `G28` (oder `PROBE`), aber lassen Sie ihn diesmal das Bett berühren, wie es sein sollte.

BL-Touch ist defekt

Sobald sich der BL-Touch in einem inkonsistenten Zustand befindet, beginnt er rot zu blinken. Sie können ihn zwingen, diesen Zustand zu verlassen, indem Sie den Befehl

```
BLTOUCH_DEBUG COMMAND=reset
```

Dies kann passieren, wenn die Kalibrierung unterbrochen wird, weil die Sonde nicht herausgezogen werden kann.

Es kann aber auch sein, dass der BL-Touch sich nicht mehr selbst kalibrieren kann. Das passiert, wenn die Schraube an der Oberseite des Geräts falsch positioniert ist oder sich der Magnetkern im Inneren des Sondenstifts verschoben hat. Wenn er sich so weit nach oben bewegt hat, dass er an der Schraube festklebt, kann er seinen Stift möglicherweise nicht mehr absenken. In diesem Fall müssen Sie die Schraube öffnen und sie mit einem Kugelschreiber vorsichtig zurückschieben. Setzen Sie den Stift wieder in den BL-Touch ein, so dass er in die herausgezogene Position fällt. Drehen Sie die Madenschraube vorsichtig wieder in die richtige Position. Sie müssen die richtige Position finden, damit sich der Stift absenken und anheben kann und das rote Licht aufleuchtet. Verwenden Sie die Befehle `reset, pin_up` und `pin_down`, um dies zu erreichen.

BL-Touch "Klone"

Viele BL-Touch "Klon"-Geräte arbeiten korrekt mit Klipper zusammen, wenn die Standardkonfiguration verwendet wird. Einige "Clone"-Geräte unterstützen jedoch nicht den `QUERY_PROBE`-Befehl und einige "Clone"-Geräte erfordern die Konfiguration von `pin_up_reports_not_triggered` oder `pin_up_touch_mode_reports_triggered`.

Wichtig! Konfigurieren Sie `pin_up_reports_not_triggered` oder `pin_up_touch_mode_reports_triggered` nicht auf False, ohne vorher diese Anweisungen zu befolgen. Konfigurieren Sie keinen dieser Werte auf einem echten BL-Touch auf False. Eine falsche Einstellung auf False kann die Abtastzeit verlängern und das Risiko einer Beschädigung des Druckers erhöhen.

Einige "Klon"-Geräte unterstützen den touch_mode nicht, so dass der Befehl `QUERY_PROBE` nicht funktioniert. Trotzdem kann es möglich sein, mit diesen Geräten eine Abtastung und Referenzfahrt durchzuführen. Bei diesen Geräten wird der `QUERY_PROBE`-Befehl während der ersten Tests initial tests nicht erfolgreich sein, der anschließende `G28`- (oder `PROBE`-) Test ist jedoch erfolgreich. Es kann möglich sein, diese "Klon"-Geräte mit Klipper zu verwenden, wenn man den `QUERY_PROBE`-Befehl nicht verwendet und die Funktion `probe_with_touch_mode` nicht aktiviert.

Einige "Klon"-Geräte sind nicht in der Lage, den internen Sensorverifizierungstest von Klipper durchzuführen. Bei diesen Geräten kann der Versuch, den Sensor in die Homeposition zu bringen oder zu testen, dazu führen, dass Klipper den Fehler "BLTouch failed to verify sensor state" meldet. Sollte dies der Fall sein, dann führe manuell die Schritte aus, um zu bestätigen, dass der Sensor-Pin funktioniert, wie im Abschnitt über die ersten Tests initial tests beschrieben. Wenn die `QUERY_PROBE`-Befehle in diesem Test immer die erwarteten Ergebnisse liefern und der Fehler "BLTouch failed to verify sensor state" immer noch auftritt, dann kann es notwendig sein, `pin_up_touch_mode_reports_triggered` in der Klipper-Konfigurationsdatei auf False zu setzen.

Einige wenige alte "Klon"-Geräte sind nicht in der Lage zu melden, wenn sie ihre Sonde erfolgreich angehoben haben. Bei diesen Geräten meldet Klipper nach jedem Home- oder Probe-Versuch den Fehler "BLTouch failed to raise probe". Man kann diese Geräte testen, indem man den Kopf weit vom Bett entfernt, `BLTOUCH_DEBUG COMMAND=pin_down` ausführt, prüft, ob sich der Stift nach unten bewegt hat, QUERY_PROBE ausführt, prüft, ob der Befehl "probe`:` open" meldet, `BLTOUCH_DEBUG COMMAND=pin_up` ausführt, prüft, ob sich der Stift nach oben bewegt hat, und `QUERY_PROBE` ausführt. Wenn der Pin oben bleibt, das Gerät nicht in einen Fehlerzustand übergeht und die erste Abfrage "probe`:` open" meldet, während die zweite Abfrage "probe`:` `TRIGGERED`" meldet, dann bedeutet das, dass `pin_up_reports_not_triggered` in der Klipper-Konfigurationsdatei auf False gesetzt werden sollte.

BL-Touch v3

Bei einigen BL-Touch v3.0 und BL-Touch 3.1 Geräten kann es erforderlich sein, `probe_with_touch_mode` in der Druckerkonfigurationsdatei zu konfigurieren.

Wenn der Signaldraht des BL-Touch v3.0 an einen Endstop-Pin (mit einem Rauschfilterkondensator) angeschlossen ist, kann der BL-Touch v3.0 während der Referenzfahrt und der Abtastung möglicherweise kein konsistentes Signal senden. Wenn die `QUERY_PROBE`-Befehle im Abschnitt über die anfänglichen Tests initial tests immer die erwarteten Ergebnisse liefern, aber der Werkzeugkopf während der G28/PROBE-Befehle nicht immer anhält, dann ist das ein Hinweis auf dieses Problem. Eine Abhilfe besteht darin, `probe_with_touch_mode: True` in der Konfigurationsdatei zu setzen.

Der BL-Touch v3.1 kann nach einem erfolgreichen Probe-Versuch fälschlicherweise in einen Fehlerzustand übergehen. Die Symptome sind ein gelegentliches Blinken des BL-Touch v3.1, das einige Sekunden lang anhält, nachdem er erfolgreich das Bett berührt hat. Klipper sollte diesen Fehler automatisch löschen und er ist im Allgemeinen harmlos. Man kann jedoch `probe_with_touch_mode` in der Konfigurationsdatei einstellen, um dieses Problem zu vermeiden.

Wichtig! Einige "Clone"-Geräte und der BL-Touch v2.0 (und früher) können eine verminderte Genauigkeit aufweisen, wenn probe_with_touch_mode auf True gesetzt ist. Wenn Sie diesen Wert auf True setzen, erhöht sich auch die Zeit, die für die Bereitstellung der Sonde benötigt wird. Wenn Sie diesen Wert auf einem "Klon" oder einem älteren BL-Touch-Gerät konfigurieren, sollten Sie die Genauigkeit des Messtasters vor und nach dem Einstellen dieses Wertes testen (verwenden Sie den Befehl `PROBE_ACCURACY` zum Testen).

Multisondierung ohne Stiftrückzug

Standardmäßig setzt Klipper die Sonde zu Beginn eines jeden Sondierungsversuchs ein und verstaut sie danach. Dieses wiederholte Ausfahren und Verstauen der Sonde kann die Gesamtzeit von Kalibrierungssequenzen, die viele Sondenmessungen beinhalten, erhöhen. Klipper unterstützt die Möglichkeit, die Sonde zwischen aufeinanderfolgenden Messungen im ausgefahrenen Zustand zu belassen, was die Gesamtzeit der Messungen reduzieren kann. Dieser Modus wird durch die Konfiguration von stow_on_each_sample auf False in der Konfigurationsdatei aktiviert.

Wichtig! Die Einstellung `stow_on_each_sample` auf False kann dazu führen, dass Klipper horizontale Werkzeugkopfbewegungen ausführt, während der Taster eingesetzt wird. Vergewissern Sie sich, dass alle Antastvorgänge genügend Z-Freiraum haben, bevor Sie diesen Wert auf False

setzen. Wenn der Abstand nicht ausreicht, kann eine horizontale Bewegung dazu führen, dass der Stift an einem Hindernis hängen bleibt und der Drucker beschädigt wird.

Wichtig! Es wird empfohlen, `probe_with_touch_mode` auf True zu setzen, wenn `stow_on_each_sample` auf False gesetzt ist. Einige "Clone"-Geräte erkennen einen nachfolgenden Bettkontakt möglicherweise nicht, wenn probe_with_touch_mode nicht eingestellt ist. Bei allen Geräten vereinfacht die Kombination dieser beiden Einstellungen die Gerätesignalisierung, was die Gesamtstabilität verbessern kann.

Beachten Sie jedoch, dass einige "Clone"-Geräte und der BL-Touch v2.0 (und früher) eine geringere Genauigkeit aufweisen können, wenn `probe_with_touch_mode` auf True gesetzt ist. Bei diesen Geräten ist es ratsam, die Genauigkeit des Messtasters vor und nach dem Setzen von `probe_with_touch_mode` zu testen (verwenden Sie dazu den Befehl `PROBE_ACCURACY`).

Kalibrierung der BL-Touch Offsets

Folgen Sie den Anweisungen in der Anleitung Probe Calibrate, um die Konfigurationsparameter x_offset, y_offset und z_offset einzustellen.

Es ist ratsam zu überprüfen, ob der Z-Offset nahe bei 1 mm liegt. Wenn dies nicht der Fall ist, sollten Sie die Sonde nach oben oder unten verschieben, um dies zu korrigieren. Sie möchten, dass die Sonde auslöst, bevor die Düse auf das Bett auftrifft, so dass ein eventuell festsitzendes Filament oder ein verzogenes Bett keine Auswirkungen auf die Sondenaktion hat. Gleichzeitig soll die zurückgezogene Position so weit wie möglich über der Düse liegen, damit sie die gedruckten Teile nicht berührt. Wenn eine Anpassung der Tasterposition vorgenommen wird, führen Sie die Taster-Kalibrierungsschritte erneut durch.

BL-Touch-Ausgabemodus

Ein BL-Touch V3.0 unterstützt die Einstellung eines 5V- oder OPEN-DRAIN-Ausgangsmodus, ein BL-Touch V3.1 unterstützt dies ebenfalls, kann dies aber auch in seinem internen EEPROM speichern. Wenn Ihre Steuerplatine den festen 5V-High-Logikpegel des 5V-Modus benötigt, können Sie den Parameter 'set_output_mode' im Abschnitt [bltouch] der Druckerkonfigurationsdatei auf "5V" setzen.

Verwenden Sie den 5V-Modus nur, wenn die Eingangsleitung Ihrer Steuerkarte 5V tolerant ist. Aus diesem Grund ist die Standardkonfiguration dieser BL-Touch-Versionen der OPEN-DRAIN-Modus. Sie könnten möglicherweise die CPU Ihrer Steuerplatine beschädigen.

Daher gilt Folgendes: Wenn eine Steuerplatine den 5V-Modus benötigt UND auf ihrer Eingangssignalleitung 5V tolerant ist UND wenn

- Sie einen BL-Touch Smart V3.0 haben, müssen Sie den 'set_output_mode: 5V' verwenden, um diese Einstellung bei jedem Start sicherzustellen, da sich der Tastkopf die erforderliche Einstellung nicht merken kann.
- Wenn Sie einen BL-Touch Smart V3.1 haben, haben Sie die Wahl zwischen der Verwendung von 'set_output_mode: 5V' zu verwenden oder den Modus einmalig mit dem Befehl 'BLTOUCH_STORE MODE=5V' manuell zu speichern und NICHT den Parameter 'set_output_mode:' zu verwenden.
- Sie haben einen anderen Tastkopf: Einige Tastköpfe haben eine Leiterbahn auf der Platine, die durchtrennt werden muss, oder einen Jumper, der gesetzt werden muss, um den Ausgangsmodus (dauerhaft) einzustellen. In diesem Fall lassen Sie den Parameter 'set_output_mode' ganz weg.

Wenn Sie eine V3.1 haben, sollten Sie das Speichern des Ausgangsmodus nicht automatisieren oder wiederholen, um zu vermeiden, dass das EEPROM des Fühlers abgenutzt wird, denn das BLTouch-EEPROM ist für etwa 100.000 Aktualisierungen gut. 100 Speicherungen pro Tag ergeben eine Betriebsdauer von etwa 3 Jahren, bevor es abgenutzt ist. Die Speicherung des Ausgabemodus in der V3.1 wurde vom Hersteller als komplizierter Vorgang konzipiert (die Werkseinstellung ist ein sicherer OPEN DRAIN-Modus) und eignet sich nicht für wiederholte Ausgaben durch einen Slicer, ein Makro oder etwas anderes, sondern sollte vorzugsweise nur bei der ersten Integration des Fühlers in eine Druckerelektronik verwendet werden.

Bed Mesh

Das Modul Bed Mesh kann verwendet werden, um Unregelmäßigkeiten der Bettoberfläche auszugleichen und so eine bessere erste Schicht über das gesamte Bett zu erzielen. Es ist zu beachten, dass eine softwarebasierte Korrektur keine perfekten Ergebnisse erzielen kann, sondern nur eine Annäherung an die Form des Bettes. Bed Mesh kann auch keine mechanischen und elektrischen Probleme kompensieren. Wenn eine Achse schief steht oder ein Messtaster nicht genau ist, erhält das Modul bed_mesh keine genauen Ergebnisse aus dem Messvorgang.

Vor der Mesh-Kalibrierung müssen Sie sicherstellen, dass der Z-Offset Ihres Messtasters kalibriert ist. Wenn Sie einen Endanschlag für die Z-Homing-Funktion verwenden, muss dieser ebenfalls kalibriert werden. Weitere Informationen finden Sie unter Kalibrierung des Messtasters und Z_ENDSTOP_CALIBRATE im Handbuch Ebene.

Grundlegende Konfiguration

Rechteckige Druckebene

Dieses Beispiel geht von einem Drucker mit einem rechteckigen Bett von 250 mm x 220 mm und einer Sonde mit einem x-Versatz von 24 mm und einem y-Versatz von 5 mm aus.

```
[bed_mesh]
Geschwindigkeit: 120
horizontale_Verschiebung_z: 5
Maschenweite_min: 35, 6
Maschenweite_max: 240, 198
```

```
probe_count: 5, 3
```

```
speed: 120
```
Standardwert: 50
Die Geschwindigkeit, mit der sich das Werkzeug zwischen den Punkten bewegt.

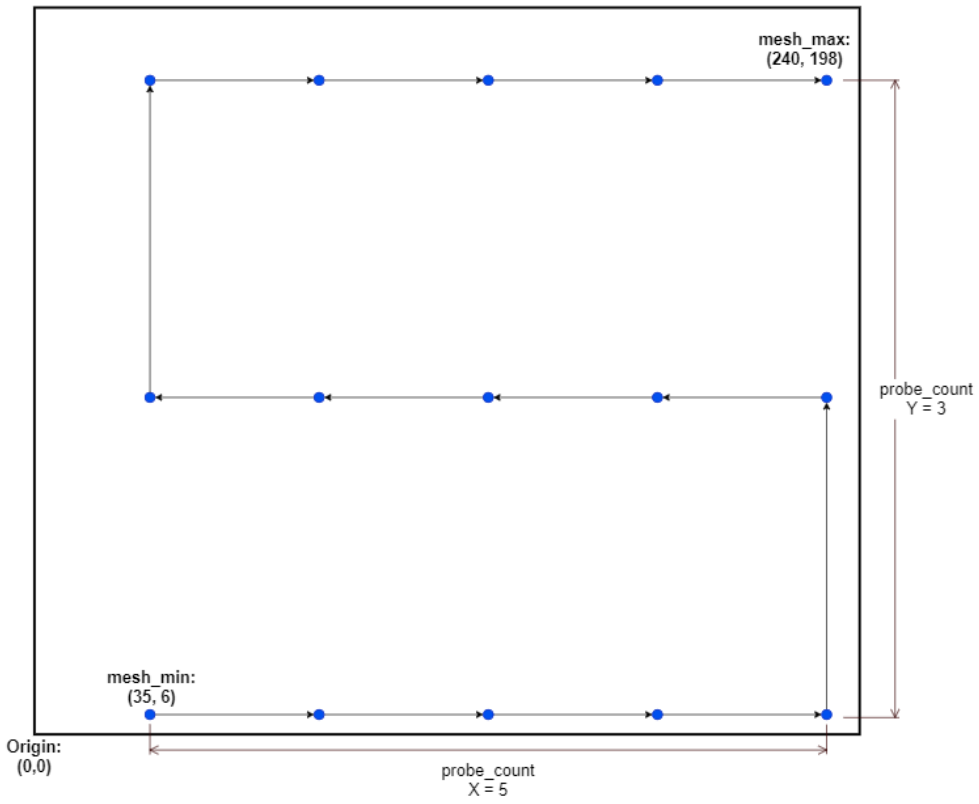
- `horizontal_move_z: 5`
  Standardwert: 5
  Die Z-Koordinate, auf die der Messtaster ansteigt, bevor er sich zwischen den Punkten bewegt.

- `mesh_min: 35, 6`
  Erforderlich
  Die erste ermittelte Koordinate, die dem Ursprung am nächsten liegt. Diese Koordinate ist relativ zum Standort der Sonde.

- `mesh_max: 240, 198`
  Erforderlich
  Die am weitesten vom Ursprung entfernte Sondierungskoordinate. Dies ist nicht notwendigerweise der letzte ermittelte Punkt, da die Ermittelung im Zick-Zack-Verfahren erfolgt. Wie bei `mesh_min` ist diese Koordinate relativ zum Standort der Sonde.

- `probe_count: 5, 3`
  Standardwert: 3, 3
  Die Anzahl der zu sondierenden Punkte auf jeder Achse, angegeben als ganzzahlige X- und Y-Werte. In diesem Beispiel werden 5 Punkte entlang der X-Achse und 3 Punkte entlang der Y-Achse gesampelt, also insgesamt 15 gesampelte Punkte. Wenn Sie ein quadratisches Raster, z. B. 3x3, wünschen, können Sie dies als einen einzigen Integer-Wert angeben, der für beide Achsen verwendet wird, d. h. probe_count: 3. Beachten Sie, dass ein Netz eine Mindestanzahl von `probe_count` von 3 entlang jeder Achse erfordert.

Die folgende Abbildung zeigt, wie die Optionen `mesh_min, mesh_max` und `probe_count` zur Erzeugung von Probepunkten verwendet werden. Die Pfeile zeigen die Richtung des Sondierungsverfahrens an, beginnend bei `mesh_min`. Zur Veranschaulichung: Wenn die Sonde bei mesh_min ist, befindet sich der Stutzen bei (11, 1), und wenn die Sonde bei `mesh_max` ist, befindet sich der Stutzen bei (206, 193).



bedmesh_rect_basic

Erweiterte Konfiguration

Nachfolgend werden die erweiterten Konfigurationsoptionen im Detail erläutert. Jedes Beispiel baut auf der oben gezeigten grundlegenden rechteckigen Bettkonfiguration auf. Jede der erweiterten Optionen gilt in gleicher Weise für runde Betten.

Mesh Interpolation

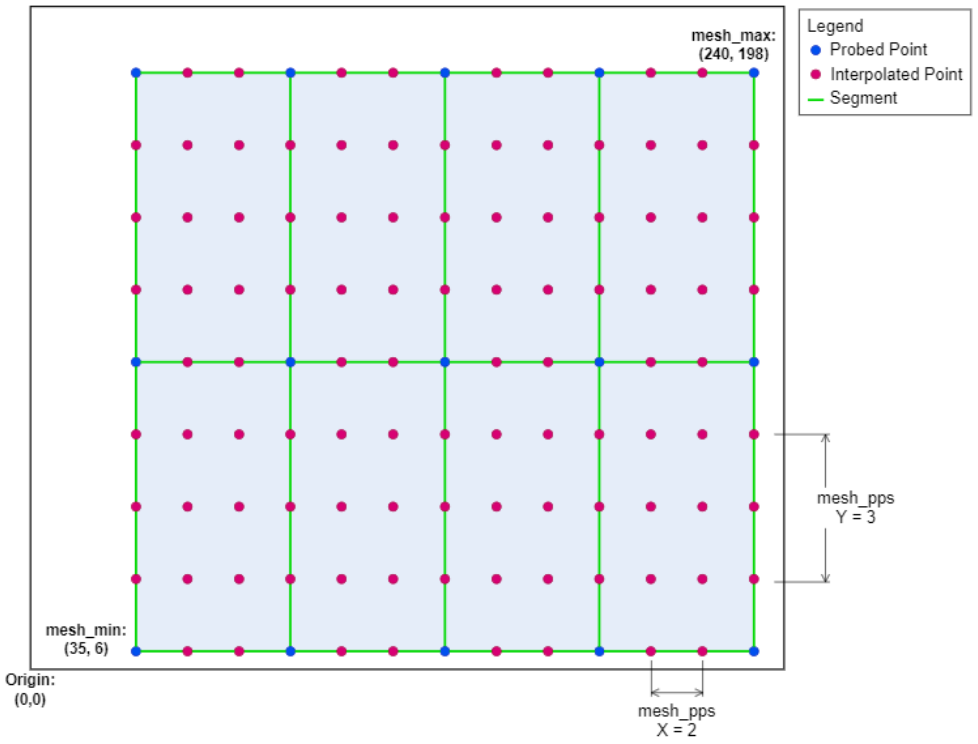
Während es möglich ist, die Sondierungsmatrix direkt mit einfacher bilinearer Interpolation abzutasten, um die Z-Werte zwischen den Sondierungspunkten zu bestimmen, ist es oft sinnvoll, zusätzliche Punkte mit Hilfe von erweiterten Interpolationsalgorithmen zu interpolieren, um die Netzdichte zu erhöhen. Diese Algorithmen fügen dem Netz eine Krümmung hinzu und versuchen so, die Materialeigenschaften des Bettes zu simulieren. Bed Mesh bietet Lagrange- und bikubische Interpolation, um dies zu erreichen.

```
[bed_mesh]
Geschwindigkeit: 120
horizontale_Verschiebung_z: 5
netz_min: 35, 6
netz_max: 240, 198
probe_count: 5, 3
netz_pps: 2, 3
Algorithmus: bikubisch
bikubische_Zugkraft: 0.2
```

- `mesh_pps: 2, 3`
  Standardwert: 2, 2
  Die Option `mesh_pps` ist eine Abkürzung für Mesh Points Per Segment. Diese Option gibt an, wie viele Punkte für jedes Segment entlang der X- und Y-Achse interpoliert werden sollen. Ein "Segment" ist der Raum zwischen den einzelnen Sondierungspunkten. Wie probe_count wird mesh_pps als X- und Y-Ganzzahlpaar angegeben, kann aber auch als einzelne Ganzzahl angegeben werden, die auf beide Achsen angewendet wird. In diesem Beispiel gibt es 4 Segmente entlang der X-Achse und 2 Segmente entlang der Y-Achse. Daraus ergeben sich 8 interpolierte Punkte entlang der X-Achse und 6 interpolierte Punkte entlang der Y-Achse, was zu einem 13x8-Netz führt. Wenn mesh_pps auf 0 gesetzt wird, ist die Interpolation des Netzes deaktiviert und die gesampelte Matrix wird direkt abgetastet.

- `Algorithmus: lagrange`
  Standardwert: lagrange
  Der zur Interpolation des Netzes verwendete Algorithmus. Kann Lagrange oder bikubisch sein. Die Lagrange-Interpolation ist auf 6 Abtastpunkte begrenzt, da bei einer größeren Anzahl von Abtastungen Oszillationen auftreten können. Für die bikubische Interpolation sind mindestens 4 Abtastpunkte entlang jeder Achse erforderlich; wenn weniger als 4 Punkte angegeben werden, wird die Lagrange-Interpolation erzwungen. Wenn mesh_pps auf 0 gesetzt ist, wird dieser Wert ignoriert, da keine Netzinterpolation durchgeführt wird.

- `bicubic_tension: 0.2`
  Standardwert: 0.2
  Wenn die Algorithmus-Option auf bikubisch eingestellt ist, kann der Wert für die Spannung angegeben werden. Je höher die Spannung ist, desto stärker wird die Steigung interpoliert. Seien Sie vorsichtig, wenn Sie dies einstellen, da höhere Werte auch mehr Überschwingen erzeugen, was dazu führt, dass die interpolierten Werte höher oder niedriger sind als Ihre Messpunkte.
  Die folgende Abbildung zeigt, wie die oben genannten Optionen zur Erzeugung eines interpolierten Netzes verwendet werden.



bedmesh_interpoliert

Splitting verschieben
Bed Mesh funktioniert, indem es gcode-Bewegungsbefehle abfängt und eine Transformation auf ihre Z-Koordinate anwendet. Lange Bewegungen müssen in kleinere Bewegungen aufgeteilt werden, um der Form des Bettes korrekt zu folgen. Die folgenden Optionen steuern das Aufteilungsverhalten.

```
[bed_mesh]
Geschwindigkeit: 120
horizontale_Bewegung_z: 5
netz_min: 35, 6
```

```
netz_max: 240, 198
probe_count: 5, 3
move_check_distance: 5
split_delta_z: .025
```

- `move_check_distance: 5`
  Standardwert: 5
  Die Mindestdistanz, die auf die gewünschte Z-Änderung geprüft wird, bevor eine Teilung durchgeführt wird. In diesem Beispiel wird eine Bewegung, die länger als 5 mm ist, vom Algorithmus durchlaufen. Alle 5 mm wird der Z-Wert des Netzes nachgeschlagen und mit dem Z-Wert der vorherigen Bewegung verglichen. Wenn das Delta den durch `split_delta_z` festgelegten Schwellenwert erreicht, wird die Bewegung geteilt und die Durchquerung fortgesetzt. Dieser Vorgang wiederholt sich, bis das Ende der Bewegung erreicht ist, wo eine letzte Anpassung vorgenommen wird. Bei Zügen, die kürzer als die `move_check_distance` sind, wird die korrekte Z-Korrektur direkt auf den Zug angewandt, ohne Traversierung oder Aufteilung.

- `split_delta_z: .025`
  Standardwert: .025
  Wie oben erwähnt, ist dies die Mindestabweichung, die erforderlich ist, um eine Zugaufteilung auszulösen. In diesem Beispiel löst jeder Z-Wert mit einer Abweichung von +/- .025 mm eine Aufteilung aus.
  Im Allgemeinen sind die Standardwerte für diese Optionen ausreichend, der Standardwert von 5 mm für move_check_distance könnte sogar zu viel sein. Ein fortgeschrittener Benutzer kann jedoch mit diesen Optionen experimentieren, um die optimale erste Schicht herauszupressen.

Mesh Fade
Wenn "Fade" aktiviert ist, wird die Z-Anpassung über eine durch die Konfiguration festgelegte Strecke abgebaut. Dies wird durch kleine Anpassungen der Schichthöhe erreicht, die je nach Form des Bettes entweder zunehmen oder abnehmen. Wenn die Ausblendung abgeschlossen ist, wird die Z-Anpassung nicht mehr angewendet, so dass die Oberseite des Drucks flach ist und nicht die Form des Bettes widerspiegelt. Die Überblendung kann auch einige unerwünschte Eigenschaften haben: Wenn Sie die Überblendung zu schnell durchführen, kann es zu sichtbaren Artefakten auf dem Druck kommen. Wenn Ihr Druckbett stark verzogen ist, kann die Überblendung außerdem die Z-Höhe des Drucks verkleinern oder vergrößern. Aus diesem Grund ist die Überblendung standardmäßig deaktiviert.

```
[bed_mesh]
Geschwindigkeit: 120
horizontal_verschieben_z: 5
netz_min: 35, 6
Maschen_max: 240, 198
probe_count: 5, 3
fade_start: 1
fade_end: 10
fade_target: 0
```

- `fade_start: 1`
  Standardwert: 1
  Die Z-Höhe, in der die Ausblendungsanpassung beginnen soll. Es ist ratsam, ein paar Ebenen tiefer zu gehen, bevor der Fade-Prozess beginnt.

- `fade_end: 10`
  Standardwert: 0
  Die Z-Höhe, in der die Überblendung beendet werden soll. Wenn dieser Wert niedriger als `fade_start` ist, wird die Überblendung deaktiviert. Dieser Wert kann angepasst werden, je nachdem wie verzogen die Druckoberfläche ist. Eine stark verzogene Oberfläche sollte über eine längere Strecke ausgeblendet werden. Bei einer nahezu ebenen Oberfläche kann dieser Wert verringert werden, um das Ausblenden zu beschleunigen. 10 mm ist ein vernünftiger Wert für den Anfang, wenn der Standardwert von 1 für `fade_start` verwendet wird.

- `fade_target: 0`
  Standardwert: Der durchschnittliche Z-Wert des Netzes
  Das `fade_target` kann man sich als zusätzlichen Z-Offset vorstellen, der nach Abschluss der Überblendung auf das gesamte Bett angewendet wird. Im Allgemeinen sollte dieser Wert 0 sein, es gibt jedoch Umstände, unter denen dies nicht der Fall sein sollte. Nehmen wir zum Beispiel an, dass Ihre Referenzpunktposition auf dem Bett ein Ausreißer ist und um 0,2 mm niedriger liegt als die durchschnittliche Messhöhe des Bettes. Wenn der Wert für `fade_target` 0 ist, wird der Druck um durchschnittlich 0,2 mm über das Bett verkleinert. Wenn Sie `fade_target` auf 0 setzen, wird der Druckbereich um 0,2 mm vergrößert, der Rest des Bettes hat jedoch eine genaue Größe. Im Allgemeinen ist es eine gute Idee, `fade_target` nicht zu konfigurieren, so dass die durchschnittliche Höhe des Netzes verwendet wird. Es kann jedoch wünschenswert sein, das Fade-Target manuell anzupassen, wenn man auf einen bestimmten Teil des Bettes drucken möchte.

Der Relative Referenzindex
Die meisten Sonden sind anfällig für Drift, d. h. für Ungenauigkeiten bei der Abtastung, die durch Wärme oder Interferenzen verursacht werden. Dies kann die Berechnung des z-Offsets der Sonde schwierig machen, insbesondere bei unterschiedlichen Betttemperaturen. Daher verwenden einige Drucker einen Endanschlag für die Referenzierung der Z-Achse und einen Messtaster für die Kalibrierung des Netzes. Diese Drucker können von der Konfiguration des relativen Referenzindexes profitieren.

```
[bed_mesh]
```

```
Geschwindigkeit: 120
horizontale_Verschiebung_z: 5
netz_min: 35, 6
netz_max: 240, 198
probe_count: 5, 3
relative_referenz_index: 7
```

- `relative_referenz_index: 7`
  Standardwert: None (deaktiviert)
  Bei der Erzeugung der Sondierungspunkte wird jedem Punkt ein Index zugewiesen. Sie können diesen Index in klippy.log oder mit Hilfe von BED_MESH_OUTPUT nachschlagen (siehe den Abschnitt über Bed Mesh GCodes weiter unten für weitere Informationen). Wenn Sie der Option `relative_reference_index` einen Index zuweisen, ersetzt der an dieser Koordinate ermittelte Wert den z_offset der Sonde. Dadurch wird diese Koordinate zur "Null"-Referenz für das Netz.
  Wenn Sie den relativen Referenzindex verwenden, sollten Sie den Index wählen, der dem Punkt auf dem Bett am nächsten liegt, an dem die Z-Endanschlagskalibrierung durchgeführt wurde. Beachten Sie, dass Sie bei der Suche nach dem Index mithilfe des Protokolls oder BED_MESH_OUTPUT die unter der Überschrift "Probe" aufgeführten Koordinaten verwenden sollten, um den richtigen Index zu finden.

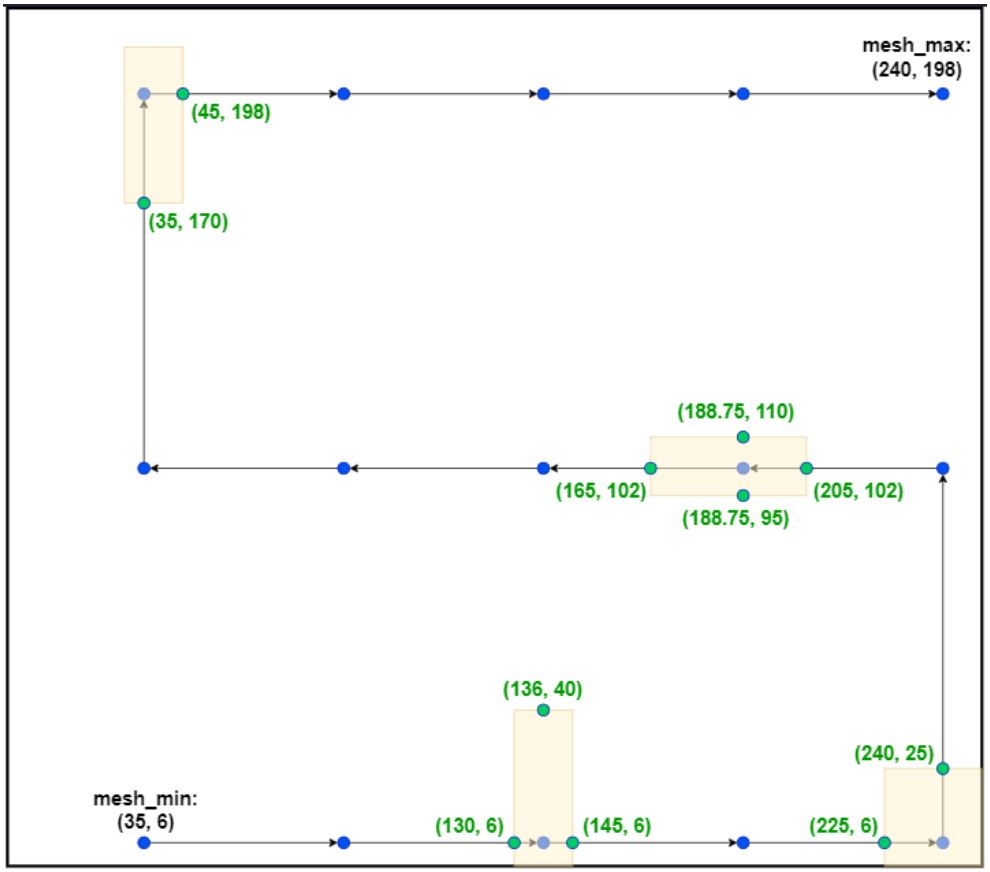
Fehlerhafte Regionen
Es ist möglich, dass einige Bereiche eines Bettes bei der Sondierung aufgrund eines "Fehlers" an bestimmten Stellen ungenaue Ergebnisse melden. Das beste Beispiel hierfür sind Betten mit einer Reihe von integrierten Magneten, die zum Festhalten von abnehmbaren Stahlblechen verwendet werden. Das Magnetfeld an und um diese Magnete kann dazu führen, dass eine induktive Sonde in einem höheren oder niedrigeren Abstand auslöst, als es sonst der Fall wäre, was zu einem Netz führt, das die Oberfläche an diesen Stellen nicht genau darstellt. Hinweis: Dies ist nicht zu verwechseln mit einer Verzerrung der Sondenposition, die ungenaue Ergebnisse für das gesamte Bett liefert.

Die Optionen faulty_region können so konfiguriert werden, dass sie diesen Effekt kompensieren. Wenn ein erzeugter Punkt innerhalb eines fehlerhaften Bereichs liegt, versucht das Bettnetz, bis zu 4 Punkte an den Grenzen dieses Bereichs zu sondieren. Diese gemessenen Werte werden gemittelt und als Z-Wert an der erzeugten (X, Y)-Koordinate in das Netz eingefügt.

```
[bed_mesh]
Geschwindigkeit: 120
horizontale_Verschiebung_z: 5
netz_min: 35, 6
netz_max: 240, 198
probe_count: 5, 3
fehlerhafte_region_1_min: 130.0, 0.0
fehlerhaftes_gebiet_1_max: 145.0, 40.0
fehlerhafte_region_2_min: 225.0, 0.0
fehlerhafte_Region_2_max: 250.0, 25.0
fehlerhafte_region_3_min: 165.0, 95.0
fehlerhafte_region_3_max: 205.0, 110.0
fehlerhafte_region_4_min: 30.0, 170.0
fehlerhafte_region_4_max: 45.0, 210.0
```

- `faulty_region_{1...99}_min`
  `faulty_region_{1..99}_max`
  Standardwert: Keine (deaktiviert)
  Fehlerhafte Regionen werden ähnlich wie das Netz selbst definiert, wobei die minimalen und maximalen (X, Y) Koordinaten für jede Region angegeben werden müssen. Eine fehlerhafte Region kann sich außerhalb eines Netzes erstrecken, die erzeugten alternativen Punkte liegen jedoch immer innerhalb der Netzgrenze. Zwei Regionen dürfen sich nicht überschneiden.

Die folgende Abbildung zeigt, wie Ersatzpunkte erzeugt werden, wenn ein erzeugter Punkt innerhalb einer fehlerhaften Region liegt. Die dargestellten Regionen entsprechen denen in der obigen Beispielkonfiguration. Die Ersatzpunkte und ihre Koordinaten sind in grüner Farbe dargestellt.



bedmesh_interpolated

Bed Mesh Gcodes

Kalibrierung

`BED_MESH_CALIBRATE PROFILE=<name> METHOD=[manuell | automatisch] [<probe_parameter>=<value>] [<mesh_parameter>=<value>]`

Standardprofil: Standard
Standardmethode: automatisch, wenn eine Sonde erkannt wird, sonst manuell

Leitet das Sondierungsverfahren für die Bettnetzkalibrierung ein.

Das Netz wird in einem Profil gespeichert, das durch den `PROFILE`-Parameter angegeben wird, oder in einem Standardprofil `default`, wenn es nicht angegeben wird. Wenn `METHOD=manual` gewählt wird, erfolgt die Sondierung manuell. Beim Umschalten zwischen automatischer und manueller Sondierung werden die erzeugten Netzpunkte automatisch angepasst.

Es ist möglich, Maschenparameter anzugeben, um den zu sondierenden Bereich zu verändern. Die folgenden Parameter sind verfügbar:

- Rechteckige Betten (kartesisch):
  `MESH_MIN`
  `MESH_MAX`
  `PROBE_ZAEHLUNG`

- Runde Betten (delta):
  `MESH_RADIUS`
  `MESH_ORIGIN`
  `RUNDE_PROBE_COUNT`

- Alle Betten:
  `RELATIVER_REFERENZ_INDEX`
  `ALGORITHM`

In der obigen Konfigurationsdokumentation finden Sie nähere Angaben dazu, wie die einzelnen Parameter auf das Netz anzuwenden sind.

Profile

`BED_MESH_PROFILE SAVE=<name> LOAD=<name> REMOVE=<name>`

Nachdem eine `BED_MESH_CALIBRATE` durchgeführt wurde, ist es möglich, den aktuellen Zustand des Netzes in einem benannten Profil zu speichern. Dadurch ist es möglich, ein Netz zu laden, ohne das Bett neu zu sondieren. Nachdem ein Profil mit `BED_MESH_PROFILE SAVE=<name>` gespeichert wurde, kann der `SAVE_CONFIG` gcode ausgeführt werden, um das Profil in printer.cfg zu schreiben.

Profile können durch die Ausführung von `BED_MESH_PROFILE LOAD=<name>` geladen werden.

Es ist zu beachten, dass bei jedem `BED_MESH_CALIBRATE` der aktuelle Zustand automatisch im Standardprofil gespeichert wird. Wenn dieses Profil existiert, wird es automatisch geladen, wenn Klipper startet. Wenn dieses Verhalten nicht erwünscht ist, kann das Standardprofil wie folgt entfernt werden:

```
BED_MESH_PROFILE REMOVE=default
```

Jedes andere gespeicherte Profil kann auf die gleiche Weise entfernt werden, wobei default durch das zu entfernende Profil ersetzt wird.

Ausgabe
`BED_MESH_OUTPUT PGP=[0 | 1]`

Gibt den aktuellen Zustand des Netzes auf dem Terminal aus. Beachten Sie, dass das Netz selbst ausgegeben wird.

Der Parameter PGP ist eine Abkürzung für "Print Generated Points". Wenn `PGP=1` gesetzt ist, werden die generierten Sondierungspunkte auf dem Terminal ausgegeben:

```
// bed_mesh: generierte Punkte
// Index | Werkzeugeinstellung | Messtaster
// 0 | (11.0, 1.0) | (35.0, 6.0)
// 1 | (62.2, 1.0) | (86.2, 6.0)
// 2 | (113.5, 1.0) | (137.5, 6.0)
// 3 | (164.8, 1.0) | (188.8, 6.0)
// 4 | (216.0, 1.0) | (240.0, 6.0)
// 5 | (216.0, 97.0) | (240.0, 102.0)
// 6 | (164.8, 97.0) | (188.8, 102.0)
// 7 | (113.5, 97.0) | (137.5, 102.0)
// 8 | (62.2, 97.0) | (86.2, 102.0)
// 9 | (11.0, 97.0) | (35.0, 102.0)
// 10 | (11.0, 193.0) | (35.0, 198.0)
// 11 | (62.2, 193.0) | (86.2, 198.0)
// 12 | (113.5, 193.0) | (137.5, 198.0)
// 13 | (164.8, 193.0) | (188.8, 198.0)
// 14 | (216.0, 193.0) | (240.0, 198.0)
```

Die "Tool Adjusted"-Punkte beziehen sich auf die Position der Düse für jeden Punkt, die "Probe"-Punkte auf die Position der Sonde. Beachten Sie, dass sich beim manuellen Antasten die "Probe"-Punkte sowohl auf die Werkzeug- als auch auf die Düsenposition beziehen.

Maschenstatus löschen
`BED_MESH_CLEAR`

Dieser Gcode kann zum Löschen des internen Mesh-Status verwendet werden.

X/Y-Offsets anwenden
`BED_MESH_OFFSET [X=<value>] [Y=<value>]`

Dies ist nützlich für Drucker mit mehreren unabhängigen Extrudern, da ein Offset notwendig ist, um eine korrekte Z-Anpassung nach einem Werkzeugwechsel zu erreichen. Die Offsets sollten relativ zum Hauptextruder angegeben werden. Das heißt, ein positiver X-Offset sollte angegeben werden, wenn der sekundäre Extruder rechts vom primären Extruder montiert ist, und ein positiver Y-Offset sollte angegeben werden, wenn der sekundäre Extruder "hinter" dem primären Extruder montiert ist.

Endstopp-Phase

Dieses Dokument beschreibt das phasengesteuerte Endstoppsystem von Klipper. Diese Funktion kann die Genauigkeit herkömmlicher Endstoppschalter verbessern. Sie ist besonders nützlich, wenn ein Trinamic-Schrittmotortreiber verwendet wird, der über eine Laufzeitkonfiguration verfügt.

Ein typischer Endanschlagschalter hat eine Genauigkeit von etwa 100 Mikrometern. (Bei jeder Referenzfahrt einer Achse kann der Schalter etwas früher oder etwas später ausgelöst werden). Obwohl dies ein relativ kleiner Fehler ist, kann er zu unerwünschten Artefakten führen. Insbesondere beim Druck der ersten Schicht eines Objekts kann sich diese Positionsabweichung bemerkbar machen. Im Gegensatz dazu können typische Schrittmotoren eine wesentlich höhere Präzision erreichen.

Der schrittphasenangepasste Endanschlagmechanismus kann die Präzision der Schrittmotoren nutzen, um die Präzision der Endanschlagschalter zu verbessern. Ein Schrittmotor bewegt sich, indem er eine Reihe von Phasen durchläuft, bis er vier "volle Schritte" vollendet hat. Ein Schrittmotor mit 16 Mikroschritten hätte also 64 Phasen, und wenn er sich in eine positive Richtung bewegt, würde er die Phasen durchlaufen: 0, 1, 2, ... 61, 62, 63, 0, 1, 2,

usw. Entscheidend ist, dass der Schrittmotor, wenn er sich an einer bestimmten Position auf einer linearen Schiene befindet, immer die gleiche Schrittphase hat. Wenn also ein Wagen den Endschalter auslöst, sollte der Schrittmotor, der diesen Wagen steuert, immer die gleiche Phase haben. Das Endstop-Phasensystem von Klipper kombiniert die Schrittmotorphase mit dem Endstop-Trigger, um die Genauigkeit des Endstopps zu verbessern.

Um diese Funktion nutzen zu können, ist es notwendig, die Phase des Schrittmotors zu kennen. Wenn man die Trinamic-Treiber TMC2130, TMC2208, TMC2224 oder TMC2660 im Runtime-Konfigurationsmodus (d.h. nicht im Standalone-Modus) verwendet, kann Klipper die Schrittmotorphase vom Treiber abfragen. (Es ist auch möglich, dieses System bei herkömmlichen Steppertreibern zu verwenden, wenn man die Steppertreiber zuverlässig zurücksetzen kann - siehe unten für Details).

Kalibrierung der Endanschlagsphasen

Wenn Sie Trinamic-Schrittmotortreiber mit Laufzeitkonfiguration verwenden, können Sie die Endanschlagsphasen mit dem Befehl `ENDSTOP_PHASE_CALIBRATE` kalibrieren. Beginnen Sie, indem Sie das Folgende zur Konfigurationsdatei hinzufügen:

```
[endstop_phase]
```

Starten Sie dann den Drucker neu und führen Sie einen G28-Befehl gefolgt von einem `ENDSTOP_PHASE_CALIBRATE`-Befehl aus. Bewegen Sie dann den Werkzeugkopf an eine neue Position und führen Sie erneut den Befehl G28 aus. Versuchen Sie, den Werkzeugkopf an mehrere verschiedene Positionen zu bewegen und führen Sie G28 an jeder Position erneut aus. Führen Sie mindestens fünf G28-Befehle aus.

Nachdem Sie die oben genannten Schritte durchgeführt haben, wird der Befehl `ENDSTOP_PHASE_CALIBRATE` oft die gleiche (oder fast die gleiche) Phase für den Stepper melden. Diese Phase kann in der Konfigurationsdatei gespeichert werden, so dass alle zukünftigen G28-Befehle diese Phase verwenden. (So wird Klipper bei zukünftigen Referenzfahrten die gleiche Position erreichen, auch wenn der Endanschlag etwas früher oder etwas später ausgelöst wird).

Um die Endstopp-Phase für einen bestimmten Schrittmotor zu speichern, führen Sie etwas wie das Folgende aus:

```
ENDSTOP_PHASE_CALIBRATE STEPPER=stepper_z
```

Führen Sie den obigen Befehl für alle Schrittmotoren aus, die Sie speichern möchten. Normalerweise würde man dies für stepper_z bei kartesischen und Corexy-Druckern und für stepper_a, stepper_b und stepper_c bei Deltadruckern verwenden. Führen Sie schließlich den folgenden Befehl aus, um die Konfigurationsdatei mit den Daten zu aktualisieren:

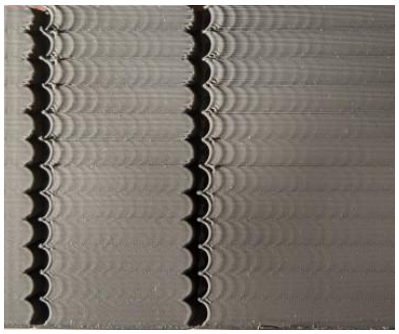
```
SAVE_CONFIG
```

Zusätzliche Hinweise

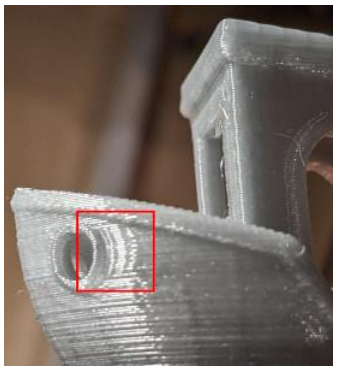
- Diese Funktion ist vor allem bei Deltadruckern und am Z-Endanschlag von kartesischen/Corexy-Druckern nützlich. Es ist möglich, diese Funktion auf die XY-Endanschläge von kartesischen Druckern anzuwenden, aber das ist nicht besonders nützlich, da ein kleiner Fehler in der X/Y-Endanschlagsposition die Druckqualität wahrscheinlich nicht beeinträchtigt. Es ist nicht zulässig, diese Funktion für die XY-Endanschläge von Corexy-Druckern zu verwenden (da die XY-Position bei Corexy-Kinematiken nicht durch einen einzigen Stepper bestimmt wird). Es ist nicht zulässig, diese Funktion auf einem Drucker zu verwenden, der einen "probe:z_virtual_endstop" Z-Endanschlag verwendet (da die Stepperphase nur stabil ist, wenn sich der Endanschlag an einer statischen Position auf einer Schiene befindet).
- Wenn der Endanschlag nach der Kalibrierung der Endanschlagsphase später bewegt oder eingestellt wird, muss der Endanschlag neu kalibriert werden. Entfernen Sie die Kalibrierungsdaten aus der Konfigurationsdatei und führen Sie die obigen Schritte erneut aus.
- Um dieses System verwenden zu können, muss der Endanschlag genau genug sein, um die Stepperposition innerhalb von zwei "vollen Schritten" zu identifizieren. Wenn ein Stepper also beispielsweise 16 Mikroschritte mit einem Schrittabstand von 0,005 mm verwendet, muss der Endstopp eine Genauigkeit von mindestens 0,160 mm haben. Wenn man Fehlermeldungen vom Typ "Endstop stepper_z incorrect phase" erhält, kann dies an einem Endstop liegen, der nicht genau genug ist. Wenn eine Neukalibrierung nicht hilft, deaktivieren Sie die Endstop-Phasenanpassung, indem Sie sie aus der Konfigurationsdatei entfernen.
- Wenn man eine herkömmliche schrittgesteuerte Z-Achse (wie bei einem kartesischen oder Corexy-Drucker) zusammen mit herkömmlichen Bettnivellierschrauben verwendet, ist es auch möglich, dieses System so zu verwenden, dass jede Druckschicht an einer "Vollschritt"-Grenze ausgeführt wird. Um diese Funktion zu aktivieren, stellen Sie sicher, dass der G-Code Slicer mit einer Schichthöhe konfiguriert ist, die ein Vielfaches eines "Full Step" ist, aktivieren Sie manuell die Option endstop_align_zero im Konfigurationsabschnitt endstop_phase (siehe Konfigurationsreferenz config reference für weitere Details) und nivellieren Sie dann die Bettschrauben neu.
- Es ist möglich, dieses System mit herkömmlichen (nicht-trinamischen) Schrittmotor-Treibern zu verwenden. Dazu muss jedoch sichergestellt werden, dass die Schrittmotortreiber jedes Mal zurückgesetzt werden, wenn der Mikrocontroller zurückgesetzt wird. (Wenn die beiden immer zusammen zurückgesetzt werden, kann Klipper die Schrittmotorphase bestimmen, indem er die Gesamtzahl der Schritte verfolgt, die er dem Schrittmotor befohlen hat). Derzeit ist die einzige Möglichkeit, dies zuverlässig zu tun, wenn sowohl der Mikrocontroller als auch die Schrittmotortreiber ausschließlich über USB mit Strom versorgt werden und dieser USB-Strom von einem Host bereitgestellt wird, der auf einem Raspberry Pi läuft. In diesem Fall kann man eine mcu-Konfiguration mit "restart_method: rpi_usb" angeben - diese Option sorgt dafür, dass der Mikrocontroller immer über einen USB-Power-Reset zurückgesetzt wird, wodurch sowohl der Mikrocontroller als auch die Schrittmotortreiber gemeinsam zurückgesetzt werden. Wenn man diesen Mechanismus verwendet, müsste man die "trigger_phase"-Konfigurationsabschnitte manuell konfigurieren (siehe Konfigurationsreferenz config reference für die Details).

Resonanz-Kompensation

Klipper unterstützt Input Shaping - eine Technik, die eingesetzt werden kann, um Ringing (auch bekannt als Echoing, Ghosting oder Rippling) in Drucken zu reduzieren. Ringing ist ein Fehler beim Flächendruck, bei dem sich typischerweise Elemente wie Kanten auf einer gedruckten Oberfläche als subtiles "Echo" wiederholen:



Ringing-Test



3D Benchy

Ringing wird durch mechanische Vibrationen im Drucker verursacht, die auf schnelle Änderungen der Druckrichtung zurückzuführen sind. Beachten Sie, dass das Schwingen in der Regel mechanische Ursachen hat: unzureichend steifer Druckerrahmen, undichte oder zu stark federnde Profile, Ausrichtungsprobleme mechanischer Teile, schwere bewegte Masse usw. Diese sollten nach Möglichkeit zuerst überprüft und behoben werden.

Beim Input shaping handelt es sich um eine offene Regeltechnik, die ein Steuersignal erzeugt, das seine eigenen Schwingungen aufhebt. Das Input Shaping erfordert einige Abstimmungen und Messungen, bevor es aktiviert werden kann. Neben dem Schwingen reduziert Input Shaping in der Regel auch die Vibrationen und Erschütterungen des Druckers im Allgemeinen und kann auch die Zuverlässigkeit des StealthChop-Modus von Trinamic-Schrittmachertreibern verbessern.

Abstimmung

Die Grundabstimmung erfordert die Messung der Schwingungsfrequenzen des Druckers durch Drucken eines Testmodells.

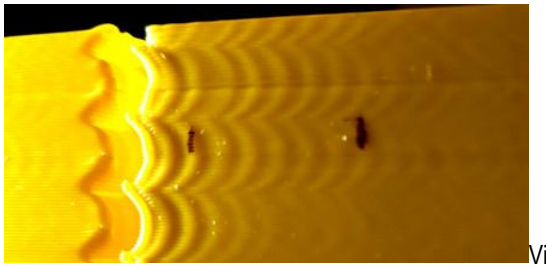
Schneiden Sie das Ringing-Testmodell, das Sie in docs/prints/ringing_tower.stl finden, im Slicer:

- Die vorgeschlagene Schichthöhe beträgt 0,2 oder 0,25 mm.
- Die Füll- und Deckschichten können auf 0 gesetzt werden.
- Verwenden Sie 1-2 Perimeter, oder noch besser den glatten Vasenmodus mit 1-2 mm Basis.
- Verwenden Sie eine ausreichend hohe Geschwindigkeit, etwa 80-100 mm/sec, für die äußeren Umfänge.
- Achten Sie darauf, dass die minimale Schichtzeit höchstens 3 Sekunden beträgt.
- Vergewissern Sie sich, dass die "dynamische Beschleunigungskontrolle" in der Schneidemaschine deaktiviert ist.
- Drehen Sie das Modell nicht. Das Modell hat X- und Y-Markierungen auf der Rückseite des Modells. Beachten Sie die ungewöhnliche Lage der Markierungen im Vergleich zu den Achsen des Druckers - es handelt sich nicht um einen Fehler. Die Markierungen können später im Abstimmungsprozess als Referenz verwendet werden, da sie zeigen, welcher Achse die Messungen entsprechen.

Schwingungsfrequenz

Messen Sie zunächst die Schwingungsfrequenz.

1. Wenn der Parameter square_corner_velocity geändert wurde, setzen Sie ihn wieder auf 5,0 zurück. Es ist nicht ratsam, diesen Wert zu erhöhen, wenn Sie den Input Shaper verwenden, da dies zu einer stärkeren Glättung der Teile führen kann - es ist besser, stattdessen einen höheren Beschleunigungswert zu verwenden.
2. Erhöhen Sie max_accel_to_decel, indem Sie den folgenden Befehl eingeben: `SET_VELOCITY_LIMIT ACCEL_TO_DECEL=7000`
3. Deaktivieren Sie die Druckvorverlegung: `SET_PRESSURE_ADVANCE ADVANCE=0`
4. Wenn Sie den Abschnitt [input_shaper] in der Datei printer.cfg bereits hinzugefügt haben, führen Sie den Befehl `SET_INPUT_SHAPER SHAPER_FREQ_X=0 SHAPER_FREQ_Y=0` aus. Wenn Sie die Fehlermeldung "Unbekannter Befehl" erhalten, können Sie sie an dieser Stelle getrost ignorieren und mit den Messungen fortfahren.
5. Führen Sie den Befehl aus: `TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500 STEP_HEIGHT=5` Grundsätzlich versuchen wir, diese Schwingungen durch das Einstellen von unterschiedlich großen Werten für die Beschleunigung stärker hervorzuheben. Mit diesem Befehl wird die Beschleunigung ab 1500 mm/sec^2 alle 5 mm erhöht: 1500 mm/sec^2, 2000 mm/sec^2, 2500 mm/sec^2 und so weiter bis zu 7000 mm/sec^2 am letzten Band.
6. Drucken Sie das Testmodell in Scheiben mit den vorgeschlagenen Parametern.
7. Sie können den Druck früher abbrechen, wenn diese Schwingungen deutlich sichtbar sind und Sie sehen, dass die Beschleunigung für Ihren Drucker zu hoch wird (z. B. wenn der Drucker zu stark zittert oder anfängt, Schritte zu überspringen).
8. Verwenden Sie die X- und Y-Markierungen auf der Rückseite des Modells als Referenz. Die Messungen von der Seite mit der X-Markierung sollten für die Konfiguration der X-Achse und die Y-Markierung für die Konfiguration der Y-Achse verwendet werden. Messen Sie den Abstand D (in mm) zwischen mehreren Schwingungen auf dem Teil mit der X-Markierung, in der Nähe der Kerben, wobei Sie vorzugsweise die erste oder zwei Schwingungen auslassen. Um den Abstand zwischen den Schwingungen leichter zu messen, markieren Sie zuerst die Schwingungen und messen dann den Abstand zwischen den Markierungen mit einem Lineal oder einem Messschieber:


Vibrationen markieren


und messen

9. Zählen Sie, wie vielen Schwingungen N der gemessene Abstand D entspricht. Wenn du dir nicht sicher bist, wie du die Schwingungen zählen sollst, sieh dir das Bild oben an, das N = 6 Schwingungen zeigt.

10. Berechnen Sie die Schwingungsfrequenz der X-Achse als V - N / D (Hz), wobei V die Geschwindigkeit für die äußeren Umfänge (mm/s) ist. Für das obige Beispiel haben wir 6 Schwingungen markiert, und der Test wurde mit einer Geschwindigkeit von 100 mm/sec gedruckt, also ist die Frequenz 100 * 6 / 12,14 ≈ 49,4 Hz.
11. Führen Sie (8) - (10) auch für die Y-Marke durch.

Beachten Sie, dass die Schwingungen auf dem Testdruck dem Muster der gebogenen Kerben folgen sollte, wie in der Abbildung oben. Wenn dies nicht der Fall ist, handelt es sich nicht wirklich um Schwingungen, sondern um einen anderen Fehler - entweder ein mechanisches oder ein Extruderproblem. Dieses Problem sollte zuerst behoben werden, bevor die Eingangsformer aktiviert und eingestellt werden.

Wenn die Messungen nicht zuverlässig sind, weil z. B. der Abstand zwischen den Schwingungen nicht stabil ist, könnte dies bedeuten, dass der Drucker mehrere Resonanzfrequenzen auf derselben Achse hat. Man kann stattdessen versuchen, das im Abschnitt Unzuverlässige Messungen von Schwingungsfrequenzen beschriebene Abstimmungsverfahren Unreliable measurements of ringing frequencies zu befolgen und trotzdem etwas aus der Input-Shaping-Technik herauszuholen.

Die Häufigkeit der Schwingungen kann von der Position des Modells innerhalb der Bauplatte und der Z-Höhe abhängen, insbesondere bei Delta-Druckern; Sie können prüfen, ob Sie Unterschiede in den Frequenzen an verschiedenen Positionen entlang der Seiten des Testmodells und in verschiedenen Höhen feststellen. Wenn dies der Fall ist, können Sie die durchschnittlichen Klingelfrequenzen über die X- und Y-Achse berechnen.

Wenn die gemessene Klingelfrequenz sehr niedrig ist (unter ca. 20-25 Hz), könnte es eine gute Idee sein, in eine Versteifung des Druckers oder eine Verringerung der beweglichen Masse zu investieren - je nachdem, was in Ihrem Fall anwendbar ist -, bevor Sie mit der weiteren Abstimmung der Eingangsformung fortfahren und die Frequenzen anschließend erneut messen. Für viele gängige Druckermodelle gibt es oft bereits Lösungen.

Beachten Sie, dass sich die Schwingungsfrequenzen ändern können, wenn Änderungen am Drucker vorgenommen werden, die sich beispielsweise auf die bewegte Masse auswirken oder die Steifigkeit des Systems verändern:

- Einige Werkzeuge werden am Werkzeugkopf installiert, entfernt oder ausgetauscht, die dessen Masse verändern, z. B. wird ein neuer (schwererer oder leichterer) Schrittmotor für den Direktextruder oder ein neues Hotend installiert, ein schwerer Lüfter mit einem Kanal wird hinzugefügt, usw.
- Riemen werden nachgespannt.
- Einige Zusätze zur Erhöhung der Rahmensteifigkeit werden installiert.
- Bei einem Bettschleuderdrucker wird ein anderes Bett installiert, oder es wird Glas hinzugefügt usw.

Wenn solche Änderungen vorgenommen werden, ist es ratsam, zumindest die Klingelfrequenzen zu messen, um festzustellen, ob sie sich verändert haben.

Konfiguration des Input Shapers

Nachdem die Klingelfrequenzen für die X- und Y-Achse gemessen wurden, können Sie den folgenden Abschnitt in die Datei printer.cfg einfügen:

```
[input_shaper]
shaper_freq_x: ...  # Frequenz für die X-Marke des Testmodells
shaper_freq_y: ...  # Frequenz für die Y-Marke des Testmodells
Für das obige Beispiel ergibt sich shaper_freq_x/y = 49,4.
```

Auswahl des Eingangs-Shapers

Klipper unterstützt mehrere Input Shaper. Sie unterscheiden sich in ihrer Empfindlichkeit gegenüber Fehlern bei der Bestimmung der Resonanzfrequenz und darin, wie viel Glättung sie in den gedruckten Teilen verursachen. Auch sollten einige der Shaper wie 2HUMP_EI und 3HUMP_EI normalerweise nicht mit shaper_freq = Resonanzfrequenz verwendet werden - sie sind aus unterschiedlichen Überlegungen heraus so konfiguriert, dass sie mehrere Resonanzen auf einmal reduzieren.

Für die meisten Drucker können entweder MZV- oder EI-Shaper empfohlen werden. In diesem Abschnitt wird ein Testverfahren beschrieben, um zwischen diesen beiden zu wählen und einige andere damit verbundene Parameter zu ermitteln.

Drucken Sie das Schwingungstestmodell wie folgt aus:

1. Starten Sie die Firmware neu: `RESTART`
2. Für den Test vorbereiten: `SET_VELOCITY_LIMIT ACCEL_TO_DECEL=7000`
3. Deaktivieren Sie die Druckvorverlegung: `SET_PRESSURE_ADVANCE ADVANCE=0`
4. Ausführen: `SET_INPUT_SHAPER SHAPER_TYPE=MZV`
5. Führen Sie den Befehl aus: `TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500 STEP_HEIGHT=5`
6. Drucken Sie das Testmodell, das mit den vorgeschlagenen Parametern geschnitten wurde.

Wenn Sie an dieser Stelle kein Schwingungen sehen, kann der MZV-Shaper zur Verwendung empfohlen werden.

Wenn Sie gewisse Schwingungen feststellen, messen Sie die Frequenzen erneut mit den Schritten (8)-(10), die im Abschnitt Schwingungsfrequenz Ringing frequency beschrieben sind. Weichen die Frequenzen erheblich von den zuvor ermittelten Werten ab, ist eine komplexere Konfiguration des Eingangs-Shapers erforderlich. Weitere Informationen finden Sie im Abschnitt Technische Details zu Input shapers. Ansonsten fahren Sie mit dem nächsten Schritt fort.
Versuchen Sie nun den EI-Eingangs-Shaper. Wiederholen Sie dazu die Schritte (1)-(6) von oben, führen Sie aber bei Schritt 4 stattdessen den folgenden Befehl aus: `SET_INPUT_SHAPER SHAPER_TYPE=EI.`

Vergleichen Sie zwei Ausdrucke mit MZV und EI input shaper. Wenn EI deutlich bessere Ergebnisse als MZV zeigt, verwenden Sie EI-Shaper, andernfalls bevorzugen Sie MZV. Beachten Sie, dass der EI-Shaper eine stärkere Glättung der gedruckten Teile bewirkt (siehe nächster Abschnitt für weitere Details). Fügen Sie den Parameter shaper_type: mzv (oder ei) zum Abschnitt [input_shaper] hinzu, z.B.:

```
[input_shaper]
shaper_freq_x: ...
shaper_freq_y: ...
shaper_type: mzv
```

Ein paar Hinweise zur Auswahl des Shapers:

- Der EI-Shaper eignet sich möglicherweise besser für Bettschleuderdrucker (wenn die Resonanzfrequenz und die daraus resultierende Glättung dies zulassen): Je mehr Filament auf dem sich bewegenden Bett abgelegt wird, desto größer wird die Masse des Bettes und desto niedriger wird die Resonanzfrequenz. Da der EI-Shaper robuster gegenüber Änderungen der Resonanzfrequenz ist, kann er beim Druck großer Teile besser funktionieren.
- Aufgrund der Natur der Delta-Kinematik können die Resonanzfrequenzen in verschiedenen Teilen des Bauvolumens sehr unterschiedlich sein. Daher kann der EI-Shaper für Delta-Drucker besser geeignet sein als MZV oder ZV und sollte für den Einsatz in Betracht gezogen werden. Wenn die Resonanzfrequenz ausreichend groß ist (mehr als 50-60 Hz), kann man sogar versuchen, den 2HUMP_EI-Shaper zu testen (indem man den oben vorgeschlagenen Test mit `SET_INPUT_SHAPER SHAPER_TYPE=2HUMP_EI` durchführt), aber prüfen Sie die Überlegungen im folgenden Abschnitt section below, bevor Sie ihn aktivieren.

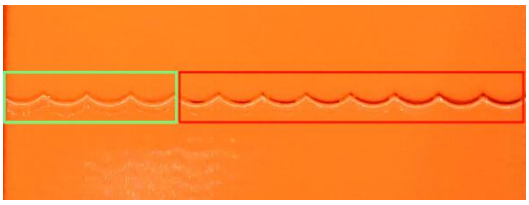
Auswählen von max_accel

Sie sollten einen gedruckten Test für den im vorherigen Schritt gewählten Shaper haben (falls nicht, drucken Sie das Testmodell mit den vorgeschlagenen Parametern suggested parameters aus, wobei der Druckvorlauf deaktiviert ist `SET_PRESSURE_ADVANCE ADVANCE=0` und der Abstimmturm als `TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500 STEP_HEIGHT=5` aktiviert ist). Beachten Sie, dass die Eingangsformung bei sehr hohen Beschleunigungen je nach Resonanzfrequenz und gewähltem Eingangs-Shaper (z. B. erzeugt der EI-Shaper mehr Glättung als der MZV-Shaper) eine zu starke Glättung und Abrundung der Teile verursachen kann. Daher sollte max_accel so gewählt werden, dass dies verhindert wird. Ein weiterer Parameter, der sich auf die Glättung auswirken kann, ist `square_corner_velocity`. Es ist daher nicht ratsam, ihn über den Standardwert von 5 mm/sec zu erhöhen, um eine stärkere Glättung zu verhindern.
Um einen geeigneten max_accel-Wert auszuwählen, untersuchen Sie das Modell für den gewählten Eingangs-Shaper. Notieren Sie sich zunächst, bei welcher Beschleunigung die Schwingungen noch gering sind - so dass Sie damit zufrieden sind.

Prüfen Sie dann die Glättung. Um dies zu erleichtern, hat das Testmodell einen kleinen Spalt in der Wand (0,15 mm):

Testspalt

Mit zunehmender Beschleunigung nimmt auch die Glättung zu, und der tatsächliche Spalt im Druck wird größer:



Shaper-Glättung

In diesem Bild nimmt die Beschleunigung von links nach rechts zu, und der Spalt beginnt ab 3500 mm/sec^2 (5. Band von links) zu wachsen. In diesem Fall ist der richtige Wert für max_accel = 3000 (mm/sec^2), um eine übermäßige Glättung zu vermeiden.

Beachten Sie die Beschleunigung, wenn der Spalt in Ihrem Testdruck noch sehr klein ist. Wenn Sie Ausbuchtungen, aber überhaupt keinen Spalt in der Wand sehen, selbst bei hohen Beschleunigungen, kann dies an einem deaktivierten Druckvorschub liegen, insbesondere bei Bowden-Extrudern. Wenn dies der Fall ist, müssen Sie den Druck mit aktivierter Druckvorverlegung wiederholen. Es kann auch das Ergebnis eines falsch kalibrierten (zu hohen) Filamentflusses sein, also ist es eine gute Idee, auch das zu überprüfen.

Wählen Sie das Minimum aus den beiden Beschleunigungswerten (aus Ringing und Smoothing) und tragen Sie es als max_accel in printer.cfg ein.

Übrigens kann es vorkommen, dass der EI-Shaper - insbesondere bei niedrigen Ringing-Frequenzen - auch bei niedrigeren Beschleunigungen eine zu starke Glättung bewirkt. In diesem Fall kann MZV die bessere Wahl sein, da es höhere Beschleunigungswerte zulässt.

Bei sehr niedrigen Schwingungsfrequenzen (~25 Hz und darunter) kann auch der MZV-Shaper eine zu starke Glättung bewirken. Wenn das der Fall ist, können Sie auch versuchen, die Schritte im Abschnitt Auswahl des Input-Shapers Choosing input shape mit dem ZV-Shaper zu wiederholen, indem Sie stattdessen den Befehl `SET_INPUT_SHAPER SHAPER_TYPE=ZV` verwenden. Der ZV-Shaper sollte eine noch geringere Glättung aufweisen als der MZV-Shaper, ist aber empfindlicher gegenüber Fehlern bei der Messung der Ringing-Frequenzen.

Eine weitere Überlegung ist, dass es bei einer zu niedrigen Resonanzfrequenz (unter 20-25 Hz) eine gute Idee sein könnte, die Steifigkeit des Druckers zu erhöhen oder die bewegte Masse zu verringern. Andernfalls könnten die Beschleunigung und die Druckgeschwindigkeit eingeschränkt werden, weil der Drucker jetzt zu stark glättet, anstatt zu schwingen.

Feinabstimmung der Resonanzfrequenzen

Beachten Sie, dass die Genauigkeit der mit dem Ringing-Testmodell gemessenen Resonanzfrequenzen für die meisten Zwecke ausreicht, so dass eine weitere Abstimmung nicht angeraten ist. Wenn Sie dennoch versuchen möchten, Ihre Ergebnisse zu überprüfen (z. B. wenn Sie nach dem Drucken eines Testmodells mit einem Eingangs-Shaper Ihrer Wahl mit denselben Frequenzen, die Sie zuvor gemessen haben, immer noch ein gewisses Schwingungen feststellen), können Sie die Schritte in diesem Abschnitt ausführen. Beachten Sie, dass dieser Abschnitt nicht weiterhilft, wenn Sie nach der Aktivierung von [input_shaper] ein Schwingungen bei unterschiedlichen Frequenzen feststellen.

Angenommen, Sie haben das Ringing-Modell mit den vorgeschlagenen Parametern in Scheiben geschnitten, dann führen Sie die folgenden Schritte für jede der Achsen X und Y durch:

1. Für den Test vorbereiten: `SET_VELOCITY_LIMIT ACCEL_TO_DECEL=7000`
2. Stellen Sie sicher, dass Pressure Advance deaktiviert ist: SET_PRESSURE_ADVANCE ADVANCE=0
3. Ausführen: `SET_INPUT_SHAPER SHAPER_TYPE=ZV`
4. Wählen Sie aus dem vorhandenen Ringing-Testmodell mit dem von Ihnen gewählten Eingangs-Shaper die Beschleunigung aus, die das Ringing ausreichend gut zeigt, und setzen Sie sie mit: `SET_VELOCITY_LIMIT ACCEL=...`
5. Berechnen Sie die notwendigen Parameter für den Befehl `TUNING_TOWER`, um den Parameter shaper_freq_x wie folgt abzustimmen: start = shaper_freq_x * 83 / 132 und factor = shaper_freq_x / 66, wobei `shaper_freq_x` hier der aktuelle Wert in `printer.cfg` ist.
6. Führen Sie den Befehl aus: `TUNING_TOWER COMMAND=SET_INPUT_SHAPER PARAMETER=SHAPER_FREQ_X START=start FACTOR=factor BAND=5` mit den in Schritt (5) berechneten `start-` und `faktor` - werten.
7. Drucken Sie das Testmodell.
8. Setzen Sie den ursprünglichen Frequenzwert zurück: `SET_INPUT_SHAPER SHAPER_FREQ_X=....`
9. Suchen Sie das Band, das am wenigsten schwingt, und zählen Sie dessen Nummer von unten nach oben, beginnend bei 1.
10. Berechnen Sie den neuen Wert für shaper_freq_x über old shaper_freq_x * (39 + 5 * #band-number) / 66.

Wiederholen Sie diese Schritte für die Y-Achse auf die gleiche Weise, wobei Sie die Verweise auf die X-Achse durch die Y-Achse ersetzen (ersetzen Sie z. B. `shaper_freq_x` durch `shaper_freq_y` in den Formeln und im Befehl `TUNING_TOWER`).

Nehmen wir als Beispiel an, Sie hätten für eine der Achsen eine Schwingungsfrequenz von 45 Hz gemessen. Dies ergibt für den Befehl TUNING_TOWER die Werte Start = 45 * 83 / 132 = 28,30 und Faktor = 45 / 66 = 0,6818. Nehmen wir nun an, dass nach dem Drucken des Testmodells das vierte Band von unten das geringste Schwingungen ergibt. Dies ergibt den aktualisierten Shaper_freq_? Wert gleich 45 * (39 + 5 * 4) / 66 ≈ 40,23.

Nachdem die beiden neuen Parameter `shaper_freq_x` und `shaper_freq_y` berechnet wurden, können Sie den Abschnitt `[input_shaper]` in `printer.cfg` mit den neuen Werten für `shaper_freq_x` und `shaper_freq_y` aktualisieren.

Pressure Advance

Wenn Sie den Druckvorschub verwenden, muss er möglicherweise neu eingestellt werden. Folgen Sie den Anweisungen instructions, um den neuen Wert zu finden, falls er sich vom vorherigen unterscheidet. Stellen Sie sicher, dass Sie Klipper neu starten, bevor Sie die Druckvorverlegung einstellen.

Unzuverlässige Messungen der Schwingungsfrequenzen
Wenn Sie nicht in der Lage sind, die Schwingungsfrequenzen zu messen, z.B. wenn der Abstand zwischen den Schwingungen nicht stabil ist, können Sie immer noch die Vorteile der Eingangsformungstechniken nutzen, aber die Ergebnisse sind möglicherweise nicht so gut wie bei einer korrekten Messung der Frequenzen und erfordern etwas mehr Abstimmung und das Drucken des Testmodells. Beachten Sie, dass eine weitere Möglichkeit darin besteht, einen Beschleunigungsmesser zu kaufen und zu installieren und die Resonanzen damit zu messen (siehe die Dokumentation, in der die benötigte Hardware und der Einrichtungsprozess beschrieben werden) - diese Option erfordert jedoch etwas Crimpen und Löten.

Zur Abstimmung fügen Sie einen leeren [input_shaper]-Abschnitt in Ihre printer.cfg ein. Wenn Sie das Klingelmodell mit den vorgeschlagenen Parametern geschnitten haben, drucken Sie das Testmodell dreimal wie folgt. Beim ersten Mal, vor dem Druck, führen Sie

```
1. RESTART
2. SET_VELOCITY_LIMIT ACCEL_TO_DECEL=7000
3. SET_PRESSURE_ADVANCE ADVANCE=0
4. SET_INPUT_SHAPER SHAPER_TYPE=2HUMP_EI SHAPER_FREQ_X=60 SHAPER_FREQ_Y=60
5. TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500
   STEP_HEIGHT=5
```

und drucken Sie das Modell aus. Dann drucken Sie das Modell erneut, aber vor dem Drucken führen Sie stattdessen aus

```
1. SET_INPUT_SHAPER SHAPER_TYPE=2HUMP_EI SHAPER_FREQ_X=50 SHAPER_FREQ_Y=50
2. TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500
   STEP_HEIGHT=5
```

Drucken Sie dann das Modell zum 3. Mal, aber führen Sie jetzt aus

```
1. SET_INPUT_SHAPER SHAPER_TYPE=2HUMP_EI SHAPER_FREQ_X=40 SHAPER_FREQ_Y=40
2. TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500
   STEP_HEIGHT=5
```

Im Wesentlichen drucken wir das Ringing-Testmodell mit TUNING_TOWER unter Verwendung des 2HUMP_EI-Shapers mit shaper_freq = 60 Hz, 50 Hz und 40 Hz.

Wenn keines der Modelle eine Verbesserung des Schwingungsverhaltens zeigt, dann sieht es leider nicht so aus, als ob die Eingangsformungstechniken in Ihrem Fall helfen können.

Andernfalls kann es sein, dass alle Modelle keine Schwingungen zeigen, oder einige zeigen die Schwingungen und einige nicht so sehr. Wählen Sie das Testmodell mit der höchsten Frequenz, das noch gute Verbesserungen des Schwingungsverhaltens zeigt. Wenn z. B. die 40-Hz- und 50-Hz-Modelle fast kein Schwingen zeigen und das 60-Hz-Modell bereits mehr Schwingungen aufweist, bleiben Sie bei 50 Hz.

Prüfen Sie nun, ob der EI-Shaper in Ihrem Fall ausreichend ist. Wählen Sie die Frequenz des EI-Shapers auf der Grundlage der Frequenz des von Ihnen gewählten 2HUMP_EI-Shapers:

Für 2HUMP_EI 60 Hz Shaper, verwenden Sie EI Shaper mit shaper_freq = 50 Hz.
Für 2HUMP_EI 50 Hz shaper, verwenden Sie EI shaper mit shaper_freq = 40 Hz.
Für 2HUMP_EI 40 Hz shaper, verwenden Sie EI shaper mit shaper_freq = 33 Hz.
Drucken Sie nun das Testmodell ein weiteres Mal, indem Sie

```
1. SET_INPUT_SHAPER SHAPER_TYPE=EI SHAPER_FREQ_X=... SHAPER_FREQ_Y=...
2. TUNING_TOWER COMMAND=SET_VELOCITY_LIMIT PARAMETER=ACCEL START=1500 STEP_DELTA=500
   STEP_HEIGHT=5
```

unter Angabe der zuvor ermittelten Werte shaper_freq_x=... und shaper_freq_y=....

Wenn der EI-Shaper vergleichbar gute Ergebnisse liefert wie der 2HUMP_EI-Shaper, bleiben Sie bei dem EI-Shaper und der zuvor ermittelten Frequenz, andernfalls verwenden Sie den 2HUMP_EI-Shaper mit der entsprechenden Frequenz. Fügen Sie die Ergebnisse in printer.cfg ein, z.B. als

```
[input_shaper]
shaper_freq_x: 50
shaper_freq_y: 50
shaper_type: 2hump_ei
```

Setzen Sie die Abstimmung mit dem Abschnitt Selecting max_accel fort.

Fehlersuche und FAQ

Ich kann keine zuverlässigen Messungen der Resonanzfrequenzen erhalten

Vergewissern Sie sich zunächst, dass es sich nicht um ein anderes Problem mit dem Drucker handelt, anstatt dieser Schwingungen. Wenn die Messungen nicht zuverlässig sind, weil z. B. der Abstand zwischen den Schwingungen nicht stabil ist, könnte dies bedeuten, dass der Drucker mehrere

Resonanzfrequenzen auf derselben Achse hat. Man kann versuchen, das im Abschnitt Unzuverlässige Messungen von Schwingungsfrequenzen Unreliable measurements of ringing frequencies beschriebene Abstimmungsverfahren zu befolgen und trotzdem etwas aus der Input-Shaper-Technik herauszuholen. Eine andere Möglichkeit besteht darin, einen Beschleunigungsmesser zu installieren, die Resonanzen damit zu messen measure und den Input Shaper anhand der Ergebnisse dieser Messungen automatisch abzustimmen.

Nach der Aktivierung von [input_shaper] erhalte ich zu glatte gedruckte Teile und feine Details gehen verloren

Prüfen Sie die Überlegungen im Abschnitt Auswahl von Selecting max_accel . Wenn die Resonanzfrequenz niedrig ist, sollte man max_accel nicht zu hoch einstellen oder den Parameter square_corner_velocity erhöhen. Es könnte auch besser sein, MZV- oder sogar ZV-Eingangsformer anstelle von EI (oder 2HUMP_EI- und 3HUMP_EI-Formern) zu wählen.

Nachdem ich einige Zeit erfolgreich gedruckt habe, ohne dass Schwingungen aufgetreten sind, scheint es wieder zu kommen

Es ist möglich, dass sich nach einiger Zeit die Resonanzfrequenzen verändert haben. Vielleicht hat sich z.B. die Spannung der Riemen geändert (die Riemen sind lockerer geworden), usw. Es ist eine gute Idee, die Resonanzfrequenzen zu überprüfen und neu zu messen, wie im Abschnitt Resonanzfrequenzen Ringing frequency beschrieben, und die Konfigurationsdatei zu aktualisieren, falls erforderlich.

Wird die Einrichtung von zwei Schlitten mit Input Shapern unterstützt?

Es gibt keine spezielle Unterstützung für doppelte Schlitten mit Input-Shapern, aber das bedeutet nicht, dass diese Einstellung nicht funktioniert. Man sollte die Abstimmung zweimal für jeden der Wagen durchführen und die Schwingungsfrequenzen für die X- und Y-Achse für jeden der Wagen unabhängig voneinander berechnen. Dann legt man die Werte für Wagen 0 in der Sektion [input_shaper] ab und ändert die Werte während des Wagenwechsels, z.B. als Teil eines Makros:

```
SET_DUAL_CARRIAGE CARRIAGE=1
SET_INPUT_SHAPER SHAPER_FREQ_X=... SHAPER_FREQ_Y=...
```

Und in ähnlicher Weise beim Zurückschalten auf Wagen 0.

Beeinflusst input_shaper die Druckzeit?

Nein, die Funktion input_shaper selbst hat so gut wie keinen Einfluss auf die Druckzeiten. Der Wert von max_accel hat jedoch sehr wohl einen Einfluss (die Einstellung dieses Parameters wird in diesem Abschnitt beschrieben this section).

Technische Details

Input Shaper

Die in Klipper verwendeten Input Shaper sind ziemlich standardisiert, und man kann einen detaillierteren Überblick in den Artikeln finden, die die entsprechenden Shaper beschreiben. Dieser Abschnitt enthält einen kurzen Überblick über einige technische Aspekte der unterstützten Input Shaper. Die folgende Tabelle zeigt einige (meist ungefähre) Parameter der einzelnen Shaper.

Input shaper	Shaper duration	Vibration reduction 20x (5% vibration tolerance)	Vibration reduction 10x (10% vibration tolerance)
ZV	0.5 / shaper_freq	N/A	± 5% shaper_freq
MZV	0.75 / shaper_freq	± 4% shaper_freq	-10%...+15% shaper_freq
ZVD	1 / shaper_freq	± 15% shaper_freq	± 22% shaper_freq
EI	1 / shaper_freq	± 20% shaper_freq	± 25% shaper_freq
2HUMP_EI	1.5 / shaper_freq	± 35% shaper_freq	± 40 shaper_freq
3HUMP_EI	2 / shaper_freq	-45...+50% shaper_freq	-50%...+55% shaper_freq

Ein Hinweis zur Schwingungsdämpfung: Die Werte in der obigen Tabelle sind Richtwerte. Wenn das Dämpfungsverhältnis des Druckers für jede Achse bekannt ist, kann der Shaper genauer konfiguriert werden und reduziert dann die Resonanzen in einem etwas breiteren Frequenzbereich. Das Dämpfungsverhältnis ist jedoch in der Regel nicht bekannt und ohne spezielle Ausrüstung schwer abzuschätzen, daher verwendet Klipper standardmäßig den Wert 0,1, der ein guter Allround-Wert ist. Die Frequenzbereiche in der Tabelle decken eine Reihe verschiedener möglicher Dämpfungsverhältnisse um diesen Wert herum ab (ca. von 0,05 bis 0,2).

Beachten Sie auch, dass EI, 2HUMP_EI und 3HUMP_EI so abgestimmt sind, dass sie Schwingungen auf 5 % reduzieren, so dass die Werte für 10 % Schwingungstoleranz nur als Referenz angegeben sind.

So verwenden Sie diese Tabelle:

Die Shaper-Dauer beeinflusst die Glättung der Teile - je größer sie ist, desto glatter sind die Teile. Diese Abhängigkeit ist nicht linear, kann aber einen Eindruck davon vermitteln, welche Former bei gleicher Frequenz mehr glätten". Die Reihenfolge nach der Glättung sieht folgendermaßen aus: ZV < MZV < ZVD ≈ EI < 2HUMP_EI < 3HUMP_EI. Außerdem ist es selten praktisch, für die Shaper 2HUMP_EI und 3HUMP_EI shaper_freq = resonance freq zu setzen (sie sollten verwendet werden, um Schwingungen für mehrere Frequenzen zu reduzieren).

Man kann einen Frequenzbereich abschätzen, in dem der Shaper Schwingungen reduziert. Zum Beispiel reduziert MZV mit shaper_freq = 35 Hz die Schwingungen auf 5 % für die Frequenzen [33,6, 36,4] Hz. 3HUMP_EI mit shaper_freq = 50 Hz reduziert die Schwingungen im Bereich [27,5, 75] Hz auf 5 %.
Anhand dieser Tabelle kann man prüfen, welchen Shaper man verwenden sollte, wenn man Schwingungen bei mehreren Frequenzen reduzieren muss. Zum Beispiel, wenn man Resonanzen bei 35 Hz und 60 Hz auf der gleichen Achse hat : a) EI Shaper muss shaper_freq = 35 / (1 - 0.2) = 43.75 Hz haben, und er wird die Resonanzen bis 43.75 * (1 + 0.2) = 52. 5 Hz, ist also nicht ausreichend; b) 2HUMP_EI shaper muss shaper_freq = 35 / (1 - 0,35) = 53,85 Hz haben und wird die Schwingungen bis 53,85 * (1 + 0,35) = 72,7 Hz reduzieren - dies ist also eine akzeptable Konfiguration. Versuchen Sie immer, eine möglichst hohe Shaper-Freq für einen bestimmten Shaper zu verwenden (vielleicht mit einer gewissen Sicherheitsmarge, so dass in diesem Beispiel Shaper-Freq ≈ 50-52 Hz am besten funktionieren würde), und versuchen Sie, einen Shaper mit einer möglichst kleinen Shaper-Dauer zu verwenden.
Wenn man Schwingungen bei mehreren sehr unterschiedlichen Frequenzen (z. B. 30 Hz und 100 Hz) reduzieren muss, kann man feststellen, dass die obige Tabelle nicht genügend Informationen liefert. In diesem Fall hat man vielleicht mehr Glück mit dem Skript scripts/graph_shaper.py, das flexibler ist.

Messung von Resonanzen

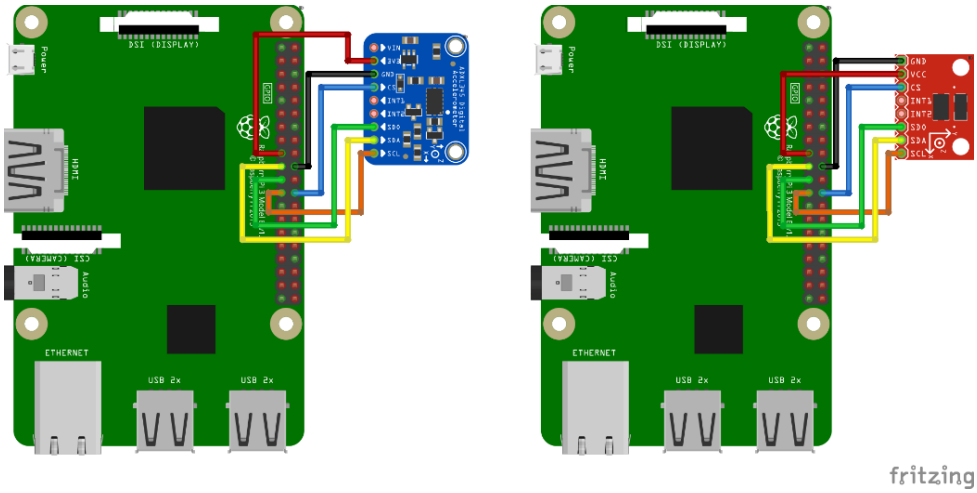
Klipper hat eine eingebaute Unterstützung für den Beschleunigungssensor ADXL345, mit dem die Resonanzfrequenzen des Druckers für verschiedene Achsen gemessen und die input shapers automatisch eingestellt werden können, um die Resonanzen zu kompensieren. Beachten Sie, dass die Verwendung des ADXL345 einige Löt- und Crimparbeiten erfordert. ADXL345 kann direkt an einen Raspberry Pi oder an eine SPI-Schnittstelle einer MCU-Platine angeschlossen werden (sie muss einigermaßen schnell sein).
Bei der Beschaffung des ADXL345 ist zu beachten, dass es eine Vielzahl von verschiedenen Platinen-Designs und Klonen davon gibt. Vergewissern Sie sich, dass die Platine den SPI-Modus unterstützt (eine kleine Anzahl von Platinen scheint fest für I2C konfiguriert zu sein, indem SDO auf GND gezogen wird), und wenn sie an eine 5-V-Drucker-MCU angeschlossen werden soll, muss sie über einen Spannungsregler und einen Level-Shifter verfügen.

Installationsanweisungen

Verdrahtung

Sie müssen den ADXL345 über SPI mit Ihrem Raspberry Pi verbinden. Beachten Sie, dass die I2C-Verbindung, die in der ADXL345-Dokumentation vorgeschlagen wird, einen zu geringen Durchsatz hat und nicht funktionieren wird. Das empfohlene Anschlussschema :

ADXL345 pin	RPi pin	RPi pin name
3V3 (or VCC)	01	3.3v DC power
GND	06	Ground
CS	24	GPIO08 (SPI0_CE0_N)
SDO	21	GPIO09 (SPI0_MISO)
SDA	19	GPIO10 (SPI0_MOSI)
SCL	23	GPIO11 (SPI0_SCLK)

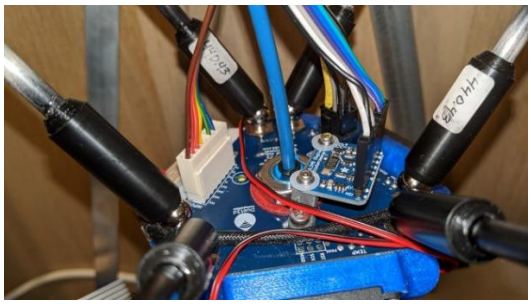


Fritzing-Schaltpläne für einige der ADXL345-Boards :

Überprüfen Sie Ihre Verdrahtung, bevor Sie den Raspberry Pi einschalten, um eine Beschädigung des Pi oder des Beschleunigungssensors zu vermeiden.

Montage des Beschleunigungssensors

Der Beschleunigungsaufnehmer muss am Toolhead befestigt werden. Man muss eine geeignete Halterung entwerfen, die zum eigenen 3D-Drucker passt. Es ist besser, die Achsen des Beschleunigungssensors mit den Achsen des Druckers auszurichten (aber wenn es bequemer ist, können die Achsen vertauscht werden - d.h. es ist nicht notwendig, die X-Achse mit der X-Achse auszurichten usw. - es sollte auch in Ordnung sein, wenn die Z-Achse des Beschleunigungssensors die X-Achse des Druckers ist usw.).



Ein Beispiel für die Montage des ADXL345 auf dem SmartEffector :

Beachten Sie, dass man bei einem Bettschwingungsdrucker zwei Halterungen konstruieren muss: eine für den Werkzeugkopf und eine für das Bett, und die Messungen zweimal durchführen muss. Siehe den entsprechenden Abschnitt section für weitere Details.

Achtung: Stellen Sie sicher, dass der Beschleunigungsaufnehmer und die Schrauben, mit denen er befestigt ist, keine Metallteile des Druckers berühren. Grundsätzlich muss die Halterung so konstruiert sein, dass die elektrische Isolierung des Aufnehmers vom Druckerrahmen gewährleistet ist. Wird dies nicht sichergestellt, kann eine Erdungsschleife im System entstehen, die die Elektronik beschädigen kann.

Softwareinstallation
Beachten Sie, dass für Resonanzmessungen und die automatische Kalibrierung des Shapers zusätzliche Softwareabhängigkeiten erforderlich sind, die nicht standardmäßig installiert sind. Zunächst müssen Sie auf Ihrem Raspberry Pi den folgenden Befehl ausführen:

```
~/klippy-env/bin/pip install -v numpy
```

um das numpy-Paket zu installieren. Beachten Sie, dass dies je nach Leistung der CPU sehr lange dauern kann, bis zu 10-20 Minuten. Seien Sie geduldig und warten Sie auf den Abschluss der Installation. In einigen Fällen, wenn das Board zu wenig RAM hat, kann die Installation fehlschlagen und Sie müssen den Swap aktivieren.
Führen Sie anschließend die folgenden Befehle aus, um die zusätzlichen Abhängigkeiten zu installieren:

```
sudo apt update
sudo apt install python3-numpy python3-matplotlib
```

Danach folgen Sie den Anweisungen im RPi Microcontroller Dokument, um die "linux mcu" auf dem Raspberry Pi RPi Microcontroller document einzurichten.
Vergewissern Sie sich, dass der Linux-SPI-Treiber aktiviert ist, indem Sie sudo raspi-config ausführen und SPI im Menü "Interfacing options" aktivieren.
Fügen Sie Folgendes in die Datei printer.cfg ein:

```
[mcu rpi]
serial: /tmp/klipper_host_mcu

[adxl345]
cs_pin: rpi:None

[resonance_tester]
akzel_chip: adxl345
probe_points:
    100, 100, 20 # ein Beispiel
```

Es wird empfohlen, mit 1 Probepunkt zu beginnen, in der Mitte des Druckbetts, leicht darüber.
Starten Sie Klipper mit dem Befehl `RESTART` neu.

Messung der Resonanzen

Überprüfen des Aufbaus
Jetzt können Sie eine Verbindung testen.

- Für "non bed-slingers" (z.B. ein Beschleunigungsaufnehmer), geben Sie in Octoprint `ACCELEROMETER_QUERY` ein
- Bei "Bettschlingern" (z.B. mehr als ein Beschleunigungssensor) geben Sie `ACCELEROMETER_QUERY CHIP=<chip>` ein, wobei `<chip>` der Name des Chips ist, wie er eingegeben wurde, z.B. CHIP=bed (siehe: Bettschlinger bed-slinger) für alle installierten Beschleunigungssensor-Chips.

Sie sollten die aktuellen Messungen des Beschleunigungsmessers sehen, einschließlich der Beschleunigung im freien Fall, z.B.

```
Recv: // adxl345 Werte (x, y, z): 470.719200, 941.438400, 9728.196800
```

Wenn Sie eine Fehlermeldung wie Invalid adxl345 id (got xx vs e5) erhalten, wobei xx eine andere ID ist, ist dies ein Hinweis auf ein Verbindungsproblem mit dem ADXL345 oder einen fehlerhaften Sensor. Überprüfen Sie die Stromversorgung, die Verdrahtung (ob sie mit den Schaltplänen übereinstimmt, kein Draht gebrochen oder lose ist usw.) und die Qualität der Lötung.
Versuchen Sie als Nächstes, MEASURE_AXES_NOISE in Octoprint auszuführen. Sie sollten einige Basiswerte für das Rauschen des Beschleunigungssensors auf den Achsen erhalten (es sollte irgendwo im Bereich von ~1-100 liegen). Zu hohe Achsengeräusche (z. B. 1000 und mehr) können auf Probleme mit dem Sensor, seiner Leistung oder zu laute, unausgewogene Lüfter in einem 3D-Drucker hinweisen.

Messung der Resonanzen
Jetzt können Sie einige praktische Tests durchführen. Führen Sie den folgenden Befehl aus:

```
TEST_RESONANCES AXIS=X
```

Beachten Sie, dass dadurch Schwingungen auf der X-Achse erzeugt werden. Außerdem wird die Eingangsformung deaktiviert, falls sie zuvor aktiviert war, da es nicht zulässig ist, die Resonanztests bei aktivierter Eingangsformung durchzuführen.

Achtung! Beobachten Sie den Drucker beim ersten Mal, um sicherzustellen, dass die Vibrationen nicht zu stark werden (mit dem Befehl `M112` kann der Test im Notfall abgebrochen werden; hoffentlich wird es nicht dazu kommen). Wenn die Vibrationen zu stark werden, können Sie versuchen, einen niedrigeren als den Standardwert für den Parameter `accel_per_hz` im Abschnitt `[resonance_tester]` anzugeben, z. B.

```
[resonance_tester]
accel_chip: adxl345
accel_per_hz: 50 # Standardwert ist 75
probe_points: ...
```

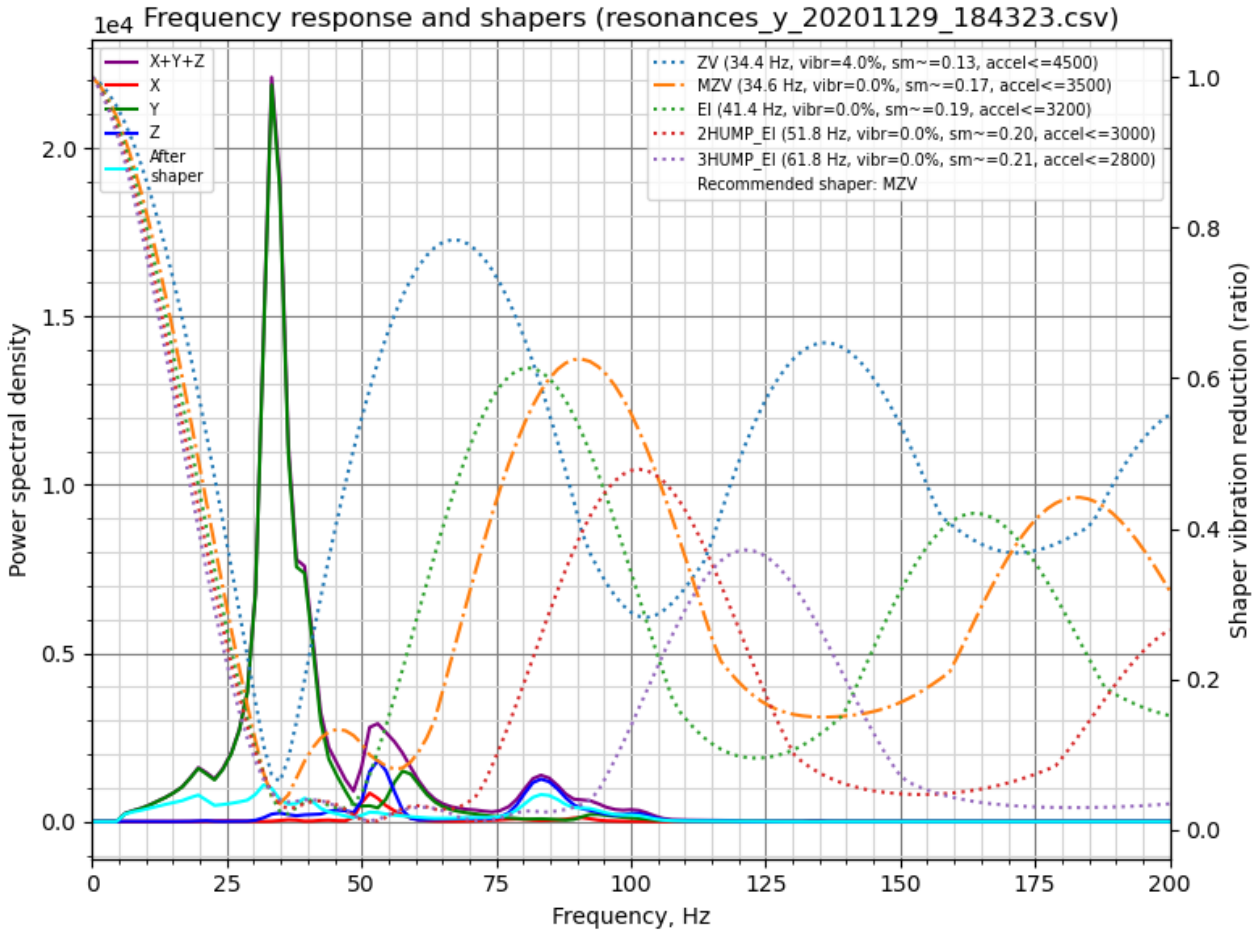
Wenn es für die X-Achse funktioniert, führen Sie es auch für die Y-Achse durch:

```
TEST_RESONANCES AXIS=Y
```

Dies erzeugt 2 CSV-Dateien (`/tmp/resonances_x_*.csv und /tmp/resonances_y_*.csv`). Diese Dateien können mit dem Standalone-Skript auf einem Raspberry Pi verarbeitet werden. Dazu führen Sie die folgenden Befehle aus:

```
~/klipper/scripts/calibrate_shaper.py /tmp/resonances_x_*.csv -o /tmp/shaper_calibrate_x.png
~/klipper/scripts/calibrate_shaper.py /tmp/resonances_y_*.csv -o /tmp/shaper_calibrate_y.png
```

Dieses Skript erzeugt die Diagramme /tmp/shaper_calibrate_x.png und /tmp/shaper_calibrate_y.png mit den Frequenzgängen. Sie erhalten auch die vorgeschlagenen Frequenzen für jeden Eingangsshaper sowie den empfohlenen Eingangsshaper für Ihr Setup. Zum Beispiel:



Resonanzen

```
Fitted shaper 'zv' frequency = 34.4 Hz (vibrations = 4.0%, smoothing ~= 0.132)
To avoid too much smoothing with 'zv', suggested max_accel <= 4500 mm/sec^2
Fitted shaper 'mzv' frequency = 34.6 Hz (vibrations = 0.0%, smoothing ~= 0.170)
To avoid too much smoothing with 'mzv', suggested max_accel <= 3500 mm/sec^2
Fitted shaper 'ei' frequency = 41.4 Hz (vibrations = 0.0%, smoothing ~= 0.188)
To avoid too much smoothing with 'ei', suggested max_accel <= 3200 mm/sec^2
Fitted shaper '2hump_ei' frequency = 51.8 Hz (vibrations = 0.0%, smoothing ~= 0.201)
To avoid too much smoothing with '2hump_ei', suggested max_accel <= 3000 mm/sec^2
Fitted shaper '3hump_ei' frequency = 61.8 Hz (vibrations = 0.0%, smoothing ~= 0.215)
To avoid too much smoothing with '3hump_ei', suggested max_accel <= 2800 mm/sec^2
Recommended shaper is mzv @ 34.6 Hz
```

Die vorgeschlagene Konfiguration kann dem Abschnitt [input_shaper] der Datei printer.cfg hinzugefügt werden, z. B:

```
[input_shaper]
shaper_freq_x: ...
shaper_type_x: ...
shaper_freq_y: 34.6
shaper_type_y: mzv

[printer]
max_accel: 3000 # sollte die geschätzte max_accel für X- und Y-Achse nicht überschreiten
```

Sie können aber auch eine andere Konfiguration wählen, die auf den erzeugten Diagrammen basiert: Spitzen in der spektralen Leistungsdichte auf den Diagrammen entsprechen den Resonanzfrequenzen des Druckers.
Beachten Sie, dass Sie alternativ die Autokalibrierung des Eingangsformers direkt directly von Klipper aus starten können, was z.B. für die Neukalibrierung re-calibration des Eingangsformers praktisch sein kann.

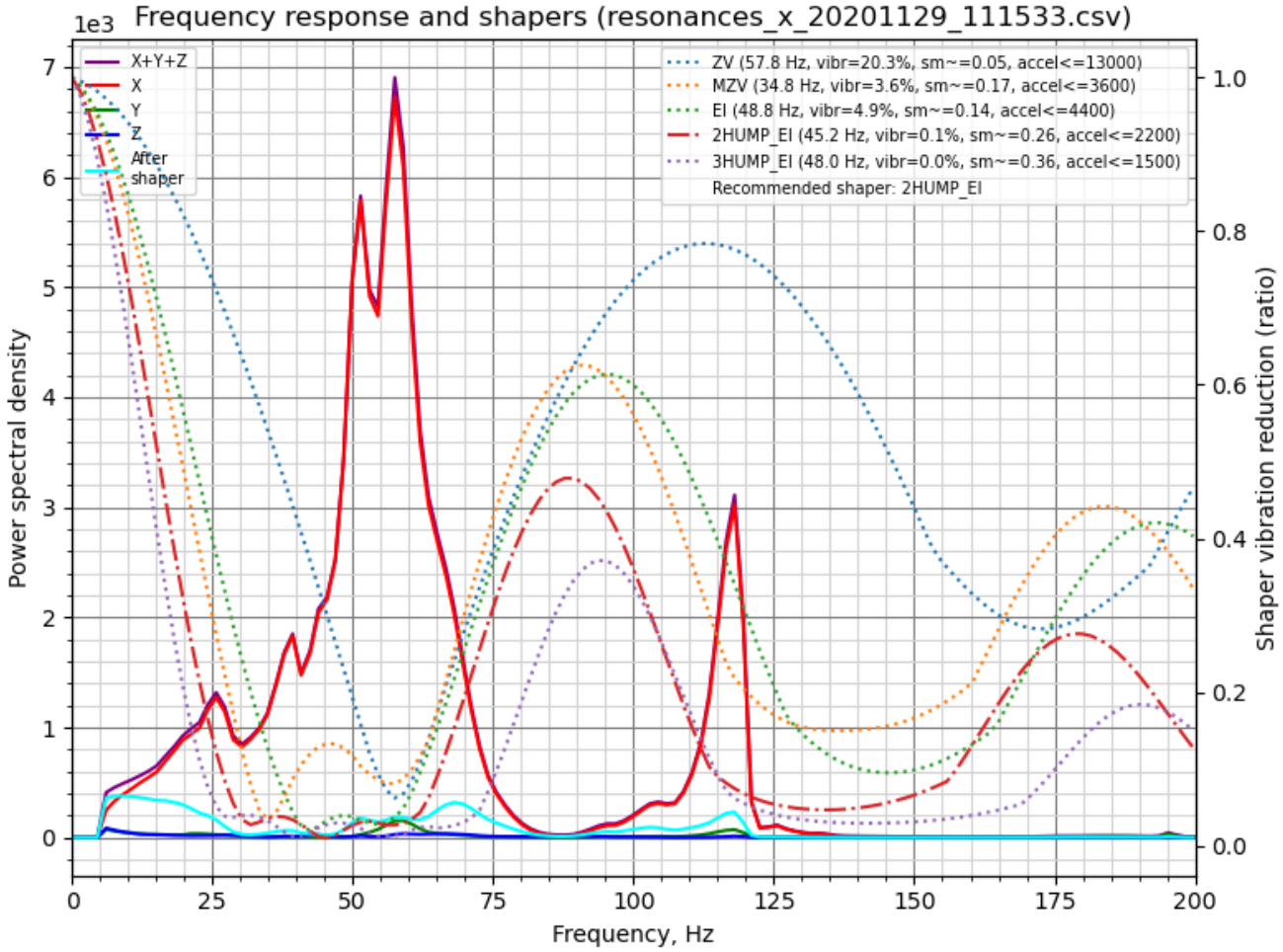
Bed-Slinger-Drucker
Wenn es sich bei Ihrem Drucker um einen Bed-Slinger-Drucker handelt, müssen Sie die Position des Beschleunigungsaufnehmers zwischen den Messungen für die X- und Y-Achse ändern: Messen Sie die Resonanzen der X-Achse mit dem Beschleunigungsaufnehmer, der am Werkzeugkopf angebracht ist, und die Resonanzen der Y-Achse - am Bett (die übliche Bed-Slinger-Einrichtung).
Sie können jedoch auch zwei Beschleunigungsmesser gleichzeitig anschließen, allerdings müssen sie an verschiedene Boards angeschlossen werden (z. B. an ein RPi und ein Drucker-MCU-Board) oder an zwei verschiedene physikalische SPI-Schnittstellen auf demselben Board (selten verfügbar). Dann können sie auf die folgende Weise konfiguriert werden:

```
[adxl345 hotend]
# Angenommen, der "hotend"-Chip ist mit einem RPi verbunden
cs_pin: rpi:None

[adxl345 bed]
# Angenommen, der `bed`-Chip ist an ein Drucker-MCU-Board angeschlossen
cs_pin: ...  # SPI-Chip-Select-Pin (CS) der Druckerplatine

[resonance_tester]
# Angenommen, der typische Aufbau des Bed-Slinger-Druckers
accel_chip_x: adxl345 hotend
accel_chip_y: adxl345 Bett
probe_points: ...
```

Dann werden die Befehle `TEST_RESONANCES AXIS=X` und `TEST_RESONANCES AXIS=Y` den richtigen Beschleunigungsmesser für jede Achse verwenden.

Maximale Glättung
Beachten Sie, dass der Input Shaper teilweise eine gewisse Glättung erzeugen kann. Die automatische Abstimmung des Eingangs-Shapers, die durch das Skript `calibrate_shaper.py` oder den Befehl `SHAPER_CALIBRATE` durchgeführt wird, versucht, die Glättung nicht zu verstärken, aber gleichzeitig die resultierenden Vibrationen zu minimieren. Manchmal wird die Shaper-Frequenz nicht optimal gewählt, oder Sie bevorzugen einfach eine geringere Glättung in Teilen auf Kosten einer größeren Restschwingung. In diesen Fällen können Sie eine Begrenzung der maximalen Glättung durch den Input-Shaper verlangen.

Betrachten wir nun die folgenden Ergebnisse der automatischen Abstimmung:



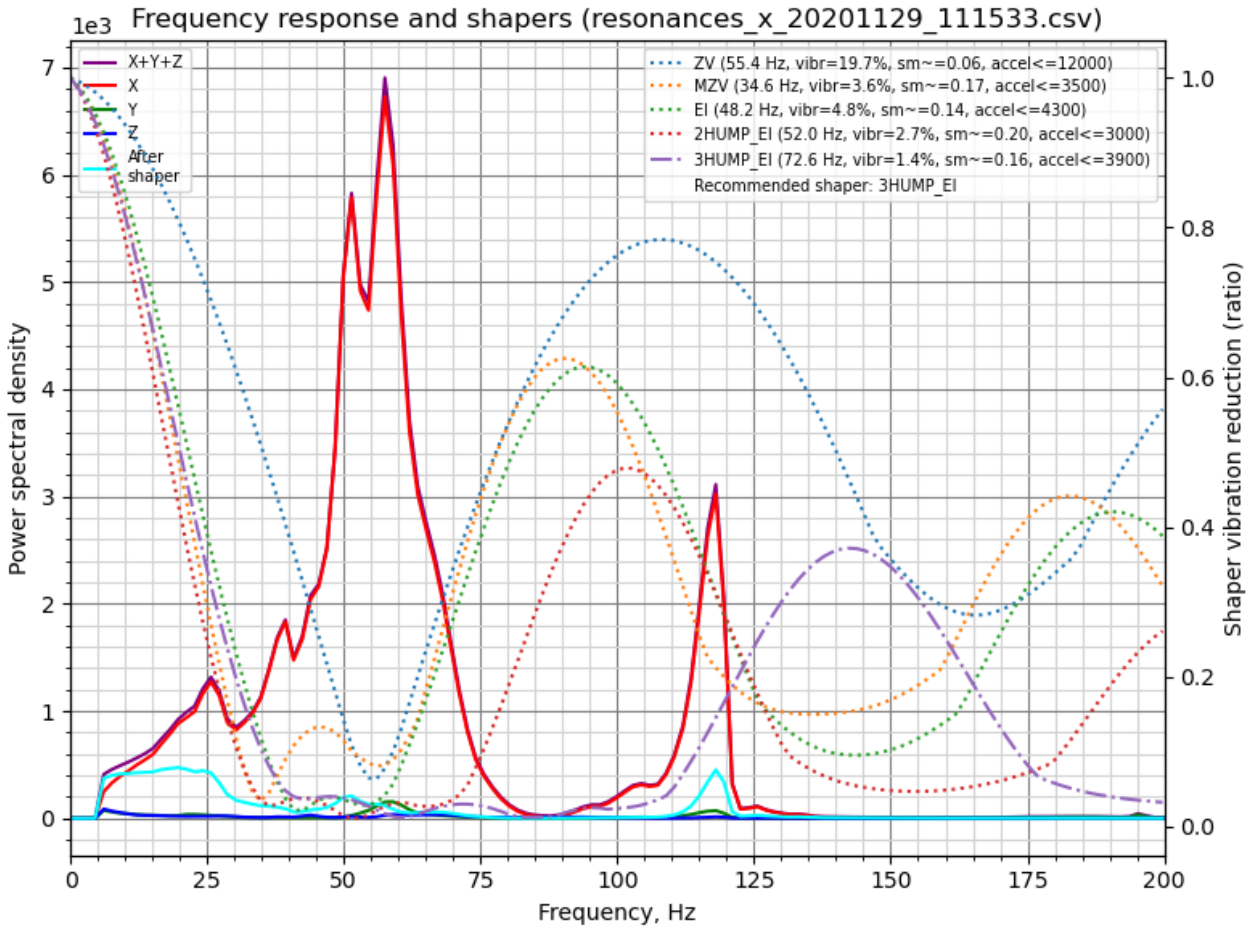

```
Angepasste Shaper 'zv' Frequenz = 57.8 Hz (Schwingungen = 20.3%, Glättung ~= 0.053)
Um eine zu starke Glättung mit 'zv' zu vermeiden, wird max_accel <= 13000 mm/sec^2 vorgeschlagen
Angepasste Shaper-Frequenz 'mzv' = 34,8 Hz (Schwingungen = 3,6%, Glättung ~= 0,168)

Um eine zu starke Glättung mit 'mzv' zu vermeiden, wird max_accel <= 3600 mm/sec^2 vorgeschlagen
Angepasste Shaper-Frequenz 'ei' = 48,8 Hz (Schwingungen = 4,9%, Glättung ~= 0,135)
Um eine zu starke Glättung mit 'ei' zu vermeiden, wird max_accel <= 4400 mm/sec^2 vorgeschlagen
Angepasster Shaper '2hump_ei' Frequenz = 45,2 Hz (Schwingungen = 0,1%, Glättung ~= 0,264)
Um eine zu starke Glättung mit '2hump_ei' zu vermeiden, wird max_accel <= 2200 mm/sec^2
vorgeschlagen
Angepasster Shaper '3hump_ei' Frequenz = 48.0 Hz (Vibrationen = 0.0%, Glättung ~= 0.356)
Um eine zu starke Glättung mit '3hump_ei' zu vermeiden, wird max_accel <= 1500 mm/sec^2
vorgeschlagen.
Empfohlener Shaper ist 2hump_ei @ 45,2 Hz
```

Beachten Sie, dass es sich bei den angegebenen Glättungswerten um abstrakte Projektionswerte handelt. Diese Werte können zum Vergleich verschiedener Konfigurationen verwendet werden: je höher der Wert, desto mehr Glättung erzeugt ein Shaper. Diese Glättungswerte stellen jedoch kein wirkliches Maß für die Glättung dar, da die tatsächliche Glättung von den Parametern `max_accel`und `square_corner_velocity` abhängt. Daher sollten Sie einige Testdrucke machen, um zu sehen, wie viel Glättung die gewählte Konfiguration tatsächlich erzeugt.

Im obigen Beispiel sind die vorgeschlagenen Shaper-Parameter nicht schlecht, aber was ist, wenn Sie eine geringere Glättung auf der X-Achse erreichen wollen? Sie können versuchen, die maximale Shaper-Glättung mit dem folgenden Befehl zu begrenzen:

```
~/klipper/scripts/calibrate_shaper.py /tmp/resonances_x_*.csv -o /tmp/shaper_calibrate_x.png --
max_smoothing=0.2
```

wodurch die Glättung auf 0,2 Punkte begrenzt wird. Nun erhalten Sie das folgende Ergebnis:



Resonanzen

```
Angepasste Shaper 'zv' Frequenz = 55.4 Hz (Schwingungen = 19.7%, Glättung ~= 0.057)
Um eine zu starke Glättung mit 'zv' zu vermeiden, wird max_accel <= 12000 mm/sec^2 vorgeschlagen
Angepasste Shaper-Frequenz 'mzv' = 34,6 Hz (Schwingungen = 3,6%, Glättung ~= 0,170)
Um eine zu starke Glättung mit 'mzv' zu vermeiden, wird max_accel <= 3500 mm/sec^2 vorgeschlagen
Angepasste Shaper-Frequenz 'ei' = 48,2 Hz (Schwingungen = 4,8%, Glättung ~= 0,139)
Um eine zu starke Glättung mit 'ei' zu vermeiden, wird max_accel <= 4300 mm/sec^2 vorgeschlagen
Angepasster Shaper '2hump_ei' Frequenz = 52,0 Hz (Schwingungen = 2,7%, Glättung ~= 0,200)
Um eine zu starke Glättung mit '2hump_ei' zu vermeiden, wird max_accel <= 3000 mm/sec^2
vorgeschlagen
Angepasster Shaper '3hump_ei' Frequenz = 72,6 Hz (Schwingungen = 1,4%, Glättung ~= 0,155)
Um eine zu starke Glättung mit '3hump_ei' zu vermeiden, wird max_accel <= 3900 mm/sec^2
vorgeschlagen.
Empfohlener Shaper ist 3hump_ei @ 72.6 Hz
```

Im Vergleich zu den zuvor vorgeschlagenen Parametern sind die Schwingungen etwas größer, aber die Glättung ist deutlich geringer als zuvor, was eine größere maximale Beschleunigung ermöglicht.

Bei der Entscheidung, welchen max_smoothing-Parameter Sie wählen sollen, können Sie einen Versuch-und-Irrtum-Ansatz verwenden. Probieren Sie verschiedene Werte aus und sehen Sie, welche Ergebnisse Sie erhalten. Beachten Sie, dass die tatsächliche Glättung, die der Input Shaper erzeugt, in erster Linie von der niedrigsten Resonanzfrequenz des Druckers abhängt: je höher die Frequenz der niedrigsten Resonanz, desto geringer die Glättung. Wenn Sie also das Skript auffordern, eine Konfiguration des Eingabeformers mit einer unrealistisch kleinen Glättung zu finden, geht dies auf Kosten eines verstärkten Klingelns bei den niedrigsten Resonanzfrequenzen (die in der Regel auch in den Ausdrucken deutlicher sichtbar sind). Überprüfen Sie daher immer die vom Skript gemeldeten projizierten Restschwingungen und stellen Sie sicher, dass sie nicht zu hoch sind.

Wenn Sie einen guten max_smoothing-Wert für beide Achsen gewählt haben, können Sie ihn in der `printer.cfg` als

```
[resonance_tester]
accel_chip: ...
probe_points: ...
max_smoothing: 0.25 # ein Beispiel
```

Wenn Sie dann in Zukunft die automatische Abstimmung des Eingangsformers mit dem Klipper-Befehl SHAPER_CALIBRATE erneut rerun ausführen, wird der gespeicherte Wert für `max_smoothing` als Referenz verwendet.

Auswählen von max_accel
Da der Input-Shaper eine gewisse Glättung in den Teilen erzeugen kann, insbesondere bei hohen Beschleunigungen, müssen Sie den Wert für `max_accel` so wählen, dass die gedruckten Teile nicht zu sehr geglättet werden. Ein Kalibrierungsskript liefert eine Schätzung für den `max_accel`-Parameter, der nicht zu viel Glättung erzeugen sollte. Beachten Sie, dass der vom Kalibrierungsskript angezeigte Wert `max_accel` nur ein theoretisches Maximum ist, bei dem der jeweilige Shaper noch arbeiten kann, ohne zu viel Glättung zu erzeugen. Es ist keineswegs eine Empfehlung, diese Beschleunigung für den Druck einzustellen. Die maximale Beschleunigung, die Ihr Drucker aushalten kann, hängt von seinen mechanischen Eigenschaften und dem maximalen Drehmoment der verwendeten Schrittmotoren ab. Daher wird empfohlen, max_accel im Abschnitt [printer] so einzustellen, dass die geschätzten Werte für die X- und Y-Achsen nicht überschritten werden, wahrscheinlich mit einer gewissen konservativen Sicherheitsspanne.

Alternativ können Sie diesen this Teil des Leitfadens für die Einstellung des Eingangsformers befolgen und das Testmodell drucken, um den Parameter max_accel experimentell zu bestimmen.

Der gleiche Hinweis gilt für die auto-calibrationdes Eingangs-Shapers mit dem Befehl `SHAPER_CALIBRATE:` Es ist immer noch notwendig, den richtigen max_accel-Wert nach der Auto-Kalibrierung zu wählen, und die vorgeschlagenen Beschleunigungsgrenzen werden nicht automatisch angewendet.

Wenn Sie eine Neukalibrierung des Shapers durchführen und die gemeldete Glättung für die vorgeschlagene Shaper-Konfiguration fast dieselbe ist wie die der vorherigen Kalibrierung, kann dieser Schritt übersprungen werden.

Testen benutzerdefinierter Achsen
Der Befehl `TEST_RESONANCES` unterstützt benutzerdefinierte Achsen. Dieser Befehl ist zwar für die Kalibrierung des Eingangsformers nicht wirklich nützlich, kann aber verwendet werden, um die Resonanzen des Druckers eingehend zu untersuchen und z. B. die Riemenspannung zu überprüfen.

Um die Riemenspannung bei CoreXY-Druckern zu überprüfen, führen Sie den Befehl

```
TEST_RESONANCES AXIS=1,1 OUTPUT=raw_data
TEST_RESONANCES AXIS=1,-1 OUTPUT=raw_data
```

und verwenden Sie `graph_accelerometer.py`, um die erzeugten Dateien zu verarbeiten, z. B.

```
~/klipper/scripts/graph_accelerometer.py -c /tmp/raw_data_axis*.csv -o /tmp/resonances.png
```

Dadurch wird `/tmp/resonances.png` erzeugt, in dem die Resonanzen verglichen werden.

Für Delta-Drucker mit der Standard-Turm-Positionierung (Turm A ~= 210 Grad, B ~= 330 Grad und C ~= 90 Grad), führen Sie

```
TEST_RESONANZEN AXIS=0,1 OUTPUT=raw_data
TEST_RESONANZEN AXIS=-0.866025404,-0.5 OUTPUT=raw_data
TEST_RESONANZEN AXIS=0.866025404,-0.5 OUTPUT=raw_data
```

und dann den gleichen Befehl verwenden

```
~/klipper/scripts/graph_accelerometer.py -c /tmp/raw_data_axis*.csv -o /tmp/resonances.png
```

um `/tmp/resonances.png` zu erzeugen und die Resonanzen zu vergleichen.

Auto-Kalibrierung des Input Shapers

Neben der manuellen Auswahl der geeigneten Parameter für die Input-Shaper-Funktion ist es auch möglich, die Autoabstimmung für den Input-Shaper direkt von Klipper aus durchzuführen. Führen Sie den folgenden Befehl über das Octoprint-Terminal aus:

```
SHAPER_CALIBRATE
```

Dies führt den vollständigen Test für beide Achsen durch und erzeugt die csv-Ausgabe (standardmäßig /tmp/calibration_data_*.csv) für den Frequenzgang und die vorgeschlagenen Eingangs-Shaper. Außerdem erhalten Sie auf der Octoprint-Konsole die vorgeschlagenen Frequenzen für jeden Eingangs-Shaper sowie den für Ihr Setup empfohlenen Eingangs-Shaper. Ein Beispiel:

```
Calculating the best input shaper parameters for y axis
Fitted shaper 'zv' frequency = 39.0 Hz (vibrations = 13.2%, smoothing ~= 0.105)
To avoid too much smoothing with 'zv', suggested max_accel <= 5900 mm/sec^2
Fitted shaper 'mzv' frequency = 36.8 Hz (vibrations = 1.7%, smoothing ~= 0.150)
To avoid too much smoothing with 'mzv', suggested max_accel <= 4000 mm/sec^2
Fitted shaper 'ei' frequency = 36.6 Hz (vibrations = 2.2%, smoothing ~= 0.240)
To avoid too much smoothing with 'ei', suggested max_accel <= 2500 mm/sec^2
Fitted shaper '2hump_ei' frequency = 48.0 Hz (vibrations = 0.0%, smoothing ~= 0.234)
To avoid too much smoothing with '2hump_ei', suggested max_accel <= 2500 mm/sec^2
```

```
Fitted shaper '3hump_ei' frequency = 59.0 Hz (vibrations = 0.0%, smoothing ~= 0.235)
To avoid too much smoothing with '3hump_ei', suggested max_accel <= 2500 mm/sec^2
Recommended shaper_type_y = mzv, shaper_freq_y = 36.8 Hz
```

Wenn Sie mit den vorgeschlagenen Parametern einverstanden sind, können Sie jetzt SAVE_CONFIG ausführen, um sie zu speichern und den Klipper neu zu starten. Beachten Sie, dass dadurch der Wert max_accel im Abschnitt [printer] nicht aktualisiert wird. Sie sollten ihn manuell aktualisieren, indem Sie die Überlegungen im Abschnitt Auswahl von Selecting max_accel befolgen.

Wenn es sich bei Ihrem Drucker um einen Bettschlingendrucker handelt, können Sie angeben, welche Achse getestet werden soll, so dass Sie den Befestigungspunkt des Beschleunigungsmessers zwischen den Tests ändern können (standardmäßig wird der Test für beide Achsen durchgeführt):

```
SHAPER_CALIBRATE AXIS=Y
```

Sie können SAVE_CONFIG zweimal ausführen - nach der Kalibrierung jeder Achse.

Wenn Sie jedoch zwei Beschleunigungsaufnehmer gleichzeitig angeschlossen haben, führen Sie einfach SHAPER_CALIBRATE ohne Angabe einer Achse aus, um den Input Shaper für beide Achsen in einem Durchgang zu kalibrieren.

Neukalibrierung des Input Shapers
Der Befehl SHAPER_CALIBRATE kann auch verwendet werden, um den Eingangs-Shaper in Zukunft neu zu kalibrieren, insbesondere wenn Änderungen am Drucker vorgenommen werden, die sich auf seine Kinematik auswirken können. Man kann entweder die vollständige Kalibrierung mit dem Befehl SHAPER_CALIBRATE erneut durchführen oder die automatische Kalibrierung auf eine einzelne Achse beschränken, indem man den Parameter AXIS= angibt, wie

```
SHAPER_CALIBRATE AXIS=X
```

Warnung! Es ist nicht ratsam, die Autokalibrierung des Shapers sehr häufig auszuführen (z.B. vor jedem Druck oder jeden Tag). Um Resonanzfrequenzen zu bestimmen, erzeugt die Autokalibrierung intensive Vibrationen auf jeder der Achsen. Im Allgemeinen sind 3D-Drucker nicht dafür ausgelegt, über einen längeren Zeitraum Vibrationen in der Nähe der Resonanzfrequenzen ausgesetzt zu sein. Dies kann den Verschleiß der Druckerkomponenten erhöhen und ihre Lebensdauer verkürzen. Es besteht auch ein erhöhtes Risiko, dass sich einige Teile lösen oder locker werden. Vergewissern Sie sich nach jedem Autotuning, dass alle Teile des Druckers (einschließlich derjenigen, die sich normalerweise nicht bewegen) sicher befestigt sind.

Außerdem ist es möglich, dass die Ergebnisse der Selbstoptimierung aufgrund von Messfehlern von einem Kalibrierungslauf zum anderen leicht abweichen. Es ist jedoch nicht zu erwarten, dass das Rauschen die Druckqualität allzu sehr beeinträchtigt. Es ist jedoch ratsam, die vorgeschlagenen Parameter zu überprüfen und einige Testdrucke zu erstellen, bevor Sie sie verwenden, um sicherzustellen, dass sie gut sind.

Offline-Verarbeitung der Daten der Beschleunigungsmesser

Es ist möglich, die Rohdaten des Beschleunigungsmessers zu generieren und offline zu verarbeiten (z. B. auf einem Host-Rechner), um z. B. Resonanzen zu finden. Führen Sie dazu die folgenden Befehle über das Octoprint-Terminal aus:

```
SET_INPUT_SHAPER SHAPER_FREQ_X=0 SHAPER_FREQ_Y=0
TEST_RESONANCES AXIS=X OUTPUT=raw_data
```

wobei eventuelle Fehler für den Befehl SET_INPUT_SHAPER ignoriert werden. Für den Befehl TEST_RESONANCES geben Sie die gewünschte Testachse an. Die Rohdaten werden in das Verzeichnis /tmp auf dem RPi geschrieben.
Die Rohdaten können auch durch zweimaliges Ausführen des Befehls ACCELEROMETER_MEASURE während einer normalen Druckertätigkeit erhalten werden - zuerst, um die Messungen zu starten, und dann, um sie zu stoppen und die Ausgabedatei zu schreiben. Weitere Einzelheiten finden Sie unter G-Codes.

Die Daten können später mit den folgenden Skripten verarbeitet werden: `scripts/graph_accelerometer.py` und `scripts/calibrate_shaper.py`. Beide akzeptieren je nach Modus eine oder mehrere csv-Rohdateien als Eingabe. Das Skript graph_accelerometer.py unterstützt mehrere Betriebsmodi:

- Plotten von Rohdaten des Beschleunigungsmessers (mit dem Parameter `-r`), nur 1 Eingabe wird unterstützt;
- Zeichnen eines Frequenzgangs (keine zusätzlichen Parameter erforderlich), wenn mehrere Eingänge angegeben sind, wird der durchschnittliche Frequenzgang berechnet;
- Vergleich des Frequenzgangs zwischen mehreren Eingängen (mit dem Parameter `-c`); Sie können zusätzlich angeben, welche Achse des Beschleunigungsaufnehmers berücksichtigt werden soll (mit den Parametern `-a x`, `-a y` oder `-a z`) (wenn kein Parameter angegeben ist, wird die Summe der Schwingungen aller Achsen verwendet);
- Zeichnen des Spektrogramms (mit dem Parameter -s), es wird nur 1 Eingang unterstützt; Sie können zusätzlich über die Parameter `-a x`, `-a y` oder `-a z` angeben, welche Achse des Beschleunigungsmessers berücksichtigt werden soll (wird keiner angegeben, wird die Summe der Schwingungen für alle Achsen verwendet).

Beachten Sie, dass das Skript graph_accelerometer.py nur die Dateien raw_data*.csv unterstützt und nicht die Dateien resonances*.csv oder calibration_data*.csv.

Zum Beispiel,

```
~/klipper/scripts/graph_accelerometer.py /tmp/raw_data_x_*.csv -o /tmp/resonances_x.png -c -a z
```

stellt den Vergleich mehrerer /tmp/raw_data_x_*.csv-Dateien für die Z-Achse mit der /tmp/resonances_x.png-Datei dar.

Das Skript shaper_calibrate.py akzeptiert 1 oder mehrere Eingaben und kann eine automatische Abstimmung des Eingangs-Shapers durchführen und die besten Parameter vorschlagen, die für alle angegebenen Eingaben gut funktionieren. Es gibt die vorgeschlagenen Parameter auf der Konsole aus und kann zusätzlich das Diagramm erzeugen, wenn der Parameter `-o output.png` angegeben wird, oder die CSV-Datei, wenn der Parameter -c `output.csv` angegeben wird.
Die Bereitstellung mehrerer Eingaben für das Skript shaper_calibrate.py kann nützlich sein, wenn z.B. ein fortgeschrittenes Tuning der Input-Shaper durchgeführt werden soll:

- Zweimaliges Ausführen von `TEST_RESONANCES AXIS=X OUTPUT=raw_data` (und Y-Achse) für eine einzelne Achse auf einem Bettschleuderdrucker, wobei der Beschleunigungsaufnehmer beim ersten Mal am Werkzeugkopf und beim zweiten Mal am Bett angebracht ist, um Achsenkreuzresonanzen zu erkennen und zu versuchen, sie mit den Eingangs-Shapern zu beseitigen.
- Zweimalige Durchführung von `TEST_RESONANZEN AXIS=Y OUTPUT=raw_data` auf einer Bettschleuder mit einem Glasbett und einer magnetischen Oberfläche (die leichter ist), um die Parameter für die Eingangsformung zu finden, die für jede Druckoberflächenkonfiguration gut funktionieren.
- Kombinieren der Resonanzdaten von mehreren Testpunkten.
- Kombination der Resonanzdaten von zwei Achsen (z. B. bei einem Bed-Slinger-Drucker zur Konfiguration des X-Achsen-Eingangsformers aus den Resonanzen der X- und Y-Achse, um die Schwingungen des Bettes zu kompensieren, falls die Düse bei der Bewegung in Richtung der X-Achse einen Druck "einfängt").

Pressure Advance (Druckvorschub)

Dieses Dokument enthält Informationen zur Einstellung der Konfigurationsvariablen "Druckvorschub" für eine bestimmte Düse und ein bestimmtes Filament. Die Druckvorverlegung kann bei der Verringerung von Schlamm hilfreich sein. Weitere Informationen darüber, wie der Druckvorschub implementiert wird, finden Sie im Dokument Kinematik kinematics.

Abstimmung des Druckvorschubs

Die Druckvorverlegung hat zwei nützliche Funktionen: Sie reduziert die Verschmutzung bei Bewegungen, bei denen nicht extrudiert wird, und sie reduziert das Blobbing bei Kurvenfahrten. In diesem Leitfaden wird die zweite Funktion (Verringerung des Blobbings bei Kurvenfahrten) als Mechanismus für die Abstimmung verwendet.
Um den Druckvorschub zu kalibrieren, muss der Drucker konfiguriert und betriebsbereit sein, da der Abstimmungstest das Drucken und Prüfen eines Testobjekts beinhaltet. Es ist ratsam, dieses Dokument vor der Durchführung des Tests vollständig zu lesen.
Verwenden Sie einen Slicer, um den G-Code für das große hohle Quadrat in docs/prints/square_tower.stl zu erzeugen. Verwenden Sie eine hohe Geschwindigkeit (z. B. 100 mm/s), keine Füllung und eine grobe Schichthöhe (die Schichthöhe sollte etwa 75 % des Düsendurchmessers betragen). Stellen Sie sicher, dass die "dynamische Beschleunigungskontrolle" im Slicer deaktiviert ist.

Bereiten Sie den Test vor, indem Sie den folgenden G-Code-Befehl eingeben:

```
SET_VELOCITY_LIMIT SQUARE_CORNER_VELOCITY=1 ACCEL=500
```

Dieser Befehl bewirkt, dass die Düse langsamer durch die Ecken fährt, um die Auswirkungen des Extruderdrucks zu verdeutlichen. Bei Druckern mit einem direkt angetriebenen Extruder führen Sie den Befehl aus:

```
TUNING_TOWER COMMAND=SET_PRESSURE_ADVANCE PARAMETER=ADVANCE START=0 FACTOR=.005
```

Für lange Bowden-Extruder verwenden Sie:

```
TUNING_TOWER COMMAND=SET_PRESSURE_ADVANCE PARAMETER=ADVANCE START=0 FACTOR=.020
```

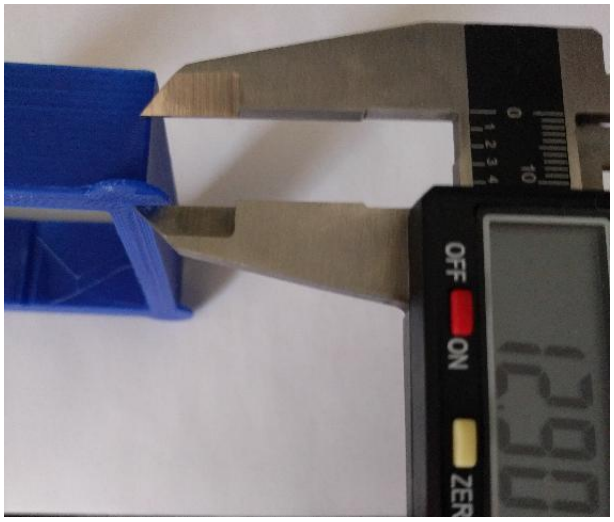
Drucken Sie dann das Objekt. Nach dem Druck sieht der Testdruck wie folgt aus:

tuning_tower

Der obige TUNING_TOWER-Befehl weist Klipper an, die pressure_advance-Einstellung auf jeder Schicht des Drucks zu ändern. Bei höheren Schichten im Druck wird ein größerer Druckvorschubwert eingestellt. Schichten unterhalb der idealen pressure_advance-Einstellung führen zu Blobbing an den Ecken, und Schichten oberhalb der idealen Einstellung können zu abgerundeten Ecken und schlechter Extrusion bis zur Ecke führen.

Man kann den Druck vorzeitig abbrechen, wenn man feststellt, dass die Ecken nicht mehr gut gedruckt werden (und so kann man den Druck von Schichten vermeiden, von denen man weiß, dass sie über dem idealen pressure_advance-Wert liegen).

Prüfen Sie den Druck und verwenden Sie dann einen digitalen Messschieber, um die Höhe zu ermitteln, die die beste Qualität der Ecken aufweist. Im Zweifelsfall sollten Sie eine niedrigere Höhe wählen.



Der Wert von pressure_advance kann dann wie folgt berechnet werden: `pressure_advance = <start> + <gemessene_Höhe> * <Faktor>. (0 + 12,90 * .020` wäre zum Beispiel `.258`).

Es ist möglich, benutzerdefinierte Einstellungen für START und FACTOR zu wählen, wenn dies hilft, die beste Einstellung für den Druckvorlauf zu finden. Achten Sie dabei darauf, dass Sie den Befehl TUNING_TOWER zu Beginn jedes Testdrucks erteilen.

Typische Werte für den Druckvorlauf liegen zwischen 0,050 und 1,000 (der obere Wert wird normalerweise nur bei Bowden-Extrudern verwendet). Wenn bei einem Druckvorschub von bis zu 1.000 keine signifikante Verbesserung eintritt, ist es unwahrscheinlich, dass der Druckvorschub die Qualität der Drucke verbessert. Kehren Sie zu einer Standardkonfiguration zurück, bei der der Druckvorschub deaktiviert ist.

Obwohl diese Optimierungsübung die Qualität der Ecken direkt verbessert, sollte man nicht vergessen, dass eine gute Druckvorschubkonfiguration auch die Schlierenbildung im gesamten Druck reduziert.

Nach Abschluss dieses Tests setzen Sie im Abschnitt [extruder] der Konfigurationsdatei pressure_advance = <calculated_value> und geben einen RESTART-Befehl aus. Der RESTART-Befehl löscht den Teststatus und setzt die Beschleunigungs- und Kurvengeschwindigkeiten auf ihre normalen Werte zurück.

Wichtige Hinweise

Der Druckvorschubwert hängt vom Extruder, der Düse und dem Filament ab. Es ist üblich, dass Filamente von verschiedenen Herstellern oder mit verschiedenen Pigmenten deutlich unterschiedliche Druckvorschubwerte erfordern. Daher sollte der Druckvorschub bei jedem Drucker und bei jeder Filamentspule kalibriert werden.

Drucktemperatur und Extrusionsgeschwindigkeit können sich auf den Druckvorlauf auswirken. Stellen Sie sicher, dass Sie den Rotationsabstand des Extruders extruder rotation_distance und die Düsentemperatur nozzle temperature abstimmen, bevor Sie den Druckvorlauf einstellen.

Der Testdruck ist so konzipiert, dass er mit einer hohen Extruder-Durchflussrate, aber ansonsten mit "normalen" Slicer-Einstellungen läuft. Eine hohe Durchflussrate wird durch eine hohe Druckgeschwindigkeit (z. B. 100 mm/s) und eine grobe Schichthöhe (typischerweise etwa 75 % des Düsendurchmessers) erreicht. Andere Slicer-Einstellungen sollten ähnlich wie die Standardeinstellungen sein (z. B. Perimeter von 2 oder 3 Linien, normaler Rückzugsbetrag). Es kann sinnvoll sein, die Geschwindigkeit des äußeren Umfangs auf die gleiche Geschwindigkeit wie den Rest des Drucks einzustellen, ist aber nicht zwingend erforderlich.
Es ist üblich, dass der Testdruck an jeder Ecke ein anderes Verhalten zeigt. Oft wird der Slicer an einer Ecke einen Lagenwechsel vornehmen, was dazu führen kann, dass sich diese Ecke deutlich von den übrigen drei Ecken unterscheidet. Wenn dies der Fall ist, ignorieren Sie diese Ecke und stellen Sie den Druckvorschub anhand der anderen drei Ecken ein. Es ist auch üblich, dass die übrigen Ecken leicht variieren. (Dies kann auf kleine Unterschiede in der Reaktion des Druckerrahmens auf Kurvenfahrten in bestimmte Richtungen zurückzuführen sein). Versuchen Sie, einen Wert zu wählen, der für alle übrigen Ecken gut funktioniert. Bevorzugen Sie im Zweifelsfall einen niedrigeren Druckvorlaufwert.

Wenn ein hoher Druckvorschubwert (z. B. über 0,200) verwendet wird, kann es vorkommen, dass der Extruder bei der Rückkehr zur normalen Beschleunigung des Druckers überspringt. Das Druckvorschubsystem berücksichtigt den Druck, indem es während der Beschleunigung zusätzliches

Filament einschiebt und dieses Filament während der Verzögerung zurückzieht. Bei einer hohen Beschleunigung und einem hohen Druckvorschub hat der Extruder möglicherweise nicht genug Drehmoment, um das benötigte Filament zu fördern. Wenn dies der Fall ist, verwenden Sie entweder einen niedrigeren Beschleunigungswert oder deaktivieren Sie die Druckvorverlegung.

Sobald der Druckvorschub in Klipper eingestellt ist, kann es immer noch nützlich sein, einen kleinen Rückzugswert im Slicer zu konfigurieren (z. B. 0,75 mm) und die "Wipe on retract"-Option des Slicers zu verwenden, falls verfügbar. Diese Slicer-Einstellungen können dazu beitragen, die durch die Filamentkohäsion (Filament, das aufgrund der Klebrigkeit des Kunststoffs aus der Düse gezogen wird) verursachte Sickerstelle zu vermeiden. Es wird empfohlen, die Option "z-lift on retract" des Slicers zu deaktivieren.

Das Druckvorschubsystem ändert weder das Timing noch den Weg des Werkzeugkopfs. Ein Druck mit aktiviertem Druckvorschub dauert genauso lange wie ein Druck ohne Druckvorschub. Der Druckvorschub verändert auch nicht die Gesamtmenge des während eines Drucks extrudierten Filaments. Der Druckvorschub führt zu einer zusätzlichen Bewegung des Extruders während der Beschleunigung und Verzögerung der Bewegung. Eine sehr hohe Druckvorschubeinstellung führt zu einer sehr großen Extruderbewegung während der Beschleunigung und Abbremsung, und keine Konfigurationseinstellung schränkt den Umfang dieser Bewegung ein.

G-Codes

Dieses Dokument beschreibt die Befehle, die Klipper unterstützt. Es handelt sich um Befehle, die man in die OctoPrint-Terminal-Registerkarte eingeben kann.

G-Code-Befehle

Klipper unterstützt die folgenden Standard-G-Code-Befehle:

- Bewegen (G0 oder G1): `G1 [X<pos>] [Y<pos>] [Z<pos>] [E<pos>] [F<speed>]`
- Verweilzeit: `G4 P<Millisekunden>`
- Bewegen zum Ursprung: `G28 [X] [Y] [Z]`
- Motoren ausschalten: `M18 oder M84`
- Warten, bis die aktuelle Bewegung beendet ist: `M400`
- Absolute/relative Abstände für die Extrusion verwenden: `M82, M83`
- Absolute/relative Koordinaten verwenden: `G90, G91`
- Position setzen: `G92 [X<pos>] [Y<pos>] [Z<pos>] [E<pos>]`
- Geschwindigkeitsfaktor-Übersteuerung in Prozent einstellen: M220 S<Prozent>
- Prozentuale Übersteuerung des Extrudierfaktors einstellen: M221 S<Prozent>
- Beschleunigung einstellen: `M204 S<value> ODER M204 P<value> T<value>`
  - Hinweis: Wenn S nicht angegeben ist und sowohl P als auch T angegeben sind, wird die Beschleunigung auf das Minimum von P und T gesetzt. Ist nur eines von P oder T angegeben, hat der Befehl keine Wirkung.
- Extrudertemperatur abfragen: `M105`
- Extrudertemperatur einstellen: `M104 [T<index>] [S<Temperatur>]`
- Extrudertemperatur einstellen und warten: `M109 [T<index>] S<Temperatur>`
  - Hinweis: M109 wartet immer darauf, dass sich die Temperatur auf den gewünschten Wert einstellt.
- Betttemperatur einstellen: `M140 [S<Temperatur>]`
- Betttemperatur einstellen und warten: `M190 S<Temperatur>`
  - Hinweis: M190 wartet immer darauf, dass sich die Temperatur auf den gewünschten Wert einstellt.
- Lüftergeschwindigkeit einstellen: `M106 S<value>`
- Lüfter ausschalten: `M107`
- Not-Aus: `M112`
- Aktuelle Position abfragen: `M114`
- Abfrage der Firmware-Version: `M115`

Für weitere Details zu den oben genannten Befehlen siehe die RepRap G-Code Dokumentation RepRap G-Code documentation.

Das Ziel von Klipper ist es, die G-Code-Befehle, die von gängiger Software von Drittanbietern (z.B. OctoPrint, Printrun, Slic3r, Cura, etc.) erzeugt werden, in deren Standardkonfigurationen zu unterstützen. Es ist nicht das Ziel, jeden möglichen G-Code-Befehl zu unterstützen. Stattdessen bevorzugt Klipper menschenlesbare "erweiterte G-Code-Befehle" "extended G-Code commands". Auch die G-Code-Terminalausgabe ist nur für Menschen lesbar - siehe das API Server document, wenn Klipper von einer externen Software aus gesteuert wird.

Wenn man einen weniger gebräuchlichen G-Code-Befehl benötigt, kann es möglich sein, diesen mit einem gcode_macro config section zu implementieren. Zum Beispiel könnte man dies verwenden, um zu implementieren: `G12, G29, G30, G31, M42, M80, M81, T1,` etc.

Zusätzliche Befehle

Klipper verwendet "erweiterte" G-Code-Befehle für die allgemeine Konfiguration und den Status. Diese erweiterten Befehle folgen alle einem ähnlichen Format - sie beginnen mit einem Befehlsnamen und können von einem oder mehreren Parametern gefolgt werden. Zum Beispiel: `SET_SERVO SERVO=myservo ANGLE=5.3`. In diesem Dokument werden die Befehle und Parameter in Großbuchstaben dargestellt, es wird jedoch nicht zwischen Groß- und Kleinschreibung unterschieden. (So führen "`SET_SERVO`" und "`set_servo`" beide denselben Befehl aus.)

Dieser Abschnitt ist nach dem Namen des Klipper-Moduls geordnet, der im Allgemeinen den in der Druckerkonfigurationsdatei printer configuration file angegebenen Abschnittsnamen folgt. Beachten Sie, dass einige Module automatisch geladen werden.

[adxl345]

Die folgenden Befehle sind verfügbar, wenn ein adxl345-Konfigurationsabschnitt aktiviert ist.

ACCELEROMETER_MEASURE

`ACCELEROMETER_MEASURE [CHIP=<config_name>] [NAME=<value>]:` Startet Beschleunigungsmesser-Messungen mit der gewünschten Anzahl von Abtastungen pro Sekunde. Wenn CHIP nicht angegeben wird, ist der Standardwert "adxl345". Der Befehl arbeitet im Start-Stopp-Modus: bei der ersten Ausführung werden die Messungen gestartet, bei der nächsten Ausführung werden sie gestoppt. Die Ergebnisse der Messungen werden in eine Datei mit dem `Namen /tmp/adxl345-<chip>-<name>.csv geschrieben, wobei <chip> der Name des Beschleunigungsmesserchips ist (my_chip_name aus [adxl345 my_chip_name])` und `<name>` der optionale Parameter NAME ist. Wenn NAME nicht angegeben wird, wird standardmäßig die aktuelle Zeit im Format "JJJJMMTT_HHMMSS" verwendet. Wenn der Beschleunigungsmesser keinen Namen in seiner Konfigurationssektion hat (einfach `[adxl345]`), wird der `<chip>`-Teil des Namens nicht generiert.

ACCELEROMETER_QUERY

`ACCELEROMETER_QUERY [CHIP=<config_name>] [RATE=<value>]:` fragt den Beschleunigungsmesser nach dem aktuellen Wert ab. Wenn CHIP nicht angegeben wird, ist der Standardwert "adxl345". Wenn RATE nicht angegeben ist, wird der Standardwert verwendet. Dieser Befehl ist nützlich, um die Verbindung zum ADXL345-Beschleunigungsmesser zu testen: einer der zurückgegebenen Werte sollte eine Beschleunigung im freien Fall sein (+/- etwas Rauschen des Chips).

ACCELEROMETER_DEBUG_READ

`ACCELEROMETER_DEBUG_READ [CHIP=<config_name>] REG=<register>:` fragt das ADXL345-Register "register" ab (z.B. 44 oder 0x2C). Kann für Debugging-Zwecke nützlich sein.

ACCELEROMETER_DEBUG_WRITE

`ACCELEROMETER_DEBUG_WRITE [CHIP=<config_name>] REG=<register> VAL=<value>:` Schreibt den Rohwert "value" in ein Register "register". Sowohl "Wert" als auch "Register" können eine dezimale oder hexadezimale Ganzzahl sein. Seien Sie vorsichtig und beziehen Sie sich auf das ADXL345-Datenblatt für die Referenz.

[angle]

Die folgenden Befehle sind verfügbar, wenn ein Winkelkonfigurationsabschnitt aktiviert ist.

ANGLE_CALIBRATE

`ANGLE_CALIBRATE CHIP=<chip_name>:` Führt eine Winkelkalibrierung für den angegebenen Sensor durch (es muss ein [angle chip_name]-Konfigurationsabschnitt vorhanden sein, der einen Stepper-Parameter angegeben hat). WICHTIG - dieses Tool befiehlt dem Schrittmotor, sich zu bewegen, ohne die normalen kinematischen Grenzwerte zu überprüfen. Idealerweise sollte der Motor vor der Kalibrierung vom Druckerwagen abgekoppelt werden. Wenn der Schrittmotor nicht vom Drucker getrennt werden kann, vergewissern Sie sich, dass sich der Schlitten in der Mitte seiner Schiene befindet, bevor Sie mit der Kalibrierung beginnen. (Der Schrittmotor kann sich während dieses Tests zwei volle Umdrehungen vorwärts oder rückwärts bewegen). Nach Abschluss dieses Tests verwenden Sie den Befehl `SAVE_CONFIG`, um die Kalibrierungsdaten in der Konfigurationsdatei zu speichern. Um dieses Tool verwenden zu können, muss das Python-Paket "numpy" installiert sein (weitere Informationen finden Sie im Dokument Resonanzmessung).

ANGLE_DEBUG_READ

`ANGLE_DEBUG_READ CHIP=<config_name> REG=<register>:` Fragt das Sensorregister "register" ab (z.B. 44 oder 0x2C). Kann für Debugging-Zwecke nützlich sein. Dies ist nur für tle5012b-Chips verfügbar.

ANGLE_DEBUG_WRITE

`ANGLE_DEBUG_WRITE CHIP=<konfig_name> REG=<register> VAL=<value>:` Schreibt den Rohwert "value" in das Register "register". Sowohl "Wert" als auch "Register" können eine dezimale oder hexadezimale Ganzzahl sein. Bitte mit Vorsicht verwenden und das Datenblatt des Sensors zu Rate ziehen. Dies ist nur für tle5012b-Chips verfügbar.

[bed_mesh]

Die folgenden Befehle sind verfügbar, wenn der Abschnitt bed_mesh config section aktiviert ist (siehe auch den bed mesh guide).

BED_MESH_CALIBRATE

`BED_MESH_CALIBRATE [METHOD=manuell] [<probe_parameter>=<value>] [<mesh_parameter>=<value>]:` Dieser Befehl sondiert das Bett mit Hilfe von generierten Punkten, die durch die Parameter in der Konfiguration angegeben werden. Nach der Abtastung wird ein Netz erzeugt und die z-Bewegung wird entsprechend dem Netz angepasst. Siehe PROBE-Befehl für Details zu den optionalen Sondenparametern. Wenn METHOD=manual angegeben ist, wird das manuelle Antastwerkzeug aktiviert - siehe den Befehl MANUAL_PROBE oben für Einzelheiten zu den zusätzlichen Befehlen, die verfügbar sind, wenn dieses Werkzeug aktiv ist.

BED_MESH_OUTPUT
`BED_MESH_OUTPUT PGP=[<0:1>]:` Dieser Befehl gibt die aktuellen sondierten z-Werte und die aktuellen Maschenwerte an das Terminal aus. Wenn PGP=1 angegeben wird, werden die von bed_mesh generierten X- und Y-Koordinaten zusammen mit den zugehörigen Indizes auf dem Terminal ausgegeben.

BED_MESH_MAP
`BED_MESH_MAP: Wie BED_MESH_OUTPUT` gibt dieser Befehl den aktuellen Zustand des Netzes auf dem Terminal aus. Anstatt die Werte in einem für Menschen lesbaren Format zu drucken, wird der Zustand im json-Format serialisiert. Dies ermöglicht es Octoprint-Plugins, die Daten einfach zu erfassen und Höhenkarten zu erzeugen, die die Oberfläche des Bettes annähernd darstellen.

BED_MESH_CLEAR
`BED_MESH_CLEAR:` Dieser Befehl löscht das Netz und entfernt alle z-Anpassungen. Es wird empfohlen, diesen Befehl in den End-G-Code einzufügen.

BED_MESH_PROFILE
`BED_MESH_PROFILE LOAD=<name> SAVE=<name> REMOVE=<name>`
`:` Dieser Befehl dient der Profilverwaltung des Mesh-Status. LOAD stellt den Mesh-Status aus dem Profil mit dem angegebenen Namen wieder her. SAVE speichert den aktuellen Mesh-Zustand in einem Profil mit dem angegebenen Namen. Remove löscht das Profil mit dem angegebenen Namen aus dem persistenten Speicher. Beachten Sie, dass nach der Ausführung von SAVE oder REMOVE der gcode SAVE_CONFIG ausgeführt werden muss, um die Änderungen im persistenten Speicher dauerhaft zu machen.

BED_MESH_OFFSET
`BED_MESH_OFFSET [X=<value>] [Y=<value>]:` Wendet X- und/oder Y-Offsets auf das Mesh-Lookup an. Dies ist nützlich für Drucker mit unabhängigen Extrudern, da ein Offset notwendig ist, um eine korrekte Z-Anpassung nach einem Werkzeugwechsel zu erreichen.

[bed_screws]
Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt bed_screws config section aktiviert ist (siehe auch den Leitfaden für manuelle Ebenen).

BED_SCREWS_ADJUST
`BED_SCREWS_ADJUST:` Dieser Befehl ruft das Werkzeug zur Einstellung der Bettschrauben auf. Er steuert die Düse an verschiedene Positionen (wie in der Konfigurationsdatei definiert) und ermöglicht es, die Bettschrauben so einzustellen, dass das Bett einen konstanten Abstand zur Düse hat.

[bed_tilt]
Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt bed_tilt config section aktiviert ist.

BED_TILT_CALIBRATE
`BED_TILT_CALIBRATE [METHOD=manuell] [<probe_parameter>=<value>]:` Dieser Befehl tastet die in der Konfiguration angegebenen Punkte ab und empfiehlt dann aktualisierte x- und y-Neigungseinstellungen. Siehe PROBE-Befehl für Details zu den optionalen Probe-Parametern. Wenn METHOD=manual angegeben ist, wird das manuelle Antastwerkzeug aktiviert - siehe den obigen Befehl MANUAL_PROBE für Einzelheiten zu den zusätzlichen Befehlen, die verfügbar sind, wenn dieses Werkzeug aktiv ist.

[bltouch]
Der folgende Befehl ist verfügbar, wenn ein bltouch-Konfigurationsabschnitt aktiviert bltouch config section ist (siehe auch das BL-Touch-Handbuch BL-Touch guide).

BLTOUCH_DEBUG
`BLTOUCH_DEBUG COMMAND=<command>:` Dieser Befehl sendet einen Befehl an das BLTouch. Dies kann für die Fehlersuche nützlich sein. Verfügbare Befehle sind`: pin_down, touch_mode, pin_up, self_test, reset`. Ein BL-Touch V3.0 oder V3.1 kann auch die Befehle `set_5V_output_mode, set_OD_output_mode, output_mode_store` unterstützen.

BLTOUCH_STORE
`BLTOUCH_STORE MODE=<output_mode>:` Speichert einen Ausgabemodus im EEPROM eines BLTouch V3.1 Verfügbare Ausgabemodi sind`: 5V, OD`

[configfile]
Das configfile-Modul wird automatisch geladen.

SAVE_CONFIG
`SAVE_CONFIG:` Mit diesem Befehl wird die Hauptkonfigurationsdatei des Druckers überschrieben und die Host-Software neu gestartet. Dieser Befehl wird in Verbindung mit anderen Kalibrierungsbefehlen verwendet, um die Ergebnisse von Kalibrierungstests zu speichern.

[delayed_gcode].
Der folgende Befehl wird aktiviert, wenn ein delayed_gcode-Konfigurationsabschnitt delayed_gcode config section aktiviert wurde (siehe auch den Vorlagenleitfaden template guide).

UPDATE_DELAYED_GCODE
`UPDATE_DELAYED_GCODE [ID=<name>] [DURATION=<seconds>]:` Aktualisiert die Verzögerungsdauer für den identifizierten [delayed_gcode] und startet den Timer für die Ausführung des Gcodes. Bei einem Wert von 0 wird die Ausführung eines anstehenden verzögerten Gcodes abgebrochen.

[display]
Der folgende Befehl ist verfügbar, wenn ein Abschnitt zur Anzeigekonfiguration display config section aktiviert ist.

SET_DISPLAY_GROUP
`SET_DISPLAY_GROUP [DISPLAY=<display>] GROUP=<group>:` Setzt die aktive Displaygruppe eines LCD-Displays. Dies ermöglicht es, mehrere Display-Datengruppen in der Konfiguration zu definieren, z.B. [display_data <group> <elementname>] und zwischen ihnen mit diesem erweiterten gcode-Befehl zu wechseln. Wenn DISPLAY nicht angegeben wird, wird standardmäßig "display" (die primäre Anzeige) verwendet.

[display_status]
Das Modul display_status wird automatisch geladen, wenn ein Display-Konfigurationsabschnitt display config section aktiviert ist. Es bietet die folgenden Standard-G-Code-Befehle:

Display Message: `M117 <message>`
Prozentsatz für die Erstellung einstellen: `M73 P<percent>`

[dual_carriage]
Der folgende Befehl ist verfügbar, wenn der Konfigurationsabschnitt dual_carriage config section aktiviert ist.

SET_DUAL_CARRIAGE
`SET_DUAL_CARRIAGE CARRIAGE=[0|1]:` Mit diesem Befehl wird der aktive Wagen eingestellt. Er wird in der Regel über die Felder activate_gcode und deactivate_gcode in einer Konfiguration mit mehreren Extrudern aufgerufen.

[endstop_phase]
Die folgenden Befehle sind verfügbar, wenn ein endstop_phase Konfigurationsabschnitt endstop_phase config section aktiviert ist (siehe auch den endstop phase Leitfaden endstop phase guide).

ENDSTOP_PHASE_KALIBRIEREN
`ENDSTOP_PHASE_CALIBRATE [STEPPER=<config_name>]:` Wenn kein STEPPER-Parameter angegeben wird, erstellt dieser Befehl eine Statistik über die Endstopp-Schrittphasen während der vergangenen Referenzfahrten. Wenn ein STEPPER-Parameter angegeben wird, sorgt er dafür, dass die angegebene Endstopp-Phaseneinstellung in die Konfigurationsdatei geschrieben wird (in Verbindung mit dem Befehl `SAVE_CONFIG`).

[extruder]
Die folgenden Befehle sind verfügbar, wenn ein Extruder-Konfigurationsabschnitt extruder config section aktiviert ist:

ACTIVATE_EXTRUDER
`ACTIVATE_EXTRUDER EXTRUDER=<config_name>:` Bei einem Drucker mit mehreren Extruder-Konfigurationsabschnitten extruder ändert dieser Befehl das aktive Hotend.

SET_PRESSURE_ADVANCE
`SET_PRESSURE_ADVANCE [EXTRUDER=<config_name>] [ADVANCE=<pressure_advance>] [SMOOTH_TIME=<pressure_advance_smooth_time>]:` Setzt die Druckvorschubparameter eines Extruder-Steppers (wie in einem extruder oder extruder_stepper-Konfigurationsabschnitt extruder_stepper definiert). Wenn EXTRUDER nicht angegeben ist, wird standardmäßig der im aktiven Hotend definierte Stepper verwendet.

SET_EXTRUDER_ROTATION_DISTANCE
`SET_EXTRUDER_ROTATION_DISTANCE EXTRUDER=<config_name> [DISTANCE=<distance>]:` Setzt einen neuen Wert für die "Rotationsdistanz" des angegebenen Extruder-Steppers (wie in einem extruder oder extruder_stepper-Konfigurationsabschnitt extruder_stepper definiert). Wenn die Rotationsdistanz eine negative Zahl ist, wird die Stepperbewegung invertiert (relativ zu der in der Konfigurationsdatei angegebenen Stepperrichtung). Geänderte Einstellungen werden beim Zurücksetzen des Klippers nicht beibehalten. Seien Sie vorsichtig, da kleine Änderungen zu einem übermäßigen Druck zwischen Extruder und Hotend führen können. Führen Sie vor der Verwendung eine ordnungsgemäße Kalibrierung mit Filament durch. Wenn der Wert 'DISTANCE' nicht angegeben wird, gibt dieser Befehl den aktuellen Rotationsabstand zurück.

SYNC_EXTRUDER_MOTION
`SYNC_EXTRUDER_MOTION EXTRUDER=<name> MOTION_QUEUE=<name>:` Dieser Befehl bewirkt, dass der durch EXTRUDER angegebene Stepper (wie in einem extruder oder extruder_stepper-Konfigurationsabschnitt extruder_stepper definiert) mit der Bewegung eines durch MOTION_QUEUE angegebenen Extruders (wie in einem Extruder-Konfigurationsabschnitt extruder definiert) synchronisiert wird. Wenn MOTION_QUEUE ein leerer String ist, wird der Stepper von allen Extruderbewegungen desynchronisiert.

SET_EXTRUDER_STEP_DISTANCE
Dieser Befehl ist veraltet und wird in naher Zukunft entfernt werden.

SYNC_STEPPER_TO_EXTRUDER
Dieser Befehl ist veraltet und wird in naher Zukunft entfernt werden.

[fan_generic]

Der folgende Befehl ist verfügbar, wenn ein fan_generic config sectionKonfigurationsabschnitt aktiviert ist.

SET_FAN_SPEED
`SET_FAN_SPEED FAN=config_name SPEED=<speed>` Dieser Befehl setzt die Geschwindigkeit eines Lüfters. "speed" muss zwischen 0.0 und 1.0 liegen.

[filament_switch_sensor]

Der folgende Befehl ist verfügbar, wenn ein filament_switch_sensor or filament_motion_sensor aktiviert ist.

QUERY_FILAMENT_SENSOR
`QUERY_FILAMENT_SENSOR SENSOR=<sensor_name>:` Fragt den aktuellen Status des Filamentsensors ab. Die auf dem Terminal angezeigten Daten hängen von dem in der Konfiguration definierten Sensortyp ab.

SET_FILAMENT_SENSOR
`SET_FILAMENT_SENSOR SENSOR=<sensor_name> ENABLE=[0|1]:` Schaltet den Filamentsensor ein/aus. Wenn ENABLE auf 0 gesetzt ist, ist der Filamentsensor deaktiviert, wenn er auf 1 gesetzt ist, ist er aktiviert.

[firmware_retraction].

Die folgenden Standard-G-Code-Befehle sind verfügbar, wenn der Konfigurationsabschnitt firmware_retraction config section aktiviert ist. Mit diesen Befehlen können Sie die in vielen Slicern verfügbare Firmware-Retraction-Funktion nutzen, um die Fadenbildung bei Nicht-Extrusionsbewegungen von einem Teil des Drucks zu einem anderen zu reduzieren. Durch eine entsprechende Konfiguration des Druckvorschubs wird die Länge des erforderlichen Rückzugs reduziert.

`G10:` Fährt den Extruder mit den aktuell konfigurierten Parametern zurück.
`G11:` Rückzug des Extruders mit den aktuell konfigurierten Parametern.

Die folgenden zusätzlichen Befehle sind ebenfalls verfügbar.

SET_RETRACTION
`SET_RETRACTION [RETRACT_LENGTH=<mm>] [RETRACT_SPEED=<mm/s>] [UNRETRACT_EXTRA_LENGTH=<mm>] [UNRETRACT_SPEED=<mm/s>]:` Passen Sie die Parameter an, die von der Firmware für den Rückzug verwendet werden. RETRACT_LENGTH bestimmt die Länge des zurückzuziehenden und nicht zurückzuziehenden Filaments. Die Geschwindigkeit des Zurückziehens wird über RETRACT_SPEED eingestellt und ist normalerweise relativ hoch. Die Geschwindigkeit des Zurückziehens wird über UNRETRACT_SPEED eingestellt und ist nicht besonders kritisch, obwohl sie oft niedriger als RETRACT_SPEED ist. In einigen Fällen ist es sinnvoll, beim Zurückziehen eine kleine zusätzliche Länge hinzuzufügen, die über UNRETRACT_EXTRA_LENGTH eingestellt wird. SET_RETRACTION wird in der Regel als Teil der Slicer-Konfiguration pro Filament gesetzt, da verschiedene Filamente unterschiedliche Parametereinstellungen erfordern.

GET_RETRACTION
`GET_RETRACTION:` Fragt die aktuellen von der Firmware Retraction verwendeten Parameter ab und zeigt sie auf dem Terminal an.

[force_move]

Das force_move-Modul wird automatisch geladen, einige Befehle erfordern jedoch die Einstellung enable_force_move in der Druckerkonfiguration printer config.

STEPPER_BUZZ
`STEPPER_BUZZ STEPPER=<config_name>:` Bewegt den angegebenen Stepper einen mm vorwärts und dann einen mm rückwärts, 10-mal wiederholt. Dies ist ein Diagnosewerkzeug, um die Konnektivität des Steppers zu überprüfen.

FORCE_MOVE
`FORCE_MOVE STEPPER=<config_name> DISTANCE=<value> VELOCITY=<value> [ACCEL=<value>]:` Mit diesem Befehl wird der angegebene Stepper zwangsweise um die angegebene Distanz (in mm) mit der angegebenen konstanten Geschwindigkeit (in mm/s) bewegt. Wenn ACCEL angegeben wird und größer als Null ist, dann wird die angegebene Beschleunigung (in mm/s^2) verwendet; andernfalls wird keine Beschleunigung durchgeführt. Es werden keine Begrenzungsprüfungen durchgeführt; es werden keine kinematischen Aktualisierungen vorgenommen; andere parallele Stepper auf einer Achse werden nicht bewegt. Seien Sie vorsichtig, da ein falscher Befehl zu Schäden führen kann! Bei Verwendung dieses Befehls wird die Low-Level-Kinematik mit ziemlicher Sicherheit in einen falschen Zustand versetzt; geben Sie anschließend ein G28 aus, um die Kinematik zurückzusetzen. Dieser Befehl ist für die Low-Level-Diagnose und Fehlersuche gedacht.

SET_KINEMATIC_POSITION
`SET_KINEMATIC_POSITION [X=<value>] [Y=<value>] [Z=<value>]:` Erzwingt, dass der Low-Level-Kinematikcode davon ausgeht, dass sich der Werkzeugkopf an der angegebenen kartesischen Position befindet. Dies ist ein Diagnose- und Debugging-Befehl; verwenden Sie SET_GCODE_OFFSET und/oder G92 für normale Achsentransformationen. Wenn eine Achse nicht angegeben wird, wird sie standardmäßig auf die Position gesetzt, auf die der Kopf zuletzt befohlen wurde. Die Einstellung einer falschen oder ungültigen Position kann zu internen Softwarefehlern führen. Dieser Befehl kann zukünftige Boundary Checks ungültig machen; geben Sie anschließend ein G28 aus, um die Kinematik zurückzusetzen.

[gcode]
Das gcode-Modul wird automatisch geladen.

RESTART
`RESTART:` Dies veranlasst die Host-Software, ihre Konfiguration neu zu laden und einen internen Reset durchzuführen. Dieser Befehl löscht weder den Fehlerstatus des Mikrocontrollers (siehe FIRMWARE_RESTART) noch lädt er neue Software (siehe the FAQ).

FIRMWARE_RESTART
`FIRMWARE_RESTART:` Dieser Befehl ähnelt dem RESTART-Befehl, löscht aber auch alle Fehlerzustände des Mikrocontrollers.

STATUS
`STATUS:` Meldet den Status der Klipper-Hostsoftware.

HELP
`HILFE:` Zeigt die Liste der verfügbaren erweiterten G-Code-Befehle an.

[gcode_arcs]
Die folgenden Standard G-Code Befehle sind verfügbar, wenn ein gcode_arcs Konfigurationsabschnitt gcode_arcs config section aktiviert ist:

Kontrollierte Bogenbewegung `(G2 oder G3): G2 [X<pos>] [Y<pos>] [Z<pos>] [E<pos>] [F<speed>] I<value> J<value>`

[gcode_macro]
Der folgende Befehl ist verfügbar, wenn ein gcode_macro-Konfigurationsabschnitt gcode_macro config section aktiviert ist (siehe auch die Anleitung für Befehlsvorlagen command templates guide).

SET_GCODE_VARIABLE
`SET_GCODE_VARIABLE MACRO=<macro_name> VARIABLE=<name> VALUE=<value>:` Dieser Befehl erlaubt es, den Wert einer gcode_macro-Variable zur Laufzeit zu ändern. Der angegebene VALUE wird als Python-Literal geparst.

[gcode_move]
Das Modul gcode_move wird automatisch geladen.

GET_POSITION
`GET_POSITION:` Gibt Informationen über die aktuelle Position des Werkzeugkopfes zurück. Weitere Informationen finden Sie in der Entwicklerdokumentation der GET_POSITION output.

SET_GCODE_OFFSET
`SET_GCODE_OFFSET [X=<pos>|X_ADJUST=<adjust>] [Y=<pos>|Y_ADJUST=<adjust>] [Z=<pos>|Z_ADJUST=<adjust>] [MOVE=1 [MOVE_SPEED=<speed>]]:` Legt einen Positionsoffset fest, der auf zukünftige G-Code-Befehle angewendet wird. Dies wird üblicherweise verwendet, um den Z-Bett-Offset virtuell zu ändern oder um Düsen-XY-Offsets beim Wechsel des Extruders einzustellen. Wenn zum Beispiel "SET_GCODE_OFFSET Z=0.2" gesendet wird, werden bei zukünftigen G-Code-Bewegungen 0,2 mm zu ihrer Z-Höhe hinzugefügt. Wenn die X_ADJUST-Parameter verwendet werden, wird die Anpassung zu einem vorhandenen Offset addiert (z. B. würde "SET_GCODE_OFFSET Z=-0.2" gefolgt von "SET_GCODE_OFFSET Z_ADJUST=0.3" zu einem Z-Offset von insgesamt 0.1 führen). Wenn "`MOVE=1`" angegeben wird, dann wird eine Werkzeugkopfbewegung ausgegeben, um den angegebenen Versatz anzuwenden (andernfalls wird der Versatz bei der nächsten absoluten G-Code-Bewegung wirksam, die die angegebene Achse angibt). Wenn "MOVE_SPEED" angegeben ist, wird die Werkzeugkopfbewegung mit der angegebenen Geschwindigkeit (in mm/s) ausgeführt; andernfalls wird die Werkzeugkopfbewegung mit der zuletzt angegebenen G-Code-Geschwindigkeit ausgeführt.

SAVE_GCODE_STATE
`SAVE_GCODE_STATE [NAME=<State_name>]:` Speichert den aktuellen Status der G-Code-Koordinatenanalyse. Das Speichern und Wiederherstellen des G-Code-Status ist in Skripten und Makros nützlich. Dieser Befehl speichert den aktuellen absoluten G-Code-Koordinatenmodus (G90/G91), den absoluten Extrudiermodus (M82/M83), den Ursprung (G92), den Offset (SET_GCODE_OFFSET), die Geschwindigkeitsübersteuerung (M220), die Extruderübersteuerung (M221), die Bewegungsgeschwindigkeit, die aktuelle XYZ-Position und die relative Extruder-"E"-Position. Wenn NAME angegeben wird, kann der gespeicherte Zustand nach der angegebenen Zeichenkette benannt werden. Wenn NAME nicht angegeben wird, ist der Standardwert "default".

RESTORE_GCODE_STATE
`RESTORE_GCODE_STATE [NAME=<State_name>] [MOVE=1 [MOVE_SPEED=<speed>]]:` Stellt einen Zustand wieder her, der zuvor mit SAVE_GCODE_STATE gespeichert wurde. Wenn "MOVE=1" angegeben wird, wird eine Werkzeugkopfbewegung ausgegeben, um zur vorherigen XYZ-Position zurückzufahren. Wenn "MOVE_SPEED" angegeben ist, wird die Werkzeugkopfbewegung mit der angegebenen Geschwindigkeit (in mm/s) ausgeführt; andernfalls wird die Werkzeugkopfbewegung mit der wiederhergestellten G-Code-Geschwindigkeit ausgeführt.

[hall_filament_width_sensor]

Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt tsl1401cl filament width sensor config section or hall filament width sensor config section aktiviert ist (siehe auch TSLI401CL Filament Width Sensor and Hall Filament Width Sensor):

QUERY_FILAMENT_WIDTH
`QUERY_FILAMENT_WIDTH:` Gibt die aktuell gemessene Filamentbreite zurück.

RESET_FILAMENT_WIDTH_SENSOR
`RESET_FILAMENT_WIDTH_SENSOR:` Löscht alle Sensormesswerte. Hilfreich nach einem Filamentwechsel.

FILAMENT_BREITENSENSOR_DEAKTIVIEREN
`DISABLE_FILAMENT_WIDTH_SENSOR:` Schaltet den Filamentbreitensensor aus und verwendet ihn nicht mehr für die Flusskontrolle.

ENABLE_FILAMENT_WIDTH_SENSOR
`ENABLE_FILAMENT_WIDTH_SENSOR:` Schalten Sie den Filamentbreitensensor ein und verwenden Sie ihn für die Flusssteuerung.

QUERY_RAW_FILAMENT_WIDTH
`QUERY_RAW_FILAMENT_WIDTH:` Liefert die aktuellen ADC-Kanalmesswerte und den RAW-Sensorwert für Kalibrierungspunkte.

ENABLE_FILAMENT_WIDTH_LOG
`ENABLE_FILAMENT_WIDTH_LOG:` Schaltet die Aufzeichnung des Durchmessers ein.

DISABLE_FILAMENT_WIDTH_LOG¶
`DISABLE_FILAMENT_WIDTH_LOG:` Durchmesserprotokollierung ausschalten.

[heater]

Das Heizungsmodul wird automatisch geladen, wenn eine Heizung in der Konfigurationsdatei definiert ist.

TURN_OFF_HEATERS
`TURN_OFF_HEATERS:` Schaltet alle Heizelemente aus.

TEMPERATURE_WAIT
`TEMPERATURE_WAIT SENSOR=<config_name> [MINIMUM=<target>] [MAXIMUM=<target>]:` Warten, bis der angegebene Temperatursensor auf oder über dem angegebenen MINIMUM und/oder auf oder unter dem angegebenen MAXIMUM liegt.

SET_HEATER_TEMPERATURE
`SET_HEATER_TEMPERATURE HEATER=<heater_name> [TARGET=<target_temperature>]:` Setzt die Zieltemperatur für eine Heizung. Wenn keine Zieltemperatur angegeben wird, ist die Zieltemperatur 0.

[idle_timeout]

Das Modul idle_timeout wird automatisch geladen.

SET_IDLE_TIMEOUT
`SET_IDLE_TIMEOUT [TIMEOUT=<timeout>]:` Ermöglicht dem Benutzer die Einstellung des Idle-Timeouts (in Sekunden).

[input_shaper].

Der folgende Befehl wird aktiviert, wenn ein input_shaper-Konfigurationsabschnitt aktiviert wurde (siehe auch die Anleitung zur Resonanzkompensation input_shaper config section).

SET_INPUT_SHAPER
`SET_INPUT_SHAPER [SHAPER_FREQ_X=<shaper_freq_x>] [SHAPER_FREQ_Y=<shaper_freq_y>] [DAMPING_RATIO_X=<damping_ratio_x>] [DAMPING_RATIO_Y=<damping_ratio_y>] [SHAPER_TYPE=<shaper>] [SHAPER_TYPE_X=<shaper_type_x>] [SHAPER_TYPE_Y=<shaper_type_y>]:` Ändern Sie die Parameter des Eingabeshapers. Beachten Sie, dass der Parameter SHAPER_TYPE den Eingangs-Shaper für die X- und Y-Achse zurücksetzt, auch wenn in der Sektion [input_shaper] unterschiedliche Shaper-Typen konfiguriert wurden. SHAPER_TYPE kann nicht zusammen mit einem der beiden Parameter SHAPER_TYPE_X und SHAPER_TYPE_Y verwendet werden. Siehe Konfigurationsreferenz config reference für weitere Details zu jedem dieser Parameter.

[manual_probe]

Das Modul manual_probe wird automatisch geladen.

MANUAL_PROBE

`MANUAL_PROBE [SPEED=<Geschwindigkeit>]:` Führt ein Hilfsskript aus, das für die Messung der Höhe der Düse an einer bestimmten Stelle nützlich ist. Wenn SPEED angegeben wird, legt es die Geschwindigkeit der TESTZ-Befehle fest (der Standardwert ist 5mm/s). Während einer manuellen Sonde sind die folgenden zusätzlichen Befehle verfügbar:

- `AKZEPTIEREN:` Dieser Befehl akzeptiert die aktuelle Z-Position und beendet das manuelle Antastwerkzeug.
- `ABORT:` Mit diesem Befehl wird das manuelle Antasten beendet.
- `TESTZ Z=<value>:` Mit diesem Befehl wird die Düse um den in "Wert" angegebenen Betrag nach oben oder unten bewegt. Zum Beispiel würde `TESTZ Z=-.1` die Düse um 0,1 mm nach unten bewegen, während TESTZ Z=.1 die Düse um 0,1 mm nach oben bewegen würde. Der Wert kann auch +, -, ++ oder -- sein, um die Düse im Vergleich zu früheren Versuchen um einen bestimmten Betrag nach oben oder unten zu bewegen.

Z_ENDSTOP_CALIBRATE

`Z_ENDSTOP_CALIBRATE [SPEED=<Geschwindigkeit>]:` Führt ein Hilfsskript aus, das für die Kalibrierung einer Z-Position_endstop-Konfigurationseinstellung nützlich ist. Siehe den Befehl MANUAL_PROBE für Details zu den Parametern und den zusätzlichen Befehlen, die verfügbar sind, während das Werkzeug aktiv ist.

Z_OFFSET_APPLY_ENDSTOP

`Z_OFFSET_APPLY_ENDSTOP:` Nimmt den aktuellen Z-Gcode-Offset (auch bekannt als Babystepping) und subtrahiert ihn von der stepper_z endstop_position. Dies dient dazu, einen häufig verwendeten Babystepping-Wert zu nehmen und ihn "dauerhaft" zu machen. Erfordert ein `SAVE_CONFIG`, um wirksam zu werden.

[manual_stepper]

Der folgende Befehl ist verfügbar, wenn ein manual_stepper Konfigurationsabschnitt aktiviert ist.

MANUAL_STEPPER

`MANUAL_STEPPER STEPPER=config_name [ENABLE=[0|1]] [SET_POSITION=<pos>] [SPEED=<speed>] [ACCEL=<accel>] [MOVE=<pos> [STOP_ON_ENDSTOP=[1|2|-1|-2]] [SYNC=0]]:` Mit diesem Befehl wird der Zustand des Steppers geändert. Verwenden Sie den ENABLE-Parameter, um den Stepper zu aktivieren/deaktivieren. Verwenden Sie den Parameter SET_POSITION, um den Stepper zu zwingen, zu denken, dass er sich an der angegebenen Position befindet. Verwenden Sie den MOVE-Parameter, um eine Bewegung zur angegebenen Position anzufordern. Wenn SPEED und/oder ACCEL angegeben wird, werden die angegebenen Werte anstelle der in der Konfigurationsdatei angegebenen Standardwerte verwendet. Wenn ACCEL mit Null angegeben wird, wird keine Beschleunigung durchgeführt. Wenn STOP_ON_ENDSTOP=1 angegeben ist, wird die Bewegung vorzeitig beendet, wenn der Endstopp als ausgelöst gemeldet wird (verwenden Sie STOP_ON_ENDSTOP=2, um die Bewegung ohne Fehler zu beenden, auch wenn der Endstopp nicht ausgelöst wird, verwenden Sie -1 oder -2, um anzuhalten, wenn der Endstopp nicht ausgelöst wird). Normalerweise werden künftige G-Code-Befehle so geplant, dass sie nach Abschluss der Stepperbewegung ausgeführt werden. Wenn jedoch bei einer manuellen Stepperbewegung SYNC=0 verwendet wird, können künftige G-Code-Befehle parallel zur Stepperbewegung ausgeführt werden.

[led]

Der folgende Befehl ist verfügbar, wenn einer der LED-Konfigurationsabschnitte led config sections aktiviert ist.

SET_LED

`SET_LED LED=<config_name> RED=<value> GREEN=<value> BLUE=<value> WHITE=<value> [INDEX=<index>] [TRANSMIT=0] [SYNC=1]:` Hiermit wird die LED-Ausgabe eingestellt. Jede Farbe <value> muss zwischen 0,0 und 1,0 liegen. Die Option WHITE ist nur für RGBW-LEDs gültig. Wenn die LED mehrere Chips in einer Verkettung unterstützt, kann man INDEX angeben, um nur die Farbe des angegebenen Chips zu ändern (1 für den ersten Chip, 2 für den zweiten usw.). Wenn INDEX nicht angegeben wird, werden alle LEDs in der Verkettung auf die angegebene Farbe eingestellt. Wenn TRANSMIT=0 angegeben ist, wird die Farbänderung erst beim nächsten SET_LED-Befehl vorgenommen, der nicht TRANSMIT=0 angibt; dies kann in Kombination mit dem INDEX-Parameter nützlich sein, um mehrere Aktualisierungen in einer Daisy-Chain zu bündeln. Standardmäßig synchronisiert der SET_LED-Befehl seine Änderungen mit anderen laufenden gcode-Befehlen. Dies kann zu einem unerwünschten Verhalten führen, wenn LEDs gesetzt werden, während der Drucker nicht druckt, da dies den Timeout für den Leerlauf zurücksetzt. Wenn ein sorgfältiges Timing nicht erforderlich ist, kann der optionale Parameter SYNC=0 angegeben werden, um die Änderungen ohne Zurücksetzen des Leerlauf-Timeouts zu übernehmen.

SET_LED_TEMPLATE

`SET_LED_TEMPLATE LED=<led_name> TEMPLATE=<template_name> [<param_x>=<literal>] [INDEX=<index>]:` Weist einer gegebenen LED ein display_template zu. Wenn man zum Beispiel einen [display_template my_led_template] Konfigurationsabschnitt definiert hat, kann man hier TEMPLATE=my_led_template zuweisen. Das display_template sollte eine durch Komma getrennte Zeichenkette mit vier Fließkommazahlen enthalten, die den Farbeinstellungen für Rot, Grün, Blau und Weiß entsprechen. Die Vorlage wird kontinuierlich ausgewertet und die LED wird automatisch auf die resultierenden Farben eingestellt. Es können display_template-Parameter festgelegt werden, die bei der Auswertung der Vorlage verwendet werden sollen (die Parameter werden als Python-Literale geparst). Wenn INDEX nicht angegeben ist, werden alle Chips in der Daisy-Chain der LED auf das Template gesetzt, andernfalls wird nur der Chip mit dem angegebenen Index aktualisiert. Wenn TEMPLATE eine leere

Zeichenkette ist, dann löscht dieser Befehl jede vorherige Vorlage, die der LED zugewiesen wurde (man kann dann SET_LED-Befehle verwenden, um die Farbeinstellungen der LED zu verwalten).

[output_pin]

Der folgende Befehl ist verfügbar, wenn ein output_pin Konfigurationsabschnitt output_pin config section aktiviert ist.

SET_PIN

`SET_PIN PIN=config_name VALUE=<value> CYCLE_TIME=<cycle_time>:` Hinweis - Hardware-PWM unterstützt den CYCLE_TIME-Parameter derzeit nicht und verwendet die in der Config definierte Zykluszeit.

[palette2]

Die folgenden Befehle sind verfügbar, wenn der Abschnitt palette2 config section aktiviert ist.
Palettendrucke funktionieren, indem spezielle OCodes (Omega Codes) in die GCode-Datei eingebettet werden:

`O1...O32:` Diese Codes werden aus dem GCode-Stream gelesen, von diesem Modul verarbeitet und an das Palette2-Gerät weitergegeben.
Die folgenden zusätzlichen Befehle sind ebenfalls verfügbar.

PALETTE_CONNECT

`PALETTE_CONNECT:` Dieser Befehl initialisiert die Verbindung mit der Palette 2.

PALETTE_DISCONNECT

`PALETTE_DISCONNECT:` Dieser Befehl trennt die Verbindung mit der Palette 2.

PALETTE_CLEAR

`PALETTE_CLEAR:` Mit diesem Befehl wird die Palette 2 angewiesen, alle Eingangs- und Ausgangspfade von Filament zu löschen.

PALETTE_CUT

`PALETTE_CUT:` Mit diesem Befehl wird die Palette 2 angewiesen, das aktuell im Spleißkern geladene Filament zu schneiden.

PALETTE_SMART_LOAD

`PALETTE_SMART_LOAD:` Dieser Befehl startet die intelligente Ladesequenz auf der Palette 2. Das Filament wird automatisch geladen, indem es die auf dem Gerät für den Drucker kalibrierte Strecke extrudiert wird, und die Palette 2 wird angewiesen, sobald das Laden abgeschlossen ist. Dieser Befehl entspricht dem Drücken von Smart Load direkt auf dem Bildschirm der Palette 2, nachdem das Laden des Filaments abgeschlossen ist.

[pid_calibrate]

Das Modul pid_calibrate wird automatisch geladen, wenn in der Konfigurationsdatei eine Heizung definiert ist.

PID_CALIBRATE

`PID_CALIBRATE HEATER=<config_name> TARGET=<Temperatur> [WRITE_FILE=1]:` Führt eine PID-Kalibrierungsprüfung durch. Die angegebene Heizung wird aktiviert, bis die angegebene Zieltemperatur erreicht ist, und dann wird die Heizung für mehrere Zyklen aus- und eingeschaltet. Wenn der Parameter WRITE_FILE aktiviert ist, wird die Datei /tmp/heattest.txt mit einem Protokoll aller während des Tests genommenen Temperaturproben erstellt.

[pause_resume]

Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt pause_resume config section aktiviert ist:

PAUSE

`PAUSE:` Unterbricht den aktuellen Druck. Die aktuelle Position wird zur Wiederherstellung bei der Wiederaufnahme gespeichert.

RESUME

`RESUME [VELOCITY=<value>]:` Setzt den Druckvorgang nach einer Pause fort, wobei zunächst die zuvor erfasste Position wiederhergestellt wird. Der Parameter VELOCITY bestimmt die Geschwindigkeit, mit der das Werkzeug zur ursprünglichen Position zurückkehren soll.

CLEAR_PAUSE

`CLEAR_PAUSE:` Löscht den aktuellen Pausenzustand, ohne den Druck fortzusetzen. Dies ist nützlich, wenn man einen Druck nach einer PAUSE abbrechen möchte. Es wird empfohlen, diese Funktion in den Start-Gcode einzufügen, um sicherzustellen, dass der Pausenzustand bei jedem Druckvorgang neu ist.

CANCEL_PRINT

`CANCEL_PRINT:` Bricht den aktuellen Druckvorgang ab.

[probe]

Die folgenden Befehle sind verfügbar, wenn ein Abschnitt zur Sondenkonfiguration oder ein Abschnitt zur bltouch-Konfiguration aktiviert ist probe config section or bltouch config section (siehe auch die Anleitung zur Sondenkalibrierung probe calibrate guide).

PROBE
`PROBE [PROBE_SPEED=<mm/s>] [LIFT_SPEED=<mm/s>] [SAMPLES=<count>] [SAMPLE_RETRACT_DIST=<mm>] [SAMPLES_TOLERANCE=<mm>] [SAMPLES_TOLERANCE_RETRIES=<count>] [SAMPLES_RESULT=median|average]:` Bewegen Sie die Düse nach unten, bis die Sonde auslöst. Wenn einer der optionalen Parameter angegeben wird, überschreibt er die entsprechende Einstellung im Abschnitt probe config section.

QUERY_PROBE
`QUERY_PROBE:` Meldung des aktuellen Status der Sonde ("ausgelöst" oder "offen").

PROBE_ACCURACY
`PROBE_ACCURACY [PROBE_SPEED=<mm/s>] [SAMPLES=<count>] [SAMPLE_RETRACT_DIST=<mm>]:` Berechnet das Maximum, das Minimum, den Durchschnitt, den Median und die Standardabweichung von mehreren Sondenproben. Standardmäßig werden 10 SAMPLES genommen. Andernfalls werden die optionalen Parameter auf ihre entsprechenden Einstellungen im Abschnitt probe config zurückgesetzt.

PROBE_CALIBRATE
`PROBE_CALIBRATE [GESCHWINDIGKEIT=<GESCHWINDIGKEIT>] [<Sondenparameter>=<value>]:` Führt ein Hilfsskript aus, das für die Kalibrierung des z_offset der Sonde nützlich ist. Einzelheiten zu den optionalen Sondenparametern finden Sie unter dem Befehl PROBE. Siehe den Befehl MANUAL_PROBE für Details zum SPEED-Parameter und zu den zusätzlichen Befehlen, die verfügbar sind, während das Tool aktiv ist. Bitte beachten Sie, dass der Befehl PROBE_CALIBRATE die Geschwindigkeitsvariable verwendet, um sich sowohl in XY- als auch in Z-Richtung zu bewegen.

Z_OFFSET_APPLY_PROBE
`Z_OFFSET_ANWENDEN_PROBE:` Nimmt den aktuellen Z-Gcode-Offset (auch bekannt als Babystepping) und subtrahiert ihn vom z_offset der Sonde. Auf diese Weise wird ein häufig verwendeter Babystepping-Wert "dauerhaft" gemacht. Erfordert ein `SAVE_CONFIG`, um wirksam zu werden.

[query_adc]
Das Modul query_adc wird automatisch geladen.

QUERY_ADC
`QUERY_ADC [NAME=<config_name>] [PULLUP=<value>]:` Meldet den letzten empfangenen Analogwert für einen konfigurierten Analogpin. Wenn NAME nicht angegeben wird, wird die Liste der verfügbaren AdC-Namen gemeldet. Wenn PULLUP angegeben wird (als Wert in Ohm), wird der analoge Rohwert zusammen mit dem äquivalenten Widerstand bei diesem Pullup gemeldet.

[query_endstops]
Das Modul query_endstops wird automatisch geladen. Die folgenden Standard-G-Code-Befehle sind derzeit verfügbar, ihre Verwendung wird jedoch nicht empfohlen:

Get Endstop Status: `M119` (Verwenden Sie stattdessen QUERY_ENDSTOPS.)

QUERY_ENDSTOPS
`QUERY_ENDSTOPS:` Prüft die Achsenendstopps und meldet, ob sie "ausgelöst" oder "offen" sind. Dieser Befehl wird in der Regel verwendet, um zu überprüfen, ob ein Endstopp korrekt funktioniert.

[resonance_tester].
Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt resonance_tester config section aktiviert ist (siehe auch die Anleitung zur Messung von Resonanzen measuring resonances guide).

MEASURE_AXES_NOISE
`MEASURE_AXES_NOISE:` Misst und gibt das Rauschen für alle Achsen aller aktivierten Beschleunigungsaufnehmer-Chips aus.

TEST_RESONANCES
`TEST_RESONANZEN AXIS=<Achse> OUTPUT=<Resonanzen,raw_data> [NAME=<name>] [FREQ_START=<min_freq>] [FREQ_END=<max_freq>] [HZ_PER_SEC=<hz_per_sec>] [CHIPS=<adxl345_chip_name>] [POINT=x,y,z] [INPUT_SHAPING=[<0:1>]]:` Führt den Resonanztest in allen konfigurierten Messpunkten für die angeforderte "Achse" durch und misst die Beschleunigung mit den für die jeweilige Achse konfigurierten Beschleunigungsmesserchips. "Achse" kann entweder X oder Y sein oder eine beliebige Richtung als AXIS=dx,dy angeben, wobei dx und dy Fließkommazahlen sind, die einen Richtungsvektor definieren (z. B. `AXIS=X, AXIS=Y` oder `AXIS=1,-1` zur Definition einer diagonalen Richtung). Beachten Sie, dass `AXIS=dx,dy` und `AXIS=-dx,-dy` gleichwertig sind. `adxl345_chip_name` kann ein oder mehrere konfigurierte adxl345-Chips sein, die durch ein Komma voneinander getrennt sind, zum Beispiel `CHIPS="adxl345, adxl345 rpi"`. Beachten Sie, dass adxl345 bei benannten adxl345-Chips weggelassen werden kann. Wenn POINT angegeben wird, werden die in `[resonance_tester]` konfigurierten Punkte außer Kraft gesetzt. Wenn `INPUT_SHAPING=0` oder nicht gesetzt ist (Standard), wird der Input Shaper für den Resonanztest deaktiviert, da es nicht zulässig ist, den Resonanztest mit aktiviertem Input Shaper durchzuführen. Der Parameter OUTPUT ist eine durch Kommata getrennte Liste der Ausgaben, die geschrieben werden sollen. Wenn `raw_data` angefordert wird, werden die Rohdaten des Beschleunigungsaufnehmers in eine Datei oder eine Reihe von Dateien `/tmp/raw_data_<axis>_[<chip_name>_][<point>_]<name>.csv` geschrieben `(<point>_` Teil des Namens wird nur erzeugt, wenn mehr als ein Messpunkt konfiguriert oder POINT angegeben ist). Wenn Resonanzen angegeben ist, wird der Frequenzgang (über alle

Messpunkte) berechnet und in die Datei /tmp/resonances_<axis>_<name>.csv geschrieben. Wenn nichts angegeben wird, ist OUTPUT auf Resonanzen voreingestellt, und NAME auf die aktuelle Zeit im Format "JJJJMMTT_HHMMSS".

SHAPER_CALIBRATE
SHAPER_CALIBRATE [AXIS=<axis>] [NAME=<name>] [FREQ_START=<min_freq>] [FREQ_END=<max_freq>] [HZ_PER_SEC=<hz_per_sec>] [MAX_SMOOTHING=<max_smoothing>]: Ähnlich wie bei TEST_RESONANCES wird der Resonanztest wie konfiguriert durchgeführt und versucht, die optimalen Parameter für den Eingangs-Shaper für die angeforderte Achse zu finden (oder sowohl für die X- als auch für die Y-Achse, wenn der Parameter AXIS nicht gesetzt ist). Wenn MAX_SMOOTHING nicht gesetzt ist, wird der Wert aus dem Abschnitt [resonance_tester] übernommen, wobei der Standardwert nicht gesetzt ist. Weitere Informationen über die Verwendung dieser Funktion finden Sie in der Anleitung Max smoothing of the measuring resonances. Die Ergebnisse der Abstimmung werden auf der Konsole ausgegeben, und die Frequenzgänge und die verschiedenen Werte der Eingangsformer werden in die CSV-Datei(en) /tmp/calibration_data_<axis>_<name>.csv geschrieben. Wenn nicht anders angegeben, ist NAME standardmäßig die aktuelle Zeit im Format "JJJJMMTT_HHMMSS". Beachten Sie, dass die vorgeschlagenen Input-Shaper-Parameter durch den Befehl SAVE_CONFIG in der Konfiguration erhalten bleiben können.

[respond]
Die folgenden Standard-G-Code-Befehle sind verfügbar, wenn der Abschnitt respond config aktiviert ist:

- M118 <message>: Echo der Nachricht mit vorangestelltem Standardpräfix (oder echo:, wenn kein Präfix konfiguriert ist).

Die folgenden zusätzlichen Befehle sind ebenfalls verfügbar.

RESPOND
RESPOND MSG="<message>": Echo der Nachricht, der das konfigurierte Standardpräfix vorangestellt wird (oder echo: wenn kein Präfix konfiguriert ist).
RESPOND TYPE=echo MSG="< message >": Echo der Nachricht mit vorangestelltem echo:.
RESPOND TYPE=command MSG="< message >": gibt die Nachricht mit vorangestelltem // aus. OctoPrint kann so konfiguriert werden, dass er auf diese Meldungen reagiert (z. B. RESPOND TYPE=command MSG=action:pause).
RESPOND TYPE=Fehler MSG="< message >": gibt die Meldung mit einem vorangestellten !! aus.
RESPOND PREFIX=<prefix> MSG="<Nachricht>": gibt die Nachricht mit vorangestelltem <prefix> aus. (Der Parameter PREFIX hat Vorrang vor dem Parameter TYPE)

[save_variables]
Der folgende Befehl wird aktiviert, wenn ein save_variables-Konfigurationsabschnitt aktiviert wurde.

SAVE_VARIABLE
SAVE_VARIABLE VARIABLE=<name> VALUE=<value>: Speichert die Variable auf der Festplatte, so dass sie über Neustarts hinweg verwendet werden kann. Alle gespeicherten Variablen werden beim Start in den Drucker printer.save_variables.variables geladen und können in Gcode-Makros verwendet werden. Der angegebene VALUE wird als Python-Literal geparst.

[screws_tilt_adjust]
Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt screws_tilt_adjust config section aktiviert ist (siehe auch den manual level guide).

SCREWS_TILT_CALCULATE
SCREWS_TILT_CALCULATE [DIRECTION=CW|CCW] [MAX_DEVIATION=<value>] [<probe_parameter>=<value>]: Mit diesem Befehl wird das Werkzeug zur Einstellung der Bettschrauben aufgerufen. Er steuert die Düse an verschiedene Stellen (wie in der Konfigurationsdatei definiert), tastet die z-Höhe ab und berechnet die Anzahl der Knopfumdrehungen, um das Bettniveau einzustellen. Wenn DIRECTION angegeben ist, werden die Drehknöpfe im Uhrzeigersinn (CW) oder gegen den Uhrzeigersinn (CCW) gedreht. Einzelheiten zu den optionalen Sondenparametern finden Sie unter dem Befehl PROBE. WICHTIG: Sie MÜSSEN immer einen G28 durchführen, bevor Sie diesen Befehl verwenden. Wenn MAX_DEVIATION angegeben wird, löst der Befehl einen G-Code-Fehler aus, wenn der Unterschied in der Schraubenhöhe relativ zur Basisschraubenhöhe größer ist als der angegebene Wert.

[sdcard_loop].
Wenn der Konfigurationsabschnitt sdcard_loop config section aktiviert ist, sind die folgenden erweiterten Befehle verfügbar.

SDCARD_LOOP_BEGIN
SDCARD_LOOP_BEGIN COUNT=<count>: Beginnt einen Schleifenabschnitt im SD-Druck. Eine Anzahl von 0 bedeutet, dass der Abschnitt in einer Endlosschleife ausgeführt werden soll.

SDCARD_LOOP_END
SDCARD_LOOP_END: Beendet einen geschleiften Abschnitt im SD-Druck.

SDCARD_LOOP_DESIST
SDCARD_LOOP_DESIST: Beendet bestehende Schleifen ohne weitere Iterationen.

[servo]

Die folgenden Befehle sind verfügbar, wenn ein Servo-Konfigurationsabschnitt servo config section aktiviert ist.

SET_SERVO

`SET_SERVO SERVO=config_name [ANGLE=<degrees> | WIDTH=<seconds>]:` Setzt die Servoposition auf den angegebenen Winkel (in Grad) oder die Pulsbreite (in Sekunden). Verwenden Sie WIDTH=0, um den Servoausgang zu deaktivieren.

[skew_correction]

Die folgenden Befehle sind verfügbar, wenn der Konfigurationsabschnitt skew_correction config section aktiviert ist (siehe auch die Anleitung zur Schräglaufkorrektur Skew Correction).

SET_SKEW

`SET_SKEW [XY=<ac_length,bd_length,ad_length>] [XZ=<ac,bd,ad>] [YZ=<ac,bd,ad>] [CLEAR=<0|1>]:` Konfiguriert das Modul [skew_correction] mit Messungen (in mm) aus einem Kalibrierungsdruck. Es können Messungen für beliebige Kombinationen von Ebenen eingegeben werden, nicht eingegebene Ebenen behalten ihren aktuellen Wert. Wenn `CLEAR=1` eingegeben wird, wird die Schräglagenkorrektur komplett deaktiviert.

GET_CURRENT_SKEW

`GET_CURRENT_SKEW:` Gibt die aktuelle Schräglage des Druckers für jede Ebene in Radiant und Grad an. Die Schräglage wird auf der Grundlage der über den `SET_SKEW` gcode bereitgestellten Parameter berechnet.

CALC_MEASURED_SKEW

`CALC_MEASURED_SKEW [AC=<ac_length>] [BD=<bd_length>] [AD=<ad_length>]:` Berechnet und meldet die Schräglage (in Radiant und Grad) auf der Grundlage eines gemessenen Drucks. Dies kann nützlich sein, um die aktuelle Schräglage des Druckers zu bestimmen, nachdem die Korrektur angewendet wurde. Es kann auch nützlich sein, bevor die Korrektur angewendet wird, um festzustellen, ob eine Schräglagenkorrektur erforderlich ist. Siehe Schräglagenkorrektur Skew Correction für Details zu Schräglagenkalibrierungsobjekten und -messungen.

SKEW PROFILE

`SKEW_PROFILE [LOAD=<name>] [SAVE=<name>] [REMOVE=<name>]:` Profilverwaltung für skew_correction. LOAD stellt den Schräglagenzustand aus dem Profil mit dem angegebenen Namen wieder her. SAVE speichert den aktuellen Schräglagenzustand in einem Profil mit dem angegebenen Namen. Remove löscht das Profil mit dem angegebenen Namen aus dem persistenten Speicher. Beachten Sie, dass nach der Ausführung von SAVE oder REMOVE der gcode SAVE_CONFIG ausgeführt werden muss, um die Änderungen im permanenten Speicher dauerhaft zu machen.

[smart_effector]

Mehrere Befehle sind verfügbar, wenn ein smart_effector-Konfigurationsabschnitt config section aktiviert ist. Bevor du die Parameter des Smart-Effektors änderst, solltest du die offizielle Dokumentation des Smart-Effektors im Duet3D Wiki lesen. Also check the probe calibration guide

SET_SMART_EFFECTOR

`SET_SMART_EFFECTOR [SENSITIVITY=<sensitivity>] [ACCEL=<accel>] [RECOVERY_TIME=<time>]:` Set the Smart Effector parameters. When `SENSITIVITY` is specified, the respective value is written to the SmartEffector EEPROM (requires control_pin to be provided). Acceptable <sensitivity> values are 0..255, the default is 50. Lower values require less nozzle contact force to trigger (but there is a higher risk of false triggering due to vibrations during probing), and higher values reduce false triggering (but require larger contact force to trigger). Since the sensitivity is written to EEPROM, it is preserved after the shutdown, and so it does not need to be configured on every printer startup. `ACCEL` and `RECOVERY_TIME` allow to override the corresponding parameters at run-time, see the config section config section of Smart Effector for more info on those parameters.

RESET_SMART_EFFECTOR

`RESET_SMART_EFFECTOR:` Resets Smart Effector sensitivity to its factory settings. Erfordert, dass control_pin in der Config-Sektion angegeben wird.

[stepper_enable]

Das Modul stepper_enable wird automatisch geladen.

SET_STEPPER_ENABLE

SET_STEPPER_ENABLE STEPPER=<config_name> ENABLE=[0|1]: Aktiviert oder deaktiviert nur den angegebenen Stepper. Dies ist ein Diagnose- und Debugging-Werkzeug und muss mit Vorsicht verwendet werden. Das Deaktivieren eines Achsenmotors setzt die Referenzfahrtinformationen nicht zurück. Das manuelle Bewegen eines deaktivierten Schrittmotors kann dazu führen, dass die Maschine den Motor außerhalb der sicheren Grenzen betreibt. Dies kann zu Schäden an Achskomponenten, heißen Enden und der Druckoberfläche führen.

[temperatur_lüfter]

Der folgende Befehl ist verfügbar, wenn ein temperature_fan config section -Abschnitt aktiviert ist.

SET_TEMPERATURE_FAN_TARGET
`SET_TEMPERATURE_FAN_TARGET temperature_fan=<temperature_fan_name> [target=<target_temperature>] [min_speed=<min_speed>] [max_speed=<max_speed>]:` Setzt die Zieltemperatur für einen temperature_fan. Wenn keine Zieltemperatur angegeben wird, wird sie auf die in der Konfigurationsdatei angegebene Temperatur gesetzt. Wenn keine Geschwindigkeiten angegeben werden, wird keine Änderung vorgenommen.

[tmcXXXX]

Die folgenden Befehle sind verfügbar, wenn einer der tmcXXXX-Konfigurationsabschnitte tmcXXXX config sections aktiviert ist.

DUMP_TMC
`DUMP_TMC STEPPER=<name>:` Dieser Befehl liest die TMC-Treiberregister und gibt deren Werte aus.

INIT_TMC
`INIT_TMC STEPPER=<name>:` Mit diesem Befehl werden die TMC-Register initialisiert. Wird benötigt, um den Treiber wieder zu aktivieren, wenn die Stromversorgung des Chips aus- und wieder eingeschaltet wird.

SET_TMC_CURRENT
`SET_TMC_CURRENT STEPPER=<name> CURRENT=<Ampere> HOLDCURRENT=<Ampere>:` Hiermit werden die Lauf- und Halteströme des TMC-Treibers eingestellt. (HOLDCURRENT ist nicht anwendbar auf tmc2660-Treiber.)

SET_TMC_FIELD
`SET_TMC_FIELD STEPPER=<name> FIELD=<field> VALUE=<value>:` Hiermit wird der Wert des angegebenen Registerfeldes des TMC-Treibers geändert. Dieser Befehl ist nur für Low-Level-Diagnosen und Debugging gedacht, da eine Änderung der Felder während der Laufzeit zu unerwünschtem und potenziell gefährlichem Verhalten Ihres Druckers führen kann. Dauerhafte Änderungen sollten stattdessen über die Druckerkonfigurationsdatei vorgenommen werden. Die angegebenen Werte werden nicht auf ihre Richtigkeit überprüft.

[toolhead]

Das Werkzeugkopfmodul wird automatisch geladen.

SET_VELOCITY_LIMIT
`SET_VELOCITY_LIMIT [VELOCITY=<value>] [ACCEL=<value>] [ACCEL_TO_DECEL=<value>] [SQUARE_CORNER_VELOCITY=<value>]:` Ändern Sie die Geschwindigkeitsgrenzen des Druckers.

[tuning_tower]

Das Modul tuning_tower wird automatisch geladen.

TUNING_TOWER
`TUNING_TOWER COMMAND=<command> PARAMETER=<name> START=<value> [SKIP=<value>] [FACTOR=<value> [BAND=<value>]] | [STEP_DELTA=<value> STEP_HEIGHT=<value>]:` Ein Werkzeug zur Einstellung eines Parameters für jede Z-Höhe während eines Druckvorgangs. Das Tool führt den gegebenen `COMMAND` mit dem gegebenen `PARAMETER` aus, dem ein Wert zugewiesen wird, der mit Z gemäß einer Formel variiert. Verwenden Sie `FACTOR`, wenn Sie ein Lineal oder einen Messschieber verwenden, um die Z-Höhe des optimalen Wertes zu messen, oder `STEP_DELTA` und `STEP_HEIGHT`, wenn das Abstimmturmmodell Schichten mit diskreten Werten hat, wie es bei Temperaturtürmen üblich ist. Wenn `SKIP=<value>` angegeben ist, beginnt der Abstimmungsprozess erst, wenn die `Z-Höhe <value>` erreicht ist, und unterhalb dieses Wertes wird der Wert auf START gesetzt; in diesem Fall ist die in den nachstehenden Formeln verwendete `z_height` tatsächlich `max(z - skip, 0)`. Es gibt drei mögliche Kombinationen von Optionen:

- `FACTOR:` Der Wert ändert sich mit einem Faktor pro Millimeter. Die verwendete Formel lautet: `value = start + faktor * z_height`. Sie können die optimale Z-Höhe direkt in die Formel einsetzen, um den optimalen Parameterwert zu ermitteln.
- `FACTOR` und `BAND:` Der Wert ändert sich mit einer durchschnittlichen Rate von Faktor pro Millimeter, aber in diskreten Schichten, in denen die Anpassung nur alle BAND-Millimeter der Z-Höhe vorgenommen wird. Die verwendete Formel lautet: `value = start + faktor * ((floor(z_height / band) + .5) * band)`.
- `STEP_DELTA` und `STEP_HEIGHT:` Der Wert ändert sich um `STEP_DELTA` alle `STEP_HEIGHT`-Millimeter. Die Formel lautet: `value = start + step_delta * floor(z_height / step_height)`. Sie können einfach die Schichten zählen oder die Beschriftungen der Tuningtower lesen, um den optimalen Wert zu ermitteln.

[virtual_sdcard]

Klipper unterstützt die folgenden Standard G-Code Befehle, wenn die virtual_sdcard config section aktiviert ist:

SD-Karte auflisten: `M20`
SD-Karte initialisieren: `M21`
SD-Datei auswählen: `M23 <filename>`
SD-Druck starten/fortsetzen: `M24`
SD-Druck anhalten: `M25`
SD-Position einstellen: `M26 S<offset>`

SD-Druckstatus melden: `M27`

Darüber hinaus sind die folgenden erweiterten Befehle verfügbar, wenn der Konfigurationsabschnitt "virtual_sdcard" aktiviert ist.

SDCARD_PRINT_FILE
`SDCARD_PRINT_FILE FILENAME=<Dateiname>:` Lädt eine Datei und startet den SD-Druck.

SDCARD_RESET_FILE
`SDCARD_RESET_FILE:` Entlade die Datei und lösche den SD-Status.

[z_tilt]
Die folgenden Befehle sind verfügbar, wenn der Abschnitt z_tilt config section aktiviert ist.

Z_TILT_ADJUST
`Z_TILT_ADJUST [<probe_parameter>=<value>]:` Dieser Befehl tastet die in der Konfiguration angegebenen Punkte ab und nimmt dann unabhängige Anpassungen an jedem Z-Stepper vor, um die Neigung auszugleichen. Siehe den PROBE-Befehl für Details zu den optionalen Probe-Parametern.

Befehlsvorlagen
Dieses Dokument enthält Informationen zur Implementierung von G-Code-Befehlssequenzen in gcode_macro (und ähnlichen) Konfigurationsabschnitten.

G-Code-Makro-Benennung
Die Groß- und Kleinschreibung spielt für den G-Code-Makronamen keine Rolle - MY_MACRO und my_macro werden gleich ausgewertet und können sowohl in Groß- als auch in Kleinschreibung aufgerufen werden. Wenn im Makronamen Zahlen verwendet werden, müssen diese am Ende des Namens stehen (z.B. TEST_MACRO25 ist gültig, aber MACRO25_TEST3 nicht).

Formatierung von G-Code in der config
Die Einrückung ist wichtig, wenn ein Makro in der Config-Datei definiert wird. Um eine mehrzeilige G-Code-Sequenz zu spezifizieren, ist es wichtig, dass jede Zeile eine korrekte Einrückung hat. Zum Beispiel:

```
[gcode_macro blink_led]
gcode:
  SET_PIN PIN=my_led VALUE=1
  G4 P2000
  SET_PIN PIN=my_led VALUE=0
```

Beachten Sie, dass die Option gcode: config immer am Anfang der Zeile beginnt und die nachfolgenden Zeilen des G-Code-Makros nie am Anfang beginnen.

Füge eine Beschreibung zu Deinem Makro hinzu
Um die Funktionalität zu verdeutlichen, kann eine kurze Beschreibung hinzugefügt werden. Beschreibung hinzufügen `description:` mit einem kurzen Text zur Beschreibung der Funktionalität. Standard ist "G-Code-Makro", wenn nichts angegeben wird. Zum Beispiel:

```
[gcode_macro blink_led]
description: Blink my_led one time
gcode:
  SET_PIN PIN=my_led VALUE=1
  G4 P2000
  SET_PIN PIN=my_led VALUE=0
```

Das Terminal zeigt die Beschreibung an, wenn Sie den `HELP`-Befehl oder die Autocomplete-Funktion verwenden.

Speichern/Wiederherstellen des Status für G-Code-Bewegungen
Leider kann die Verwendung der Befehlssprache G-Code eine Herausforderung darstellen. Der Standardmechanismus zum Verschieben des Werkzeugkopfes ist der `G1`-Befehl (der `G0`-Befehl ist ein Alias für G1 und kann austauschbar mit diesem verwendet werden). Dieser Befehl stützt sich jedoch auf den "G-Code-Parsing-Status", der durch `M82, M83, G90, G91, G92` und frühere `G1`-Befehle eingerichtet wurde. Bei der Erstellung eines G-Code-Makros ist es ratsam, den G-Code-Parsing-Status immer explizit zu setzen, bevor ein `G1`-Befehl erteilt wird. (Andernfalls besteht die Gefahr, dass der `G1`-Befehl eine unerwünschte Anforderung stellt.)

Ein üblicher Weg, dies zu erreichen, ist, die `G1`-Bewegungen in `SAVE_GCODE_STATE, G91` und `RESTORE_GCODE_STATE` zu verpacken. Zum Beispiel:

```
[gcode_macro MOVE_UP]
gcode:
```

```
  SAVE_GCODE_STATE NAME=my_move_up_state
  G91
  G1 Z10 F300
  RESTORE_GCODE_STATE NAME=my_move_up_state
```

Der Befehl `G91` versetzt den G-Code-Parsing-Status in den "relativen Bewegungsmodus", und der Befehl `RESTORE_GCODE_STATE` stellt den Zustand wieder her, der vor der Eingabe des Makros bestand. Achten Sie darauf, dass Sie beim ersten `G1`-Befehl eine explizite Geschwindigkeit angeben (über den Parameter `F`).

Vorlagenerweiterung

Der Abschnitt gcode_macro gcode`:` config wird mit der Jinja2-Vorlagensprache ausgewertet. Sie können Ausdrücke zur Laufzeit auswerten, indem Sie sie in `{ }`-Zeichen einschließen oder bedingte Anweisungen verwenden, die in `{% %}` eingeschlossen sind. Weitere Informationen über die Syntax finden Sie in der Jinja2-Dokumentation.

Ein Beispiel für ein komplexes Makro`:`

```
[gcode_macro clean_nozzle]
gcode:
  {% set wipe_count = 8 %}
  SAVE_GCODE_STATE NAME=clean_nozzle_state
  G90
  G0 Z15 F300
  {% for wipe in range(wipe_count) %}
    {% for coordinate in [(275, 4),(235, 4)] %}
      G0 X{coordinate[0]} Y{coordinate[1] + 0.25 * wipe} Z9.7 F12000
    {% endfor %}
  {% endfor %}
  RESTORE_GCODE_STATE NAME=clean_nozzle_state
```

Makro-Parameter

Es ist oft nützlich, die dem Makro beim Aufruf übergebenen Parameter zu überprüfen. Diese Parameter sind über die Pseudovariable `params` verfügbar. Zum Beispiel, wenn das Makro`:`

```
[gcode_macro SET_PERCENT]
gcode:
  M117 Jetzt bei { params.VALUE|float * 100 }%
```

als `SET_PERCENT VALUE=.2` aufgerufen würde, würde es `M117 Now at 20%` auswerten. Beachten Sie, dass Parameternamen immer in Großbuchstaben geschrieben werden, wenn sie im Makro ausgewertet werden, und dass sie immer als Zeichenketten übergeben werden. Bei mathematischen Berechnungen müssen sie explizit in Ganzzahlen oder Fließkommazahlen umgewandelt werden.

Es ist üblich, die Jinja2-Direktive set zu verwenden, um einen Standardparameter zu verwenden und das Ergebnis einem lokalen Namen zuzuweisen. Zum Beispiel`:`

```
[gcode_macro SET_BED_TEMPERATURE]
gcode:
  {% set bed_temp = params.TEMPERATURE|default(40)|float %}
  M140 S{bed_temp}
```

Die "rawparams"-Variable

Über die Pseudovariable "`rawparams`" kann auf die vollständigen, unparsedierten Parameter für das laufende Makro zugegriffen werden.

Dies ist sehr nützlich, wenn Sie das Verhalten bestimmter Befehle wie des `M117` ändern wollen. Zum Beispiel`:`

```
[gcode_macro M117]
rename_existing: M117.1
gcode:
  M117.1 { rawparams }
  M118 { rawparams }
```

Die "printer"-Variable

Es ist möglich, den aktuellen Zustand des Druckers über die Pseudovariable "`printer`" zu überprüfen (und zu ändern). Zum Beispiel`:`

```
[gcode_macro slow_fan]
gcode:
  M106 S{ printer.fan.speed * 0.9 * 255}
```

Die verfügbaren Felder sind im Dokument Statusreferenz Status Reference definiert.

Wichtig! Makros werden zunächst vollständig ausgewertet und erst dann werden die daraus resultierenden Befehle ausgeführt. Wenn ein Makro einen Befehl ausgibt, der den Zustand des Druckers ändert, sind die Ergebnisse dieser Zustandsänderung während der Auswertung des Makros nicht sichtbar. Dies kann auch zu einem subtilen Verhalten führen, wenn ein Makro Befehle erzeugt, die andere Makros aufrufen, da das aufgerufene Makro erst beim Aufruf ausgewertet wird (also nach der vollständigen Auswertung des aufrufenden Makros).

Nach der Konvention ist der Name, der unmittelbar auf `printer` folgt, der Name eines Konfigurationsabschnitts. So bezieht sich z.B. `printer.fan` auf das Lüfterobjekt, das durch den Konfigurationsabschnitt `[fan]` erzeugt wird. Es gibt einige Ausnahmen von dieser Regel - vor allem die Objekte `gcode_move` und toolhead. Wenn der Konfigurationsabschnitt Leerzeichen enthält, kann man über den Zugriffswert `[ ]` darauf zugreifen - zum Beispiel: `printer["generic_heater my_chamber_heater"].temperature`.

Beachten Sie, dass die Jinja2-Direktive set einem Objekt in der Druckerhierarchie einen lokalen Namen zuweisen kann. Dies kann die Lesbarkeit von Makros verbessern und die Tipparbeit reduzieren. Zum Beispiel:

```
[gcode_macro QUERY_HTU21D]
gcode:
    {% set sensor = printer["htu21d my_sensor"] %}
    M117 Temp:{sensor.temperature} Luftfeuchtigkeit:{sensor.humidity}
```

Actions

Es gibt einige Befehle, die den Zustand des Druckers ändern können. Zum Beispiel würde `{ action_emergency_stop() }` den Drucker in den Zustand des Herunterfahrens versetzen. Beachten Sie, dass diese Aktionen zu dem Zeitpunkt ausgeführt werden, zu dem das Makro ausgewertet wird, was eine beträchtliche Zeitspanne vor der Ausführung der generierten g-code-Befehle sein kann.

Verfügbare "action"-Befehle:

- `action_respond_info(msg):` Schreibt die angegebene Nachricht an das Pseudoterminal /tmp/printer. Jede Zeile von `msg` wird mit einem Präfix "// " gesendet.
- `action_raise_error(msg):` Bricht das aktuelle Makro (und alle aufrufenden Makros) ab und schreibt die angegebene Fehlermeldung an das Pseudoterminal /tmp/printer. Die erste Zeile von `msg` wird mit einem "!! "Präfix und die nachfolgenden Zeilen mit einem "//"-Präfix versehen.
- `action_emergency_stop(msg):` Versetzt den Drucker in den Zustand des Herunterfahrens. Der `msg`-Parameter ist optional, er kann nützlich sein, um den Grund für das Herunterfahren zu beschreiben.
- `action_call_remote_method(method_name):` Ruft eine Methode auf, die von einem entfernten Client registriert wurde. Wenn die Methode Parameter benötigt, sollten diese über Schlüsselwortargumente angegeben werden, d.h.:
  `action_call_remote_method("print_stuff", my_arg="hello_world")`

Variablen

Der Befehl `SET_GCODE_VARIABLE` kann nützlich sein, um den Zustand zwischen Makroaufrufen zu speichern. Variablennamen dürfen keine Großbuchstaben enthalten. Zum Beispiel:

```
[gcode_macro start_probe]
variable_bed_temp: 0
gcode:
  # Zieltemperatur in der Variable bed_temp speichern
  SET_GCODE_VARIABLE MACRO=start_probe VARIABLE=bed_temp VALUE={printer.heater_bed.target}
  # Bettheizung deaktivieren
  M140
  # Probe durchführen
  PROBE
  # Makro finish_probe nach Abschluss der Sonde aufrufen
  finish_probe

[gcode_macro finish_probe]
gcode:
  # Temperatur wiederherstellen
  M140 S{printer["gcode_makro start_probe"].bed_temp}
```

Achten Sie bei der Verwendung von SET_GCODE_VARIABLE auf das Timing der Makroauswertung und der Befehlsausführung.

Verzögerte Gcodes

Die Konfigurationsoption `[delayed_gcode]` kann verwendet werden, um eine verzögerte Gcode-Sequenz auszuführen:

```
[delayed_gcode clear_display]
gcode:
  M117

[gcode_macro load_filament]
gcode:
 G91
 G1 E50
 G90
 M400
 M117 load complete!
 UPDATE_DELAYED_GCODE ID=clear_display DURATION=10
```

Wenn das obige `load_filament`-Makro ausgeführt wird, wird nach Abschluss der Extrusion die Meldung "`Load Complete!`" angezeigt. Die letzte Zeile des gcode aktiviert den delayed_gcode "clear_display", der nach 10 Sekunden ausgeführt wird.

Die Konfigurationsoption `initial_duration` kann so eingestellt werden, dass der delayed_gcode beim Starten des Druckers ausgeführt wird. Der Countdown beginnt, wenn der Drucker in den Zustand "bereit" übergeht. Der folgende delayed_gcode wird zum Beispiel 5 Sekunden nach der Bereitschaft des Druckers ausgeführt und initialisiert das Display mit einer "`Welcome!`"-Meldung:

```
[delayed_gcode welcome]
initial_duration: 5.
gcode:
  M117 Welcome!
```

Es ist möglich, dass sich ein verzögerter Gcode wiederholt, indem er in der Gcode-Option aktualisiert wird:

```
[delayed_gcode report_temp]
initial_duration: 2.
gcode:
  {action_respond_info("Extruder Temp: %.1f" % (printer.extruder0.temperature))}
  UPDATE_DELAYED_GCODE ID=report_temp DURATION=2
```

Der obige delayed_gcode sendet alle 2 Sekunden "// Extruder Temp: [ex0_temp]" an Octoprint. Dies kann mit dem folgenden gcode abgebrochen werden:

```
UPDATE_DELAYED_GCODE ID=report_temp DURATION=0
```

Menüvorlagen

Wenn ein Anzeigekonfigurationsabschnitt display config section aktiviert ist, ist es möglich, das menu mit Menükonfigurationsabschnitten anzupassen.

Die folgenden schreibgeschützten Attribute sind in Menüvorlagen verfügbar:

- `menu.width` - Elementbreite (Anzahl der Anzeigespalten)
- `menu.ns` - Namespace des Elements
- `menu.event` - Name des Ereignisses, das das Skript ausgelöst hat
- `menu.input` - Eingabewert, nur im Kontext des Eingabeskripts verfügbar

Die folgenden Aktionen sind in Menüvorlagen verfügbar:

- menu.back(`force, update`): führt den Befehl menu back aus, optionale boolesche Parameter `<force>` und `<update>`.
  - Wenn `<force>` auf True gesetzt ist, wird auch die Bearbeitung gestoppt. Der Standardwert ist False.
  - Wenn `<update>` auf False gesetzt ist, werden übergeordnete Container-Elemente nicht aktualisiert. Der Standardwert ist True.
- `menu.exit(force)` - führt den Befehl zum Verlassen des Menüs aus, optionaler boolescher Parameter `<force>` Standardwert False.
  - Wenn `<force>` auf True gesetzt wird, dann wird auch die Bearbeitung beendet. Der Standardwert ist False.

Variablen auf Festplatte speichern

Wenn ein save_variables Konfigurationsabschnitt save_variables config section aktiviert wurde, kann `SAVE_VARIABLE VARIABLE=<name> VALUE=<value>` verwendet werden, um die Variable auf der Festplatte zu speichern, so dass sie über Neustarts hinweg verwendet werden kann. Alle gespeicherten Variablen werden beim Start in das Diktat printer.save_variables.variables geladen und können in Gcode-Makros verwendet werden. Um übermäßig lange Zeilen zu vermeiden, können Sie Folgendes am Anfang des Makros hinzufügen:

```
{% set svv = printer.save_variables.variables %}
```

So könnte man beispielsweise den Zustand des 2-in-1-out-Hotends speichern und beim Starten eines Drucks sicherstellen, dass der aktive Extruder anstelle von T0 verwendet wird:

```
[gcode_macro T1]
gcode:
  ACTIVATE_EXTRUDER extruder=extruder1
  SAVE_VARIABLE VARIABLE=aktuellerExtruder VALUE='"extruder1"'

[gcode_macro T0]
gcode:
  ACTIVATE_EXTRUDER extruder=extruder
  SAVE_VARIABLE VARIABLE=aktuellerExtruder VALUE='"extruder"'

[gcode_macro START_GCODE]
gcode:
  {% set svv = printer.save_variables.variables %}
  ACTIVATE_EXTRUDER extruder={svv.currentextruder}
```

Status-Referenz

Dieses Dokument ist eine Referenz der Druckerstatusinformationen, die in Klipper-Makros, Anzeigefeldern macros, display fields und über den API-Server API Server verfügbar sind.

Die Felder in diesem Dokument können sich ändern - wenn Sie ein Attribut verwenden, stellen Sie sicher, dass Sie das Dokument Config Changes document überprüfen, wenn Sie die Klipper Software aktualisieren.

Winkel

Die folgenden Informationen sind in angle some_nameObjekten verfügbar:

- `Temperatur:` Die letzte Temperaturmessung (in Celsius) von einem tle5012b magnetischen Hall-Sensor. Dieser Wert ist nur verfügbar, wenn es sich bei dem Winkelsensor um einen tle5012b-Chip handelt und wenn Messungen im Gange sind (andernfalls wird None gemeldet).

bed_mesh

Die folgenden Informationen sind im bed_mesh -Objekt verfügbar:

- `profile_name, mesh_min, mesh_max, probed_matrix, mesh_matrix:` Informationen über das derzeit aktive bed_mesh.
- `profiles:` Die Menge der aktuell definierten Profile, die mit BED_MESH_PROFILE eingerichtet wurden.

configfile

Die folgenden Informationen sind im `configfile`-Objekt verfügbar (dieses Objekt ist immer verfügbar):

- `settings.<section>.<option>:` Gibt die angegebene Einstellung der Konfigurationsdatei (oder den Standardwert) beim letzten Start oder Neustart der Software zurück. (Einstellungen, die während der Laufzeit geändert wurden, werden hier nicht berücksichtigt).
- `config.<section>.<option>:` Gibt die rohe Einstellung der Konfigurationsdatei zurück, wie sie von Klipper während des letzten Starts oder Neustarts der Software gelesen wurde. (Alle zur Laufzeit geänderten Einstellungen werden hier nicht berücksichtigt.) Alle Werte werden als Strings zurückgegeben.
- `save_config_pending:` Gibt true zurück, wenn es Aktualisierungen gibt, die ein `SAVE_CONFIG`-Befehl auf der Festplatte festhalten kann.
- `warnings:` Eine Liste von Warnungen über Konfigurationsoptionen. Jeder Eintrag in der Liste ist ein Wörterbuch, das einen Typ und ein Nachrichtenfeld (beides Strings) enthält. Je nach Art der Warnung können zusätzliche Felder verfügbar sein.

display_status

Die folgenden Informationen sind im display_status-Objekt verfügbar (dieses Objekt ist automatisch verfügbar, wenn ein display-config-Abschnitt definiert ist):

- `progress:` Der Fortschrittswert des letzten M73 G-Code Befehls (oder `virtual_sdcard.progress`, wenn kein neues `M73` empfangen wurde).
- `message:` Die im letzten `M117`-G-Code-Befehl enthaltene Nachricht.

endstop_phase¶

Die folgenden Informationen sind im endstop_phase-Objekt verfügbar:

- `last_home.<Schrittmotorname>.phase:` Die Phase des Schrittmotors am Ende des letzten Home-Versuchs.
- `last_home.<Schrittmachername>.phases:` Die Gesamtzahl der am Schrittmotor verfügbaren Phasen.
- `last_home.<Schrittmachername>.mcu_position:` Die (vom Mikrocontroller verfolgte) Position des Schrittmotors am Ende des letzten Referenzfahrtversuchs. Die Position ist die Gesamtzahl der Schritte in Vorwärtsrichtung abzüglich der Gesamtzahl der Schritte in Rückwärtsrichtung seit dem letzten Neustart des Mikrocontrollers.

exclude_object

Die folgenden Informationen sind im Objekt exclude_object verfügbar:

- `objects:` Ein Array der bekannten Objekte, wie sie durch den Befehl `EXCLUDE_OBJECT_DEFINE` bereitgestellt werden. Dies ist die gleiche Information, die der Befehl `EXCLUDE_OBJECT VERBOSE=1` liefert. Die Felder `center` und `polygon` sind nur vorhanden, wenn sie im ursprünglichen `EXCLUDE_OBJECT_DEFINE` angegeben wurden.

Hier ist ein JSON-Beispiel:

```
[
  {
    "polygon": [
      [ 156.25, 146.2511675 ],
      [ 156.25, 153.7488325 ],
      [ 163.75, 153.7488325 ],
      [ 163.75, 146.2511675 ]
    ],
    "name": "CYLINDER_2_STL_ID_2_COPY_0",
    "center": [ 160, 150 ]
  },
  {
    "polygon": [
      [ 146.25, 146.2511675 ],
      [ 146.25, 153.7488325 ],
      [ 153.75, 153.7488325 ],
      [ 153.75, 146.2511675 ]
    ],
    "name": "CYLINDER_2_STL_ID_1_COPY_0",
    "center": [ 150, 150 ]
  }
]
```

`excluded_objects:` Ein Array von Strings, das die Namen der ausgeschlossenen Objekte auflistet.
`Current object:` Der Name des Objekts, das gerade gedruckt wird.

fan

Die folgenden Informationen sind in den Objekten fanfan, heater_fan some_name und controller_fan some_name verfügbar:

- `speed:` Die Lüfterdrehzahl als Fließkommazahl zwischen 0,0 und 1,0.
- `rpm:` Die gemessene Lüfterdrehzahl in Umdrehungen pro Minute, wenn der Lüfter einen pwm_pin definiert hat.

filament_switch_sensor

Die folgenden Informationen sind in den Objekten filament_switch_sensor some_name verfügbar:

- `enabled:` Gibt True zurück, wenn der Schaltersensor derzeit aktiviert ist.
- `filament_detected:` Gibt True zurück, wenn sich der Sensor in einem ausgelösten Zustand befindet.

filament_motion_sensor

Die folgenden Informationen sind in den Objekten filament_motion_sensor some_nameverfügbar:

- `enabled:` Gibt True zurück, wenn der Bewegungssensor derzeit aktiviert ist.
- `filament_detected:` Gibt True zurück, wenn sich der Sensor in einem ausgelösten Zustand befindet.

firmware_retraction

Die folgenden Informationen sind im Objekt firmware_retractionverfügbar:

- `retract_length, retract_speed, unretract_extra_length, unretract_speed:` Die aktuellen Einstellungen für das firmware_retraction-Modul. Diese Einstellungen können von der Konfigurationsdatei abweichen, wenn sie durch einen `SET_RETRACTION`-Befehl geändert werden.

gcode_macro

Die folgenden Informationen sind in gcode_macro some_name-Objekten verfügbar:

- `<Variable>:` Der aktuelle Wert einer gcode_macro variable-Variable.

gcode_move

Die folgenden Informationen sind im Objekt gcode_move verfügbar (dieses Objekt ist immer verfügbar):

- `gcode_position:` Die aktuelle Position des Werkzeugkopfes in Bezug auf den aktuellen G-Code-Ursprung. Das heißt, Positionen, die man direkt an einen `G1`-Befehl senden könnte. Es ist möglich, auf die x-, y-, z- und e-Komponenten dieser Position zuzugreifen (z. B. `gcode_position.x`).
- `Position:` Die letzte befohlene Position des Werkzeugkopfes unter Verwendung des in der Konfigurationsdatei angegebenen Koordinatensystems. Es ist möglich, auf die x-, y-, z- und e-Komponenten dieser Position zuzugreifen (z. B. `position.x`).
- `homing_origin:` Der Ursprung des Gcode-Koordinatensystems (relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem), das nach einem `G28`-Befehl verwendet werden soll. Mit dem Befehl `SET_GCODE_OFFSET` kann diese Position geändert werden. Es ist möglich, auf die x-, y- und z-Komponenten dieser Position zuzugreifen (z. B. `homing_origin.x`).
- `speed:` Die zuletzt mit einem `G1`-Befehl eingestellte Geschwindigkeit (in mm/s).
- `speed_factor:` Der "Geschwindigkeitsfaktor-Override", wie er durch einen `M220`-Befehl gesetzt wurde. Dies ist ein Fließkommawert, so dass 1,0 keine Übersteuerung bedeutet und 2,0 beispielsweise die geforderte Geschwindigkeit verdoppeln würde.
- `extrude_factor:` Der "Extrude-Faktor-Override", der durch einen `M221`-Befehl festgelegt wird. Hierbei handelt es sich um einen Fließkommawert, so dass 1,0 keine Übersteuerung bedeutet und 2,0 beispielsweise eine Verdoppelung der angeforderten Extrusion bedeuten würde.
- `absolute_coordinates:` Gibt True zurück, wenn es sich um einen absoluten `G90`-Koordinatenmodus handelt, oder False, wenn es sich um einen relativen `G91`-Modus handelt.
- `absolute_extrude:` Gibt True zurück, wenn es sich um den absoluten Extrudiermodus `M82` handelt, oder False, wenn es sich um den relativen Modus `M83` handelt.

hall_filament_width_sensor

Die folgenden Informationen sind im hall_filament_width_sensor-Objekt verfügbar:

- `is_active:` Gibt True zurück, wenn der Sensor derzeit aktiv ist.
- `Durchmesser:` Der letzte Messwert des Sensors in mm.
- `Raw:` Der letzte rohe ADC-Messwert des Sensors.

heater

Die folgenden Informationen sind für Heizungsobjekte wie extruder, heater_bed, and heater_generic:

- `temperature:` Die zuletzt gemeldete Temperatur (in Celsius als Fließkommazahl) für die angegebene Heizung.
- `target:` Die aktuelle Zieltemperatur (in Celsius als Float) für das angegebene Heizelement.
- `power:` Die letzte Einstellung des PWM-Pins (ein Wert zwischen 0,0 und 1,0), der mit dem Heizer verbunden ist.
- `can_extrude:` Wenn der Extruder extrudieren kann (definiert durch `min_extrude_temp`), verfügbar nur für extruder

heaters

Die folgenden Informationen sind im Objekt `heaters` verfügbar (dieses Objekt ist verfügbar, wenn eine Heizung definiert ist):

- `available_heaters:` Liefert eine Liste aller derzeit verfügbaren Heizungen mit den vollständigen Namen ihrer Konfigurationsabschnitte, z. B. `["extruder", "heater_bed", "heater_generic my_custom_heater"]`.
- `verfügbare_sensoren:` Gibt eine Liste aller derzeit verfügbaren Temperatursensoren anhand ihrer vollständigen Namen im Konfigurationsabschnitt zurück, z. B. `["extruder", "heater_bed", "heater_generic my_custom_heater", "temperature_sensor electronics_temp"]`.

idle_timeout

Die folgenden Informationen sind im Objekt idle_timeout verfügbar (dieses Objekt ist immer verfügbar):

- `state:` Der aktuelle Zustand des Druckers, wie er vom Modul idle_timeout erfasst wird. Es handelt sich um eine der folgenden Zeichenketten: `"Idle", "Printing", "Ready"`.
- `printing_time:` Die Zeit (in Sekunden), die sich der Drucker im Zustand "Printing" befunden hat (wie vom Modul idle_timeout erfasst).

led

Die folgenden Informationen sind für jeden in printer.cfg definierten Konfigurationsabschnitt [led led_name], [neopixel led_name], [dotstar led_name], [pca9533 led_name] und [pca9632 led_name] verfügbar:

- `color_data:` Eine Liste von Farblisten, die die RGBW-Werte für eine LED in der Kette enthält. Jeder Wert wird als Float von 0,0 bis 1,0 dargestellt. Jede Farbliste enthält 4 Elemente (rot, grün, blau, weiß), auch wenn die untergeordnete LED weniger Farbkanäle unterstützt. Auf den Blauwert (drittes Element in der Farbliste) des zweiten Neopixels in einer Kette kann zum Beispiel `unter printer["neopixel <config_name>"].color_data[1][2]` zugegriffen werden.

mcu

Die folgenden Informationen sind in den Objekten mcu and mcu some_name verfügbar:

- `mcu_version:` Die vom Mikrocontroller gemeldete Klipper-Code-Version.
- `mcu_build_versions:` Informationen über die zur Generierung des Mikrocontroller-Codes verwendeten Build-Tools (wie vom Mikrocontroller gemeldet).
- `mcu_constants.<konstante_name>:` Vom Mikrocontroller gemeldete Kompilierzeitkonstanten. Die verfügbaren Konstanten können sich zwischen Mikrocontroller-Architekturen und mit jeder Code-Revision unterscheiden.
- `last_stats.<statistics_name>:` Statistikinformationen über die Mikrocontroller-Verbindung.

motion_report

Die folgenden Informationen sind im motion_report-Objekt verfügbar (dieses Objekt ist automatisch verfügbar, wenn ein Stepper-Konfigurationsabschnitt definiert ist):

- `live_position:` Die angeforderte Werkzeugkopfposition, interpoliert auf die aktuelle Zeit.
- `live_velocity:` Die angeforderte Werkzeugkopfgeschwindigkeit (in mm/s) zum aktuellen Zeitpunkt.
- `live_extruder_velocity:` Die angeforderte Extruder-Geschwindigkeit (in mm/s) zum aktuellen Zeitpunkt.

output_pin

Die folgenden Informationen sind in output_pin some_name-Objekten verfügbar:
`value:` Der "value" des Pins, wie er durch einen `SET_PIN`-Befehl festgelegt wurde.

palette2

Die folgenden Informationen sind im Objekt palette2verfügbar:

- `ping:` Höhe des letzten gemeldeten Palette-2-Pings in Prozent.
- `remaining_load_length:` Beim Starten eines Palette-2-Drucks ist dies die Menge an Filament, die in den Extruder geladen werden muss.
- `is_splicing:` Wahr, wenn die Palette 2 Filament spleißt.

pause_resume

Die folgenden Informationen sind im Objekt pause_resumeverfügbar:

- `is_paused:` Gibt true zurück, wenn ein PAUSE-Befehl ohne ein entsprechendes RESUME ausgeführt wurde.

print_stats

Die folgenden Informationen sind im print_stats-Objekt verfügbar (dieses Objekt ist automatisch verfügbar, wenn ein virtual_sdcard-Konfigurationsabschnitt definiert ist):

- `filename, total_duration, print_duration, filament_used, state, message:` Geschätzte Informationen über den aktuellen Druck, wenn ein virtual_sdcard-Druck aktiv ist.

probe

Die folgenden Informationen sind im probe-Objekt verfügbar (dieses Objekt ist auch verfügbar, wenn ein bltouch config-Abschnitt definiert ist):

- `last_query:` Gibt True zurück, wenn die Sonde beim letzten QUERY_PROBE-Befehl als "ausgelöst" gemeldet wurde. Hinweis: Wenn dies in einem Makro verwendet wird, muss der QUERY_PROBE-Befehl aufgrund der Reihenfolge der Vorlagenexpansion vor dem Makro ausgeführt werden, das diese Referenz enthält.
- `last_z_result:` Gibt den Z-Ergebniswert des letzten PROBE-Befehls zurück. Wird dies in einem Makro verwendet, muss der PROBE-Befehl (oder ein ähnlicher Befehl) aufgrund der Reihenfolge der Vorlagenexpansion vor dem Makro ausgeführt werden, das diesen Verweis enthält.

quad_gantry_level

Die folgenden Informationen sind im Objekt quad_gantry_level verfügbar (dieses Objekt ist verfügbar, wenn `quad_gantry_level` definiert ist):

- `applied:` True, wenn der Gantry-Leveling-Prozess ausgeführt und erfolgreich abgeschlossen wurde.

query_endstops

Die folgenden Informationen sind im Objekt query_endstops verfügbar (dieses Objekt ist verfügbar, wenn ein Endstop definiert ist):

- `last_query["<endstop>"]:` Gibt True zurück, wenn der angegebene Endstop während des letzten QUERY_ENDSTOP-Befehls als "ausgelöst" gemeldet wurde. Hinweis: Wenn dies in einem Makro verwendet wird, muss der QUERY_ENDSTOP-Befehl aufgrund der Reihenfolge der Schablonenexpansion vor dem Makro ausgeführt werden, das diese Referenz enthält.

servo

Die folgenden Informationen sind in servo some_name-Objekten verfügbar:

- `printer["servo <config_name>"].value:` Die letzte Einstellung des PWM-Pins (ein Wert zwischen 0,0 und 1,0), der dem Servo zugeordnet ist.

system_stats

Die folgenden Informationen sind im Objekt system_stats verfügbar (dieses Objekt ist immer verfügbar):

- `sysload, cputime, memavail:` Informationen über das Host-Betriebssystem und die Prozessauslastung.

temperature sensors

Die folgenden Informationen sind verfügbar in

bme280 config_section_name, htu21d config_section_name, lm75 config_section_name, and temperature_host config_section_name Objekte:

- `temperature:` Die zuletzt vom Sensor gelesene Temperatur.
- `humidity, pressure, gas:` Die zuletzt vom Sensor gelesenen Werte (nur bei den Sensoren bme280, htu21d und lm75).

temperatur_lüfter

Die folgenden Informationen sind in temperature_fan some_name Objekten verfügbar:

Temperatur: Die zuletzt vom Sensor gelesene Temperatur.
target: Die Zieltemperatur für den Lüfter.

temperature_sensor

Die folgenden Informationen sind in temperature_fan some_name -Objekten verfügbar:

- `temperature:` Die zuletzt vom Sensor gelesene Temperatur.
- `measured_min_temp, measured_max_temp:` Die niedrigste und höchste Temperatur, die der Sensor seit dem letzten Neustart der Klipper-Host-Software gemessen hat.

tmc drivers

Die folgenden Informationen sind in TMC stepper driver -Objekten verfügbar (z. B. [tmc2208 stepper_x]):

- `mcu_phase_offset:` Die Mikrocontroller-Schrittmacherposition, die der "Null"-Phase des Treibers entspricht. Dieses Feld kann Null sein, wenn der Phasenoffset nicht bekannt ist.
- `phase_offset_position:` Die "befohlene Position", die der "Null"-Phase des Treibers entspricht. Dieses Feld kann Null sein, wenn der Phasenversatz nicht bekannt ist.
- `drv_status:` Die Ergebnisse der letzten Abfrage des Fahrerstatus. (Es werden nur Felder ungleich Null gemeldet.) Dieses Feld ist Null, wenn der Treiber nicht aktiviert ist (und daher nicht regelmäßig abgefragt wird).
- `run_current:` Der aktuell eingestellte Laufstrom.
- `hold_current:` Der aktuell eingestellte Haltestrom.

toolhead

Die folgenden Informationen sind im Toolhead-Objekt verfügbar (dieses Objekt ist immer verfügbar):

- `position:` Die letzte befohlene Position des Werkzeugkopfes relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem. Es ist möglich, auf die x-, y-, z- und e-Komponenten dieser Position zuzugreifen (z.B. `position.x`).
- `Extruder:` Der Name des derzeit aktiven Extruders. In einem Makro könnte man zum Beispiel `printer[printer.toolhead.extruder].target` verwenden, um die Zieltemperatur des aktuellen Extruders zu erhalten.
- `homed_axes:` Die aktuellen kartesischen Achsen, die sich in einem "homed" Zustand befinden. Dies ist eine Zeichenkette, die eine oder mehrere der Angaben "x", "y", "z" enthält.
- `axis_minimum, axis_maximum:` Die Achsenverfahrgrenzen (mm) nach der Referenzfahrt. Es ist möglich, auf die x-, y- und z-Komponenten dieses Grenzwertes zuzugreifen (z. B. `axis_minimum.x, axis_maximum.z`).
- `max_velocity, max_accel, max_accel_to_decel, square_corner_velocity:` Die aktuellen Druckgrenzen, die in Kraft sind. Diese können von den Einstellungen in der Konfigurationsdatei abweichen, wenn sie zur Laufzeit durch den Befehl `SET_VELOCITY_LIMIT` (oder `M204`) geändert werden.
- `Stalls` (Abbrüche): Die Gesamtzahl der Male (seit dem letzten Neustart), in denen der Drucker angehalten werden musste, weil sich der Werkzeugkopf schneller bewegte, als Bewegungen aus der G-Code-Eingabe gelesen werden konnten.

dual_carriage
Die folgenden Informationen sind in dual_carriage auf einem hybrid_corexy oder hybrid_corexz Roboter verfügbar

- `Modus:` Der aktuelle Modus. Mögliche Werte sind: "FULL_CONTROL"
- `active_carriage:` Der aktuell aktive Wagen. Mögliche Werte sind: "CARRIAGE_0", "CARRIAGE_1"

virtual_sdcard
Die folgenden Informationen sind im Objekt dual_carriage verfügbar:

- `is_active:` Gibt True zurück, wenn ein Druck aus der Datei gerade aktiv ist.
- `progress:` Eine Schätzung des aktuellen Druckfortschritts (basierend auf Dateigröße und Dateiposition).
- `file_path:` Ein vollständiger Pfad zur Datei der aktuell geladenen Datei.
- `datei_position:` Die aktuelle Position (in Bytes) eines aktiven Drucks.
- `file_size:` Die Dateigröße (in Bytes) der aktuell geladenen Datei.

webhooks
Die folgenden Informationen sind im `webhooks`-Objekt verfügbar (dieses Objekt ist immer verfügbar):

- `state:` Gibt eine Zeichenkette zurück, die den aktuellen Klipper-Status angibt. Mögliche Werte sind: "ready", "startup", "shutdown", "error".
- `state_message:` Eine menschenlesbare Zeichenkette, die zusätzlichen Kontext über den aktuellen Klipper-Status liefert.

z_tilt
Die folgenden Informationen sind im z_tilt-Objekt verfügbar (dieses Objekt ist verfügbar, wenn `z_tilt` definiert ist):

- `applied:` True, wenn der z-tilt Nivellierungsprozess ausgeführt und erfolgreich abgeschlossen wurde.

TMC-Treiber
Dieses Dokument enthält Informationen zur Verwendung von Trinamic-Schrittmotortreibern TMC driver config reference im SPI/UART-Modus auf Klipper.

Klipper kann Trinamic-Treiber auch in ihrem "Standalone-Modus" verwenden. Wenn sich die Treiber jedoch in diesem Modus befinden, ist keine spezielle Klipper-Konfiguration erforderlich und die in diesem Dokument beschriebenen erweiterten Klipper-Funktionen sind nicht verfügbar.

Zusätzlich zu diesem Dokument sollten Sie unbedingt die Referenz zur TMC-Treiberkonfiguration TMC driver config reference lesen.

Abstimmung des Motorstroms
Ein höherer Treiberstrom erhöht die Positioniergenauigkeit und das Drehmoment. Ein höherer Strom erhöht jedoch auch die vom Schrittmotor und dem Schrittmotortreiber erzeugte Wärme. Wenn der Schrittmotortreiber zu heiß wird, schaltet er sich ab und Klipper meldet einen Fehler. Wenn der Schrittmotor zu heiß wird, verliert er an Drehmoment und Positioniergenauigkeit. (Wenn er sehr heiß wird, kann er auch Kunststoffteile schmelzen, die an ihm oder in seiner Nähe angebracht sind).

Als allgemeiner Tuning-Tipp sollten Sie höhere Stromwerte bevorzugen, solange der Schrittmotor nicht zu heiß wird und der Schrittmotortreiber keine Warnungen oder Fehler meldet. Im Allgemeinen ist es in Ordnung, wenn sich der Schrittmotor warm anfühlt, aber er sollte nicht so heiß werden, dass es schmerzhaft ist, ihn zu berühren.

Vorzugsweise keinen hold_current angeben
Wenn man einen `hold_current` konfiguriert, kann der TMC-Treiber den Strom zum Schrittmotor reduzieren, wenn er feststellt, dass sich der Schrittmotor nicht bewegt. Die Änderung des Motorstroms kann jedoch selbst zu einer Motorbewegung führen. Dies kann aufgrund von "Rastkräften" innerhalb des Schrittmotors (der Permanentmagnet im Rotor zieht an den Eisenzähnen im Stator) oder aufgrund von externen Kräften auf den Achsschlitten geschehen.

Bei den meisten Schrittmotoren bringt eine Stromreduzierung während normaler Druckvorgänge keine nennenswerten Vorteile, da nur wenige Druckvorgänge einen Schrittmotor lange genug im Leerlauf lassen, um die Funktion `hold_current` zu aktivieren. Außerdem ist es unwahrscheinlich, dass man bei den wenigen Druckbewegungen, bei denen ein Schrittmotor ausreichend lange im Leerlauf ist, subtile Druckartefakte erzeugen möchte.

Wenn man die Stromzufuhr zu den Motoren während der Druckstartroutinen reduzieren möchte, sollte man die Ausgabe von SET_TMC_CURRENT-Befehlen in einem START_PRINT macro-Makro in Betracht ziehen, um den Strom vor und nach normalen Druckbewegungen anzupassen.

Einige Drucker mit dedizierten Z-Motoren, die während normaler Druckbewegungen (kein bed_mesh, kein bed_tilt, keine Z skew_correction, keine "Vasenmodus"-Drucke usw.) im Leerlauf sind, können feststellen, dass die Z-Motoren mit einem `hold_current` kühler laufen. Wenn Sie dies implementieren, müssen Sie diese Art der unkontrollierten Z-Achsenbewegung während der Nivellierung des Bettes, der Bettantastung, der Kalibrierung

des Tasters und Ähnlichem berücksichtigen. Die Parameter `driver_TPOWERDOWN` und `driver_IHOLDDELAY` sollten ebenfalls entsprechend kalibriert werden. Wenn Sie unsicher sind, geben Sie lieber `hold_current` an.

Einstellung des "spreadCycle" vs. "stealthChop" Modus

Standardmäßig setzt Klipper die TMC-Treiber in den "spreadCycle"-Modus. Wenn der Treiber "stealthChop" unterstützt, kann er durch Hinzufügen von `stealthchop_threshold: 999999` in den TMC-Konfigurationsabschnitt einfügt.

Im Allgemeinen bietet der SpreadCycle-Modus ein größeres Drehmoment und eine höhere Positionsgenauigkeit als der StealthChop-Modus. Allerdings kann der stealthChop-Modus bei einigen Druckern deutlich weniger hörbare Geräusche erzeugen.

Tests, bei denen die Modi verglichen wurden, haben gezeigt, dass sich die "Positionsverzögerung" bei Bewegungen mit konstanter Geschwindigkeit um etwa 75 % eines Vollschritts erhöht, wenn der stealthChop-Modus verwendet wird (bei einem Drucker mit 40 mm Rotationsabstand und 200 Schritten pro Umdrehung erhöhte sich die Positionsabweichung bei Bewegungen mit konstanter Geschwindigkeit beispielsweise um ~0,150 mm). Diese "Verzögerung beim Erreichen der angeforderten Position" kann sich jedoch nicht als signifikanter Druckfehler erweisen und man kann das ruhigere Verhalten des stealthChop-Modus vorziehen.

Es wird empfohlen, immer den Modus "spreadCycle" (ohne Angabe von `stealthchop_threshold`) oder immer den Modus "stealthChop" (mit `stealthchop_threshold auf 999999`) zu verwenden. Leider liefern die Treiber oft schlechte und verwirrende Ergebnisse, wenn der Modus geändert wird, während der Motor eine Geschwindigkeit ungleich Null hat.

Die TMC-Interpolationseinstellung führt zu kleinen Positionsabweichungen

Die Interpolationseinstellung des TMC-Treibers kann das hörbare Geräusch der Druckerbewegung um den Preis eines kleinen systemischen Positionsfehlers reduzieren. Dieser systemische Positionsfehler resultiert aus der Verzögerung des Treibers bei der Ausführung von "Schritten", die Klipper ihm sendet. Bei Bewegungen mit konstanter Geschwindigkeit führt diese Verzögerung zu einem Positionsfehler von fast einem halben konfigurierten Mikroschritt (genauer gesagt ist der Fehler ein halber Mikroschrittabstand minus ein 512stel eines Vollschrittabstands). Bei einer Achse mit einem Rotationsabstand von 40 mm, 200 Schritten pro Umdrehung und 16 Mikroschritten beträgt der systembedingte Fehler bei Bewegungen mit konstanter Geschwindigkeit beispielsweise ~0,006 mm.

Für eine optimale Positionsgenauigkeit sollten Sie den SpreadCycle-Modus verwenden und die Interpolation deaktivieren (setzen Sie `interpolate: False` in der TMC-Treiberkonfiguration). Bei dieser Konfiguration kann man die Mikroschritt-Einstellung erhöhen, um die hörbaren Geräusche während der Stepperbewegung zu reduzieren. Typischerweise hat eine Mikroschritt-Einstellung von `64` oder `128` ein ähnliches hörbares Geräusch wie die Interpolation, und zwar ohne einen systemischen Positionsfehler.

Wenn der StealthChop-Modus verwendet wird, ist die Positionsungenauigkeit der Interpolation im Vergleich zur Positionsungenauigkeit des StealthChop-Modus gering. Daher wird die Einstellung der Interpolation im stealthChop-Modus nicht als sinnvoll erachtet, und man kann die Interpolation in ihrem Standardzustand belassen.

Sensorlose Referenzfahrt

Die sensorlose Referenzfahrt ermöglicht die Referenzfahrt einer Achse ohne einen physikalischen Endschalter. Stattdessen wird der Schlitten auf der Achse in die mechanische Begrenzung bewegt, wodurch der Schrittmotor Schritte verliert. Der Schrittmotortreiber erkennt die verlorenen Schritte und meldet dies an die steuernde MCU (Klipper) durch Umschalten eines Pins. Diese Information kann von Klipper als Endanschlag für die Achse verwendet werden.

Diese Anleitung behandelt die Einrichtung der sensorlosen Referenzfahrt für die X-Achse Ihres (kartesischen) Druckers. Es funktioniert jedoch genauso mit allen anderen Achsen (die einen Endanschlag benötigen). Sie sollten sie jeweils für eine Achse konfigurieren und abstimmen.

Beschränkungen

Vergewissern Sie sich, dass Ihre mechanischen Komponenten in der Lage sind, die Belastung durch das wiederholte Anstoßen des Schlittens an die Achsengrenze zu bewältigen. Insbesondere Spindeln können eine große Kraft erzeugen. Das Homing einer Z-Achse durch Anstoßen der Düse an die Druckoberfläche ist möglicherweise keine gute Idee. Die besten Ergebnisse erzielen Sie, wenn Sie sicherstellen, dass der Achsschlitten einen festen Kontakt mit der Achsenbegrenzung hat.

Außerdem ist die sensorlose Referenzfahrt für Ihren Drucker möglicherweise nicht genau genug. Während die Referenzierung der X- und Y-Achse auf einer kartesischen Maschine gut funktionieren kann, ist die Referenzierung der Z-Achse im Allgemeinen nicht genau genug und kann zu einer inkonsistenten Höhe der ersten Schicht führen. Die sensorlose Referenzfahrt eines Deltadruckers ist aufgrund der fehlenden Genauigkeit nicht ratsam.

Außerdem ist die Blockiererkennung des Schrittmotortreibers abhängig von der mechanischen Belastung des Motors, dem Motorstrom und der Motortemperatur (Spulenwiderstand).

Die sensorlose Referenzfahrt funktioniert am besten bei mittleren Motordrehzahlen. Bei sehr langsamen Drehzahlen (weniger als 10 U/min) erzeugt der Motor keine nennenswerte Gegen-EMK, und das TMC kann einen Motorstillstand nicht zuverlässig erkennen. Außerdem nähert sich die Gegen-EMK des Motors bei sehr hohen Drehzahlen der Versorgungsspannung des Motors an, so dass das TMC keinen Motorstillstand mehr erkennen kann. Es ist ratsam, einen Blick in das Datenblatt Ihres spezifischen TMCs zu werfen. Dort finden Sie auch weitere Einzelheiten zu den Einschränkungen dieses Aufbaus.

Voraussetzungen

Um die sensorlose Referenzfahrt zu nutzen, sind einige Voraussetzungen erforderlich:

1. Ein stallGuard-fähiger TMC-Schrittmachertreiber (tmc2130, tmc2209, tmc2660, oder tmc5160).
2. SPI / UART-Schnittstelle des TMC-Treibers mit Mikrocontroller verdrahtet (Stand-Alone-Modus funktioniert nicht).
3. Der entsprechende "DIAG"- oder "SG_TST"-Pin des TMC-Treibers muss mit dem Mikrocontroller verbunden sein.
4. Die Schritte im Dokument config checks müssen ausgeführt werden, um sicherzustellen, dass die Schrittmotoren konfiguriert sind und ordnungsgemäß funktionieren.

Abstimmung

Das hier beschriebene Verfahren umfasst sechs Hauptschritte:

1. Wählen Sie eine Referenzfahrtgeschwindigkeit.
2. Konfigurieren Sie die Datei printer.cfg, um die sensorlose Referenzfahrt zu aktivieren.
3. Finden Sie die Stallguard-Einstellung mit der höchsten Empfindlichkeit, die eine erfolgreiche Referenzfahrt ermöglicht.
4. Ermitteln Sie die Stallguard-Einstellung mit der niedrigsten Empfindlichkeit, die eine erfolgreiche Referenzfahrt mit einer einzigen Berührung ermöglicht.
5. Aktualisieren Sie die Datei `printer.cfg` mit der gewünschten stallguard-Einstellung.
6. Erstellen oder aktualisieren Sie die Makros in der Datei `printer.cfg`, um eine konsistente Referenzfahrt durchzuführen.

Referenzfahrtgeschwindigkeit wählen

Die Referenzfahrtgeschwindigkeit ist eine wichtige Entscheidung bei der sensorlosen Referenzfahrt. Es ist wünschenswert, eine langsame Referenzfahrtgeschwindigkeit zu verwenden, damit der Wagen keine übermäßige Kraft auf den Rahmen ausübt, wenn er das Ende der Schiene berührt. Allerdings können die TMC-Treiber bei sehr langsamen Geschwindigkeiten einen Strömungsabriss nicht zuverlässig erkennen.

Ein guter Ausgangspunkt für die Referenzfahrtgeschwindigkeit ist, dass der Schrittmotor alle zwei Sekunden eine volle Umdrehung macht. Bei vielen Achsen ist dies der Rotation_distance geteilt durch zwei. Zum Beispiel:

```
[stepper_x]
rotation_distance: 40
homing_speed: 20
...
```

printer.cfg für sensorlose Referenzfahrt konfigurieren

Die Einstellung `homing_retract_dist` muss im Konfigurationsabschnitt `stepper_x` auf Null gesetzt werden, um die zweite Referenzfahrt zu deaktivieren. Der zweite Referenzfahrtversuch bringt bei der sensorlosen Referenzfahrt keinen Mehrwert, er funktioniert nicht zuverlässig und verwirrt den Abstimmungsprozess.

Vergewissern Sie sich, dass im TMC-Treiber-Abschnitt der Konfiguration keine `hold_current`-Einstellung angegeben ist (wenn eine hold_current-Einstellung vorgenommen wird, stoppt der Motor, nachdem der Kontakt hergestellt wurde, während der Schlitten gegen das Ende der Schiene gedrückt wird, und eine Reduzierung des Stroms in dieser Position kann dazu führen, dass sich der Schlitten bewegt - das führt zu einer schlechten Leistung und verwirrt den Abstimmungsprozess).

Es ist notwendig, die Pins für die sensorlose Referenzfahrt zu konfigurieren und die anfänglichen "stallguard"-Einstellungen zu konfigurieren. Eine tmc2209-Beispielkonfiguration für eine X-Achse könnte wie folgt aussehen:

```
[tmc2209 stepper_x]
diag_pin: ^PA1 # Einstellung auf MCU-Pin, der mit dem TMC DIAG-Pin verbunden ist
driver_SGTHRS: 255 # 255 ist der empfindlichste Wert, 0 der am wenigsten empfindliche
...

[stepper_x]
endstop_pin: tmc2209_stepper_x:virtual_endstop
homing_retract_dist: 0
...
```

Eine tmc2130- oder tmc5160-Konfiguration könnte beispielsweise so aussehen:

```
[tmc2130 stepper_x]
diag1_pin: ^!PA1 # Pin verbunden mit TMC DIAG1 Pin (oder diag0_pin / DIAG0 Pin verwenden)
driver_SGT: -64 # -64 ist der empfindlichste Wert, 63 ist der unempfindlichste
...

[stepper_x]
endstop_pin: tmc2130_stepper_x:virtual_endstop
homing_retract_dist: 0
```

```
...
```

Eine tmc2660-Konfiguration könnte beispielsweise so aussehen:

```
[tmc2660 stepper_x]
driver_SGT: -64 # -64 ist der empfindlichste Wert, 63 ist der am wenigsten empfindliche
...

[stepper_x]
endstop_pin: ^PA1 # Pin verbunden mit TMC SG_TST Pin
homing_retract_dist: 0
...
```

Die obigen Beispiele zeigen nur die Einstellungen für die sensorlose Referenzfahrt. In der Konfigurationsreferenz finden Sie alle verfügbaren Optionen.

Höchste Empfindlichkeit für erfolgreiche Referenzfahrt finden
Platzieren Sie den Schlitten nahe der Mitte der Schiene. Verwenden Sie den Befehl SET_TMC_FIELD, um die höchste Empfindlichkeit einzustellen. Für tmc2209:

```
SET_TMC_FIELD STEPPER=stepper_x FIELD=SGTHRS VALUE=255
```

Für tmc2130, tmc5160 und tmc2660:

```
SET_TMC_FIELD STEPPER=stepper_x FIELD=sgt VALUE=-64
```

Geben Sie dann einen `G28 X0`-Befehl ein und überprüfen Sie, ob sich die Achse überhaupt nicht bewegt. Wenn sich die Achse bewegt, geben Sie einen `M112`-Befehl aus, um den Drucker anzuhalten - irgendetwas stimmt mit der Verdrahtung oder der Konfiguration der diag/sg_tst-Pins nicht und muss korrigiert werden, bevor Sie fortfahren.

Verringern Sie dann kontinuierlich die Empfindlichkeit der VALUE-Einstellung und führen Sie die `SET_TMC_FIELD G28 X0`-Befehle erneut aus, um die höchste Empfindlichkeit zu finden, die dazu führt, dass der Schlitten erfolgreich bis zum Endanschlag fährt und anhält. (Bei tmc2209-Treibern ist dies die Verringerung von SGTHRS, bei anderen Treibern die Erhöhung von sgt). Achten Sie darauf, dass Sie jeden Versuch mit dem Wagen in der Nähe der Mitte der Schiene beginnen (geben Sie bei Bedarf M84 aus und bewegen Sie den Wagen dann manuell in die Mitte). Es sollte möglich sein, die höchste Empfindlichkeit zu finden, die zuverlässig funktioniert (Einstellungen mit höherer Empfindlichkeit führen zu geringer oder keiner Bewegung). Notieren Sie den gefundenen Wert als maximum_sensitivity. (Wenn die kleinstmögliche Empfindlichkeit (SGTHRS=0 oder sgt=63) erreicht wird, ohne dass sich der Sägeschlitten bewegt, stimmt etwas mit der Verdrahtung oder Konfiguration des diag/sg_tst-Pins nicht und muss korrigiert werden, bevor Sie fortfahren).

Bei der Suche nach der maximalen Empfindlichkeit kann es sinnvoll sein, zu verschiedenen VALUE-Einstellungen zu springen (um den VALUE-Parameter zu halbieren). Wenn Sie dies tun, sollten Sie darauf vorbereitet sein, einen `M112`-Befehl zu geben, um den Drucker anzuhalten, da eine Einstellung mit einer sehr niedrigen Empfindlichkeit dazu führen kann, dass die Achse wiederholt gegen das Ende der Schiene "knallt".

Achten Sie darauf, dass Sie zwischen den einzelnen Referenzfahrtversuchen einige Sekunden warten. Nachdem der TMC-Treiber einen Stillstand erkannt hat, kann es eine Weile dauern, bis er seine interne Anzeige löscht und in der Lage ist, einen weiteren Stillstand zu erkennen.

Wenn während dieser Einstellungstests ein `G28 X0`-Befehl nicht bis zur Achsengrenze fährt, sollten Sie vorsichtig sein, wenn Sie reguläre Fahrbefehle (z. B. `G1`) erteilen. Klipper wird die Position des Schlittens nicht richtig verstehen und ein Fahrbefehl kann zu unerwünschten und verwirrenden Ergebnissen führen.

Finde die niedrigste Empfindlichkeit, die mit einer Berührung referenziert
Bei der Referenzfahrt mit dem gefundenen Wert für die *maximale Empfindlichkeit* sollte sich die Achse bis zum Ende der Schiene bewegen und mit einer "einzigen Berührung" stoppen, d.h. es sollte kein "Klicken" oder "Knallen" zu hören sein. (Wenn bei maximaler Empfindlichkeit ein Knall- oder Klickgeräusch zu hören ist, ist die Referenzfahrtgeschwindigkeit möglicherweise zu niedrig, der Treiberstrom zu gering oder die sensorlose Referenzfahrt ist für die Achse nicht geeignet).

Der nächste Schritt besteht darin, den Schlitten wieder kontinuierlich in eine Position nahe der Mitte der Schiene zu bewegen, die Empfindlichkeit zu verringern und die `SET_TMC_FIELD G28 X0`-Befehle auszuführen - das Ziel ist nun, die niedrigste Empfindlichkeit zu finden, die immer noch dazu führt, dass der Schlitten mit einer "einzigen Berührung" erfolgreich referenziert. Das heißt, er "knallt" oder "klickt" nicht, wenn er das Ende der Schiene berührt. Notieren Sie den gefundenen Wert als minimum_sensitivity.

Aktualisieren Sie printer.cfg mit dem Empfindlichkeitswert
Nachdem Sie die *maximale und minimale Empfindlichkeit* ermittelt haben, verwenden Sie einen Taschenrechner, um die empfohlene Empfindlichkeit als *minimale Empfindlichkeit + (maximale Empfindlichkeit - minimale Empfindlichkeit)/3* zu berechnen. Die empfohlene Empfindlichkeit sollte im Bereich zwischen dem Minimum und dem Maximum liegen, aber etwas näher am Minimum. Runden Sie den endgültigen Wert auf den nächsten ganzzahligen Wert.

Für tmc2209 setzen Sie dies in der Konfiguration als `driver_SGTHRS`, für andere TMC-Treiber setzen Sie dies in der Konfiguration als `driver_SGT`.

Wenn der Bereich zwischen *maximum_sensitivity und minimum_sensitivity* klein ist (z.B. weniger als 5), kann dies zu einer instabilen Referenzfahrt führen. Eine schnellere Referenzfahrtgeschwindigkeit kann den Bereich vergrößern und den Betrieb stabiler machen.

Beachten Sie, dass bei einer Änderung des Treiberstroms, der Referenzfahrtgeschwindigkeit oder einer nennenswerten Änderung an der Druckerhardware der Abstimmungsprozess erneut durchgeführt werden muss.

Verwendung von Makros bei der Referenzfahrt
Nachdem die sensorlose Referenzfahrt abgeschlossen ist, wird der Schlitten gegen das Ende der Schiene gedrückt und der Stepper übt eine Kraft auf den Rahmen aus, bis der Schlitten weggefahren ist. Es ist ratsam, ein Makro zu erstellen, das die Achse referenziert und den Schlitten sofort vom Ende der Schiene wegbewegt.

Es ist ratsam, dass das Makro mindestens 2 Sekunden pausiert, bevor die sensorlose Referenzfahrt gestartet wird (oder anderweitig sicherstellt, dass 2 Sekunden lang keine Bewegung am Stepper stattgefunden hat). Ohne eine Verzögerung ist es möglich, dass das interne "stall"-Flag des Treibers noch von einer vorherigen Bewegung gesetzt ist.

Es kann auch nützlich sein, dass dieses Makro den Treiberstrom vor der Referenzfahrt setzt und einen neuen Strom setzt, nachdem sich der Schlitten wegbewegt hat.

Ein Beispielmakro könnte etwa so aussehen:

```
[gcode_macro SENSORLESS_HOME_X]
gcode:
    {% set HOME_CUR = 0.700 %}
    {% set driver_config = printer.configfile.settings['tmc2209 stepper_x'] %}
    {% set RUN_CUR = driver_config.run_current %}
    # Strom für sensorlose Referenzfahrt einstellen
    SET_TMC_CURRENT STEPPER=stepper_x CURRENT={HOME_CUR}
    # Pause, um sicherzustellen, dass das Treiber-Stall-Flag gelöscht ist
    G4 P2000
    # Home
    G28 X0
    # Wegfahren
    G90
    G1 X5 F1200
    # Strom während des Drucks einstellen
    SET_TMC_CURRENT STEPPER=stepper_x CURRENT={RUN_CUR}
```

Das resultierende Makro kann von einem homing_override config section Konfigurationsabschnitt oder von einem START_PRINT macro aufgerufen werden.

Wenn der Treiberstrom während der Referenzfahrt geändert wird, sollte der Abstimmungsprozess erneut durchgeführt werden.

Tipps zur sensorlosen Referenzfahrt auf dem CoreXY

Es ist möglich, die sensorlose Referenzierung auf den X- und Y-Schlitten eines CoreXY-Druckers zu verwenden. Klipper verwendet den `[stepper_x]` Stepper zur Erkennung von Strömungsabrissen bei der Referenzfahrt des X-Schlittens und den `[stepper_y]` Stepper zur Erkennung von Strömungsabrissen bei der Referenzfahrt des Y-Schlittens.

Benutzen Sie die oben beschriebene Tuning-Anleitung, um die geeignete "Stall Sensitivity" für jeden Schlitten zu finden, aber beachten Sie die folgenden Einschränkungen:

1. Wenn Sie die sensorlose Referenzfahrt auf CoreXY verwenden, stellen Sie sicher, dass für keinen der beiden Stepper ein `hold_current` konfiguriert ist.
2. Stellen Sie während der Abstimmung sicher, dass sich sowohl der X- als auch der Y-Schlitten vor jedem Referenzfahrtversuch in der Mitte ihrer Schienen befinden.
3. Verwenden Sie nach Abschluss der Abstimmung bei der Referenzfahrt von X und Y Makros, um sicherzustellen, dass zuerst eine Achse referenziert wird, bewegen Sie dann diesen Schlitten von der Achsengrenze weg, halten Sie mindestens 2 Sekunden lang inne und starten Sie dann die Referenzfahrt des anderen Schlittens. Durch die Bewegung von der Achse weg wird vermieden, dass eine Achse referenziert wird, während die andere gegen die Achsengrenze gedrückt wird (was die Erkennung des Stillstands verfälschen könnte). Die Pause ist notwendig, um sicherzustellen, dass das Stall-Flag des Treibers gelöscht wird, bevor die Referenzfahrt erneut gestartet wird.

Abfrage und Diagnose von Treibereinstellungen

Der Befehl `DUMP_TMC command ist ein nützliches Werkzeug für die Konfiguration und Diagnose der Treiber. Er meldet alle Felder, die von Klipper konfiguriert wurden, sowie alle Felder, die vom Treiber abgefragt werden können.

Alle gemeldeten Felder sind im Trinamic-Datenblatt für jeden Treiber definiert. Diese Datenblätter sind auf der Trinamic website zu finden. Besorgen Sie sich das Trinamic-Datenblatt für den Treiber und lesen Sie es, um die Ergebnisse von DUMP_TMC zu interpretieren.

Konfigurieren der driver_XXX-Einstellungen

Klipper unterstützt die Konfiguration vieler Low-Level-Treiberfelder mit Hilfe von driver_XXX-Einstellungen. Die TMC driver config reference enthält eine vollständige Liste der Felder, die für jede Art von Treiber verfügbar sind.

Darüber hinaus können fast alle Felder zur Laufzeit mit dem Befehl SET_TMC_FIELD command geändert werden.

Jedes dieser Felder ist im Trinamic-Datenblatt für jeden Treiber definiert. Diese Datenblätter sind auf der Trinamic website zu finden.

Beachten Sie, dass in den Trinamic-Datenblättern manchmal Formulierungen verwendet werden, die eine Einstellung auf hoher Ebene (z. B. "Hysterese Ende") mit einem Feldwert auf niedriger Ebene (z. B. "HEND") verwechseln können. In Klipper setzen driver_XXX und SET_TMC_FIELD immer den Low-Level-Feldwert, der tatsächlich in den Treiber geschrieben wird. Wenn also zum Beispiel das Trinamic-Datenblatt besagt, dass ein Wert von 3 in das HEND-Feld geschrieben werden muss, um ein "Hysterese-Ende" von 0 zu erhalten, dann setzen Sie driver_HEND=3, um den High-Level-Wert von 0 zu erhalten.

Allgemeine Fragen

Kann ich den stealthChop-Modus auf einem Extruder mit Druckvorschub verwenden?

Viele Leute verwenden den "stealthChop"-Modus erfolgreich mit dem Druckvorschub von Klipper. Klipper implementiert einen sanften Druckvorschub smooth pressure advance, der keine momentanen Geschwindigkeitsänderungen verursacht.

Der "stealthChop"-Modus kann jedoch zu einem geringeren Motordrehmoment und/oder zu einer höheren Motorwärme führen. Es kann sein, dass dieser Modus für Ihren speziellen Drucker nicht geeignet ist.

Ich erhalte immer wieder die Fehlermeldung "Unable to read tmc uart 'stepper_x' register IFCNT"?

Dies tritt auf, wenn Klipper nicht in der Lage ist, mit einem tmc2208 oder tmc2209 Treiber zu kommunizieren.

Vergewissern Sie sich, dass die Motorleistung aktiviert ist, da der Schrittmotortreiber im Allgemeinen Motorleistung benötigt, bevor er mit dem Mikrocontroller kommunizieren kann.

Wenn dieser Fehler nach dem ersten Flashen von Klipper auftritt, kann es sein, dass der Schrittmotor-Treiber zuvor in einem Zustand programmiert wurde, der nicht mit Klipper kompatibel ist. Um den Status zurückzusetzen, trennen Sie den Drucker für einige Sekunden von der Stromversorgung (ziehen Sie sowohl den USB- als auch den Netzstecker).

Ansonsten ist dieser Fehler typischerweise das Ergebnis einer falschen UART-Pin-Verkabelung oder einer falschen Klipper-Konfiguration der UART-Pin-Einstellungen.

Ich erhalte immer wieder die Fehlermeldung "Unable to write tmc spi 'stepper_x' register ..."?

Dies tritt auf, wenn Klipper nicht in der Lage ist, mit einem tmc2130 oder tmc5160 Treiber zu kommunizieren.

Vergewissern Sie sich, dass die Motorleistung aktiviert ist, da der Schrittmotortreiber im Allgemeinen Motorleistung benötigt, bevor er mit dem Mikrocontroller kommunizieren kann.

Andernfalls ist dieser Fehler typischerweise das Ergebnis einer falschen SPI-Verdrahtung, einer falschen Klipper-Konfiguration der SPI-Einstellungen oder einer unvollständigen Konfiguration von Geräten auf einem SPI-Bus.

Beachten Sie, dass wenn der Treiber an einem gemeinsamen SPI-Bus mit mehreren Geräten ist, stellen Sie sicher, dass Sie jedes Gerät auf diesem gemeinsamen SPI-Bus in Klipper vollständig konfigurieren. Wenn ein Gerät auf einem gemeinsam genutzten SPI-Bus nicht konfiguriert ist, kann es fälschlicherweise auf Befehle reagieren, die nicht für es bestimmt sind, und die Kommunikation mit dem beabsichtigten Gerät stören. Wenn es ein Gerät auf einem gemeinsam genutzten SPI-Bus gibt, das nicht in Klipper konfiguriert werden kann, dann verwenden Sie eine static_digital_output config section section, um den CS-Pin des unbenutzten Geräts auf high zu setzen (so dass es nicht versucht, den SPI-Bus zu nutzen). Der Schaltplan des Boards ist oft eine nützliche Referenz, um herauszufinden, welche Geräte an einem SPI-Bus angeschlossen sind und welche Pins ihnen zugeordnet sind.

Warum habe ich die Fehlermeldung "TMC meldet Fehler: ..." erhalten?

Diese Art von Fehler zeigt an, dass der TMC-Treiber ein Problem erkannt und sich selbst deaktiviert hat. Das heißt, der Treiber hat aufgehört, seine Position zu halten und ignoriert Bewegungsbefehle. Wenn Klipper feststellt, dass ein aktiver Treiber sich selbst deaktiviert hat, wird der Drucker in einen "Shutdown"-Status überführt.

Es ist auch möglich, dass ein TMC einen Fehler meldet, der aufgrund von SPI-Fehlern, die die Kommunikation mit dem Treiber verhindern (bei tmc2130, tmc5160 oder tmc2660), zum Abschalten führt. In diesem Fall ist es üblich, dass der gemeldete Treiberstatus `00000000` oder `ffffffff` anzeigt - zum Beispiel: `TMC reports error Fehler: DRV_STATUS: ffffffff ...` ODER `TMC reports error Fehler: READRSP@RDSEL2: 00000000 ....` Ein solcher Fehler kann auf ein Problem mit der SPI-Verdrahtung oder auf einen Selbst-Reset oder einen Fehler des TMC-Treibers zurückzuführen sein.

Einige häufige Fehler und Tipps zu deren Diagnose:

TMC reports error: ... ot=1(OvertempError!)
Dies bedeutet, dass der Motortreiber sich selbst deaktiviert hat, weil er zu heiß geworden ist. Typische Lösungen sind, den Schrittmotorstrom zu verringern, die Kühlung des Schrittmotortreibers zu erhöhen und/oder die Kühlung des Schrittmotors zu erhöhen.

TMC report error: ... ShortToGND OR LowSideShort
Dies zeigt an, dass der Treiber sich selbst deaktiviert hat, weil er einen sehr hohen Stromfluss durch den Treiber festgestellt hat. Dies kann auf ein loses oder kurzgeschlossenes Kabel zum Schrittmotor oder im Schrittmotor selbst hinweisen.

Dieser Fehler kann auch auftreten, wenn Sie den StealthChop-Modus verwenden und der TMC-Treiber die mechanische Belastung des Motors nicht genau vorhersagen kann. (Wenn der Treiber eine schlechte Vorhersage macht, kann er zu viel Strom durch den Motor schicken und seine eigene Überstromerkennung auslösen.) Um dies zu testen, deaktivieren Sie den stealthChop-Modus und prüfen Sie, ob die Fehler weiterhin auftreten.

TMC reports error: ... reset=1(Reset) OR CS_ACTUAL=0(Reset?) OR SE=0(Reset?)¶
Dies zeigt an, dass sich der Treiber mitten im Druckvorgang zurückgesetzt hat. Dies kann auf Spannungs- oder Verdrahtungsprobleme zurückzuführen sein.

TMC reports error:... uv_cp=1(Unterspannung!)¶
Dies zeigt an, dass der Treiber ein Unterspannungsereignis erkannt und sich selbst deaktiviert hat. Dies kann auf Verdrahtungs- oder Stromversorgungsprobleme zurückzuführen sein.

Wie stelle ich den spreadCycle/coolStep/etc.-Modus auf meinen Treibern ein?

Auf der Trinamic-Website finden Sie Anleitungen zum Konfigurieren der Treiber. Diese Anleitungen sind oft technisch, nicht sehr anspruchsvoll und können spezielle Hardware erfordern. Nichtsdestotrotz sind sie die beste Quelle für Informationen.

Referenzfahrt und Abtastung mit mehreren Mikrocontrollern

Klipper unterstützt einen Mechanismus für die Referenzfahrt mit einem Endanschlag, der an einen Mikrocontroller angeschlossen ist, während die Schrittmotoren an einen anderen Mikrocontroller angeschlossen sind. Diese Unterstützung wird als "Multi-MCU-Homing" bezeichnet. Diese Funktion wird auch verwendet, wenn sich ein Z-Taster auf einem anderen Mikrocontroller befindet als die Z-Schrittmotoren.

Diese Funktion kann nützlich sein, um die Verdrahtung zu vereinfachen, da es bequemer sein kann, einen Endanschlag oder eine Sonde an einen näheren Mikrocontroller anzuschließen. Die Verwendung dieser Funktion kann jedoch zu einem "Überschwingen" der Schrittmotoren während der Referenzfahrt und des Antastvorgangs führen.

Das Überschwingen ist auf mögliche Verzögerungen bei der Nachrichtenübertragung zwischen dem Mikrocontroller, der den Endanschlag überwacht, und den Mikrocontrollern, die die Schrittmotoren bewegen, zurückzuführen. Der Klipper-Code ist so konzipiert, dass diese Verzögerung nicht mehr als 25 ms beträgt. (Wenn die Referenzfahrt mit mehreren Mikrocontrollern aktiviert ist, senden die Mikrocontroller periodische Statusmeldungen und prüfen, ob die entsprechenden Statusmeldungen innerhalb von 25 ms empfangen werden).

Wenn die Referenzfahrt beispielsweise mit 10 mm/s erfolgt, ist eine Überschreitung von bis zu 0,250 mm möglich (10 mm/s * 0,025 s = = 0,250 mm). Bei der Konfiguration der Multi-MCU-Referenzfahrt sollte diese Art von Überschwingen berücksichtigt werden. Die Verwendung langsamerer Referenzier- oder Antastgeschwindigkeiten kann das Überschwingen reduzieren.

Das Überschwingen des Schrittmotors sollte sich nicht negativ auf die Präzision der Referenzfahrt und des Antastverfahrens auswirken. Der Klipper-Code erkennt das Überschwingen und berücksichtigt es in seinen Berechnungen. Es ist jedoch wichtig, dass das Hardware-Design in der Lage ist, das Überschwingen zu verarbeiten, ohne die Maschine zu beschädigen.

Sollte Klipper ein Kommunikationsproblem zwischen den Mikrocontrollern während der Multi-MCU-Referenzfahrt feststellen, wird ein Fehler "Kommunikations-Timeout während der Referenzfahrt" ausgegeben.

Beachten Sie, dass eine Achse mit mehreren Steppern (z. B. `stepper_z` und `stepper_z1`) auf demselben Mikrocontroller sein muss, um die Multi-MCU-Referenzierung zu verwenden. Befindet sich zum Beispiel ein Endanschlag auf einem anderen Mikrocontroller als `stepper_z`, muss `stepper_z1` auf demselben Mikrocontroller wie stepper_z sein.

Slicer

Dieses Dokument enthält einige Tipps zur Konfiguration einer "Slicer"-Anwendung für die Verwendung mit Klipper. Übliche Slicer, die mit Klipper verwendet werden, sind Slic3r, Cura, Simplify3D, etc.

Stellen Sie den G-Code Flavor auf Marlin

Viele Slicer haben eine Option, um den "G-Code flavor" zu konfigurieren. Die Standardeinstellung ist häufig "Marlin" und das funktioniert gut mit Klipper. Die Einstellung "Smoothieware" funktioniert auch gut mit Klipper.

Klipper gcode_macro

Slicer erlauben oft die Konfiguration von "Start G-Code" und "End G-Code" Sequenzen. Es ist oft praktisch, stattdessen benutzerdefinierte Makros in der Klipper-Konfigurationsdatei zu definieren - wie z.B.: `[gcode_macro START_PRINT]` und `[gcode_macro END_PRINT]`. Dann kann man einfach START_PRINT und END_PRINT in der Konfiguration des Slicers ausführen. Das Definieren dieser Aktionen in der Klipper-Konfiguration kann es einfacher machen, die Start- und Endschritte des Druckers zu optimieren, da Änderungen kein erneutes Slicen erfordern.

Siehe sample-macros.cfg für Beispiele von START_PRINT und END_PRINT Makros.

Einzelheiten zur Definition eines gcode_macro finden Sie in der Konfigurationsreferenz config reference.

Große Rückzugseinstellungen erfordern möglicherweise die Einstellung von Klipper

Die maximale Geschwindigkeit und Beschleunigung von Rückzugsbewegungen wird in Klipper durch die Konfigurationseinstellungen `max_extrude_only_velocity` und `max_extrude_only_accel` gesteuert. Diese Einstellungen haben einen Standardwert, der bei vielen Druckern gut funktionieren sollte. Wenn man jedoch einen großen Rückzug im Slicer konfiguriert hat (z. B. 5 mm oder mehr), kann man feststellen, dass sie die gewünschte Geschwindigkeit des Rückzugs einschränken.

Wenn Sie einen großen Rückzug verwenden, sollten Sie stattdessen den Druckvorlauf pressure advance von Klipper einstellen. Andernfalls, wenn man feststellt, dass der Werkzeugkopf während des Rückzugs und des Ansetzens zu "pausieren" scheint, sollte man in Erwägung ziehen, `max_extrude_only_velocity` und `max_extrude_only_accel` in der Klipper-Konfigurationsdatei explizit zu definieren.

Aktivieren Sie nicht das "Ausrollen"

Die Funktion "Auslaufen" wird wahrscheinlich zu einer schlechten Druckqualität mit Klipper führen. Erwägen Sie stattdessen die Verwendung des Druckvorschubs pressure advance von Klipper.

Insbesondere, wenn der Slicer die Extrusionsrate zwischen den Zügen drastisch ändert, wird Klipper zwischen den Zügen eine Verlangsamung und Beschleunigung durchführen. Dadurch wird das Blobbing wahrscheinlich nicht besser, sondern schlechter.

Im Gegensatz dazu ist es in Ordnung (und oft hilfreich), die Einstellungen "Retract", "Wipe" und/oder "Wipe on retract" eines Slicers zu verwenden.

Verwenden Sie bei Simplify3d nicht die Einstellung "extra restart distance".

Diese Einstellung kann zu drastischen Änderungen der Extrusionsraten führen, die die Überprüfung des maximalen Extrusionsquerschnitts durch Klipper auslösen können. Erwägen Sie stattdessen die Verwendung von Klippers Druckvorlauf pressure advance oder der regulären Simplify3d-Rücklaufeinstellung.

Deaktivieren Sie "PreloadVE" auf KISSlicer

Wenn Sie KISSlicer verwenden, setzen Sie "PreloadVE" auf Null. Erwägen Sie stattdessen den Druckvorschub pressure advance von Klipper zu verwenden.

Deaktivieren Sie alle "erweiterten Extruderdruck"-Einstellungen

Einige Slicer werben mit einer "erweiterten Extruderdruck"-Funktion. Es wird empfohlen, diese Optionen bei der Verwendung von Klipper zu deaktivieren, da sie wahrscheinlich zu einer schlechten Druckqualität führen. Ziehen Sie stattdessen die Druckvorverlegung pressure advance von Klipper in Betracht.

Insbesondere können diese Slicer-Einstellungen die Firmware anweisen, wilde Änderungen an der Extrusionsrate vorzunehmen, in der Hoffnung, dass die Firmware diese Anforderungen annähernd erfüllt und der Drucker ungefähr den gewünschten Extruderdruck erreicht. Klipper hingegen arbeitet mit präzisen kinematischen Berechnungen und Timing. Wenn Klipper angewiesen wird, signifikante Änderungen an der Extrusionsrate vorzunehmen, plant er die entsprechenden Änderungen der Geschwindigkeit, der Beschleunigung und der Extruderbewegung - was nicht in der Absicht des Slicers liegt. Der Slicer kann sogar übermäßige Extrusionsraten befehlen, bis zu dem Punkt, an dem er die Prüfung des maximalen Extrusionsquerschnitts von Klipper auslöst.

Im Gegensatz dazu ist es in Ordnung (und oft hilfreich), die Einstellungen "Zurückziehen", "Wischen" und/oder "Wischen beim Zurückziehen" eines Slicers zu verwenden.

Schräglagenkorrektur

Die softwarebasierte Schräglagenkorrektur kann helfen, Maßungenauigkeiten zu beheben, die sich aus einer nicht perfekt rechtwinkligen Druckerbaugruppe ergeben. Beachten Sie, dass es bei einer erheblichen Schräglage Ihres Druckers dringend empfohlen wird, den Drucker zunächst mit mechanischen Mitteln so rechtwinklig wie möglich zu machen, bevor Sie die softwarebasierte Korrektur anwenden.

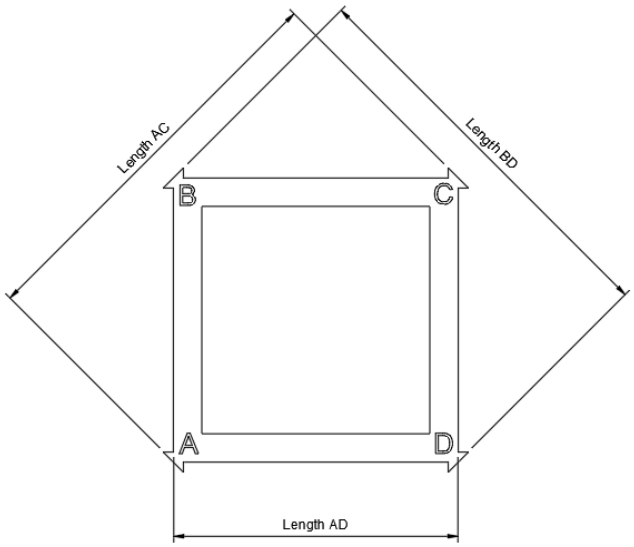
Drucken Sie ein Kalibrierungsobjekt

Der erste Schritt zur Korrektur der Schräglage besteht darin, ein Kalibrierungsobjekt calibration object entlang der zu korrigierenden Ebene zu drucken. Es gibt auch ein Kalibrierungsobjekt calibration object, das alle Ebenen in einem Modell umfasst. Das Objekt soll so ausgerichtet werden, dass die Ecke A zum Ursprung der Ebene zeigt.

Stellen Sie sicher, dass während des Drucks keine Schräglagenkorrektur vorgenommen wird. Dies können Sie erreichen, indem Sie entweder das Modul `[skew_correction]` aus der Datei printer.cfg entfernen oder einen `SET_SKEW CLEAR=1` gcode ausgeben.

Nehmen Sie Ihre Messungen vor

Das Modul [skew_correcton] erfordert 3 Messungen für jede zu korrigierende Ebene: die Länge von Ecke A bis Ecke C, die Länge von Ecke B bis Ecke D und die Länge von Ecke A bis Ecke D. Bei der Messung der Länge AD werden die Abflachungen an den Ecken, die einige Testobjekte bieten, nicht berücksichtigt.



skew_lengths

Konfigurieren Sie Ihren Schräglauf

Stellen Sie sicher, dass `[skew_correction]` in printer.cfg enthalten ist. Sie können nun den SET_SKEW gcode verwenden, um skew_correcton zu konfigurieren. Wenn Ihre gemessenen Längen entlang von XY beispielsweise wie folgt sind:

```
Länge AC = 140,4
Länge BD = 142,8
Länge AD = 99,8
```

SET_SKEW kann verwendet werden, um die Schräglagenkorrektur für die XY-Ebene zu konfigurieren.

```
SET_SKEW XY=140.4,142.8,99.8
```

Sie können auch Messungen für XZ und YZ in den gcode einfügen:

```
SET_SKEW XY=140.4,142.8,99.8 XZ=141.6,141.4,99.8 YZ=142.4,140.5,99.5
```

Das Modul [skew_correction] unterstützt auch die Profilverwaltung in ähnlicher Weise wie [bed_mesh]. Nachdem Sie die Schräglage mit dem SET_SKEW gcode eingestellt haben, können Sie den SKEW_PROFILE gcode verwenden, um sie zu speichern:

```
SKEW_PROFILE SAVE=my_skew_profile
```

Nach diesem Befehl werden Sie aufgefordert, einen SAVE_CONFIG gcode einzugeben, um das Profil im permanenten Speicher zu speichern. Wenn kein Profil mit dem Namen my_skew_profile vorhanden ist, wird ein neues Profil erstellt. Wenn das benannte Profil bereits existiert, wird es überschrieben.

Sobald Sie ein Profil gespeichert haben, können Sie es laden:

```
SKEW_PROFILE LOAD=my_skew_profile
```

Es ist auch möglich, ein altes oder veraltetes Profil zu entfernen:

```
SKEW_PROFILE REMOVE=mein_skew_profil
```

Nach dem Entfernen eines Profils werden Sie aufgefordert, ein SAVE_CONFIG auszuführen, damit diese Änderung bestehen bleibt.

Überprüfen Ihrer Korrektur

Nachdem die Schräglagenkorrektur konfiguriert wurde, können Sie das Kalibrierteil mit aktivierter Korrektur erneut drucken. Verwenden Sie den folgenden gcode, um Ihre Schräglage auf jeder Ebene zu überprüfen. Die Ergebnisse sollten niedriger sein als die mit GET_CURRENT_SKEW ermittelten Werte.

```
CALC_MEASURED_SKEW AC=<ac_length> BD=<bd_length> AD=<ad_length>
```

Vorsichtsmaßnahmen

Aufgrund der Natur der Schräglagenkorrektur wird empfohlen, die Schräglagenkorrektur in Ihrem Start-Gcode zu konfigurieren, und zwar nach der Referenzfahrt und jeder Art von Bewegung, die sich in der Nähe des Randes des Druckbereichs abspielt, wie z. B. eine Reinigung oder Düsenwischung. Sie können dazu die Gcodes `SET_SKEW` oder `SKEW_PROFILE` verwenden. Es wird auch empfohlen, einen `SET_SKEW CLEAR=1` in Ihrem End-Gcode zu verwenden.

Denken Sie daran, dass es möglich ist, dass `[skew_correction]` eine Korrektur erzeugt, die das Werkzeug auf der X- und/oder Y-Achse über die Grenzen des Druckers hinaus bewegt. Es wird empfohlen, bei der Verwendung von `[skew_correction]` die Teile nicht an den Kanten anzuordnen.

Objekte ausschließen

Das `[exclude_object]`-Modul erlaubt es Klipper, Objekte auszuschließen, während ein Druckvorgang läuft. Um diese Funktion zu aktivieren, fügen Sie einen exclude_object Konfigurationsabschnitt exclude_object config section ein (siehe auch die Befehlsreferenz command reference und die sample-macros.cfg sample-macros.cfg Datei für ein Marlin/RepRapFirmware kompatibles M486 G-Code Makro).

Im Gegensatz zu anderen 3D-Drucker-Firmware-Optionen verwendet ein Drucker, auf dem Klipper läuft, eine Reihe von Komponenten, und die Benutzer haben viele Optionen, aus denen sie wählen können. Um eine einheitliche Benutzererfahrung zu gewährleisten, wird das [exclude_object]-Modul eine Art Vertrag oder API einrichten. Der Vertrag regelt den Inhalt der gcode-Datei, wie der interne Zustand des Moduls gesteuert wird und wie dieser Zustand den Clients zur Verfügung gestellt wird.

Workflow-Übersicht

Ein typischer Arbeitsablauf für das Drucken einer Datei könnte wie folgt aussehen:

1. Das Slicing ist abgeschlossen und die Datei wird zum Drucken hochgeladen. Während des Hochladens wird die Datei verarbeitet und `[exclude_object]`-Markierungen werden der Datei hinzugefügt. Alternativ können Slicer so konfiguriert werden, dass sie Objektausschlussmarkierungen nativ oder in einem eigenen Vorverarbeitungsschritt vorbereiten.
2. Wenn der Druck beginnt, setzt Klipper den `[exclude_object]` status zurück.
3. Wenn Klipper den `EXCLUDE_OBJECT_DEFINE`-Block verarbeitet, wird der Status mit den bekannten Objekten aktualisiert und an die Clients weitergegeben.
4. Der Client kann diese Informationen verwenden, um dem Benutzer eine Benutzeroberfläche zu präsentieren, so dass der Fortschritt verfolgt werden kann. Klipper aktualisiert den Status, um das aktuell druckende Objekt einzuschließen, das der Client für Anzeigezwecke verwenden kann.
5. Wenn der Benutzer verlangt, dass ein Objekt gelöscht wird, gibt der Client einen `EXCLUDE_OBJECT NAME=<name>` Befehl an Klipper.
6. Wenn Klipper den Befehl verarbeitet, wird das Objekt zur Liste der ausgeschlossenen Objekte hinzugefügt und der Status für den Client aktualisiert.
7. Der Client erhält den aktualisierten Status von Klipper und kann diese Information verwenden, um den Status des Objekts in der Benutzeroberfläche wiederzugeben.
8. Nach Beendigung des Druckvorgangs bleibt der Status `[exclude_object]` so lange erhalten, bis er durch eine andere Aktion zurückgesetzt wird.

Die GCode-Datei

Die spezielle GCode-Verarbeitung, die benötigt wird, um das Ausschließen von Objekten zu unterstützen, passt nicht zu den Kernzielen von Klipper. Daher erfordert dieses Modul, dass die Datei verarbeitet wird, bevor sie zum Drucken an Klipper gesendet wird. Die Verwendung eines Nachbearbeitungsskripts im Slicer oder die Verarbeitung der Datei durch eine Middleware beim Hochladen sind zwei Möglichkeiten zur Vorbereitung der Datei für Klipper. Ein Referenz-Nachbearbeitungsskript ist sowohl als ausführbare Datei als auch als Python-Bibliothek verfügbar, siehe cancelobject-preprocessor.

Objekt-Definitionen

Der Befehl `EXCLUDE_OBJECT_DEFINE` wird verwendet, um eine Zusammenfassung jedes Objekts in der zu druckenden gcode-Datei zu erstellen. Liefert eine Zusammenfassung eines Objekts in der Datei. Objekte müssen nicht definiert werden, um von anderen Befehlen referenziert zu werden. Der Hauptzweck dieses Befehls besteht darin, der Benutzeroberfläche Informationen zur Verfügung zu stellen, ohne die gesamte Gcodedatei analysieren zu müssen.

Die Objektdefinitionen werden benannt, damit die Benutzer ein auszuschließendes Objekt leicht auswählen können, und es können zusätzliche Metadaten angegeben werden, um grafische Stornoanzeigen zu ermöglichen. Derzeit definierte Metadaten umfassen eine `CENTER` X,Y-Koordinate und eine `POLYGON`-Liste von X,Y-Punkten, die einen minimalen Umriss des Objekts darstellen. Dabei kann es sich um eine einfache Bounding Box oder eine komplizierte Hülle handeln, um detailliertere Visualisierungen der gedruckten Objekte zu zeigen. Vor allem, wenn gcode-Dateien mehrere Teile mit überlappenden Begrenzungsbereichen enthalten, sind die Mittelpunkte visuell schwer zu unterscheiden. POLYGONS muss ein json kompatibles Array von Punkt `[X,Y]` Tupeln ohne Leerzeichen sein. Zusätzliche Parameter werden als Strings in der Objektdefinition gespeichert und in Status-Updates bereitgestellt.

```
EXCLUDE_OBJECT_DEFINE NAME=calibration_pyramid CENTER=50,50 POLYGON=[[40,40],[50,60],[60,40]]
```

Alle verfügbaren G-Code-Befehle sind in der G-Code Reference dokumentiert

Statusinformationen

Der Status dieses Moduls wird den Clients durch den Status exclude_object status mitgeteilt.

Der Status wird zurückgesetzt, wenn:

- Die Klipper-Firmware neu gestartet wird.
- `die [virtual_sdcard]` zurückgesetzt wird. Insbesondere wird diese von Klipper zu Beginn eines Drucks zurückgesetzt.
- Wenn ein `EXCLUDE_OBJECT_DEFINE RESET=1` Befehl ausgegeben wird.

Die Liste der definierten Objekte wird in dem Statusfeld exclude_object.objects dargestellt. In einer gut definierten gcode-Datei wird dies mit `EXCLUDE_OBJECT_DEFINE`-Befehlen am Anfang der Datei durchgeführt. Dadurch erhalten die Clients Objektnamen und Koordinaten, so dass die Benutzeroberfläche auf Wunsch eine grafische Darstellung der Objekte liefern kann.

Im Laufe des Druckvorgangs wird das Statusfeld exclude_object.current_object aktualisiert, wenn Klipper die Befehle `EXCLUDE_OBJECT_START` und `EXCLUDE_OBJECT_END` verarbeitet. Das Feld current_object wird auch dann gesetzt, wenn das Objekt bereits ausgeschlossen wurde. Undefinierte Objekte, die mit `EXCLUDE_OBJECT_START` markiert sind, werden zu den bekannten Objekten hinzugefügt, um den UI-Hinweis zu unterstützen, ohne zusätzliche Metadaten.
Wenn `EXCLUDE_OBJECT`-Befehle ausgegeben werden, wird die Liste der ausgeschlossenen Objekte im `exclude_object.excluded_objects-Array` bereitgestellt. Da Klipper die Verarbeitung von anstehendem gcode vorwegnimmt, kann es zu einer Verzögerung zwischen der Ausgabe des Befehls und der Aktualisierung des Status kommen.

Verwendung von PWM-Werkzeugen

Dieses Dokument beschreibt, wie man einen PWM-gesteuerten Laser oder eine Spindel mit output_pin und einigen Makros einrichtet.

Wie funktioniert das?

Mit der Wiederverwendung des Lüfter-PWM-Ausgangs des Druckkopfs können Sie Laser oder Spindeln steuern. Dies ist nützlich, wenn Sie umschaltbare Druckköpfe verwenden, zum Beispiel den E3D-Toolchanger oder eine DIY-Lösung. Normalerweise können Cam-Tools wie LaserWeb so konfiguriert werden, dass sie `M3-M5`-Befehle verwenden, die für Spindeldrehzahl CW (`M3 S[0-255]`), Spindeldrehzahl CCW (`M4 S[0-255]`) und Spindelstopp (`M5`) stehen.

Achtung! Halten Sie beim Betrieb eines Lasers alle erdenklichen Sicherheitsvorkehrungen ein! Diodenlaser sind normalerweise invertiert. Das bedeutet, dass der Laser beim Neustart der MCU für die Zeit, die die MCU braucht, um wieder hochzufahren, voll eingeschaltet ist. Vorsichtshalber wird empfohlen, bei eingeschaltetem Laser immer eine geeignete Laserschutzbrille der richtigen Wellenlänge zu tragen und den Laser abzuschalten, wenn er nicht benötigt wird. Außerdem sollten Sie eine Sicherheitszeitüberschreitung konfigurieren, damit das Tool anhält, wenn Ihr Host oder Ihre MCU einen Fehler feststellt.

Eine Beispielkonfiguration finden Sie unter config/sample-pwm-tool.cfg.

Aktuelle Beschränkungen

Die Häufigkeit der PWM-Aktualisierungen ist begrenzt. Obwohl es sehr präzise ist, kann eine PWM-Aktualisierung nur alle 0,1 Sekunden erfolgen, was es für die Rastergravur fast unbrauchbar macht. Es gibt jedoch einen experimentellen Zweig mit seinen eigenen Kompromissen. Langfristig ist geplant, diese Funktionalität in den main-line klipper zu integrieren.

Befehle
`M3/M4 S<Wert> :` Einstellen des PWM-Tastverhältnisses. Werte zwischen 0 und 255. `M5` : PWM-Ausgang auf Abschaltwert stoppen.

Laserweb Konfiguration
Wenn Sie Laserweb verwenden, wäre eine funktionierende Konfiguration wie folgt:

```
GCODE START:
    M5            ; Disable Laser
    G21           ; Set units to mm
    G90           ; Absolute positioning
    G0 Z0 F7000   ; Set Non-Cutting speed

GCODE END:
    M5            ; Disable Laser
    G91           ; relative
    G0 Z+20 F4000 ;
    G90           ; absolute

GCODE HOMING:
```

```
    M5            ; Disable Laser
    G28           ; Home all axis

TOOL ON:
    M3 $INTENSITY

TOOL OFF:
    M5            ; Disable Laser

LASER INTENSITY:
    S
```

Dokumentation für Entwickler

Code-Übersicht

Dieses Dokument beschreibt das allgemeine Codelayout und den Hauptcodefluss von Klipper.

Verzeichnis-Layout

Das Verzeichnis src/ enthält den C-Quelltext für den Mikrocontroller-Code. Die Verzeichnisse src/atsam/, src/atsamd/, src/avr/, src/linux/, src/lpc176x/, src/pru/ und src/stm32/ enthalten architekturspezifischen Mikrocontroller-Code. Das Verzeichnis src/simulator/ enthält Code-Stubs, mit denen der Mikrocontroller auf anderen Architekturen testkompiliert werden kann. Das Verzeichnis src/generic/ enthält Hilfscode, der auf verschiedenen Architekturen nützlich sein kann. Der Build sorgt dafür, dass Includes von "board/somefile.h" zuerst im Verzeichnis der aktuellen Architektur (z.B. src/avr/somefile.h) und dann im generischen Verzeichnis (z.B. src/generic/somefile.h) gesucht werden.

Das Verzeichnis klippy/ enthält die Host-Software. Der größte Teil der Wirtssoftware ist in Python geschrieben, das Verzeichnis klippy/chelper/ enthält jedoch einige C-Code-Helfer. Das Verzeichnis klippy/kinematics/ enthält den Code für die Roboterkinematik. Das Verzeichnis klippy/extras/ enthält die erweiterbaren "Module" des Host-Codes.

Das Verzeichnis lib/ enthält externe Bibliotheken von Drittanbietern, die für die Erstellung einiger Ziele erforderlich sind.

Das Verzeichnis config/ enthält Beispiel-Druckerkonfigurationsdateien.

Das Verzeichnis scripts/ enthält Build-Time-Skripte, die für die Kompilierung des Mikrocontroller-Codes nützlich sind.

Das Verzeichnis test/ enthält automatische Testfälle.

Während der Kompilierung wird möglicherweise ein Verzeichnis out/ erstellt. Dieses Verzeichnis enthält temporäre Build-Time-Objekte. Das endgültige Mikrocontroller-Objekt, das gebaut wird, ist out/klipper.elf.hex auf AVR und out/klipper.bin auf ARM.

Mikrocontroller-Codefluss

Die Ausführung des Mikrocontroller-Codes beginnt im architekturspezifischen Code (z. B. src/avr/main.c), der letztendlich sched_main() in src/sched.c aufruft. Der sched_main()-Code beginnt mit der Ausführung aller Funktionen, die mit dem Makro DECL_INIT() gekennzeichnet sind. Anschließend werden wiederholt alle Funktionen ausgeführt, die mit dem Makro DECL_TASK() gekennzeichnet sind.

Eine der wichtigsten Aufgabenfunktionen ist command_dispatch() in src/command.c. Diese Funktion wird vom platinenspezifischen Ein-/Ausgabecode (z. B. src/avr/serial.c, src/generic/serial_irq.c) aufgerufen und führt die mit den Befehlen im Eingabestrom verbundenen Befehlsfunktionen aus. Befehlsfunktionen werden mit dem Makro DECL_COMMAND() deklariert (weitere Informationen finden Sie im Protokolldokument protocol).

Task-, Init- und Befehlsfunktionen werden immer mit aktivierten Unterbrechungen ausgeführt (sie können jedoch bei Bedarf vorübergehend die Unterbrechungen deaktivieren). Diese Funktionen sollten lange Pausen oder Verzögerungen vermeiden oder Arbeiten ausführen, die eine längere Zeit dauern. (Lange Verzögerungen in diesen "Task"-Funktionen führen zu Planungsfehlern bei anderen "Tasks" - Verzögerungen von mehr als 100us können sich bemerkbar machen, Verzögerungen von mehr als 500us können zu erneuten Befehlsübertragungen führen, Verzögerungen von mehr als 100ms können zu Watchdog-Reboots führen.) Diese Funktionen planen die Arbeit zu bestimmten Zeiten, indem sie Timer einplanen.

Timer-Funktionen werden durch den Aufruf von sched_add_timer() (zu finden in src/sched.c) geplant. Der Zeitplanungscode sorgt dafür, dass die angegebene Funktion zur gewünschten Uhrzeit aufgerufen wird. Timer-Interrupts werden zunächst in einem architekturspezifischen Interrupt-Handler (z. B. src/avr/timer.c) behandelt, der sched_timer_dispatch() in src/sched.c aufruft. Der Timer-Interrupt führt zur Ausführung der Scheduler-Timer-Funktionen. Timer-Funktionen werden immer mit deaktivierten Interrupts ausgeführt. Die Timer-Funktionen sollten immer innerhalb weniger Mikrosekunden abgeschlossen sein. Nach Beendigung des Timer-Ereignisses kann die Funktion sich selbst neu terminieren.

Wird ein Fehler festgestellt, kann der Code shutdown() aufrufen (ein Makro, das sched_shutdown() in src/sched.c aufruft). Der Aufruf von shutdown() bewirkt, dass alle mit dem Makro DECL_SHUTDOWN() gekennzeichneten Funktionen ausgeführt werden. Shutdown-Funktionen werden immer mit deaktivierten Interrupts ausgeführt.

Ein Großteil der Funktionalität des Mikrocontrollers beinhaltet die Arbeit mit General-Purpose Input/Output Pins (GPIO). Um den architekturspezifischen Low-Level-Code vom High-Level-Taskcode zu abstrahieren, werden alle GPIO-Ereignisse in architekturspezifischen Wrappern implementiert (z. B.

src/avr/gpio.c). Der Code wird mit gccs "-flto -fwhole-program"-Optimierung kompiliert, die eine hervorragende Arbeit beim Inlining von Funktionen über Kompilierungseinheiten hinweg leistet, so dass die meisten dieser winzigen GPIO-Funktionen in ihre Aufrufer eingebettet sind und es keine Laufzeitkosten für ihre Verwendung gibt.

Übersicht über den Klippy-Code

Der Host-Code (Klippy) soll auf einem kostengünstigen Computer (z. B. einem Raspberry Pi) laufen, der mit dem Mikrocontroller gekoppelt ist. Der Code ist hauptsächlich in Python geschrieben, verwendet jedoch CFFI, um einige Funktionen in C-Code zu implementieren.

Die erste Ausführung beginnt in klippy/klippy.py. Diese liest die Befehlszeilenargumente, öffnet die Druckerkonfigurationsdatei, instanziiert die wichtigsten Druckerobjekte und startet die serielle Verbindung. Die Hauptausführung der G-Code-Befehle erfolgt in der Methode process_commands() in klippy/gcode.py. Dieser Code übersetzt die G-Code-Befehle in Aufrufe von Druckerobjekten, die die Aktionen häufig in Befehle übersetzen, die auf dem Mikrocontroller ausgeführt werden (wie über das Makro DECL_COMMAND im Mikrocontroller-Code deklariert).

Es gibt vier Threads im Klippy-Host-Code. Der Hauptthread verarbeitet die eingehenden gcode-Befehle. Ein zweiter Thread (der vollständig im C-Code von klippy/chelper/serialqueue.c untergebracht ist) verarbeitet Low-Level-IO mit der seriellen Schnittstelle. Der dritte Thread wird für die Verarbeitung von Antwortmeldungen des Mikrocontrollers im Python-Code verwendet (siehe klippy/serialhdl.py). Der vierte Thread schreibt Debug-Meldungen in das Protokoll (siehe klippy/queuelogger.py), so dass die anderen Threads beim Schreiben des Protokolls nicht blockieren.

Codefluss eines Bewegungsbefehls

Eine typische Druckerbewegung beginnt, wenn ein "G1"-Befehl an den Klippy-Host gesendet wird, und sie ist abgeschlossen, wenn die entsprechenden Schrittimpulse auf dem Mikrocontroller erzeugt werden. In diesem Abschnitt wird der Codeablauf eines typischen Fahrbefehls beschrieben. Das Kinematik-Dokument kinematics enthält weitere Informationen über die Mechanik von Bewegungen.

- Die Verarbeitung eines Fahrbefehls beginnt in gcode.py. Das Ziel von gcode.py ist es, G-Code in interne Aufrufe zu übersetzen. Ein G1-Befehl ruft cmd_G1() in klippy/extras/gcode_move.py auf. Der Code von gcode_move.py behandelt Änderungen des Ursprungs (z.B. G92), Änderungen der relativen und absoluten Positionen (z.B. G90) und Änderungen der Einheiten (z.B. F6000=100mm/s). Der Code-Pfad für eine Bewegung ist: `_process_data() -> _process_commands() -> cmd_G1()`. Schließlich wird die Klasse ToolHead aufgerufen, um den eigentlichen Auftrag auszuführen: `cmd_G1() -> ToolHead.move()`

- Die Klasse ToolHead (in toolhead.py) kümmert sich um die "Vorausschau" und verfolgt den Zeitplan der Druckaktionen. Der Hauptcodepfad für eine Bewegung ist: `ToolHead.move() -> MoveQueue.add_move() -> MoveQueue.flush() -> Move.set_junction() -> ToolHead._process_moves()`.
    - ToolHead.move() erzeugt ein Move()-Objekt mit den Parametern der Bewegung (im kartesischen Raum und in Einheiten von Sekunden und Millimetern).
    - Die Kinematikklasse hat die Möglichkeit, jede Bewegung zu überprüfen (`ToolHead.move() -> kin.check_move()`). Die Kinematikklassen befinden sich im Verzeichnis klippy/kinematics/. Der check_move()-Code kann einen Fehler auslösen, wenn die Bewegung nicht gültig ist. Wenn check_move() erfolgreich abgeschlossen wird, muss die zugrundeliegende Kinematik in der Lage sein, die Bewegung zu verarbeiten.
    - MoveQueue.add_move() setzt das Move-Objekt in die Warteschlange "look-ahead".
    - MoveQueue.flush() bestimmt die Start- und Endgeschwindigkeiten jeder Bewegung.
    - Move.set_junction() implementiert den "Trapezoid-Generator" auf einen Zug. Der "Trapezgenerator" unterteilt jede Bewegung in drei Teile: eine Phase mit konstanter Beschleunigung, gefolgt von einer Phase mit konstanter Geschwindigkeit, gefolgt von einer Phase mit konstanter Verzögerung. Jede Bewegung enthält diese drei Phasen in dieser Reihenfolge, aber einige Phasen können von Null Dauer sein.
    - Wenn ToolHead._process_moves() aufgerufen wird, ist alles über die Bewegung bekannt - die Startposition, die Endposition, die Beschleunigung, die Start-/Gleit-/Endgeschwindigkeit und die bei der Beschleunigung/Gleit-/Abbremsung zurückgelegte Strecke. Alle Informationen werden in der Klasse Move() gespeichert und liegen im kartesischen Raum in Einheiten von Millimetern und Sekunden vor.
- Klipper verwendet einen iterative solver, um die Schrittzeiten für jeden Stepper zu erzeugen. Aus Gründen der Effizienz werden die Schrittimpulszeiten in C-Code erzeugt. Die Bewegungen werden zunächst in eine "Trapezbewegungs-Warteschlange" gestellt: `ToolHead._process_moves() -> trapq_append()` (in klippy/chelper/trapq.c). Die Schrittzeiten werden dann generiert: `ToolHead._process_moves() -> ToolHead._update_move_time() -> MCU_Stepper.generate_steps() -> itersolve_generate_steps() -> itersolve_gen_steps_range()` (in klippy/chelper/itersolve.c). Das Ziel des iterativen Solvers ist es, Schrittzeiten zu finden, die von einer Funktion vorgegeben werden, die eine Schrittposition aus einer Zeit berechnet. Dies geschieht durch wiederholtes "Raten" verschiedener Zeiten, bis die Formel für die Stepperposition die gewünschte Position des nächsten Schritts auf dem Stepper liefert. Das Feedback, das sich aus jeder Schätzung ergibt, wird zur Verbesserung künftiger Schätzungen verwendet, so dass der Prozess schnell zur gewünschten Zeit konvergiert. Die Formeln für die kinematische Stepperposition befinden sich im Verzeichnis klippy/chelper/ (z. B. kin_cart.c, kin_corexy.c, kin_delta.c, kin_extruder.c).

- Beachten Sie, dass der Extruder in seiner eigenen kinematischen Klasse behandelt wird: `ToolHead._process_moves() -> PrinterExtruder.move()`. Da die Move()-Klasse die genaue Bewegungszeit angibt und die Schrittimpulse mit einem bestimmten Timing an den Mikrocontroller gesendet werden, sind die von der Extruderklasse erzeugten Schrittbewegungen mit der Kopfbewegung synchron, obwohl der Code getrennt ist.

- Nachdem der iterative Löser die Schrittzeiten berechnet hat, werden sie zu einem Array hinzugefügt: `itersolve_gen_steps_range() -> stepcompress_append()` (in klippy/chelper/stepcompress.c). Das Array (struct stepcompress.queue) speichert die entsprechenden

Mikrocontroller-Taktzählerzeiten für jeden Schritt. Hier entspricht der "Mikrocontroller-Taktzähler"-Wert direkt dem Hardware-Zähler des Mikrocontrollers - er ist relativ zu dem Zeitpunkt, an dem der Mikrocontroller zuletzt eingeschaltet wurde.

- Der nächste wichtige Schritt ist die Komprimierung der Schritte: `stepcompress_flush() -> compress_bisect_add()` (in klippy/chelper/stepcompress.c). Dieser Code generiert und kodiert eine Reihe von "queue_step"-Befehlen des Mikrocontrollers, die der Liste der Schrittzeiten entsprechen, die im vorherigen Schritt erstellt wurde. Diese "queue_step"-Befehle werden dann in eine Warteschlange gestellt, nach Prioritäten geordnet und an den Mikrocontroller gesendet (über stepcompress.c:steppersync und serialqueue.c:serialqueue).

- Die Verarbeitung der queue_step-Befehle auf dem Mikrocontroller beginnt in src/command.c, das den Befehl parst und `command_queue_step()` aufruft. Der command_queue_step()-Code (in src/stepper.c) fügt lediglich die Parameter jedes queue_step-Befehls an eine Warteschlange pro Stepper an. Im Normalbetrieb wird der queue_step-Befehl mindestens 100 ms vor dem ersten Schritt geparst und in die Warteschlange gestellt. Schließlich erfolgt die Erzeugung von Stepper-Ereignissen in `stepper_event()`. Sie wird vom Hardware-Timer-Interrupt zum geplanten Zeitpunkt des ersten Schrittes aufgerufen. Der Code stepper_event() erzeugt einen Schrittimpuls und plant sich dann neu, um zum Zeitpunkt des nächsten Schrittimpulses für die angegebenen queue_step-Parameter zu laufen. Die Parameter für jeden queue_step-Befehl sind "interval", "count" und "add". Auf einer hohen Ebene führt stepper_event() folgendes aus, "count" mal: `do_step(); next_wake_time = last_wake_time + interval; interval += add;`

Die obigen Ausführungen mögen wie eine Menge Komplexität erscheinen, um eine Bewegung auszuführen. Die einzigen wirklich interessanten Teile befinden sich jedoch in den Klassen ToolHead und kinematic. In diesem Teil des Codes werden die Bewegungen und ihre Zeitvorgaben festgelegt. Der restliche Teil der Verarbeitung ist meist nur Kommunikation und Klempnerarbeit.

Hinzufügen eines Host-Moduls

Der Klippy-Hostcode verfügt über eine dynamische Modulladefunktion. Wenn in der Konfigurationsdatei des Druckers ein Abschnitt mit dem Namen "[my_module]" gefunden wird, versucht die Software automatisch, das Python-Modul klippy/extras/my_module.py zu laden. Dieses Modulsystem ist die bevorzugte Methode, um neue Funktionen zu Klipper hinzuzufügen.

Der einfachste Weg, ein neues Modul hinzuzufügen, ist, ein bestehendes Modul als Referenz zu verwenden - siehe klippy/extras/servo.py als Beispiel.

Das Folgende kann auch nützlich sein:

- Die Ausführung des Moduls beginnt in der Funktion `load_config()` auf Modulebene (für Konfigurationsabschnitte der Form [my_module]) oder in `load_config_prefix()` (für Konfigurationsabschnitte der Form [my_module my_name]). Dieser Funktion wird ein "config"-Objekt übergeben und sie muss ein neues "printer-Objekt" zurückgeben, das mit dem angegebenen Konfigurationsabschnitt verknüpft ist.
- Während des Prozesses der Instanziierung eines neuen Druckerobjekts kann das config-Objekt verwendet werden, um Parameter aus dem angegebenen config-Abschnitt zu lesen. Dies geschieht mit den Methoden `config.get()`, `config.getfloat()`, `config.getint()`, etc. Achten Sie darauf, dass Sie während der Erstellung des Druckerobjekts alle Werte aus der Config lesen - wenn der Benutzer einen Config-Parameter angibt, der in dieser Phase nicht gelesen wird, wird davon ausgegangen, dass es sich um einen Tippfehler in der Config handelt, und es wird ein Fehler ausgegeben.
- Verwenden Sie die Methode `config.get_printer()`, um einen Verweis auf die Hauptklasse "printer" zu erhalten. Diese "printer"-Klasse speichert Verweise auf alle "Druckerobjekte", die instanziiert wurden. Verwenden Sie die Methode `printer.lookup_object()`, um Verweise auf andere Druckerobjekte zu finden. Fast die gesamte Funktionalität (sogar die kinematischen Kernmodule) sind in einem dieser Druckerobjekte gekapselt. Beachten Sie jedoch, dass bei der Instanziierung eines neuen Moduls noch nicht alle anderen Druckerobjekte instanziiert wurden. Die Module "gcode" und "pins" werden immer verfügbar sein, aber für andere Module ist es eine gute Idee, die Suche zu verschieben.
- Registrieren Sie Event-Handler mit der Methode `printer.register_event_handler()`, wenn der Code bei "Ereignissen", die von anderen Druckerobjekten ausgelöst werden, aufgerufen werden soll. Jeder Ereignisname ist eine Zeichenkette und besteht vereinbarungsgemäß aus dem Namen des Hauptquellmoduls, das das Ereignis auslöst, sowie einem Kurznamen für die stattfindende Aktion (z. B. "klippy:connect"). Die Parameter, die an jeden Event-Handler übergeben werden, sind spezifisch für das jeweilige Ereignis (wie auch die Ausnahmebehandlung und der Ausführungskontext). Zwei gängige Startup-Ereignisse sind:
    - klippy:connect - Dieses Ereignis wird erzeugt, nachdem alle Druckerobjekte instanziiert wurden. Es wird üblicherweise verwendet, um andere Druckerobjekte zu suchen, die Konfigurationseinstellungen zu überprüfen und einen ersten "Handshake" mit der Druckerhardware durchzuführen.
    - klippy:ready - Dieses Ereignis wird ausgelöst, nachdem alle Connect-Handler erfolgreich abgeschlossen wurden. Es zeigt an, dass der Drucker in einen Zustand übergeht, in dem er für den normalen Betrieb bereit ist. Lösen Sie keinen Fehler in diesem Callback aus.
- Wenn ein Fehler in der Benutzerkonfiguration auftritt, sollten Sie diesen während der load_config() oder "connect event"-Phase auslösen. Verwenden Sie entweder `raise config.error("mein Fehler")` oder `raise printer.config_error`("mein Fehler"), um den Fehler zu melden.
- Verwenden Sie das Modul "pins", um einen Pin auf einem Mikrocontroller zu konfigurieren. Dies geschieht in der Regel mit etwas Ähnlichem wie `printer.lookup_object("pins").setup_pin("pwm", config.get("my_pin"))`. Das zurückgegebene Objekt kann dann zur Laufzeit befohlen werden.
- Wenn das Druckerobjekt eine get_status()-Methode definiert, kann das Modul Statusinformationen status information über macrosund über den API Serverr exportieren. Die Methode `get_status()` muss ein Python-Dictionary zurückgeben, dessen Schlüssel Strings und dessen Werte Integers, Floats, Strings, Listen, Dictionaries, True, False oder None sind. Es können auch Tupel (und benannte Tupel) verwendet werden (diese werden beim Zugriff über den API-Server als Listen angezeigt). Listen und Dictionaries, die exportiert werden, müssen als "unveränderlich" behandelt werden - wenn sich ihr Inhalt ändert, muss ein neues Objekt von `get_status()` zurückgegeben werden, andernfalls wird der API-Server diese Änderungen nicht erkennen.

- Wenn das Modul Zugriff auf die Systemzeitsteuerung oder externe Dateideskriptoren benötigt, verwenden Sie `printer.get_reactor()`, um Zugriff auf die globale Klasse "event reactor" zu erhalten. Diese Reactor-Klasse ermöglicht es, Timer zu planen, auf Eingaben in Dateideskriptoren zu warten und den Host-Code "schlafen" zu legen.
- Verwenden Sie keine globalen Variablen. Der gesamte Status sollte in dem Druckerobjekt gespeichert werden, das von der Funktion `load_config()` zurückgegeben wird. Dies ist wichtig, da sonst der RESTART-Befehl möglicherweise nicht wie erwartet funktioniert. Wenn externe Dateien (oder Sockets) geöffnet sind, sollten Sie aus ähnlichen Gründen einen "klippy:disconnect"-Ereignishandler registrieren und diese über diesen Callback schließen.
- Vermeiden Sie den Zugriff auf die internen Mitgliedsvariablen (oder den Aufruf von Methoden, die mit einem Unterstrich beginnen) von anderen Druckerobjekten. Die Einhaltung dieser Konvention macht es einfacher, zukünftige Änderungen zu verwalten.
- Es wird empfohlen, allen Mitgliedsvariablen im Python-Konstruktor von Python-Klassen einen Wert zuzuweisen. (Und damit die Fähigkeit von Python, dynamisch neue Mitgliedsvariablen zu erstellen, nicht zu nutzen.)
- Wenn eine Python-Variable einen Fließkommawert speichern soll, ist es empfehlenswert, diese Variable immer mit Fließkomma-Konstanten zuzuweisen und zu manipulieren (und niemals Ganzzahl-Konstanten zu verwenden). Ziehen Sie zum Beispiel `self.speed = 1.` gegenüber `self.speed = 1` vor, und ziehen Sie `self.speed = 2. * x` gegenüber `self.speed = 2 * x`. Die konsequente Verwendung von Fließkommawerten kann schwer zu debuggende Macken in Python-Typkonvertierungen vermeiden.
- Wenn Sie das Modul zur Aufnahme in den Klipper-Hauptcode einreichen, stellen Sie sicher, dass Sie einen Copyright-Vermerk am Anfang des Moduls platzieren. Siehe die bestehenden Module für das bevorzugte Format.

Hinzufügen neuer Kinematiken

Dieser Abschnitt enthält einige Tipps, wie man Klipper um zusätzliche Arten von Druckerkinematiken erweitern kann. Diese Art von Aktivität erfordert ein ausgezeichnetes Verständnis der mathematischen Formeln für die Zielkinematik. Sie erfordert auch Kenntnisse in der Softwareentwicklung - obwohl man eigentlich nur die Hostsoftware aktualisieren muss.

Nützliche Schritte:

1. Beginnen Sie mit dem Studium des Abschnitts "code flow of a move" und des Kinematics document.
2. Schauen Sie sich die vorhandenen kinematischen Klassen im Verzeichnis klippy/kinematics/ an. Die Kinematik-Klassen haben die Aufgabe, eine Bewegung in kartesischen Koordinaten in die Bewegung auf jedem Stepper umzuwandeln. Man sollte in der Lage sein, eine dieser Dateien als Ausgangspunkt zu kopieren.
3. Implementieren Sie die kinematischen C-Stepper-Positionsfunktionen für jeden Stepper, falls sie nicht bereits vorhanden sind (siehe kin_cart.c, kin_corexy.c und kin_delta.c in klippy/chelper/). Die Funktion sollte `move_get_coord()` aufrufen, um eine gegebene Bewegungszeit (in Sekunden) in eine kartesische Koordinate (in Millimetern) umzuwandeln, und dann die gewünschte Stepperposition (in Millimetern) aus dieser kartesischen Koordinate berechnen.
4. Implementieren Sie die Methode `calc_position()` in der neuen Kinematikklasse. Diese Methode berechnet die Position des Werkzeugkopfs in kartesischen Koordinaten aus der Position der einzelnen Stepper. Sie muss nicht effizient sein, da sie in der Regel nur bei Referenzfahrt- und Antastvorgängen aufgerufen wird.
5. Andere Methoden. Implementieren Sie die Methoden `check_move()`, `get_status()`, `get_steppers()`, `home()` und `set_position()`. Diese Funktionen werden in der Regel für kinematikspezifische Überprüfungen verwendet. Zu Beginn der Entwicklung kann man hier jedoch auch Standardcode verwenden.
6. Implementieren Sie Testfälle. Erstellen Sie eine G-Code-Datei mit einer Reihe von Bewegungen, die wichtige Fälle für die gegebene Kinematik testen können. Folgen Sie der debugging documentation, um diese G-Code-Datei in Mikrocontroller-Befehle umzuwandeln. Dies ist nützlich, um Eckfälle zu üben und auf Regressionen zu prüfen.

Portierung auf einen neuen Mikrocontroller

Dieser Abschnitt enthält einige Tipps zur Portierung des Klipper-Mikrocontroller-Codes auf eine neue Architektur. Diese Art von Aktivität erfordert gute Kenntnisse der Embedded-Entwicklung und praktischen Zugang zum Ziel-Mikrocontroller.

Nützliche Schritte:

1. Beginnen Sie damit, alle Bibliotheken von Drittanbietern zu identifizieren, die während der Portierung verwendet werden sollen. Übliche Beispiele sind "CMSIS"-Wrapper und "HAL"-Bibliotheken der Hersteller. Der gesamte Code von Drittanbietern muss mit der GNU GPLv3 kompatibel sein. Der Code von Drittanbietern sollte in das Klipper lib/ Verzeichnis übertragen werden. Aktualisieren Sie die lib/README-Datei mit Informationen darüber, wo und wann die Bibliothek bezogen wurde. Es ist vorzuziehen, den Code unverändert in das Klipper-Repository zu kopieren, aber wenn Änderungen erforderlich sind, sollten diese Änderungen explizit in der lib/README-Datei aufgeführt werden.
2. Erstellen Sie ein neues Architektur-Unterverzeichnis im src/-Verzeichnis und fügen Sie die anfängliche Unterstützung für Kconfig und Makefile hinzu. Verwenden Sie die vorhandenen Architekturen als Leitfaden. Der src/simulator bietet ein grundlegendes Beispiel für einen minimalen Startpunkt.
3. Die erste Hauptaufgabe bei der Programmierung besteht darin, die Kommunikationsunterstützung für die Zielplatine einzurichten. Dies ist der schwierigste Schritt bei einem neuen Port. Sobald die grundlegende Kommunikation funktioniert, sind die verbleibenden Schritte in der Regel viel einfacher. In der Regel wird in der Anfangsphase der Entwicklung ein serielles Gerät vom Typ UART verwendet, da diese Art von Hardware-Geräten im Allgemeinen einfacher zu aktivieren und zu steuern ist. In dieser Phase sollten Sie den Hilfscode aus dem Verzeichnis src/generic/ ausgiebig nutzen (überprüfen Sie, wie src/simulator/Makefile den generischen C-Code in den Build einbindet). Es ist auch notwendig, timer_read_time() (das die aktuelle Systemuhr zurückgibt) in dieser Phase zu definieren, aber es ist nicht notwendig, Timer-Irq-Handling vollständig zu unterstützen.
4. Machen Sie sich mit dem Werkzeug console.py vertraut (wie im debugging document beschrieben) und überprüfen Sie damit die Verbindung zum Mikrocontroller. Dieses Tool übersetzt das Low-Level-Kommunikationsprotokoll des Mikrocontrollers in eine für Menschen lesbare Form.

5. Hinzufügen von Unterstützung für Timer-Dispatch von Hardware-Interrupts. Siehe Klipper commit 970831ee als Beispiel für die Schritte 1-5 für die LPC176x Architektur.
6. Grundlegende Unterstützung für GPIO-Eingänge und -Ausgänge einführen. Siehe Klipper commit c78b9076 als Beispiel hierfür.
7. Zusätzliche Peripheriegeräte einrichten - siehe zum Beispiel Klipper Commit 65613aed, c812a40a und c381d03a.
8. Erstelle eine Klipper-Beispielkonfigurationsdatei im Verzeichnis config/. Testen Sie den Mikrocontroller mit dem Hauptprogramm klippy.py.
9. Erwäge das Hinzufügen von Build-Testfällen in das test/-Verzeichnis.

Zusätzliche Tipps zur Kodierung:

1. Vermeiden Sie die Verwendung von "C-Bitfeldern" für den Zugriff auf IO-Register; bevorzugen Sie direkte Lese- und Schreiboperationen von 32bit-, 16bit- oder 8bit-Ganzzahlen. Die C-Sprachenspezifikationen geben nicht eindeutig an, wie der Compiler C-Bitfelder implementieren muss (z. B. Endianness und Bit-Layout), und es ist schwierig zu bestimmen, welche IO-Operationen beim Lesen oder Schreiben eines C-Bitfelds auftreten.
2. Bevorzugen Sie es, explizite Werte in IO-Register zu schreiben, anstatt Lese-Änderungs-Schreib-Operationen zu verwenden. Das heißt, wenn Sie ein Feld in einem IO-Register aktualisieren, dessen andere Felder bekannte Werte haben, ist es besser, den gesamten Inhalt des Registers explizit zu schreiben. Explizite Schreibvorgänge erzeugen einen kleineren, schnelleren und leichter zu debuggenden Code.

Koordinatensysteme

Intern verfolgt Klipper hauptsächlich die Position des Werkzeugkopfes in kartesischen Koordinaten, die relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem sind. Das heißt, der größte Teil des Klipper-Codes wird nie eine Änderung des Koordinatensystems erfahren. Wenn der Benutzer eine Änderung des Ursprungs anfordert (z. B. ein G92-Befehl), wird dieser Effekt durch die Übersetzung zukünftiger Befehle in das primäre Koordinatensystem erzielt.

In manchen Fällen ist es jedoch nützlich, die Position des Werkzeugkopfes in einem anderen Koordinatensystem zu erhalten, und Klipper verfügt über mehrere Werkzeuge, die dies erleichtern. Dies kann mit dem Befehl GET_POSITION ermittelt werden. Zum Beispiel:

```
Send: GET_POSITION
Empfangen: // mcu: stepper_a:-2060 stepper_b:-1169 stepper_c:-1613
Empfangen: // stepper: stepper_a:457.254159 stepper_b:466.085669 stepper_c:465.382132
Empfangen: // Kinematik: X:8.339144 Y:-3.131558 Z:233.347121
Empfangen: // Werkzeugkopf: X:8.338078 Y:-3.123175 Z:233.347878 E:0.000000
Empfangen: // gcode: X:8.338078 Y:-3.123175 Z:233.347878 E:0.000000
Recv: // gcode base: X:0.000000 Y:0.000000 Z:0.000000 E:0.000000
Recv: // gcode homing: X:0.000000 Y:0.000000 Z:0.000000
```

Die "mcu"-Position (`stepper.get_mcu_position()` im Code) ist die Gesamtzahl der Schritte, die der Mikrocontroller in positiver Richtung ausgegeben hat, abzüglich der Anzahl der Schritte, die seit dem letzten Reset des Mikrocontrollers in negativer Richtung ausgegeben wurden. Wenn sich der Roboter zum Zeitpunkt der Abfrage in Bewegung befindet, umfasst der gemeldete Wert die auf dem Mikrocontroller gepufferten Bewegungen, nicht aber die Bewegungen in der Vorausschau-Warteschlange.

Die "Stepper"-Position (`stepper.get_commanded_position()`) ist die Position des angegebenen Steppers, wie sie vom Kinematik-Code verfolgt wird. Dies entspricht im Allgemeinen der Position (in mm) des Schlittens entlang seiner Schiene, relativ zu dem in der Konfigurationsdatei angegebenen position_endstop. (Einige Kinematiken verfolgen die Stepperpositionen im Bogenmaß statt in Millimetern.) Wenn der Roboter in Bewegung ist, wenn die Abfrage gestellt wird, dann schließt der gemeldete Wert die auf dem Mikrocontroller gepufferten Bewegungen ein, aber nicht die Bewegungen in der Look-Ahead-Warteschlange. Man kann die Aufrufe `toolhead.flush_step_generation()` oder `toolhead.wait_moves()` verwenden, um den Vorausschau- und Schrittgenerierungscode vollständig zu löschen.

Die "kinematische" Position (`kin.calc_position()`) ist die kartesische Position des Werkzeugkopfes, die aus den "Stepper"-Positionen abgeleitet wird und relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem ist. Dies kann aufgrund der Granularität der Schrittmotoren von der angeforderten kartesischen Position abweichen. Wenn der Roboter in Bewegung ist, während die "Stepper"-Positionen ermittelt werden, enthält der gemeldete Wert die auf dem Mikrocontroller gepufferten Bewegungen, aber nicht die Bewegungen in der Vorausschau-Warteschlange. Man kann die Aufrufe toolhead.flush_step_generation() oder toolhead.wait_moves() verwenden, um den Vorausschau- und Schrittgenerierungscode vollständig zu löschen.

Die "Werkzeugkopf"-Position (`toolhead.get_position()`) ist die zuletzt angeforderte Position des Werkzeugkopfs in kartesischen Koordinaten relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem. Wenn sich der Roboter zum Zeitpunkt der Abfrage in Bewegung befindet, umfasst der gemeldete Wert alle angeforderten Bewegungen (auch diejenigen, die sich in Puffern befinden und darauf warten, an die Schrittmotortreiber ausgegeben zu werden).

Die "gcode"-Position ist die letzte von einem `G1`- (oder `G0`-) Befehl angeforderte Position in kartesischen Koordinaten relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem. Sie kann sich von der "Werkzeugkopf"-Position unterscheiden, wenn eine G-Code-Transformation (z. B. bed_mesh, bed_tilt, skew_correction) in Kraft ist. Sie kann von den tatsächlichen Koordinaten abweichen, die im letzten G1-Befehl angegeben wurden, wenn der G-Code-Ursprung geändert wurde (z. B. `G92, SET_GCODE_OFFSET, M221`). Der Befehl `M114` (`gcode_move.get_status()['gcode_position']`) meldet die letzte g-code Position relativ zum aktuellen g-code Koordinatensystem.

Die "gcode base" ist die Position des g-code Ursprungs in kartesischen Koordinaten relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem. Befehle wie `G92, SET_GCODE_OFFSET`, und `M221` ändern diesen Wert.

Die "gcode homing" ist die Position, die für den g-code Ursprung (in kartesischen Koordinaten relativ zu dem in der Konfigurationsdatei angegebenen Koordinatensystem) nach einem `G28` home Befehl verwendet werden soll. Mit dem Befehl `SET_GCODE_OFFSET` kann dieser Wert geändert werden.

Zeit

Grundlegend für den Betrieb von Klipper ist die Handhabung von Uhren, Zeiten und Zeitstempeln. Klipper führt Aktionen auf dem Drucker aus, indem er Ereignisse plant, die in naher Zukunft eintreten sollen. Um z.B. einen Lüfter einzuschalten, könnte der Code eine Änderung an einem GPIO-Pin in 100ms planen. Es ist selten, dass der Code versucht, eine sofortige Aktion auszuführen. Daher ist der Umgang mit der Zeit in Klipper entscheidend für den korrekten Betrieb.

Es gibt drei Arten von Zeiten, die intern in der Klipper-Host-Software verfolgt werden:

- Systemzeit. Die Systemzeit verwendet die monotone Uhr des Systems - es handelt sich um eine Gleitkommazahl, die in Sekunden gespeichert wird, und sie ist (im Allgemeinen) relativ zu dem Zeitpunkt, an dem der Host-Computer zuletzt gestartet wurde. Die Systemzeit wird in der Software nur in begrenztem Umfang verwendet - sie wird hauptsächlich bei der Interaktion mit dem Betriebssystem eingesetzt. Innerhalb des Host-Codes werden Systemzeiten häufig in Variablen namens eventtime oder curtime gespeichert.
- Druckzeit. Die Druckzeit wird mit der Hauptuhr des Mikrocontrollers synchronisiert (der Mikrocontroller, der in der "[mcu]"-Konfigurationssektion definiert ist). Es handelt sich um eine Fließkommazahl, die in Sekunden gespeichert wird und sich auf den Zeitpunkt des letzten Neustarts der Hauptmikrosteuerung bezieht. Es ist möglich, eine "Druckzeit" in die Hardware-Uhr des Haupt-Mikrocontrollers umzuwandeln, indem man die Druckzeit mit der statisch konfigurierten Frequenzrate der Mikrosteuerung multipliziert. Der High-Level-Host-Code verwendet Druckzeiten zur Berechnung fast aller physischen Aktionen (z. B. Kopfbewegung, Heizungswechsel usw.). Innerhalb des Host-Codes werden die Druckzeiten im Allgemeinen in Variablen namens print_time oder move_time gespeichert.
- MCU-Uhr. Dies ist der Hardware-Taktzähler auf jedem Mikrocontroller. Er wird als Ganzzahl gespeichert und seine Aktualisierungsrate ist relativ zur Frequenz des jeweiligen Mikrocontrollers. Die Host-Software wandelt ihre internen Zeiten in Uhren um, bevor sie an die MCU übertragen werden. Der mcu-Code verfolgt die Zeit immer nur in Ticks. Innerhalb des Host-Codes werden die Taktwerte als 64-Bit-Ganzzahlen verfolgt, während der Mikrosteuerungscode 32-Bit-Ganzzahlen verwendet. Innerhalb des Host-Codes werden die Uhrwerte im Allgemeinen in Variablen gespeichert, deren Namen Clock oder Ticks enthalten.

Die Konvertierung zwischen den verschiedenen Zeitformaten wird hauptsächlich im klippy/clocksync.py Code implementiert.

Bei der Durchsicht des Codes sollten einige Dinge beachtet werden:

- 32-Bit- und 64-Bit-Taktgeber: Um die Bandbreite zu verringern und die Effizienz des Mikrocontrollers zu verbessern, werden die Takte auf dem Mikrocontroller als 32-Bit-Integer verfolgt. Beim Vergleich zweier Takte im Mikrocontroller-Code muss immer die Funktion `timer_is_before()` verwendet werden, um sicherzustellen, dass Integer-Rollover ordnungsgemäß behandelt werden. Die Host-Software konvertiert 32-Bit-Takte in 64-Bit-Takte, indem sie die höherwertigen Bits des letzten von der MCU empfangenen Zeitstempels anhängt - keine Nachricht von der MCU liegt mehr als 2^31 Ticks in der Zukunft oder Vergangenheit, so dass diese Konvertierung nie mehrdeutig ist. Der Host konvertiert von 64-Bit-Takten in 32-Bit-Takte, indem er einfach die höherwertigen Bits abschneidet. Um sicherzustellen, dass es bei dieser Umwandlung keine Mehrdeutigkeit gibt, puffert der Code klippy/chelper/serialqueue.c Nachrichten so lange, bis sie innerhalb von 2^31 Taktschlägen von ihrer Zielzeit liegen.
- Mehrere Mikrocontroller: Die Host-Software unterstützt die Verwendung mehrerer Mikrocontroller an einem einzigen Drucker. In diesem Fall wird die "MCU-Uhr" jedes Mikrocontrollers separat verfolgt. Der clocksync.py-Code behandelt die Taktdrift zwischen den Mikrocontrollern, indem er die Art der Konvertierung von "Druckzeit" in "MCU-Takt" ändert. Bei sekundären Mikrocontrollern wird die MCU-Frequenz, die bei dieser Umwandlung verwendet wird, regelmäßig aktualisiert, um die gemessene Drift zu berücksichtigen.

Kinematik

Dieses Dokument gibt einen Überblick darüber, wie Klipper die Roboterbewegung implementiert (seine kinematics). Der Inhalt kann sowohl für Entwickler, die an der Klipper-Software arbeiten wollen, als auch für Benutzer, die die Mechanik ihrer Maschinen besser verstehen wollen, von Interesse sein.

Beschleunigung

Klipper implementiert ein konstantes Beschleunigungsschema, wann immer der Druckkopf seine Geschwindigkeit ändert - die Geschwindigkeit wird allmählich an die neue Geschwindigkeit angepasst, anstatt ruckartig auf sie zu beschleunigen. Klipper erzwingt immer eine Beschleunigung zwischen dem Werkzeugkopf und dem Druck. Das Filament, das den Extruder verlässt, kann sehr empfindlich sein - schnelle Rucke und/oder Änderungen der Extrudergeschwindigkeit führen zu schlechter Qualität und schlechter Bettanhaftung. Selbst wenn nicht extrudiert wird, kann ein schnelles Ruckeln des Druckkopfs, wenn er sich auf gleicher Höhe mit dem Druck befindet, zu einer Unterbrechung des gerade aufgebrachten Filaments führen. Eine Begrenzung der Geschwindigkeitsänderungen des Druckkopfs (relativ zum Druck) verringert das Risiko einer Unterbrechung des Drucks.

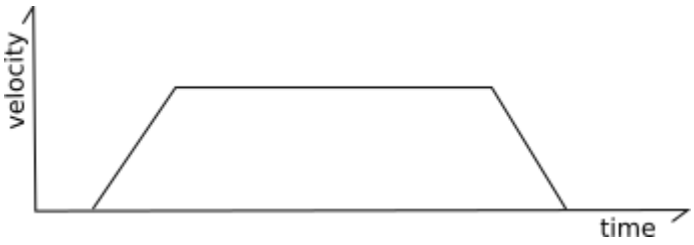
Es ist auch wichtig, die Beschleunigung zu begrenzen, damit die Schrittmotoren nicht aussetzen oder die Maschine übermäßig belasten. Klipper begrenzt das Drehmoment jedes Schrittmotors durch die Begrenzung der Beschleunigung des Druckkopfs. Die Erzwingung der Beschleunigung am Druckkopf begrenzt natürlich auch das Drehmoment der Schrittmotoren, die den Druckkopf bewegen (das Umgekehrte ist nicht immer der Fall).

Klipper implementiert konstante Beschleunigung. Die Schlüsselformel für eine konstante Beschleunigung lautet:

```
Geschwindigkeit(Zeit) = Start_Geschwindigkeit + Beschleunigung*Zeit
velocity(time) = start_velocity + accel*time
```

Trapezförmiger Generator

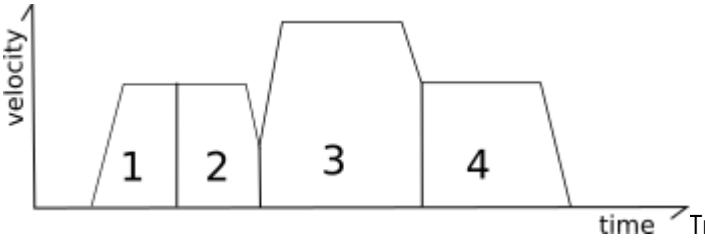
Klipper verwendet einen traditionellen "Trapezgenerator", um die Bewegung jedes Zuges zu modellieren - jeder Zug hat eine Startgeschwindigkeit, er beschleunigt mit konstanter Beschleunigung auf eine Reisegeschwindigkeit, er fährt mit konstanter Geschwindigkeit und verlangsamt dann mit konstanter Beschleunigung auf die Endgeschwindigkeit.



trapezförmig

Er wird "Trapezgenerator" genannt, weil ein Geschwindigkeitsdiagramm der Bewegung wie ein Trapez aussieht.

Die Reisegeschwindigkeit ist immer größer oder gleich der Startgeschwindigkeit und der Endgeschwindigkeit. Die Beschleunigungsphase kann von Null Dauer sein (wenn die Startgeschwindigkeit gleich der Reisegeschwindigkeit ist), die Reisephase kann von Null Dauer sein (wenn der Zug nach der Beschleunigung sofort zu verlangsamen beginnt) und/oder die Verzögerungsphase kann von Null Dauer sein (wenn die Endgeschwindigkeit gleich der Reisegeschwindigkeit ist).



Trapezoide

Vorausschau

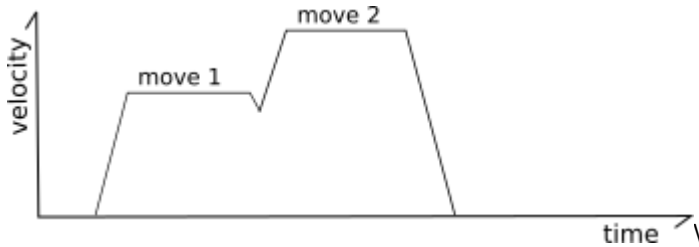
Das "Look-ahead"-System wird verwendet, um die Kurvengeschwindigkeiten zwischen den Zügen zu bestimmen.

Betrachten Sie die folgenden zwei Züge, die in einer XY-Ebene liegen:

Ecke

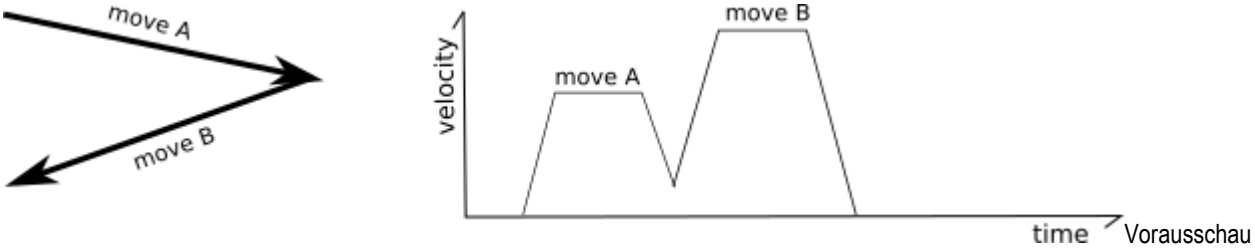
In der obigen Situation ist es möglich, nach der ersten Bewegung vollständig abzubremsen und dann zu Beginn der nächsten Bewegung vollständig zu beschleunigen, aber das ist nicht ideal, da die gesamte Beschleunigung und Abbremsung die Druckzeit stark verlängern würde und die häufigen Änderungen des Extruderflusses zu einer schlechten Druckqualität führen würden.

Um dieses Problem zu lösen, stellt der "Vorausschau"-Mechanismus mehrere ankommende Bewegungen in eine Warteschlange und analysiert die Winkel zwischen den Bewegungen, um eine angemessene Geschwindigkeit zu bestimmen, die während der "Kreuzung" zwischen zwei Bewegungen erreicht werden kann. Wenn die nächste Bewegung fast in dieselbe Richtung geht, muss der Kopf nur ein wenig langsamer werden (wenn überhaupt).



Vorausschauende Betrachtung

Wenn der nächste Zug jedoch einen spitzen Winkel bildet (der Kopf fährt beim nächsten Zug fast in die entgegengesetzte Richtung), dann ist nur eine geringe Übergangsgeschwindigkeit zulässig.



Vorausschau

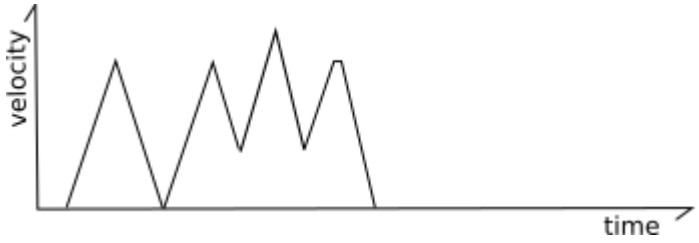
Die Kreuzungsgeschwindigkeiten werden mit Hilfe der "angenäherten Zentripetalbeschleunigung" bestimmt. Die beste Beschreibung stammt vom Autor described by the author. In Klipper werden die Kreuzungsgeschwindigkeiten jedoch konfiguriert, indem man die gewünschte Geschwindigkeit angibt, die eine 90°-Ecke haben sollte (die "quadratische Eckgeschwindigkeit"), und die Kreuzungsgeschwindigkeiten für andere Winkel werden davon abgeleitet.

Schlüsselformel für die Vorausschau:

```
end_velocity^2 = start_velocity^2 + 2*accel*move_distance
```

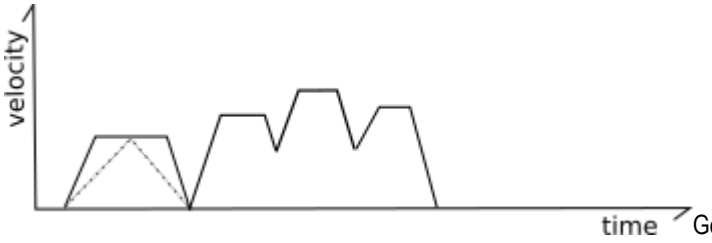
Geglättete Vorausschau

Klipper implementiert auch einen Mechanismus zur Glättung der Bewegungen von kurzen "Zickzack"-Bewegungen. Betrachte die folgenden Züge:



zigzag

Der häufige Wechsel von Beschleunigung und Abbremsung kann zu Vibrationen führen, die die Maschine belasten und die Geräuschentwicklung erhöhen. Um dies zu reduzieren, verfolgt Klipper sowohl die reguläre Beschleunigung der Bewegung als auch eine virtuelle "Beschleunigungs- und Abbremsrate". Mit diesem System wird die Höchstgeschwindigkeit dieser kurzen "Zickzack"-Bewegungen begrenzt, um die Bewegung des Druckers zu glätten:



Geglättet

Konkret berechnet der Code, wie hoch die Geschwindigkeit jedes Zuges wäre, wenn er auf diese virtuelle "Beschleunigung-zu-Verzögerung"-Rate (standardmäßig die Hälfte der normalen Beschleunigungsrate) begrenzt wäre. In der obigen Abbildung stellen die gestrichelten grauen Linien diese virtuelle Beschleunigungsrate für den ersten Zug dar. Wenn ein Zug mit dieser virtuellen Beschleunigungsrate nicht seine volle Reisegeschwindigkeit erreichen kann, wird seine Höchstgeschwindigkeit auf die maximale Geschwindigkeit reduziert, die er mit dieser virtuellen Beschleunigungsrate erreichen kann. Bei den meisten Zügen liegt der Grenzwert bei oder über den bestehenden Grenzen des Zuges, und es wird keine Verhaltensänderung herbeigeführt. Bei kurzen Zickzack-Bewegungen wird jedoch die Höchstgeschwindigkeit reduziert. Beachten Sie, dass die tatsächliche Beschleunigung innerhalb des Zuges dadurch nicht verändert wird - der Zug verwendet weiterhin das normale Beschleunigungsschema bis zu seiner angepassten Höchstgeschwindigkeit.

Schritte generieren

Sobald der Vorausschau-Prozess abgeschlossen ist, ist die Druckkopfbewegung für die gegebene Bewegung vollständig bekannt (Zeit, Startposition, Endposition, Geschwindigkeit an jedem Punkt) und es ist möglich, die Schrittzeiten für die Bewegung zu erzeugen. Dieser Prozess wird innerhalb der "kinematischen Klassen" im Klipper-Code durchgeführt. Außerhalb dieser kinematischen Klassen wird alles in Millimetern, Sekunden und im kartesischen Koordinatenraum verfolgt. Es ist die Aufgabe der kinematischen Klassen, von diesem allgemeinen Koordinatensystem auf die Hardwarespezifika des jeweiligen Druckers umzurechnen.

Klipper verwendet einen iterative solver, um die Schrittzeiten für jeden Stepper zu erzeugen. Der Code enthält die Formeln zur Berechnung der idealen kartesischen Koordinaten des Kopfes zu jedem Zeitpunkt und die kinematischen Formeln zur Berechnung der idealen Stepperpositionen auf der Grundlage dieser kartesischen Koordinaten. Mit diesen Formeln kann Klipper den idealen Zeitpunkt bestimmen, an dem sich der Stepper an jeder Schrittposition befinden sollte. Die vorgegebenen Schritte werden dann zu diesen berechneten Zeiten geplant.

Die Schlüsselformel, um zu bestimmen, wie weit eine Bewegung bei konstanter Beschleunigung gehen soll, lautet:

```
move_distance = (start_velocity + .5 * accel * move_time) * move_time
```

und die Schlüsselformel für eine Bewegung mit konstanter Geschwindigkeit ist:

```
move_distance = cruise_velocity * move_time
```

Die Schlüsselformel für die Bestimmung der kartesischen Koordinate einer Bewegung bei gegebener Bewegungsdistanz lautet:

```
cartesian_x_position = start_x + move_distance * total_x_movement / total_movement
cartesian_y_position = start_y + move_distance * total_y_movement / total_movement
```

```
cartesian_z_position = start_z + move_distance * total_z_movement / total_movement
```

Kartesische Roboter

Die Erzeugung von Schritten für kartesische Drucker ist der einfachste Fall. Die Bewegung auf jeder Achse ist direkt mit der Bewegung im kartesischen Raum verbunden.

Die wichtigsten Formeln:

```
stepper_x_position = kartesische_x_position
stepper_y_position = kartesische_y_position
stepper_z_position = kartesische_z_position
```

CoreXY-Roboter¶

Die Erzeugung von Schritten auf einer CoreXY-Maschine ist nur wenig komplexer als bei einfachen kartesischen Robotern. Die wichtigsten Formeln lauten:

```
stepper_a_position = kartesische_x_position + kartesische_y_position
stepper_b_position = kartesische_x_position - kartesische_y_position
stepper_z_position = kartesische_z_position
```

Delta-Roboter

Die Schrittgenerierung bei einem Delta-Roboter basiert auf dem Satz des Pythagoras:

```
stepper_position = (sqrt(arm_length^2

                         - (kartesische_x_Position - Turm_x_Position)^2
                         - (kartesische_y_Position - Turm_y_Position)^2)
                   + kartesische_z_Position)
```

Grenzen der Schrittmotorbeschleunigung

Bei der Delta-Kinematik ist es möglich, dass eine Bewegung, die im kartesischen Raum beschleunigt wird, eine Beschleunigung eines bestimmten Schrittmotors erfordert, die größer ist als die Beschleunigung der Bewegung. Dies kann vorkommen, wenn ein Schrittmotorarm mehr horizontal als vertikal ausgerichtet ist und die Bewegungslinie in der Nähe des Turms dieses Schrittmotors verläuft. Obwohl diese Bewegungen eine höhere Schrittmotorbeschleunigung als die maximal konfigurierte Bewegungsbeschleunigung des Druckers erfordern könnten, wäre die effektive Masse, die von diesem Schrittmotor bewegt wird, geringer. Die höhere Stepperbeschleunigung führt also nicht zu einem wesentlich höheren Stepper-Drehmoment und wird daher als unbedenklich angesehen.

Um jedoch Extremfälle zu vermeiden, erzwingt Klipper eine maximale Begrenzung der Stepperbeschleunigung auf das Dreifache der konfigurierten maximalen Bewegungsbeschleunigung des Druckers. (Ebenso ist die maximale Geschwindigkeit des Steppers auf das Dreifache der maximalen Bewegungsgeschwindigkeit begrenzt). Um diese Begrenzung durchzusetzen, sind die maximale Beschleunigung und Geschwindigkeit bei Bewegungen am äußersten Rand des Bauraums (wo ein Stepperarm nahezu horizontal sein kann) geringer.

Extruder-Kinematik

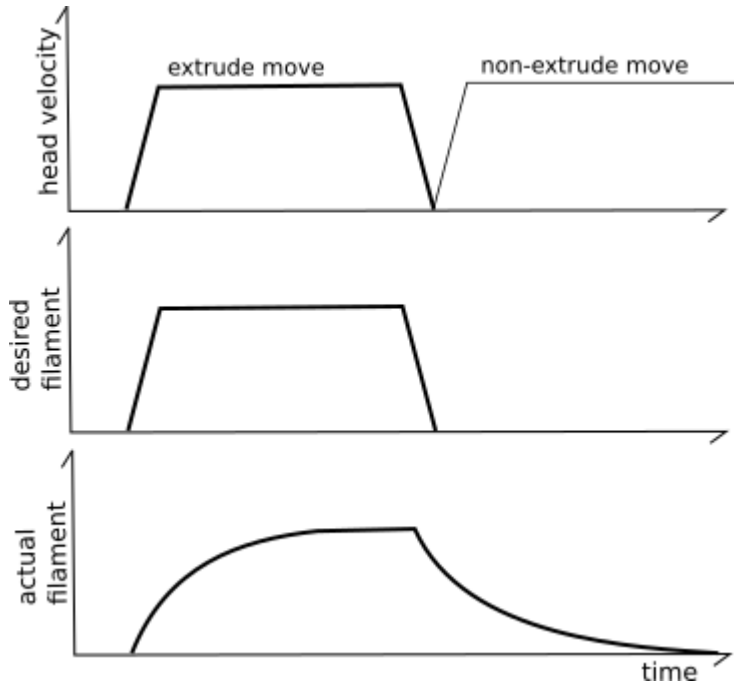
Klipper implementiert die Extruderbewegung in einer eigenen kinematischen Klasse. Da das Timing und die Geschwindigkeit jeder Druckkopfbewegung für jede Bewegung vollständig bekannt sind, ist es möglich, die Schrittzeiten für den Extruder unabhängig von den Schrittzeitberechnungen der Druckkopfbewegung zu berechnen.

Die grundlegende Extruderbewegung ist einfach zu berechnen. Die Schrittzeitgenerierung verwendet dieselben Formeln, die auch kartesische Roboter verwenden:

```
stepper_position = requested_e_position
```

Pressure Advance

Experimente haben gezeigt, dass es möglich ist, die Modellierung des Extruders über die grundlegende Extruderformel hinaus zu verbessern. Im Idealfall sollte im Verlauf einer Extrusionsbewegung an jedem Punkt der Bewegung das gleiche Volumen an Filament abgelegt werden und nach der Bewegung sollte kein Volumen extrudiert werden. Leider kommt es häufig vor, dass die grundlegenden Extrusionsformeln dazu führen, dass zu Beginn der Extrusionsbewegung zu wenig Filament aus dem Extruder austritt und nach dem Ende der Extrusion zu viel Filament extrudiert wird. Dies wird oft als "Sickern" bezeichnet.



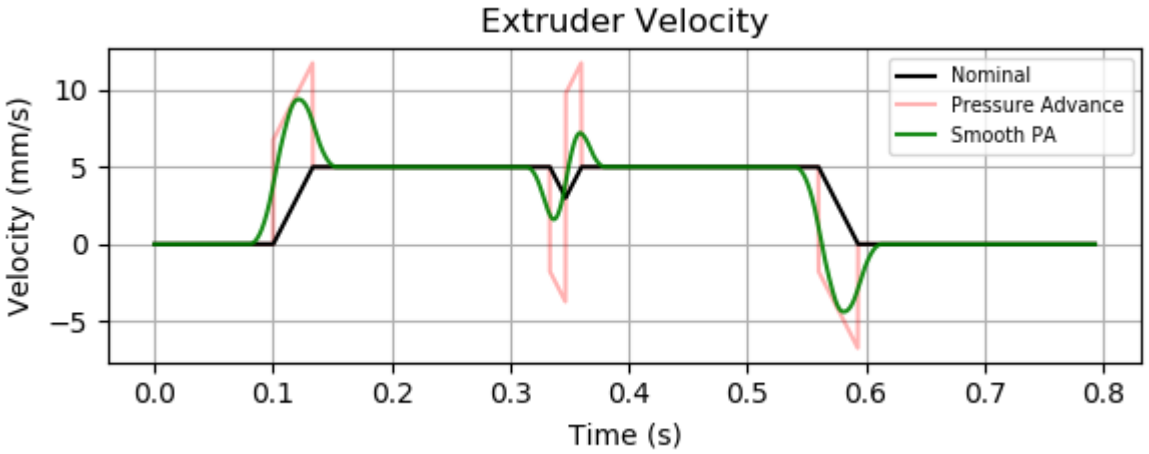
Ooze

Das System des "Druckvorschubs" versucht, diesem Umstand Rechnung zu tragen, indem es ein anderes Modell für den Extruder verwendet. Anstatt naiv zu glauben, dass jeder mm^3 Filament, der in den Extruder eingespeist wird, dazu führt, dass diese Menge an mm^3 sofort aus dem Extruder austritt, wird ein Modell verwendet, das auf dem Druck basiert. Der Druck nimmt zu, wenn das Filament in den Extruder gedrückt wird (wie im Hooke'schen Gesetz Hooke's law), und der zum Extrudieren erforderliche Druck wird von der Durchflussmenge durch die Düsenöffnung bestimmt (wie im Poiseuille'schen Gesetz Poiseuille's law). Der Grundgedanke ist, dass die Beziehung zwischen Faden, Druck und Durchflussmenge mit einem linearen Koeffizienten modelliert werden kann:

```
pa_position = nominal_position + pressure_advance_coefficient * nominal_velocity
```

Informationen zur Ermittlung dieses Druckvorkommens-Koeffizienten finden Sie im Dokument Druckvorkommen.

Die grundlegende Formel für den Druckvorschub pressure advance kann dazu führen, dass der Extrudermotor plötzliche Geschwindigkeitsänderungen vornimmt. Um dies zu vermeiden, führt Klipper eine "Glättung" der Extruderbewegung ein.



druck-vorschub

Das obige Diagramm zeigt ein Beispiel für zwei Extrusionsbewegungen mit einer von Null abweichenden Kurvengeschwindigkeit zwischen ihnen. Beachten Sie, dass das Druckvorschubsystem bewirkt, dass während der Beschleunigung zusätzliches Filament in den Extruder gedrückt wird. Je höher der gewünschte Filamentdurchsatz ist, desto mehr Filament muss während der Beschleunigung eingeschoben werden, um den Druck zu berücksichtigen. Beim Abbremsen des Kopfes wird das zusätzliche Filament zurückgezogen (der Extruder hat dann eine negative Geschwindigkeit).

Die "Glättung" wird durch einen gewichteten Durchschnitt der Extruderposition über eine kleine Zeitspanne (wie durch den Konfigurationsparameter pressure_advance_smooth_time festgelegt) implementiert. Diese Mittelwertbildung kann sich über mehrere G-Code-Bewegungen erstrecken. Beachten Sie, wie sich der Extrudermotor vor dem nominalen Start der ersten Extrusionsbewegung in Bewegung setzt und sich nach dem nominalen Ende der letzten Extrusionsbewegung weiter bewegt.

Schlüsselformel für den "geglätteten Druckvorschub":

```
smooth_pa_position(t) =
```

```
    ( definitives_integral(pa_position(x) * (smooth_time/2 - abs(t - x)) * dx,
                           from=t-smooth_time/2, to=t+smooth_time/2)
     / (glatte_Zeit/2)^2 )
```

Protokoll

Das Klipper-Nachrichtenprotokoll wird für die Low-Level-Kommunikation zwischen der Klipper-Host-Software und der Klipper-Mikrocontroller-Software verwendet. Auf einer hohen Ebene kann man sich das Protokoll als eine Reihe von Befehls- und Antwortstrings vorstellen, die komprimiert, übertragen und dann auf der Empfangsseite verarbeitet werden. Ein Beispiel für eine Reihe von Befehlen in unkomprimiertem, für Menschen lesbarem Format könnte wie folgt aussehen:

```
set_digital_out pin=PA3 value=1
set_digital_out pin=PA7 Wert=1
schedule_digital_out oid=8 clock=4000000 value=0
queue_step oid=7 interval=7458 count=10 add=331
queue_step oid=7 interval=11717 count=4 add=1281
```

Informationen zu den verfügbaren Befehlen finden Sie im Dokument mcu commands. Im Dokument debugging finden Sie Informationen darüber, wie Sie eine G-Code-Datei in die entsprechenden menschenlesbaren Mikrocontroller-Befehle übersetzen können.

Diese Seite bietet eine allgemeine Beschreibung des Klipper-Nachrichtenprotokolls selbst. Es beschreibt, wie Nachrichten deklariert, im Binärformat kodiert (das "Kompressionsschema") und übertragen werden.

Das Ziel des Protokolls ist es, einen fehlerfreien Kommunikationskanal zwischen dem Host und dem Mikrocontroller zu ermöglichen, der eine geringe Latenzzeit, eine geringe Bandbreite und eine geringe Komplexität für den Mikrocontroller aufweist.

Mikrocontroller-Schnittstelle

Das Klipper-Übertragungsprotokoll kann als ein RPC-Mechanismus zwischen Mikrocontroller und Host betrachtet werden. Die Software des Mikrocontrollers deklariert die Befehle, die der Host aufrufen kann, sowie die Antwortnachrichten, die er erzeugen kann. Der Host verwendet diese Informationen, um dem Mikrocontroller zu befehlen, Aktionen durchzuführen und die Ergebnisse zu interpretieren.

Deklaration von Befehlen

Die Mikrocontroller-Software deklariert einen "Befehl", indem sie das Makro DECL_COMMAND() im C-Code verwendet. Zum Beispiel:

```
DECL_COMMAND(command_update_digital_out, "update_digital_out oid=%c value=%c");
```

Der obige Befehl deklariert einen Befehl namens "update_digital_out". Dies ermöglicht dem Host, diesen Befehl "aufzurufen", was die Ausführung der C-Funktion command_update_digital_out() im Mikrocontroller zur Folge hat. Die obige Darstellung zeigt auch, dass der Befehl zwei Integer-Parameter benötigt. Wenn der C-Code command_update_digital_out() ausgeführt wird, wird ihm ein Array übergeben, das diese beiden Ganzzahlen enthält - die erste entspricht der "oid" und die zweite dem "value".

Im Allgemeinen werden die Parameter mit einer Syntax im Stil von printf() beschrieben (z. B. "%u"). Die Formatierung entspricht direkt der menschenlesbaren Ansicht der Befehle (z. B. "update_digital_out oid=7 value=1"). Im obigen Beispiel ist "value=" ein Parametername und "%c" zeigt an, dass der Parameter eine ganze Zahl ist. Intern wird der Parametername nur zur Dokumentation verwendet. In diesem Beispiel wird "%c" auch als Dokumentation verwendet, um anzuzeigen, dass die erwartete Ganzzahl 1 Byte groß ist (die angegebene Ganzzahlgröße hat keinen Einfluss auf das Parsing oder die Kodierung).

Der Mikrocontroller-Build sammelt alle mit DECL_COMMAND() deklarierten Befehle, bestimmt ihre Parameter und sorgt dafür, dass sie aufrufbar sind.

Antworten deklarieren

Um Informationen vom Mikrocontroller an den Host zu senden, wird eine "Antwort" erzeugt. Diese werden mit dem C-Makro sendf() deklariert und übertragen. Zum Beispiel:

```
sendf("status clock=%u status=%c", sched_read_time(), sched_is_shutdown());
```

Das obige Beispiel sendet eine "Status"-Antwortnachricht, die zwei ganzzahlige Parameter ("clock" und "status") enthält. Der Mikrocontroller-Build findet automatisch alle sendf()-Aufrufe und generiert Kodierer für sie. Der erste Parameter der Funktion sendf() beschreibt die Antwort und hat das gleiche Format wie die Befehlsdeklarationen.

Der Host kann für jede Antwort eine Callback-Funktion registrieren lassen. Befehle ermöglichen es dem Host also, C-Funktionen im Mikrocontroller aufzurufen, und Antworten ermöglichen es der Mikrocontroller-Software, Code im Host aufzurufen.

Das Makro sendf() sollte nur von Befehls- oder Task-Handlern aufgerufen werden, nicht aber von Interrupts oder Timern. Der Code muss kein sendf() als Antwort auf einen empfangenen Befehl ausgeben, er ist in der Anzahl der Aufrufe von sendf() nicht beschränkt und kann sendf() jederzeit von einem Task-Handler aus aufrufen.

Ausgabeantworten
Um die Fehlersuche zu vereinfachen, gibt es auch eine output() C-Funktion. Zum Beispiel:

```
output("Der Wert von %u ist %s mit der Größe %u.", x, buf, buf_len);
```

Die Funktion output() wird ähnlich wie printf() verwendet - sie ist dazu gedacht, beliebige Nachrichten für den menschlichen Konsum zu erzeugen und zu formatieren.

Deklarieren von Aufzählungen
Aufzählungen ermöglichen es dem Host-Code, String-Identifikatoren für Parameter zu verwenden, die der Mikrocontroller als Ganzzahlen behandelt. Sie werden im Code des Mikrocontrollers deklariert - zum Beispiel:

```
DECL_ENUMERATION("spi_bus", "spi", 0);

DECL_ENUMERATION_RANGE("pin", "PC0", 16, 8);
```

Im ersten Beispiel definiert das Makro DECL_ENUMERATION() eine Aufzählung für jede Befehls-/Antwortnachricht mit dem Parameternamen "spi_bus" oder einem Parameternamen mit dem Suffix "_spi_bus". Für diese Parameter ist die Zeichenkette "spi" ein gültiger Wert und wird mit einem Integer-Wert von Null übertragen.

Es ist auch möglich, einen Aufzählungsbereich zu deklarieren. Im zweiten Beispiel würde ein "pin"-Parameter (oder jeder Parameter mit dem Suffix "_pin") PC0, PC1, PC2, ..., PC7 als gültige Werte akzeptieren. Die Zeichenketten werden mit den Ganzzahlen 16, 17, 18, ..., 23 übertragen.

Konstanten deklarieren
Auch Konstanten können exportiert werden. Zum Beispiel das Folgende:

```
DECL_CONSTANT("SERIAL_BAUD", 250000);
```

würde eine Konstante mit dem Namen "SERIAL_BAUD" mit einem Wert von 250000 vom Mikrocontroller zum Host exportieren. Es ist auch möglich, eine Konstante zu deklarieren, die eine Zeichenkette ist - zum Beispiel:

```
DECL_CONSTANT_STR("MCU", "pru");
```

Low-Level-Nachrichtenkodierung
Um den oben genannten RPC-Mechanismus zu erreichen, wird jeder Befehl und jede Antwort für die Übertragung in ein binäres Format kodiert. Dieser Abschnitt beschreibt das Übertragungssystem.

Nachrichtenblöcke
Alle Daten, die vom Host zum Mikrocontroller und umgekehrt gesendet werden, sind in "Nachrichtenblöcken" enthalten. Ein Nachrichtenblock hat einen zwei Byte langen Header und einen drei Byte langen Trailer. Das Format eines Nachrichtenblocks ist:

```
<1 Byte Länge><1 Byte Sequenz><n Byte Inhalt><2 Byte Crc><1 Byte Sync>
```

Das Längenbyte enthält die Anzahl der Bytes im Nachrichtenblock einschließlich der Header- und Trailer-Bytes (die Mindestlänge der Nachricht beträgt also 5 Bytes). Die maximale Nachrichtenblocklänge beträgt derzeit 64 Byte. Das Sequenzbyte enthält eine 4-Bit-Sequenznummer in den niederwertigen Bits und die höherwertigen Bits enthalten immer 0x10 (die höherwertigen Bits sind für die zukünftige Verwendung reserviert). Die Content-Bytes enthalten beliebige Daten und ihr Format wird im folgenden Abschnitt beschrieben. Die crc-Bytes enthalten einen 16-Bit-CCCITT-CRC des Nachrichtenblocks einschließlich der Header-Bytes, aber ohne die Trailer-Bytes. Das Sync-Byte ist 0x7e.

Das Format des Nachrichtenblocks orientiert sich an den HDLC-Nachrichtenrahmen. Wie bei HDLC kann der Nachrichtenblock optional ein zusätzliches Sync-Zeichen am Anfang des Blocks enthalten. Anders als bei HDLC ist ein Sync-Zeichen nicht ausschließlich für die Rahmung vorgesehen und kann auch im Inhalt des Nachrichtenblocks vorhanden sein.

Inhalt des Meldungsblocks
Jeder vom Host an den Mikrocontroller gesendete Nachrichtenblock enthält eine Reihe von null oder mehr Nachrichtenbefehlen in seinem Inhalt. Jeder Befehl beginnt mit einer VLQ (Variable Length Quantity)-kodierten Befehls-ID, gefolgt von null oder mehr VLQ-Parametern für den jeweiligen Befehl.

So könnten beispielsweise die folgenden vier Befehle in einem einzigen Nachrichtenblock untergebracht werden:

```
update_digital_out oid=6 value=1
update_digital_out oid=5 Wert=0
get_config
get_clock
```

und kodiert in die folgenden acht VLQ-Ganzzahlen:

```
<id_update_digital_out><6><1><id_update_digital_out><5><0><id_get_config><id_get_clock>
```

Um den Inhalt der Nachricht zu kodieren und zu analysieren, müssen sich sowohl der Host als auch der Mikrocontroller auf die Befehlsnummern und die Anzahl der Parameter für jeden Befehl einigen. Im obigen Beispiel wüssten also sowohl der Host als auch der Mikrocontroller, dass "id_update_digital_out" immer von zwei Parametern gefolgt wird und "id_get_config" und "id_get_clock" null Parameter haben. Host und Mikrocontroller teilen sich ein "Datenverzeichnis", das die Befehlsbeschreibungen (z. B. "update_digital_out oid=%c value=%c") auf ihre ganzzahligen Befehls-IDs abbildet. Bei der Verarbeitung der Daten weiß der Parser, dass er eine bestimmte Anzahl von VLQ-kodierten Parametern nach einer bestimmten Befehls-ID erwarten muss.

Die Nachrichteninhalte für Blöcke, die vom Mikrocontroller an den Host gesendet werden, folgen demselben Format. Die Bezeichner in diesen Nachrichten sind "Antwort-IDs", aber sie dienen demselben Zweck und folgen denselben Kodierungsregeln. In der Praxis enthalten die vom Mikrocontroller an den Host gesendeten Nachrichtenblöcke nie mehr als eine Antwort im Inhalt des Nachrichtenblocks.

Quantitäten variabler Länge

Im wikipedia article finden Sie weitere Informationen über das allgemeine Format von VLQ-kodierten Ganzzahlen. Klipper verwendet ein Kodierungsschema, das sowohl positive als auch negative ganze Zahlen unterstützt. Ganzzahlen nahe bei Null benötigen weniger Bytes zur Kodierung und positive Ganzzahlen werden typischerweise mit weniger Bytes kodiert als negative Ganzzahlen. Die folgende Tabelle zeigt die Anzahl der Bytes, die jede ganze Zahl zur Kodierung benötigt:

Ganzzahl Kodierte Größe

Integer	Encoded size
-32 .. 95	1
-4096 .. 12287	2
-524288 .. 1572863	3
-67108864 .. 201326591	4
-2147483648 .. 4294967295	5

Zeichenketten variabler Länge

Abweichend von den obigen Kodierungsregeln wird ein Parameter für einen Befehl oder eine Antwort, der eine dynamische Zeichenkette ist, nicht als einfache VLQ-Ganzzahl kodiert. Stattdessen wird er kodiert, indem die Länge als VLQ-kodierte Ganzzahl übertragen wird, gefolgt vom eigentlichen Inhalt:

```
<VLQ-kodierte Länge><n-Byte-Inhalt>
```

Anhand der Befehlsbeschreibungen im Datenwörterbuch können sowohl der Host als auch der Mikrocontroller erkennen, welche Befehlsparameter eine einfache VLQ-Kodierung und welche Parameter eine String-Kodierung verwenden.

Datenwörterbuch

Damit eine sinnvolle Kommunikation zwischen Mikrocontroller und Host möglich ist, müssen sich beide Seiten auf ein "Datenwörterbuch" einigen. Dieses Datenwörterbuch enthält die ganzzahligen Bezeichner für Befehle und Antworten zusammen mit ihren Beschreibungen.

Der Mikrocontroller-Build verwendet den Inhalt der Makros DECL_COMMAND() und sendf(), um das Datenwörterbuch zu erstellen. Der Build weist jedem Befehl und jeder Antwort automatisch eindeutige Bezeichner zu. Mit diesem System können sowohl der Host- als auch der Mikrocontroller-Code nahtlos beschreibende, für den Menschen lesbare Namen verwenden, während gleichzeitig eine minimale Bandbreite genutzt wird.

Der Host fragt das Datenwörterbuch ab, wenn er sich zum ersten Mal mit dem Mikrocontroller verbindet. Sobald der Host das Datenwörterbuch vom Mikrocontroller heruntergeladen hat, verwendet er dieses Datenwörterbuch, um alle Befehle zu kodieren und alle Antworten des Mikrocontrollers zu parsen. Der Host muss also mit einem dynamischen Datenwörterbuch umgehen. Um die Software des Mikrocontrollers einfach zu halten, verwendet der Mikrocontroller jedoch immer sein statisches (einkompiliertes) Datenwörterbuch.

Das Datenwörterbuch wird abgefragt, indem "Identify"-Befehle an den Mikrocontroller gesendet werden. Der Mikrocontroller antwortet auf jeden Identify-Befehl mit einer "identify_response"-Nachricht. Da diese beiden Befehle vor der Abfrage des Datenverzeichnisses benötigt werden, sind ihre Integer-IDs und Parametertypen sowohl im Mikrocontroller als auch im Host fest codiert. Die ID der "identify_response"-Antwort ist 0, die ID des "identify"-Befehls ist 1. Abgesehen von den fest kodierten IDs werden der "identify"-Befehl und seine Antwort auf dieselbe Weise deklariert und übertragen wie andere Befehle und Antworten. Kein anderer Befehl oder keine andere Antwort ist fest kodiert.

Das Format des übertragenen Datenwörterbuchs selbst ist ein zlib-komprimierter JSON-String. Der Erstellungsprozess des Mikrocontrollers erzeugt den String, komprimiert ihn und speichert ihn im Textbereich des Mikrocontroller-Flashs. Das Datenwörterbuch kann viel größer sein als die maximale Größe des Nachrichtenblocks - der Host lädt es herunter, indem er mehrere Identifizierungsbefehle sendet, die fortschreitende Teile des Datenwörterbuchs anfordern. Sobald alle Teile erhalten sind, setzt der Host die Teile zusammen, dekomprimiert die Daten und parst den Inhalt.

Neben Informationen über das Kommunikationsprotokoll enthält das Datenwörterbuch auch die Softwareversion, Aufzählungen (wie durch DECL_ENUMERATION definiert) und Konstanten (wie durch DECL_CONSTANT definiert).

Nachrichtenfluss

Die vom Host an den Mikrocontroller gesendeten Nachrichtenbefehle sollen fehlerfrei sein. Der Mikrocontroller prüft die CRC- und Sequenznummern in jedem Nachrichtenblock, um sicherzustellen, dass die Befehle korrekt und in der richtigen Reihenfolge sind. Der Mikrocontroller verarbeitet Nachrichtenblöcke immer in der richtigen Reihenfolge - sollte er einen Block in falscher Reihenfolge empfangen, verwirft er ihn und alle anderen Blöcke in falscher Reihenfolge, bis er Blöcke in der richtigen Reihenfolge erhält.

Der Low-Level-Host-Code implementiert ein automatisches System zur erneuten Übertragung von verlorenen und beschädigten Nachrichtenblöcken, die an den Mikrocontroller gesendet werden. Um dies zu erleichtern, sendet der Mikrocontroller nach jedem erfolgreich empfangenen Nachrichtenblock einen "ack message block". Der Host plant nach dem Senden jedes Blocks eine Zeitüberschreitung ein und sendet erneut, wenn die Zeitüberschreitung abläuft, ohne ein entsprechendes "ack" zu erhalten. Stellt der Mikrocontroller einen beschädigten oder nicht ordnungsgemäßen Block fest, kann er außerdem einen "nak message block" senden, um eine schnelle Neuübertragung zu ermöglichen.

Ein "ack" ist ein Nachrichtenblock mit leerem Inhalt (d. h. ein 5-Byte-Nachrichtenblock) und einer Sequenznummer, die größer ist als die zuletzt empfangene Host-Sequenznummer. Ein "nak" ist ein Nachrichtenblock mit leerem Inhalt und einer Sequenznummer, die kleiner als die letzte empfangene Host-Sequenznummer ist.

Das Protokoll ermöglicht ein "Fenster"-Übertragungssystem, so dass der Host viele ausstehende Nachrichtenblöcke gleichzeitig in der Luft haben kann. (Dies gilt zusätzlich zu den vielen Befehlen, die in einem bestimmten Nachrichtenblock enthalten sein können). Dies ermöglicht eine maximale Bandbreitennutzung auch bei Übertragungsverzögerungen. Der Timeout-, Retransmit-, Windowing- und Ack-Mechanismus ist von ähnlichen Mechanismen in TCP inspiriert.

In der anderen Richtung sind die vom Mikrocontroller zum Host gesendeten Nachrichtenblöcke so konzipiert, dass sie fehlerfrei sind, aber sie haben keine gesicherte Übertragung. (Antworten sollten nicht verfälscht werden, aber sie können verloren gehen.) Dies geschieht, um die Implementierung im Mikrocontroller einfach zu halten. Es gibt kein automatisches System für die erneute Übertragung von Antworten - es wird erwartet, dass der High-Level-Code in der Lage ist, mit einer gelegentlich fehlenden Antwort umzugehen (in der Regel durch erneutes Anfordern des Inhalts oder Einrichten eines wiederkehrenden Zeitplans für die Übertragung von Antworten). Das Feld für die Sequenznummer in den an den Host gesendeten Nachrichtenblöcken ist immer um eins größer als die zuletzt empfangene Sequenznummer der vom Host empfangenen Nachrichtenblöcke. Es wird nicht zur Verfolgung von Sequenzen von Antwortnachrichtenblöcken verwendet.

API-Server

Dieses Dokument beschreibt die Anwendungsprogrammierschnittstelle (API) von Klipper. Diese Schnittstelle ermöglicht es externen Anwendungen, die Klipper-Host-Software abzufragen und zu steuern.

Aktivieren des API-Sockets

Um den API-Server nutzen zu können, muss die klippy.py Host-Software mit dem Parameter -a gestartet werden. Zum Beispiel:

```
~/klippy-env/bin/python ~/klipper/klippy/klippy.py ~/printer.cfg -a /tmp/klippy_uds -l
/tmp/klippy.log
```

Dies veranlasst die Host-Software, ein Unix Domain Socket zu erstellen. Ein Client kann dann eine Verbindung zu diesem Socket öffnen und Befehle an Klipper senden.

Siehe das Moonraker-Projekt für ein populäres Werkzeug, das HTTP-Anfragen an den Unix Domain Socket des Klipper API-Servers weiterleiten kann.

Anfrageformat

Nachrichten, die über den Socket gesendet und empfangen werden, sind JSON kodierte Strings, die mit einem ASCII 0x03 Zeichen abgeschlossen werden:

```
<json_object_1><0x03><json_object_2><0x03>...
```

Klipper enthält ein Werkzeug scripts/whconsole.py, das die obige Nachrichtenformulierung durchführen kann. Zum Beispiel:

~/klipper/scripts/whconsole.py /tmp/klippy_uds

Dieses Werkzeug kann eine Reihe von JSON-Befehlen von stdin lesen, sie an Klipper senden und die Ergebnisse melden. Das Tool erwartet, dass jeder JSON-Befehl in einer einzigen Zeile steht, und es fügt automatisch den 0x03-Terminator an, wenn es eine Anfrage sendet. (Der Klipper-API-Server verlangt keine Zeilenumbrüche).

API-Protokoll

Das auf dem Kommunikationssocket verwendete Befehlsprotokoll ist von json-rpc inspiriert.

Eine Anfrage könnte wie folgt aussehen:

```
{"id": 123, "method": "info", "params": {}}
```

und eine Antwort könnte wie folgt aussehen:

```
{"id": 123, "result": {"state_message": "Drucker ist bereit", "klipper_path": "/home/pi/klipper",
"config_file": "/home/pi/printer.cfg", "software_version": "v0.8.0-823-g883b1cb6", "hostname":
"octopi", "cpu_info": "4 core ARMv7 Processor rev 4 (v7l)", "state": "ready", "python_path":
"/home/pi/klippy-env/bin/python", "log_file": "/tmp/klippy.log"}}
```

Jede Anfrage muss ein JSON-Wörterbuch sein. (In diesem Dokument wird der Python-Begriff "Dictionary" verwendet, um ein "JSON-Objekt" zu beschreiben - eine Abbildung von Schlüssel/Wert-Paaren, die in `{}` enthalten sind).

Das Anforderungswörterbuch muss einen "method"-Parameter enthalten, der der Stringname eines verfügbaren Klipper-"Endpunkts" ist.

Das Anforderungswörterbuch kann einen "params"-Parameter enthalten, der vom Typ Wörterbuch sein muss. Die "params" liefern dem Klipper-"Endpunkt", der die Anfrage bearbeitet, zusätzliche Parameterinformationen. Sein Inhalt ist spezifisch für den "Endpunkt".

Das Anfrage-Dictionary kann einen "id"-Parameter enthalten, der von einem beliebigen JSON-Typ sein kann. Wenn "id" vorhanden ist, wird Klipper auf die Anfrage mit einer Antwortnachricht antworten, die diese "id" enthält. Wenn "id" weggelassen wird (oder auf einen JSON "null" Wert gesetzt wird), dann wird Klipper keine Antwort auf die Anfrage geben. Eine Antwortnachricht ist ein JSON-Dictionary, das "id" und "result" enthält. Das "Ergebnis" ist immer ein Wörterbuch - sein Inhalt ist spezifisch für den "Endpunkt", der die Anfrage bearbeitet.

Wenn die Verarbeitung einer Anfrage zu einem Fehler führt, enthält die Antwortnachricht ein "error"-Feld anstelle eines "result"-Feldes. Beispiel: Die Anfrage`: {"id": 123, "method": "gcode/script", "params": {"script": "G1 X200"}}` könnte zu einer Fehlerantwort wie der folgenden führen`: {"id": 123, "error": {"message": "Muss Achse zuerst starten: 200.000 0.000 0.000 [0.000]", "error": "WebRequestError"}}`

Klipper beginnt immer mit der Bearbeitung von Anfragen in der Reihenfolge, in der sie empfangen werden. Es kann jedoch vorkommen, dass einige Anfragen nicht sofort abgeschlossen werden, was dazu führen kann, dass die zugehörige Antwort nicht in der richtigen Reihenfolge zu den Antworten anderer Anfragen gesendet wird. Eine JSON-Anfrage unterbricht niemals die Verarbeitung von zukünftigen JSON-Anfragen.

Abonnements
Einige Klipper-"Endpunkt"-Anfragen erlauben es, zukünftige asynchrone Update-Nachrichten zu "abonnieren".

Zum Beispiel:

```
{"id": 123, "method": "gcode/subscribe_output", "params": {"response_template":{"key": 345}}}
```

kann zunächst mit antworten:

```
{"id": 123, "result": {}}
```

und Klipper dazu veranlassen, zukünftige Nachrichten ähnlich wie diese zu senden:

```
{"params": {"response": "ok B:22.8 /0.0 T0:22.4 /0.0"}, "key": 345}
```

Eine Abonnementanfrage akzeptiert ein "response_template"-Dictionary im "params"-Feld der Anfrage. Dieses "response_template"-Wörterbuch wird als Vorlage für künftige asynchrone Nachrichten verwendet - es kann beliebige Schlüssel/Wertpaare enthalten. Wenn diese zukünftigen asynchronen Nachrichten gesendet werden, fügt Klipper ein "params"-Feld mit einem Wörterbuch mit "endpoint"-spezifischen Inhalten zur Antwortvorlage hinzu und sendet diese Vorlage. Wenn kein "response_template"-Feld angegeben wird, wird standardmäßig ein leeres Wörterbuch (`{}`) verwendet.

Verfügbare "Endpunkte"

Konventionell haben Klipper "Endpunkte" die Form <Modulname>/<Einzelner_Name>. Wenn eine Anfrage an einen "Endpunkt" gestellt wird, muss der vollständige Name im Parameter "method" des Anfrageverzeichnisses angegeben werden (z.B. `{"method"="gcode/restart"}`).

info

Der "info"-Endpunkt wird verwendet, um System- und Versionsinformationen von Klipper zu erhalten. Er wird auch verwendet, um die Versionsinformationen des Clients an Klipper zu übermitteln. Zum Beispiel: `{"id": 123, "method": "info", "params": { "client_info": {"version": "v1"}}}`

Wenn vorhanden, muss der Parameter "client_info" ein Wörterbuch sein, das jedoch einen beliebigen Inhalt haben kann. Klienten wird empfohlen, den Namen des Klienten und seine Softwareversion anzugeben, wenn sie sich zum ersten Mal mit dem Klipper-API-Server verbinden.

notfall_stopp

Der "emergency_stop"-Endpunkt wird verwendet, um Klipper anzuweisen, in einen "shutdown"-Zustand überzugehen. Er verhält sich ähnlich wie der G-Code `M112` Befehl. Zum Beispiel: `{"id": 123, "Methode": "emergency_stop"}`

register_remote_method

Dieser Endpunkt ermöglicht es Kunden, Methoden zu registrieren, die von klipper aus aufgerufen werden können. Bei Erfolg wird ein leeres Objekt zurückgegeben.

Zum Beispiel: `{"id": 123, "method": "register_remote_method", "params": {"response_template": {"action": "run_paneldue_beep"}, "remote_method": "paneldue_beep"}}` wird zurückgegeben: `{"id": 123, "result": {}}`

Die Remote-Methode paneldue_beep kann nun von Klipper aus aufgerufen werden. Wenn die Methode Parameter benötigt, sollten diese als Schlüsselwortargumente angegeben werden. Nachfolgend ein Beispiel, wie die Methode von einem gcode_macro aufgerufen werden kann:

```
[gcode_macro PANELDUE_BEEP]
gcode:
  {action_call_remote_method("paneldue_beep", frequency=300, duration=1.0)}
```

Wenn das gcode-Makro PANELDUE_BEEP ausgeführt wird, würde Klipper etwas wie das Folgende über den Socket senden: `{"action": "run_paneldue_beep", "params": {"frequency": 300, "Dauer": 1.0}}`

objects/list

Dieser Endpunkt fragt die Liste der verfügbaren Drucker-"Objekte" ab, die man (über den Endpunkt "objects/query") abfragen kann. Zum Beispiel: `{"id": 123, "method": "objects/list"}` könnte zurückgeben: `{"id": 123, "Ergebnis": {"objects": ["webhooks", "configfile", "heaters", "gcode_move", "query_endstops", "idle_timeout", "toolhead", "extruder"]}}`

objects/query

Dieser Endpunkt ermöglicht die Abfrage von Informationen aus Druckerobjekten. Zum Beispiel: `{"id": 123, "method": "objects/query", "params": {"objects": {"toolhead": ["position"], "webhooks": null}}`, könnte zurückkehren: `{"id": 123, "result": {"status": {"webhooks": {"state": "ready", "state_message": "Drucker ist bereit"}, "toolhead": {"position": [0.0, 0.0, 0.0, 0.0]}}, "eventtime": 3051555.377933684}}`

Der Parameter "objects" in der Anfrage muss ein Wörterbuch sein, das die abzufragenden Druckerobjekte enthält - der Schlüssel enthält den Namen des Druckerobjekts und der Wert ist entweder "null" (zur Abfrage aller Felder) oder eine Liste von Feldnamen.

Die Antwortnachricht enthält ein "Status"-Feld, das ein Wörterbuch mit den abgefragten Informationen enthält - der Schlüssel enthält den Namen des Druckerobjekts und der Wert ist ein Wörterbuch mit dessen Feldern. Die Antwortnachricht enthält außerdem ein Feld "eventtime", das den Zeitstempel des Abfragezeitpunkts enthält.

Die verfügbaren Felder sind im Dokument Statusreferenz dokumentiert.

objects/subscribe

Dieser Endpunkt ermöglicht es, Informationen von Druckerobjekten abzufragen und dann zu abonnieren. Die Anfrage und die Antwort des Endpunkts sind identisch mit dem Endpunkt "objects/query". Zum Beispiel: `{"id": 123, "method": "objects/subscribe", "params": {"objects":{"toolhead": ["position"], "webhooks": ["state"]}, "response_template":{}}}` könnte zurückkehren: `{"id": 123, "result": {"status": {"webhooks": {"state": "ready"}, "toolhead": {"position": [0.0, 0.0, 0.0, 0.0]}}, "eventtime": 3052153.382083195}}` und führen zu folgenden asynchronen Meldungen wie: `{"params": {"status": {"webhooks": {"state": "shutdown"}}, "eventtime": 3052165.418815847}}`

gcode/help

Dieser Endpunkt ermöglicht die Abfrage verfügbarer G-Code-Befehle, für die eine Hilfezeichenfolge definiert ist. Zum Beispiel: `{"id": 123, "method": "gcode/help"}` könnte zurückgeben: `{"id": 123, "result": {"RESTORE_GCODE_STATE": "Restore a previously saved G-Code state", "PID_CALIBRATE": "Run PID calibration test", "QUERY_ADC": "Report the last value of an analog pin", ...}}`

gcode/script

Mit diesem Endpunkt können Sie eine Reihe von G-Code-Befehlen ausführen. Zum Beispiel: `{"id": 123, "method": "gcode/script", "params": {"script": "G90"}}`

Wenn das bereitgestellte G-Code-Skript einen Fehler auslöst, wird eine Fehlerantwort erzeugt. Wenn der G-Code-Befehl jedoch eine Terminalausgabe erzeugt, wird diese Terminalausgabe nicht in der Antwort angegeben. (Verwenden Sie den Endpunkt "gcode/subscribe_output", um eine G-Code-Terminalausgabe zu erhalten.)

Wenn zum Zeitpunkt des Empfangs dieser Anfrage bereits ein G-Code-Befehl verarbeitet wird, wird das bereitgestellte Skript in die Warteschlange gestellt. Diese Verzögerung kann erheblich sein (z. B. wenn ein G-Code-Befehl "Warten auf Temperatur" läuft). Die JSON-Antwortnachricht wird gesendet, wenn die Verarbeitung des Skripts vollständig abgeschlossen ist.

gcode/restart

Dieser Endpunkt ermöglicht es, einen Neustart anzufordern - er ist vergleichbar mit der Ausführung des G-Code-Befehls "RESTART". Zum Beispiel: {"id": 123, "method": "gcode/restart"}

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst nach Abschluss aller anstehenden G-Code-Befehle abgeschlossen.

gcode/firmware_restart

Dies ist ähnlich wie der Endpunkt "gcode/restart" - er implementiert den G-Code-Befehl "FIRMWARE_RESTART". Zum Beispiel: `{"id": 123, "Methode": "gcode/firmware_restart"}`

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst nach Abschluss aller anstehenden G-Code-Befehle abgeschlossen.

gcode/subscribe_output

Dieser Endpunkt wird verwendet, um G-Code Terminalnachrichten zu abonnieren, die von Klipper generiert werden. Zum Beispiel: `{"id": 123, "method": "gcode/subscribe_output", "params": {"response_template":{}}}` könnte später asynchrone Nachrichten wie die folgenden erzeugen: `{"params": {"response": "// Klipper state: Shutdown"}}`

Dieser Endpunkt soll die menschliche Interaktion über eine "Terminalfenster"-Schnittstelle unterstützen. Vom Parsen von Inhalten aus der G-Code-Terminalausgabe wird abgeraten. Verwenden Sie den Endpunkt "objects/subscribe", um Aktualisierungen des Klipper-Status zu erhalten.

motion_report/dump_stepper

Dieser Endpunkt wird verwendet, um Klippers internen Stepper queue_step Befehlsstrom für einen Stepper zu abonnieren. Das Abrufen dieser Low-Level-Bewegungsaktualisierungen kann für Diagnose- und Debugging-Zwecke nützlich sein. Die Verwendung dieses Endpunkts kann die Systemlast von Klipper erhöhen.

Eine Anfrage kann wie folgt aussehen: `{"id": 123, "method": "motion_report/dump_stepper", "params": {"name": "stepper_x", "response_template": {}}}` und könnte zurückgeben: `{"id": 123, "result": {"header": ["interval", "count", "add"]}}` und könnte später asynchrone Nachrichten wie die folgenden erzeugen: `{"params": {"first_clock": 179601081, "first_time": 8.98, "first_position": 0, "last_clock": 219686097, "last_time": 10.984, "data": [[179601081, 1, 0], [29573, 2, -8685], [16230, 4, -1525], [10559, 6, -160], [10000, 976, 0], [10000, 1000, 0], [10000, 1000, 0], [10000, 1000, 0], [9855, 5, 187], [11632, 4, 1534], [20756, 2, 9442]]}}`

Das Feld "header" in der anfänglichen Abfrageantwort wird zur Beschreibung der Felder in den späteren "Daten"-Antworten verwendet.

motion_report/dump_trapq

Dieser Endpunkt wird verwendet, um Klippers interne "trapezförmige Bewegungswarteschlange" zu abonnieren. Das Abrufen dieser Low-Level-Bewegungsupdates kann für Diagnose- und Fehlerbehebungszwecke nützlich sein. Die Verwendung dieses Endpunktes kann die Systemlast von Klipper erhöhen.

Eine Anfrage kann wie folgt aussehen: `{"id": 123, "method": "motion_report/dump_trapq", "params": {"name": "toolhead", "response_template":{}}} und könnte zurückgeben: {"id": 1, "result": {"header": ["time", "duration", "start_velocity", "acceleration", "start_position", "direction"]}}` und könnte später asynchrone Nachrichten wie die folgenden erzeugen: `{"params": {"data": [[4.05, 1.0, 0.0, 0.0, [300.0, 0.0, 0.0], [0.0, 0.0, 0.0]], [5.054, 0.001, 0.0, 3000.0, [300.0, 0.0, 0.0], [-1.0, 0.0, 0.0]]]}}`

Das Feld "header" in der anfänglichen Abfrageantwort wird zur Beschreibung der Felder in den späteren "Daten"-Antworten verwendet.

adxl345/dump_adxl345

Dieser Endpunkt wird verwendet, um ADXL345-Beschleunigungsmesserdaten zu abonnieren. Der Erhalt dieser Low-Level-Bewegungsaktualisierungen kann für Diagnose- und Fehlerbehebungszwecke nützlich sein. Die Verwendung dieses Endpunktes kann die Systemlast von Klipper erhöhen.

Eine Anfrage kann wie folgt aussehen: `{"id": 123, "method": "adxl345/dump_adxl345", "params": {"sensor": "adxl345", "response_template": {}}}` und könnte zurückgeben: `{"id": 123, "result":{"header":["time", "x_acceleration", "y_acceleration", "z_acceleration"]}}` und könnte später asynchrone Nachrichten wie diese erzeugen: `{"params":{"overflows":0,"data":[[3292.432935,-535.44309,-1529.8374,9561.4], [3292.433256,-382.45935,-1606.32927,9561.48375]]}}`

Das "Header"-Feld in der anfänglichen Abfrageantwort wird zur Beschreibung der Felder in späteren "Daten"-Antworten verwendet.

angle/dump_angle

Dieser Endpunkt wird verwendet, um Winkelsensordaten angle sensor data zu abonnieren. Der Erhalt dieser Low-Level-Bewegungsaktualisierungen kann für Diagnose- und Fehlerbehebungszwecke nützlich sein. Die Verwendung dieses Endpunktes kann die Systemlast von Klipper erhöhen.

Eine Anfrage kann wie folgt aussehen: `{"id": 123, "method": "angle/dump_angle", "params": {"sensor": "my_angel_sensor", "response_template": {}}}` und könnte zurückgeben: `{"id": 123,`

`"result":{"header":["time", "angle"]}}` und könnte später asynchrone Nachrichten wie die folgenden erzeugen: `{"params":{"position_offset":3.151562,"errors":0, "data":[[1290.951905,-5063],[1290.952321,-5065]]}}`

Das Feld "header" in der anfänglichen Abfrageantwort wird verwendet, um die Felder zu beschreiben, die in späteren "data"-Antworten zu finden sind.

pause_resume/cancel

Dieser Endpunkt ist vergleichbar mit der Ausführung des G-Code-Befehls "PRINT_CANCEL". Zum Beispiel: `{"id": 123, "method": "pause_resume/cancel"}`

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst nach Abschluss aller anhängigen G-Code-Befehle abgeschlossen.

pause_resume/pause

Dieser Endpunkt ist vergleichbar mit der Ausführung des G-Code-Befehls "PAUSE". Zum Beispiel: `{"id": 123, "method": "pause_resume/pause"}`

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst abgeschlossen, wenn alle anstehenden G-Code-Befehle beendet sind.

pause_resume/resume

Dieser Endpunkt ist vergleichbar mit der Ausführung des G-Code-Befehls "RESUME". Zum Beispiel: `{"id": 123, "method": "pause_resume/resume"}`

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst nach Abschluss aller anhängigen G-Code-Befehle abgeschlossen.

query_endstops/status

Dieser Endpunkt fragt die aktiven Endpunkte ab und gibt deren Status zurück. Zum Beispiel: `{"id": 123, "method": "query_endstops/status"}` könnte zurückgeben: `{"id": 123, "Ergebnis": {"y": "offen", "x": "open", "z": "TRIGGERED"}}`

Wie der Endpunkt "gcode/script" wird auch dieser Endpunkt erst nach Abschluss aller anstehenden G-Code-Befehle abgeschlossen.

MCU-Befehle

Dieses Dokument enthält Informationen zu den Low-Level-Mikrocontroller-Befehlen, die von der Klipper "Host"-Software gesendet und von der Klipper Mikrocontroller-Software verarbeitet werden. Dieses Dokument ist weder eine verbindliche Referenz für diese Befehle, noch ist es eine exklusive Liste aller verfügbaren Befehle.

Dieses Dokument kann für Entwickler nützlich sein, die daran interessiert sind, die Low-Level-Mikrocontroller-Befehle zu verstehen.

Weitere Informationen über das Format der Befehle und ihre Übertragung finden Sie im Protokolldokument protocol. Die Befehle werden hier in ihrer "printf"-Syntax beschrieben - für diejenigen, die mit diesem Format nicht vertraut sind, sei darauf hingewiesen, dass eine "%..."-Sequenz durch eine Ganzzahl ersetzt werden sollte. Zum Beispiel könnte eine Beschreibung mit "count=%c" durch den Text "count=10" ersetzt werden. Beachten Sie, dass Parameter, die als "Aufzählungen" gelten (siehe obiges Protokolldokument), einen String-Wert annehmen, der für den Mikrocontroller automatisch in einen Integer-Wert umgewandelt wird. Dies ist bei Parametern mit dem Namen "pin" (oder mit dem Suffix "_pin") üblich.

Startup-Befehle

Es kann notwendig sein, bestimmte einmalige Aktionen durchzuführen, um den Mikrocontroller und seine Peripheriegeräte zu konfigurieren. In diesem Abschnitt werden die zu diesem Zweck verfügbaren Befehle aufgeführt. Im Gegensatz zu den meisten Mikrocontroller-Befehlen werden diese Befehle sofort ausgeführt, sobald sie empfangen werden, und sie erfordern keine besondere Einrichtung.

Allgemeine Startbefehle:

- `set_digital_out pin=%u value=%c :` Dieser Befehl konfiguriert den angegebenen Pin sofort als digitalen GPIO-Ausgang und setzt ihn entweder auf einen niedrigen Pegel (Wert=0) oder einen hohen Pegel (Wert=1). Dieser Befehl kann für die Konfiguration des Anfangswertes von LEDs und für die Konfiguration des Anfangswertes von Stepper-Treiber-Micro-Stepping-Pins nützlich sein.

- `set_pwm_out pin=%u cycle_ticks=%u value=%hu :` Mit diesem Befehl wird der angegebene Pin sofort so konfiguriert, dass er eine hardwarebasierte Pulsweitenmodulation (PWM) mit der angegebenen Anzahl von cycle_ticks verwendet. Die "cycle_ticks" ist die Anzahl der MCU-Taktticks, die jeder Ein- und Ausschaltzyklus dauern soll. Ein cycle_ticks-Wert von 1 kann verwendet werden, um die schnellstmögliche Zykluszeit anzufordern. Der Parameter "value" liegt zwischen 0 und 255, wobei 0 einen vollständig ausgeschalteten Zustand und 255 einen vollständig eingeschalteten Zustand angibt. Dieser Befehl kann nützlich sein, um CPU- und Düsenlüfter zu aktivieren.

Mikrocontroller-Konfiguration auf niedriger Ebene

Die meisten Befehle im Mikrocontroller erfordern eine Ersteinrichtung, bevor sie erfolgreich aufgerufen werden können. Dieser Abschnitt gibt einen Überblick über den Konfigurationsprozess. Dieser und die folgenden Abschnitte sind wahrscheinlich nur für Entwickler von Interesse, die an den internen Details von Klipper interessiert sind.

Wenn der Host zum ersten Mal eine Verbindung zum Mikrocontroller herstellt, beginnt er immer mit der Beschaffung eines Datenverzeichnisses (siehe protocol für weitere Informationen). Nachdem das Datenwörterbuch erhalten wurde, prüft der Host, ob der Mikrocontroller in einem "konfigurierten" Zustand ist und konfiguriert ihn, falls nicht. Die Konfiguration umfasst die folgenden Phasen:

- `get_config :` Der Host prüft zunächst, ob der Mikrocontroller bereits konfiguriert ist. Der Mikrocontroller antwortet auf diesen Befehl mit einer "config"-Antwortnachricht. Die Software des Mikrocontrollers startet beim Einschalten immer in einem unkonfigurierten Zustand. Sie verbleibt in diesem Zustand, bis der Host die Konfigurationsprozesse abschließt (indem er einen finalize_config-Befehl erteilt). Wenn der Mikrocontroller bereits in einer früheren Sitzung konfiguriert wurde (und mit den gewünschten Einstellungen konfiguriert ist), ist keine weitere Aktion des Hosts erforderlich und der Konfigurationsprozess wird erfolgreich beendet.

- `allocate_oids count=%c :` Dieser Befehl wird erteilt, um dem Mikrocontroller die maximale Anzahl von Objekt-IDs (oid) mitzuteilen, die der Host benötigt. Dieser Befehl darf nur einmal erteilt werden. Eine oid ist eine ganzzahlige Kennung, die jedem Stepper, jedem Endanschlag und jedem planbaren gpio-Pin zugewiesen wird. Der Host bestimmt im Voraus die Anzahl der Oids, die er für den Betrieb der Hardware benötigt, und gibt diese an den Mikrocontroller weiter, damit dieser genügend Speicher zuweisen kann, um ein Mapping von Oid zu internem Objekt zu speichern.

- `config_XXX oid=%c ... :` Konventionell erstellt jeder Befehl, der mit dem Präfix "config_" beginnt, ein neues Mikrocontroller-Objekt und weist ihm die angegebene oid zu. Zum Beispiel konfiguriert der Befehl config_digital_out den angegebenen Pin als digitalen GPIO-Ausgang und erstellt ein internes Objekt, das der Host verwenden kann, um Änderungen an dem angegebenen GPIO zu planen. Der oid-Parameter, der an den config-Befehl übergeben wird, wird vom Host ausgewählt und muss zwischen null und der maximalen Anzahl liegen, die im Befehl allocate_oids angegeben wurde. Die config-Befehle dürfen nur ausgeführt werden, wenn sich der Mikrocontroller nicht in einem konfigurierten Zustand befindet (d. h. bevor der Host finalize_config sendet) und nachdem der Befehl allocate_oids gesendet wurde.

- `finalize_config crc=%u :` Mit dem Befehl finalize_config wird der Mikrocontroller von einem unkonfigurierten Zustand in einen konfigurierten Zustand überführt. Der an den Mikrocontroller übergebene crc-Parameter wird gespeichert und in "config"-Antwortnachrichten an den Host zurückgegeben. Vereinbarungsgemäß nimmt der Host eine 32-Bit-CRC der von ihm angeforderten Konfiguration und prüft zu Beginn nachfolgender Kommunikationssitzungen, ob die im Mikrocontroller gespeicherte CRC genau mit seiner gewünschten CRC übereinstimmt. Wenn die CRC nicht übereinstimmt, weiß der Host, dass der Mikrocontroller nicht in dem vom Host gewünschten Zustand konfiguriert wurde.

Allgemeine Mikrocontroller-Objekte

In diesem Abschnitt werden einige häufig verwendete Konfigurationsbefehle aufgeführt.

- `config_digital_out oid=%c pin=%u value=%c default_value=%c max_duration=%u :` Dieser Befehl erstellt ein internes Mikrocontroller-Objekt für den angegebenen GPIO 'pin'. Der Pin wird im digitalen Ausgabemodus konfiguriert und auf einen Anfangswert gesetzt, der durch 'value' festgelegt wird (0 für low, 1 für high). Die Erstellung eines digital_out-Objekts ermöglicht es dem Host, GPIO-Aktualisierungen für den angegebenen Pin zu bestimmten Zeiten zu planen (siehe den unten beschriebenen Befehl queue_digital_out). Sollte die Mikrocontroller-Software in den Shutdown-Modus gehen, werden alle konfigurierten digital_out-Objekte auf 'default_value' gesetzt. Der Parameter "max_duration" dient zur Durchführung einer Sicherheitsprüfung - ist er ungleich Null, so ist dies die maximale Anzahl von Ticks, die der Host den gegebenen GPIO ohne weitere Aktualisierungen auf einen anderen Wert als den Standardwert setzen darf. Wenn z.B. der default_value gleich Null und die max_duration gleich 16000 ist, dann muss der Host, wenn er den GPIO auf einen Wert von Eins setzt, innerhalb von 16000 Ticks eine weitere Aktualisierung des GPIO-Pins (entweder auf Null oder Eins) einplanen. Diese Sicherheitsfunktion kann bei Heizungspins verwendet werden, um sicherzustellen, dass der Host die Heizung nicht aktiviert und dann offline geht.

- `config_pwm_out oid=%c pin=%u cycle_ticks=%u value=%hu default_value=%hu max_duration=%u :` Dieser Befehl erzeugt ein internes Objekt für hardwarebasierte PWM-Pins, für die der Host Updates planen kann. Seine Verwendung ist analog zu config_digital_out - siehe die Beschreibung der Befehle 'set_pwm_out' und 'config_digital_out' für die Parameterbeschreibung.

- `config_analog_in oid=%c pin=%u :` Dieser Befehl wird verwendet, um einen Pin im analogen Eingangs-Sampling-Modus zu konfigurieren. Einmal konfiguriert, kann der Pin mit dem Befehl query_analog_in (siehe unten) in regelmäßigen Abständen abgetastet werden.

- `config_stepper oid=%c step_pin=%c dir_pin=%c invert_step=%c step_pulse_ticks=%u :` Dieser Befehl erzeugt ein internes Stepper-Objekt. Die Parameter 'step_pin' und 'dir_pin' geben die Schritt- bzw. Richtungspins an; dieser Befehl konfiguriert sie im digitalen Ausgangsmodus. Der Parameter 'invert_step' gibt an, ob ein Schritt bei einer steigenden Flanke (invert_step=0) oder einer fallenden Flanke (invert_step=1) erfolgt. Der Parameter "step_pulse_ticks" gibt die Mindestdauer des Schrittimpulses an. Wenn die CPU die Konstante 'STEPPER_BOTH_EDGE=1' exportiert, dann wird durch die Einstellung von step_pulse_ticks=0 und invert_step=-1 das Steppen sowohl bei der steigenden als auch bei der fallenden Flanke des Step-Pins eingestellt.

- `config_endstop oid=%c pin=%c pull_up=%c stepper_count=%c :` Dieser Befehl erzeugt ein internes "endstop" Objekt. Er wird verwendet, um die Endstopp-Pins zu spezifizieren und um "Homing"-Operationen zu ermöglichen (siehe den Befehl endstop_home weiter unten). Der Befehl konfiguriert den angegebenen Pin im digitalen Eingangsmodus. Der Parameter 'pull_up' bestimmt, ob die von der Hardware

bereitgestellten Pullup-Widerstände für den Pin (falls vorhanden) aktiviert werden. Der Parameter 'stepper_count' gibt die maximale Anzahl von Steppern an, die dieser Endstop während einer Referenzfahrt anhalten muss (siehe endstop_home unten).

- `config_spi oid=%c bus=%u pin=%u mode=%u rate=%u shutdown_msg=%*s` : Dieser Befehl erstellt ein internes SPI-Objekt. Er wird mit den Befehlen spi_transfer und spi_send verwendet (siehe unten). Der "bus" identifiziert den zu verwendenden SPI-Bus (wenn der Mikrocontroller mehr als einen SPI-Bus zur Verfügung hat). Der "pin" gibt den Chip-Select-Pin (CS) für das Gerät an. Der "mode" ist der SPI-Modus (sollte zwischen 0 und 3 liegen). Der Parameter "rate" gibt die SPI-Busrate an (in Zyklen pro Sekunde). Der Parameter "shutdown_msg" schließlich ist ein SPI-Befehl, der an das angegebene Gerät zu senden ist, wenn der Mikrocontroller in einen Shutdown-Zustand übergeht.

- `config_spi_without_cs oid=%c bus=%u mode=%u rate=%u shutdown_msg=%*s` : Dieser Befehl ist ähnlich wie config_spi, jedoch ohne CS-Pin-Definition. Er ist nützlich für SPI-Geräte, die keine Chip-Select-Leitung haben.

Allgemeine Befehle

In diesem Abschnitt werden einige häufig verwendete Laufzeitbefehle aufgeführt. Er ist wahrscheinlich nur für Entwickler von Interesse, die einen Einblick in Klipper gewinnen wollen.

- `set_digital_out_pwm_cycle oid=%c cycle_ticks=%u` : Dieser Befehl konfiguriert einen digitalen Ausgangspin (wie mit config_digital_out erstellt) für die Verwendung von "Software-PWM". Die 'cycle_ticks' ist die Anzahl der Taktticks für den PWM-Zyklus. Da das Schalten des Ausgangs in der Software des Mikrocontrollers implementiert wird, wird empfohlen, dass 'cycle_ticks' einer Zeit von 10ms oder mehr entspricht.

- `queue_digital_out oid=%c clock=%u on_ticks=%u` : Mit diesem Befehl wird ein Wechsel zu einem digitalen GPIO-Ausgangspin zur angegebenen Taktzeit geplant. Um diesen Befehl zu verwenden, muss ein 'config_digital_out'-Befehl mit dem gleichen 'oid'-Parameter während der Mikrocontroller-Konfiguration ausgegeben worden sein. Wenn 'set_digital_out_pwm_cycle' aufgerufen wurde, ist 'on_ticks' die Einschaltdauer (in Ticks) für den Pwm-Zyklus. Andernfalls sollte 'on_ticks' entweder 0 (für niedrige Spannung) oder 1 (für hohe Spannung) sein.

- `queue_pwm_out oid=%c clock=%u value=%hu` : Plant eine Änderung an einem Hardware-PWM-Ausgangspin. Siehe die Befehle 'queue_digital_out' und 'config_pwm_out' für weitere Informationen.

- `query_analog_in oid=%c clock=%u sample_ticks=%u sample_count=%c rest_ticks=%u min_value=%hu max_value=%hu` : Dieser Befehl richtet einen wiederkehrenden Zeitplan für analoge Eingangsabtastungen ein. Um diesen Befehl zu verwenden, muss ein 'config_analog_in'-Befehl mit demselben 'oid'-Parameter während der Mikrocontroller-Konfiguration erteilt worden sein. Die Abtastungen beginnen ab der Uhrzeit 'clock', der erhaltene Wert wird alle 'rest_ticks' Uhrenticks gemeldet, die Anzahl der Überabtastungen wird mit 'sample_count' angegeben, und zwischen den Überabtastungen wird eine Pause von 'sample_ticks' Anzahl von Uhrenticks eingelegt. Die Parameter 'min_value' und 'max_value' implementieren eine Sicherheitsfunktion - die Mikrocontroller-Software überprüft, ob der abgetastete Wert (nach einer Überabtastung) immer innerhalb des angegebenen Bereichs liegt. Dies ist für die Verwendung mit Pins gedacht, die an Thermistoren angeschlossen sind, die Heizungen steuern - es kann verwendet werden, um zu überprüfen, ob eine Heizung innerhalb eines Temperaturbereichs liegt.

- `get_clock` : Dieser Befehl veranlasst den Mikrocontroller, eine "Clock"-Antwortnachricht zu erzeugen. Der Host sendet diesen Befehl einmal pro Sekunde, um den Wert der Mikrocontroller-Uhr zu ermitteln und die Abweichung zwischen Host- und Mikrocontroller-Uhr abzuschätzen. Dadurch kann der Host die Uhr des Mikrocontrollers genau einschätzen.

Stepper-Befehle

- `queue_step oid=%c interval=%u count=%hu add=%hi` : Dieser Befehl plant 'count' Anzahl der Schritte für den angegebenen Stepper, mit 'interval' Anzahl der Ticks zwischen jedem Schritt. Der erste Schritt ist 'interval' die Anzahl der Ticks seit dem letzten geplanten Schritt für den angegebenen Stepper. Wenn 'add' ungleich Null ist, wird das Intervall nach jedem Schritt um den Betrag von 'add' angepasst. Dieser Befehl fügt die angegebene Intervall/Count/Add-Sequenz an eine Warteschlange pro Stepper an. Im Normalbetrieb können sich Hunderte dieser Sequenzen in der Warteschlange befinden. Neue Sequenzen werden an das Ende der Warteschlange angehängt, und sobald jede Sequenz ihre "Anzahl" an Schritten vollendet hat, wird sie aus der Warteschlange entfernt. Mit diesem System kann der Mikrocontroller potenziell Hunderttausende von Schritten in eine Warteschlange stellen - und das mit zuverlässigen und vorhersehbaren Zeitplänen.

- `set_next_step_dir oid=%c dir=%c` : Dieser Befehl legt den Wert des dir_pin fest, den der nächste queue_step-Befehl verwenden wird.

- `reset_step_clock oid=%c clock=%u` : Normalerweise ist das Step-Timing relativ zum letzten Step für einen bestimmten Stepper. Dieser Befehl setzt die Uhr zurück, so dass der nächste Schritt relativ zur angegebenen 'clock'-Zeit erfolgt. Der Host sendet diesen Befehl normalerweise nur zu Beginn eines Druckvorgangs.

- `stepper_get_position oid=%c` : Dieser Befehl veranlasst den Mikrocontroller, eine Antwortnachricht "stepper_position" mit der aktuellen Position des Steppers zu erzeugen. Die Position ist die Gesamtzahl der mit dir=1 erzeugten Schritte abzüglich der Gesamtzahl der mit dir=0 erzeugten Schritte.

- `endstop_home oid=%c clock=%u sample_ticks=%u sample_count=%c rest_ticks=%u pin_value=%c` : Dieser Befehl wird während der "Homing"-Operationen des Steppers verwendet. Um diesen Befehl zu verwenden, muss ein 'config_endstop'-Befehl mit

demselben 'oid'-Parameter während der Mikrocontroller-Konfiguration erteilt worden sein. Wenn dieser Befehl aufgerufen wird, tastet der Mikrocontroller den Endstop-Pin alle 'rest_ticks'-Takte ab und prüft, ob er einen Wert gleich 'pin_value' hat. Stimmt der Wert überein (und bleibt er für 'sample_count' weitere Abtastungen im Abstand von 'sample_ticks' übereinstimmend), wird die Bewegungswarteschlange für den zugehörigen Stepper gelöscht und der Stepper kommt sofort zum Stillstand. Der Host verwendet diesen Befehl, um die Referenzfahrt zu implementieren - der Host weist den Endstop an, auf den Endstop-Trigger zu warten, und gibt dann eine Reihe von queue_step-Befehlen aus, um einen Stepper zum Endstop zu bewegen. Sobald der Stepper den Endstopp erreicht, wird der Trigger erkannt, die Bewegung gestoppt und der Host benachrichtigt.

Warteschlange bewegen

Jeder queue_step-Befehl verwendet einen Eintrag in der "move queue" des Mikrocontrollers. Diese Warteschlange wird zugewiesen, wenn der Befehl "finalize_config" empfangen wird, und die Anzahl der verfügbaren Warteschlangeneinträge wird in den "config"-Antwortmeldungen gemeldet.

Es liegt in der Verantwortung des Hosts, sicherzustellen, dass in der Warteschlange Platz ist, bevor er einen queue_step-Befehl sendet. Dazu berechnet der Host, wann jeder queue_step-Befehl abgeschlossen ist, und plant neue queue_step-Befehle entsprechend ein.

SPI-Befehle

- `spi_transfer oid=%c data=%*s` : Dieser Befehl veranlasst den Mikrocontroller, "Daten" an das durch "oid" angegebene SPI-Gerät zu senden, und erzeugt eine "spi_transfer_response"-Antwortnachricht mit den während der Übertragung zurückgegebenen Daten.

- `spi_send oid=%c data=%*s` : Dieser Befehl ist ähnlich wie "spi_transfer", erzeugt aber keine "spi_transfer_response"-Nachricht.

CANBUS-Protokoll

Dieses Dokument beschreibt das Protokoll, das Klipper zur Kommunikation über den CAN bus verwendet. Siehe CANBUS.md für Informationen zur Konfiguration von Klipper mit CAN-Bus.

Mikrocontroller-ID-Zuweisung
Klipper verwendet nur CAN 2.0A Standard-CAN-Bus-Pakete, die auf 8 Datenbytes und einen 11-Bit-CAN-Bus-Identifier beschränkt sind. Um eine effiziente Kommunikation zu unterstützen, wird jedem Mikrocontroller während der Laufzeit eine eindeutige 1-Byte-CAN-Bus-Nodeid (`canbus_nodeid`) für den allgemeinen Klipper-Befehls- und Antwortverkehr zugewiesen. Klipper-Befehlsnachrichten, die vom Host zum Mikrocontroller gehen, verwenden die CAN-Bus-ID `canbus_nodeid * 2 + 256`, während Klipper-Antwortnachrichten vom Mikrocontroller zum Host `canbus_nodeid * 2 + 256 + 1` verwenden.

Jeder Mikrocontroller hat einen werkseitig zugewiesenen eindeutigen Chip-Identifikator, der bei der ID-Zuweisung verwendet wird. Dieser Bezeichner kann die Länge eines CAN-Pakets überschreiten, daher wird eine Hash-Funktion verwendet, um eine eindeutige 6-Byte-ID (`canbus_uuid`) aus der Werks-ID zu erzeugen.

Admin-Nachrichten

Admin-Nachrichten werden für die ID-Zuweisung verwendet. Admin-Nachrichten, die vom Host an den Mikrocontroller gesendet werden, verwenden die CAN-Bus-ID `0x3f0` und Nachrichten, die vom Mikrocontroller an den Host gesendet werden, verwenden die CAN-Bus-ID `0x3f1`. Alle Mikrocontroller hören auf Nachrichten mit der id `0x3f0`; diese id kann als "Broadcast-Adresse" betrachtet werden.

CMD_QUERY_UNASSIGNED message

Dieser Befehl fragt alle Mikrocontroller ab, denen noch keine `canbus_nodeid` zugewiesen wurde. Nicht zugewiesene Mikrocontroller antworten mit einer RESP_NEED_NODEID-Antwortnachricht.

Das CMD_QUERY_UNASSIGNED-Meldungsformat ist: `<1 Byte message_id = 0x00>`

CMD_SET_NODEID message

Dieser Befehl weist dem Mikrocontroller eine canbus_nodeid mit einer bestimmten `canbus_uuid` zu.

Das CMD_SET_NODEID-Meldungsformat ist: `<1 Byte message_id = 0x01><6 Byte canbus_uuid><1 Byte canbus_nodeid>`

RESP_NEED_NODEID-message

Das Format der RESP_NEED_NODEID-Nachricht ist: <`1-Byte message_id = 0x20><6-Byte canbus_uuid>`

Datenpakete

Ein Mikrocontroller, dem über den Befehl CMD_SET_NODEID eine Nodeid zugewiesen wurde, kann Datenpakete senden und empfangen.

Die Paketdaten in Nachrichten mit der Empfangs-CAN-Bus-ID des Knotens (`canbus_nodeid * 2 + 256`) werden einfach an einen Puffer angehängt, und wenn eine vollständige mcu-Protokollnachricht mcu protocol message gefunden wird, wird ihr Inhalt geparst und verarbeitet. Die Daten werden wie ein Bytestrom behandelt - es gibt keine Anforderung, dass der Beginn eines Klipper-Nachrichtenblocks mit dem Beginn eines CAN-Bus-Pakets übereinstimmt.

In ähnlicher Weise werden Antworten auf mcu-Protokollnachrichten vom Mikrocontroller an den Host gesendet, indem die Nachrichtendaten in ein oder mehrere Pakete mit der Sende-CAN-Bus-ID des Knotens (`canbus_nodeid * 2 + 256 + 1`) kopiert werden.

Fehlersuche

Dieses Dokument beschreibt einige der Klipper-Debugging-Werkzeuge.

Ausführen der Regressionstests

Das Klipper GitHub Repository verwendet "github actions", um eine Reihe von Regressionstests auszuführen. Es kann nützlich sein, einige dieser Tests lokal auszuführen.

Der Quellcode "whitespace check" kann mit ausgeführt werden:

```
./scripts/check_whitespace.sh
```

Die Klippy Regressionstest-Suite benötigt "Datenwörterbücher" von vielen Plattformen. Der einfachste Weg, sie zu erhalten, ist, sie von github herunterzuladen download them from github. Sobald die Datenwörterbücher heruntergeladen sind, führen Sie die Regressionstestsuite wie folgt aus:

```
tar xfz klipper-dict-20??????.tar.gz
~/klippy-env/bin/python ~/klipper/scripts/test_klippy.py -d dict/ ~/klipper/test/klippy/*.test
```

Manuelles Senden von Befehlen an den Mikrocontroller

Normalerweise würde der Host klippy.py Prozess verwendet werden, um gcode Befehle in Klipper Mikrocontroller Befehle zu übersetzen. Es ist jedoch auch möglich, diese MCU-Befehle manuell zu senden (Funktionen, die im Klipper-Quellcode mit dem Makro DECL_COMMAND() gekennzeichnet sind). Um dies zu tun, führe aus:

```
~/klippy-env/bin/python ./klippy/console.py /tmp/pseudoserial
```

Weitere Informationen zu den Funktionen des Tools finden Sie unter dem Befehl "HELP" im Tool.

Es sind einige Befehlszeilenoptionen verfügbar. Für weitere Informationen führen Sie aus: `~/klippy-env/bin/python ./klippy/console.py -help`

Übersetzen von gcode-Dateien in Mikrocontroller-Befehle

Der Klippy-Hostcode kann in einem Batch-Modus ausgeführt werden, um die mit einer gcode-Datei verbundenen Low-Level-Mikrocontroller-Befehle zu erzeugen. Die Untersuchung dieser Low-Level-Befehle ist nützlich, wenn man versucht, die Aktionen der Low-Level-Hardware zu verstehen. Es kann auch nützlich sein, die Unterschiede in den Mikrocontroller-Befehlen nach einer Codeänderung zu vergleichen.

Um Klippy in diesem Batch-Modus laufen zu lassen, ist ein einmaliger Schritt notwendig, um das Mikrocontroller-"Datenwörterbuch" zu erzeugen. Dies geschieht durch Kompilieren des Mikrocontroller-Codes, um die Datei ***out/klipper.dict*** zu erhalten:

```
make menuconfig
make
```

Danach kann Klipper im Batch-Modus gestartet werden (siehe installation für die notwendigen Schritte zur Erstellung der virtuellen Python-Umgebung und einer printer.cfg-Datei):

```
~/klippy-env/bin/python ./klippy/klippy.py ~/printer.cfg -i test.gcode -o test.serial -v -d
out/klipper.dict
```

Der obige Befehl erzeugt eine Datei test.serial mit der binären seriellen Ausgabe. Diese Ausgabe kann in lesbaren Text übersetzt werden mit:

```
~/klippy-env/bin/python ./klippy/parsedump.py out/klipper.dict test.serial > test.txt
```

Die resultierende Datei test.txt enthält eine menschenlesbare Liste von Mikrocontroller-Befehlen.

Der Batch-Modus deaktiviert bestimmte Antwort-/Anfragebefehle, um zu funktionieren. Infolgedessen gibt es einige Unterschiede zwischen den tatsächlichen Befehlen und der obigen Ausgabe. Die erzeugten Daten sind für Tests und Inspektionen nützlich; sie sind nicht geeignet, um sie an einen echten Mikrocontroller zu senden.

Bewegungsanalyse und Datenprotokollierung

Klipper unterstützt die Aufzeichnung seiner internen Bewegungshistorie, die später analysiert werden kann. Um diese Funktion zu nutzen, muss Klipper mit aktiviertem API Server gestartet werden.

Die Datenprotokollierung wird mit dem Werkzeug data_logger.py aktiviert. Zum Beispiel:

```
~/klipper/scripts/motan/data_logger.py /tmp/klippy_uds mylog
```

Dieser Befehl stellt eine Verbindung zum Klipper API Server her, abonniert Status- und Bewegungsinformationen und protokolliert die Ergebnisse. Es werden zwei Dateien erzeugt - eine komprimierte Datendatei und eine Indexdatei (z.B. `mylog.json.gz und mylog.index.gz`). Nach dem Start der Protokollierung können Sie Ausdrucke und andere Aktionen durchführen - die Protokollierung wird im Hintergrund fortgesetzt. Wenn Sie mit der Protokollierung fertig sind, drücken Sie `ctrl-c`, um das Werkzeug `data_logger.py` zu beenden.

Die resultierenden Dateien können mit dem Tool `motan_graph.py` gelesen und grafisch dargestellt werden. Um Graphen auf einem Raspberry Pi zu erzeugen, muss einmalig das Paket "matplotlib" installiert werden:

```
sudo apt-get update
sudo apt-get install python-matplotlib
```

Es kann jedoch bequemer sein, die Datendateien zusammen mit dem Python-Code in das Verzeichnis `scripts/motan/` auf einen Desktop-Rechner zu kopieren. Die Skripte zur Bewegungsanalyse sollten auf jedem Rechner laufen, auf dem eine aktuelle Version von Python und Matplotlib installiert ist.

Diagramme können mit einem Befehl wie dem folgenden erzeugt werden:

```
~/klipper/scripts/motan/motan_graph.py mylog -o mygraph.png
```

Man kann die Option `-g` verwenden, um die Datensätze anzugeben, die grafisch dargestellt werden sollen (sie nimmt ein Python-Literal, das eine Liste von Listen enthält). Zum Beispiel:

```
~/klipper/scripts/motan/motan_graph.py mylog -g '[["trapq(toolhead,velocity)"],
["trapq(toolhead,accel)"]]'
```

Die Liste der verfügbaren Datensätze kann mit der Option `-l` gefunden werden - zum Beispiel:

```
~/klipper/scripts/motan/motan_graph.py -l
```

Es ist auch möglich, matplotlib Plot-Optionen für jeden Datensatz anzugeben:

```
~/klipper/scripts/motan/motan_graph.py mylog -g '[["trapq(toolhead,velocity)?color=red&alpha=0.4"]]'
```

Viele matplotlib Optionen sind verfügbar; einige Beispiele sind "color", "label", "alpha", und "linestyle".

Das `motan_graph.py` Tool unterstützt eine Reihe weiterer Kommandozeilenoptionen - verwenden Sie die `--help` Option, um eine Liste zu sehen. Es kann auch sinnvoll sein, das motan_graph.py Skript selbst anzusehen/zu verändern.

Die vom Tool `data_logger.py` erzeugten Rohdatenprotokolle folgen dem im API-Server beschriebenen Format. Es kann nützlich sein, die Daten mit einem Unix-Befehl wie dem folgenden zu prüfen: `gunzip < mylog.json.gz | tr '\03' '\n' | less`

Erzeugen von Ladediagrammen

Die Klippy-Protokolldatei (/tmp/klippy.log) speichert Statistiken über Bandbreite, Mikrocontroller-Last und Host-Pufferlast. Es kann nützlich sein, diese Statistiken nach einem Druck grafisch darzustellen.

Um ein Diagramm zu erstellen, muss einmalig das Paket "matplotlib" installiert werden:

```
sudo apt-get update
sudo apt-get install python-matplotlib
```

Dann können Graphen mit erzeugt werden:

```
~/klipper/scripts/graphstats.py /tmp/klippy.log -o loadgraph.png
```

Die resultierende Datei loadgraph.png kann dann betrachtet werden.

Es können verschiedene Graphen erzeugt werden. Für weitere Informationen führen Sie aus: `~/klipper/scripts/graphstats.py -help`

Extrahieren von Informationen aus der Datei klippy.log

Die Klippy-Protokolldatei (/tmp/klippy.log) enthält auch Debugging-Informationen. Es gibt ein logextract.py-Skript, das bei der Analyse eines Mikrocontroller-Shutdowns oder eines ähnlichen Problems nützlich sein kann. Es wird normalerweise mit einem Befehl wie:

```
mkdir work_directory
cd work_directory
cp /tmp/klippy.log .
~/klipper/scripts/logextract.py ./klippy.log
```

Das Skript extrahiert die Druckerkonfigurationsdatei und extrahiert Informationen über das Herunterfahren der MCU. Die Informationen eines MCU-Shutdowns (falls vorhanden) werden nach Zeitstempel geordnet, um die Diagnose von Ursache und Wirkung zu erleichtern.

Testen mit simulavr

Mit dem Tool simulavr kann ein Atmel ATmega-Mikrocontroller simuliert werden. In diesem Abschnitt wird beschrieben, wie man Test-Gcode-Dateien mit simulavr ausführen kann. Es wird empfohlen, das Tool auf einem Desktop-Rechner (nicht auf einem Raspberry Pi) auszuführen, da es eine hohe CPU-Leistung benötigt.

Um simulavr zu verwenden, laden Sie das simulavr-Paket herunter und kompilieren Sie mit Python-Unterstützung. Beachten Sie, dass das Build-System möglicherweise einige Pakete (wie z. B. swig) installiert haben muss, um das Python-Modul zu erstellen.

```
git clone git://git.savannah.nongnu.org/simulavr.git
cd simulavr
make python
make build
```

Stellen Sie sicher, dass eine Datei wie ***./build/pysimulavr/_pysimulavr.*.so*** nach der obigen Kompilierung vorhanden ist:

```
ls ./build/pysimulavr/_pysimulavr.*.so
```

Dieser Befehl sollte eine bestimmte Datei melden (z.B. ./build/pysimulavr/_pysimulavr.cpython-39-x86_64-linux-gnu.so) und nicht einen Fehler.

Wenn Sie auf einem Debian-basierten System (Debian, Ubuntu, etc.) arbeiten, können Sie die folgenden Pakete installieren und *.deb-Dateien für die systemweite Installation von simulavr erzeugen:

```
sudo apt update
sudo apt install g++ make cmake swig rst2pdf help2man texinfo
make cfgclean python debian
sudo dpkg -i build/debian/python3-simulavr*.deb
```

Um Klipper für die Verwendung in simulavr zu kompilieren, führen Sie aus:

```
cd /pfad/zu/klipper
make menuconfig
```

und kompiliere die Mikrocontroller-Software für einen AVR atmega644p und wähle SIMULAVR Software-Emulationsunterstützung. Dann kann man Klipper kompilieren (make ausführen) und dann die Simulation starten mit:

```
PYTHONPATH=/path/to/simulavr/build/pysimulavr/ ./scripts/avrsim.py out/klipper.elf
```

Beachten Sie, dass Sie, wenn Sie python3-simulavr systemweit installiert haben, den PYTHONPATH nicht setzen müssen, sondern den Simulator einfach als

```
./scripts/avrsim.py out/klipper.elf
```

Dann kann man, während simulavr in einem anderen Fenster läuft, folgendes ausführen, um gcode aus einer Datei (z.B. "test.gcode") zu lesen, ihn mit Klippy zu verarbeiten und ihn an Klipper zu senden, der in simulavr läuft (siehe Installation für die notwendigen Schritte, um die virtuelle Python-Umgebung aufzubauen):

```
~/klippy-env/bin/python ./klippy/klippy.py config/generic-simulavr.cfg -i test.gcode -v
```

Verwendung von simulavr mit gtkwave

Eine nützliche Funktion von simulavr ist die Möglichkeit, Dateien zur Erzeugung von Signalwellen mit dem genauen Timing von Ereignissen zu erstellen. Um dies zu tun, folgen Sie den obigen Anweisungen, aber führen Sie avrsim.py mit einer Befehlszeile wie der folgenden aus:

```
PYTHONPATH=/path/to/simulavr/src/python/ ./scripts/avrsim.py out/klipper.elf -t
PORTA.PORT,PORTC.PORT
```

Damit wird eine Datei avrsim.vcd mit Informationen über jede Änderung an den GPIOs von PORTA und PORTB erstellt. Diese Datei könnte dann mit gtkwave angezeigt werden mit:

```
gtkwave avrsim.vcd
```

Benchmarks

Dieses Dokument beschreibt Klipper-Benchmarks.

Mikrocontroller-Benchmarks

Dieser Abschnitt beschreibt den Mechanismus, mit dem die Klipper Mikrocontroller-Schrittraten-Benchmarks erzeugt werden.

Das primäre Ziel der Benchmarks ist es, einen konsistenten Mechanismus zur Messung der Auswirkungen von Kodierungsänderungen innerhalb der Software bereitzustellen. Ein sekundäres Ziel ist es, High-Level-Metriken für den Vergleich der Leistung zwischen Chips und zwischen Software-Plattformen bereitzustellen.

Der Schrittraten-Benchmark wurde entwickelt, um die maximale Schrittrate zu ermitteln, die die Hardware und Software erreichen können. Diese Benchmark-Schrittrate ist im täglichen Gebrauch nicht erreichbar, da Klipper im realen Einsatz noch andere Aufgaben erfüllen muss (z.B. Mikrocontroller/Host-Kommunikation, Temperaturmessung, Endstop-Prüfung).

Im Allgemeinen werden die Pins für die Benchmark-Tests so gewählt, dass LEDs oder andere harmlose Pins blinken. ***Vergewissern Sie sich immer, dass es sicher ist, die konfigurierten Pins anzusteuern, bevor Sie einen Benchmark durchführen.*** Es wird nicht empfohlen, einen tatsächlichen Stepper während eines Benchmarks anzusteuern.

Schrittfrequenz-Benchmark-Test

Der Test wird mit dem Tool console.py durchgeführt (beschrieben in Debugging.md). Der Mikrocontroller wird für die jeweilige Hardwareplattform konfiguriert (siehe unten) und dann wird der folgende Befehl per Cut-and-Paste in das Terminalfenster von console.py eingefügt:

```
SET start_clock {clock+freq}
SET ticks 1000

reset_step_clock oid=0 clock={start_clock}
set_next_step_dir oid=0 dir=0
queue_step oid=0 interval={ticks} count=60000 add=0
set_next_step_dir oid=0 dir=1
queue_step oid=0 interval=3000 count=1 add=0

reset_step_clock oid=1 clock={start_clock}
set_next_step_dir oid=1 dir=0
queue_step oid=1 interval={ticks} count=60000 add=0
set_next_step_dir oid=1 dir=1
queue_step oid=1 interval=3000 count=1 add=0

reset_step_clock oid=2 clock={start_clock}
set_next_step_dir oid=2 dir=0
queue_step oid=2 interval={ticks} count=60000 add=0
set_next_step_dir oid=2 dir=1
queue_step oid=2 interval=3000 count=1 add=0
```

Die obige Funktion testet drei Stepper, die gleichzeitig steppen. Wenn die Ausführung des obigen Beispiels zu einem Fehler "Rescheduled timer in the past" oder "Stepper too far in past" führt, dann bedeutet dies, dass der ticks-Parameter zu niedrig ist (er führt zu einer zu schnellen Stepprate). Ziel ist es, die niedrigste Einstellung des ticks-Parameters zu finden, die zuverlässig zu einem erfolgreichen Abschluss des Tests führt. Es sollte möglich sein, den ticks-Parameter zu halbieren, bis ein stabiler Wert gefunden ist.

Bei einem Fehlschlag kann man durch Kopieren und Einfügen Folgendes tun, um den Fehler in Vorbereitung auf den nächsten Test zu löschen:

clear_shutdown

Um die Single-Stepper-Benchmarks zu erhalten, wird dieselbe Konfigurationssequenz verwendet, aber nur der erste Block des obigen Tests wird per Copy-and-Paste in das Fenster console.py eingefügt.

Um die Benchmarks im Dokument Features zu erstellen, wird die Gesamtzahl der Schritte pro Sekunde berechnet, indem die Anzahl der aktiven Stepper mit der nominalen Mikroprozessorfrequenz multipliziert und durch den Parameter final ticks geteilt wird. Die Ergebnisse werden auf das nächste K gerundet. Ein Beispiel: drei aktive Stepper:

```
ECHO Testergebnis ist: {"%.0fK" % (3. * freq / ticks / 1000.)}
```

Die Benchmarks werden mit Parametern ausgeführt, die für TMC-Treiber geeignet sind. Für Mikrocontroller, die STEPPER_BOTH_EDGE=1 unterstützen (wie in der MCU-Konfigurationszeile beim ersten Start von console.py angegeben), verwenden Sie step_pulse_duration=0 und invert_step=-1, um optimiertes Stepping auf beiden Flanken des Schrittimpulses zu ermöglichen. Für andere Mikrocontroller verwenden Sie eine step_pulse_duration, die 100ns entspricht.

```
AVR-Schrittfrequenz-Benchmark
```

Die folgende Konfigurationssequenz wird auf AVR-Chips verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PA5 dir_pin=PA4 invert_step=0 step_pulse_ticks=32
config_stepper oid=1 step_pin=PA3 dir_pin=PA2 invert_step=0 step_pulse_ticks=32
config_stepper oid=2 step_pin=PC7 dir_pin=PC6 invert_step=0 step_pulse_ticks=32
finalize_config crc=0
```

Der Test wurde zuletzt am Commit 59314d99 mit der gcc-Version avr-gcc (GCC) 5.4.0 durchgeführt. Sowohl die 16Mhz- als auch die 20Mhz-Tests wurden mit simulavr durchgeführt, das für einen atmega644p konfiguriert ist (frühere Tests haben bestätigt, dass die simulavr-Ergebnisse mit denen eines 16Mhz at90usb und eines 16Mhz atmega2560 übereinstimmen).

avr	ticks
1 stepper	102
3 stepper	486

Arduino Due Schrittfrequenz Benchmark

Die folgende Konfigurationssequenz wird auf dem Due verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PB27 dir_pin=PA21 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PB26 dir_pin=PC30 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PA21 dir_pin=PC30 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt am Commit `59314d99` mit der gcc-Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.

sam3x8e	ticks
1 stepper	66
3 stepper	257

Duet Maestro Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem Duet Maestro verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PC26 dir_pin=PC18 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PC26 dir_pin=PA8 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PC26 dir_pin=PB4 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt am Commit `59314d99` mit der gcc-Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.

sam4s8c	ticks
1 stepper	71
3 stepper	260

Duet Wifi Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem Duet Wifi verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PD6 dir_pin=PD11 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PD7 dir_pin=PD12 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PD8 dir_pin=PD13 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt am Commit `59314d99` mit gcc Version `gcc Version 10.3.1 20210621 (release) (GNU Arm Embedded Toolchain 10.3-2021.07)` durchgeführt.

sam4e8e	ticks
1 stepper	48
3 stepper	215

Beaglebone PRU Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf der PRU verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=gpio0_23 dir_pin=gpio1_12 invert_step=0 step_pulse_ticks=20
config_stepper oid=1 step_pin=gpio1_15 dir_pin=gpio0_26 invert_step=0 step_pulse_ticks=20
config_stepper oid=2 step_pin=gpio0_22 dir_pin=gpio2_1 invert_step=0 step_pulse_ticks=20
finalize_config crc=0
```

Der Test wurde zuletzt am Commit `59314d99` mit der Gcc-Version `pru-gcc (GCC) 8.0.0 20170530` (experimental) durchgeführt.

pru	ticks
1 stepper	231
3 stepper	847

STM32F042 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem STM32F042 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PA1 dir_pin=PA2 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PA3 dir_pin=PA2 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PB8 dir_pin=PA2 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.

stm32f042	ticks
1 stepper	59
3 stepper	249

STM32F103 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem STM32F103 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PC13 dir_pin=PB5 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PB3 dir_pin=PB6 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PA4 dir_pin=PB7 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.
stm32f103 tickt
1 Stepper 61
3 Schrittmotor 264

STM32F4 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem STM32F4 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PA5 dir_pin=PB5 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PB2 dir_pin=PB6 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PB3 dir_pin=PB7 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.
Die STM32F407-Ergebnisse wurden erzielt, indem ein STM32F407-Binary auf einem STM32F446 (und somit mit einem 168Mhz-Takt) ausgeführt wurde.

stm32f446	ticks
1 stepper	46
3 stepper	205
stm32f407	**ticks**
1 stepper	46
3 stepper	205

STM32G0B1 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem STM32G0B1 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PB13 dir_pin=PB12 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PB10 dir_pin=PB2 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PB0 dir_pin=PC5 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt mit Commit `247cd753` mit der gcc-Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt.

stm32g0b1	ticks
1 stepper	58
3 stepper	243

LPC176x Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem LPC176x verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=P1.20 dir_pin=P1.18 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=P1.21 dir_pin=P1.18 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=P1.23 dir_pin=P1.18 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` durchgeführt. Die 120Mhz LPC1769 Ergebnisse wurden durch Übertaktung eines LPC1768 auf 120Mhz erzielt.

lpc1768	ticks
1 stepper	52
3 stepper	222
lpc1769	**ticks**
1 stepper	51
3 stepper	222

SAMD21 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird für den SAMD21 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PA27 dir_pin=PA20 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PB3 dir_pin=PA21 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PA17 dir_pin=PA21 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` auf einem SAMD21G18 Mikrocontroller durchgeführt.

samd21	ticks
1 stepper	70
3 stepper	306

SAMD51 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem SAMD51 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=PA22 dir_pin=PA20 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=PA22 dir_pin=PA21 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=PA22 dir_pin=PA19 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` auf einem SAMD51J19A Mikrocontroller durchgeführt.

samd51	ticks
1 stepper	39
3 stepper	191
1 stepper (200Mhz)	39

samd51	ticks
3 stepper (200Mhz)	181

RP2040 Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf dem RP2040 verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=gpio25 dir_pin=gpio3 invert_step=-1 step_pulse_ticks=0
config_stepper oid=1 step_pin=gpio26 dir_pin=gpio4 invert_step=-1 step_pulse_ticks=0
config_stepper oid=2 step_pin=gpio27 dir_pin=gpio5 invert_step=-1 step_pulse_ticks=0
finalize_config crc=0
```

Der Test wurde zuletzt unter Commit `59314d99` mit gcc Version `arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0` auf einem Raspberry Pi Pico Board durchgeführt.

rp2040	ticks
1 stepper	5
3 stepper	22

Linux MCU Schrittfrequenz-Benchmark

Die folgende Konfigurationssequenz wird auf einem Raspberry Pi verwendet:

```
allocate_oids count=3
config_stepper oid=0 step_pin=gpio2 dir_pin=gpio3 invert_step=0 step_pulse_ticks=5
config_stepper oid=1 step_pin=gpio4 dir_pin=gpio5 invert_step=0 step_pulse_ticks=5
config_stepper oid=2 step_pin=gpio6 dir_pin=gpio17 invert_step=0 step_pulse_ticks=5
finalize_config crc=0
```

Der Test wurde zuletzt am Commit `59314d99` mit der gcc-Version `gcc (Raspbian 8.3.0-6+rpi1) 8.3.0` auf einem Raspberry Pi 3 (Revision a02082) durchgeführt. Es war schwierig, stabile Ergebnisse in diesem Benchmark zu erhalten.

Linux (RPi3)	ticks
1 stepper	160
3 stepper	380

Befehlsversand-Benchmark

Der Befehlsversand-Benchmark testet, wie viele "Dummy"-Befehle der Mikrocontroller verarbeiten kann. Er ist in erster Linie ein Test des Hardware-Kommunikationsmechanismus. Der Test wird mit dem Tool console.py durchgeführt (beschrieben in Debugging.md). Der folgende Text wird in das Terminalfenster von console.py kopiert und eingefügt:

```
DELAY {Uhr + 2*freq} get_uptime
FLOOD 100000 0.0 debug_nop
get_uptime
```

Bestimmen Sie nach Abschluss des Tests die Differenz zwischen den in den beiden "Uptime"-Antwortmeldungen gemeldeten Uhrzeiten. Die Gesamtzahl der Befehle pro Sekunde ist dann `100000 * mcu_frequency / clock_diff`.

Beachten Sie, dass dieser Test die USB/CPU-Kapazität eines Raspberry Pi sättigen kann. Wenn er auf einem Raspberry Pi, Beaglebone oder einem ähnlichen Host-Computer läuft, erhöhen Sie die Verzögerung (z. B. `DELAY {clock + 20*freq} get_uptime`). Sofern zutreffend, wurden die unten aufgeführten Benchmarks mit console.py auf einem Desktop-Rechner ausgeführt, wobei das Gerät über einen Hochgeschwindigkeits-Hub angeschlossen war.

MCU	Rate	Build	Build compiler
stm32f042 (CAN)	18K	c105adc8	arm-none-eabi-gcc (GNU Tools 7-2018-q3-update) 7.3.1
atmega2560 (serial)	23K	b161a69e	avr-gcc (GCC) 4.8.1
sam3x8e (serial)	23K	b161a69e	arm-none-eabi-gcc (Fedora 7.1.0-5.fc27) 7.1.0
at90usb1286 (USB)	75K	01d2183f	avr-gcc (GCC) 5.4.0
samd21 (USB)	223K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
pru (shared memory)	260K	c5968a08	pru-gcc (GCC) 8.0.0 20170530 (experimental)
stm32f103 (USB)	355K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
sam3x8e (USB)	418K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
lpc1768 (USB)	534K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0

MCU	Rate	Build	Build compiler
lpc1769 (USB)	628K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
sam4s8c (USB)	650K	8d4a5c16	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
samd51 (USB)	864K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
stm32f446 (USB)	870K	01d2183f	arm-none-eabi-gcc (Fedora 7.4.0-1.fc30) 7.4.0
rp2040 (USB)	873K	c5667193	arm-none-eabi-gcc (Fedora 10.2.0-4.fc34) 10.2.0

Host-Benchmarks

Es ist möglich, Timing-Tests auf der Host-Software unter Verwendung des "Batch-Modus"-Verarbeitungsmechanismus durchzuführen (beschrieben in Debugging.md). Dazu wählt man in der Regel eine große und komplexe G-Code-Datei und misst, wie lange die Host-Software für die Verarbeitung benötigt. Zum Beispiel:

```
time ~/klippy-env/bin/python ./klippy/klippy.py config/example-cartesian.cfg -i
something_complex.gcode -o /dev/null -d out/klipper.dict
```

Zu Klipper beitragen

Danke, dass du zu Klipper beiträgst! Dieses Dokument beschreibt den Prozess, wie man Änderungen an Klipper vornimmt.

Bitte besuchen Sie die Kontaktseite contact page für Informationen über das Melden eines Problems oder für Details zur Kontaktaufnahme mit den Entwicklern.

Überblick über den Beitragsprozess

Beiträge zu Klipper folgen im Allgemeinen einem sehr einfachen Prozess:

1. Ein Einsender beginnt damit, einen GitHub Pull Request zu erstellen, wenn ein Beitrag bereit für die breite Anwendung ist.
2. Wenn ein reviewer verfügbar ist, um den Beitrag review zu prüfen, wird er sich dem Pull Request auf GitHub zuordnen. Ziel der Überprüfung ist es, nach Fehlern zu suchen und zu prüfen, ob die Einreichung den dokumentierten Richtlinien entspricht.
3. Nach erfolgreicher Prüfung wird der Prüfer die Prüfung auf GitHub "genehmigen" und ein Betreuer maintainer wird die Änderung in den Klipper-Masterzweig übertragen.

Wenn Sie an Erweiterungen arbeiten, sollten Sie in Erwägung ziehen, ein Thema auf Klipper Discourse zu starten (oder dazu beizutragen). Eine fortlaufende Diskussion im Forum kann die Sichtbarkeit der Entwicklungsarbeit verbessern und kann andere anziehen, die daran interessiert sind, die neue Arbeit zu testen.

Was Sie bei einer Überprüfung erwarten können

Beiträge zu Klipper werden vor dem Zusammenführen geprüft. Das primäre Ziel des Überprüfungsprozesses ist es, nach Fehlern zu suchen und zu prüfen, ob die eingereichten Beiträge den in der Klipper-Dokumentation angegebenen Richtlinien entsprechen.

Es wird davon ausgegangen, dass es viele Wege gibt, eine Aufgabe zu erfüllen; es ist nicht die Absicht der Überprüfung, die "beste" Implementierung zu diskutieren. Wenn möglich, sind Diskussionen, die sich auf Fakten und Messungen konzentrieren, vorzuziehen.

Die meisten Einreichungen führen zu einer Rückmeldung im Rahmen einer Überprüfung. Seien Sie darauf vorbereitet, Feedback einzuholen, weitere Details zu liefern und die Einreichung bei Bedarf zu aktualisieren.

Worauf ein Gutachter in der Regel achten wird:

1. **Ist der Beitrag frei von Fehlern und kann er auf breiter Basis eingesetzt werden?**

Von den Einreichern wird erwartet, dass sie ihre Änderungen vor der Einreichung testen. Die Prüfer suchen nach Fehlern, aber sie testen die eingereichten Beiträge im Allgemeinen nicht. Eine angenommene Einreichung wird oft innerhalb weniger Wochen nach der Annahme an Tausende von Druckern verteilt. Die Qualität der Einsendungen hat daher Priorität.

Das Haupt-Repository von Klipper3d/klipper GitHub akzeptiert keine experimentellen Arbeiten. Einreicher sollten Experimente, Fehlersuche und Tests in ihren eigenen Repositories durchführen. Der Klipper Discourse Server ist ein guter Ort, um auf neue Arbeiten aufmerksam zu machen und Nutzer zu finden, die daran interessiert sind, Feedback aus der Praxis zu geben.

Einreichungen müssen alle Regressionstests regression test cases bestehen.

Eingesandter Code sollte keinen übermäßigen Debugging-Code, Debugging-Optionen oder Laufzeit-Debugging-Logging enthalten.

Kommentare im eingereichten Code sollten sich auf die Verbesserung der Code-Wartung konzentrieren. Einreichungen sollten weder "auskommentierten Code" noch übermäßige Kommentare enthalten, die frühere Implementierungen beschreiben. Es sollten keine übermäßigen "ToDo"-Kommentare enthalten sein.

Bei Aktualisierungen der Dokumentation sollte nicht angegeben werden, dass es sich um eine "laufende Arbeit" handelt.

2. Bietet die Einreichung einen hohen Nutzen für reale Benutzer, die reale Aufgaben ausführen?

Die Prüfer müssen, zumindest für sich selbst, grob bestimmen, "wer die Zielgruppe ist", eine grobe Skala für "die Größe dieser Zielgruppe", den "Nutzen", den sie erhalten werden, wie der "Nutzen gemessen wird" und die "Ergebnisse dieser Messtests". In den meisten Fällen ist dies sowohl für den Einreicher als auch für den Gutachter offensichtlich und wird bei einer Begutachtung nicht explizit angegeben.

Bei Einreichungen für den Master-Klipper-Zweig wird erwartet, dass sie ein nennenswertes Zielpublikum haben. Als Faustregel gilt, dass die eingereichten Beiträge eine Zielgruppe von mindestens 100 realen Nutzern ansprechen sollten.

Wenn ein Gutachter nach Details über den "Nutzen" einer Einsendung fragt, betrachten Sie dies bitte nicht als Kritik. Die realen Vorteile einer Änderung zu verstehen, ist ein natürlicher Bestandteil einer Überprüfung.

Bei der Erörterung von Vorteilen ist es besser, "Fakten und Messungen" zu diskutieren. Im Allgemeinen suchen die Prüfer nicht nach Antworten der Form "jemand könnte Option X nützlich finden", noch suchen sie nach Antworten der Form "diese Einreichung fügt eine Funktion hinzu, die Firmware X implementiert". Stattdessen ist es im Allgemeinen vorzuziehen, Einzelheiten darüber zu erörtern, wie die Qualitätsverbesserung gemessen wurde und was die Ergebnisse dieser Messungen waren - zum Beispiel: "Tests auf Acme X1000-Druckern zeigen verbesserte Ecken, wie auf dem Bild zu sehen ist ...", oder zum Beispiel: "Die Druckzeit für ein reales Objekt X auf einem Foomatic X900-Drucker ging von 4 Stunden auf 3,5 Stunden zurück". Es versteht sich von selbst, dass Tests dieser Art viel Zeit und Mühe kosten können. Einige der bemerkenswertesten Funktionen von Klipper wurden monatelang diskutiert, überarbeitet, getestet und dokumentiert, bevor sie in den Hauptzweig aufgenommen wurden.

Alle neuen Module, Konfigurationsoptionen, Befehle, Befehlsparameter und Dokumente sollten "high impact" haben. Wir wollen die Benutzer nicht mit Optionen belasten, die sie nicht vernünftig konfigurieren können, noch wollen wir sie mit Optionen belasten, die keinen nennenswerten Nutzen bringen.

Ein Prüfer kann um Klärung bitten, wie ein Benutzer eine Option konfigurieren soll - eine ideale Antwort enthält Details zum Prozess - z. B. "Benutzer des MegaX500 sollen die Option X auf 99,3 setzen, während Benutzer des Elite100Y die Option X mit der Prozedur ... kalibrieren sollen".

Wenn das Ziel einer Option darin besteht, den Code modularer zu gestalten, dann verwenden Sie lieber Codekonstanten als benutzerseitige Konfigurationsoptionen.

Neue Module, neue Optionen und neue Parameter sollten keine ähnliche Funktionalität wie bestehende Module bieten - wenn die Unterschiede willkürlich sind, ist es besser, das bestehende System zu verwenden oder den bestehenden Code zu überarbeiten.

3. Ist das Copyright der Einreichung klar, nicht unentgeltlich und kompatibel?

Neue C- und Python-Dateien sollten eine eindeutige Copyright-Erklärung enthalten. Das bevorzugte Format finden Sie in den vorhandenen Dateien. Es wird davon abgeraten, ein Urheberrecht für eine bestehende Datei zu erklären, wenn kleinere Änderungen an dieser Datei vorgenommen werden.

Code, der aus Quellen Dritter stammt, muss mit der Klipper-Lizenz (GNU GPLv3) kompatibel sein. Großer Code von Drittanbietern sollte in das lib/ Verzeichnis eingefügt werden (und dem in lib/README beschriebenen Format folgen).

Die Einreicher müssen eine Signed-off-by line-Zeile mit ihrem vollen Namen angeben. Sie zeigt an, dass der Einreicher mit dem Herkunftsnachweis des Entwicklers einverstanden ist developer certificate of origin.

4. Entspricht der eingereichte Code den in der Klipper-Dokumentation beschriebenen Richtlinien?

Insbesondere sollte der Code den Richtlinien in Code_Overview.md folgen und die Konfigurationsdateien sollten den Richtlinien in Example_Configs.md folgen.

5. Wird die Klipper-Dokumentation aktualisiert, um neue Änderungen zu berücksichtigen?

Zumindest muss die Referenzdokumentation mit den entsprechenden Änderungen des Codes aktualisiert werden:

- Alle Befehle und Befehlsparameter müssen in G-Codes.md dokumentiert werden.
- Alle benutzerorientierten Module und ihre Konfigurationsparameter müssen in Config_Reference.md dokumentiert werden.
- Alle exportierten "Statusvariablen" müssen in Status_Reference.md. dokumentiert werden.
- Alle neuen "Webhooks" und ihre Parameter müssen in API_Server.md dokumentiert werden.
- Jede Änderung, die eine nicht rückwärtskompatible Änderung an einem Befehl oder einer Konfigurationsdatei-Einstellung vornimmt, muss in Config_Changes.md dokumentiert werden.

Neue Dokumente sollten zu Overview.md hinzugefügt werden und in den Website-Index docs/_klipper3d/mkdocs.yml aufgenommen werden.

6. Sind die Commits wohlgeformt, behandeln sie ein einziges Thema pro Commit und sind sie unabhängig?

- Commit-Nachrichten sollten dem bevorzugten Format preferred format folgen.

- Commits dürfen keinen Merge-Konflikt haben. Neue Hinzufügungen zum Klipper-Masterzweig werden immer über einen "rebase" oder "squash and rebase" durchgeführt. Es ist im Allgemeinen nicht notwendig, dass die Einreicher ihre Beiträge bei jeder Aktualisierung des Klipper-Master-Repositorys neu zusammenführen. Sollte es jedoch einen Merge-Konflikt geben, wird empfohlen, den Konflikt mit `git rebase` zu lösen.

- Jeder Commit sollte eine einzelne, übergeordnete Änderung betreffen. Große Änderungen sollten in mehrere unabhängige Commits aufgeteilt werden. Jeder Commit sollte "für sich stehen", damit Werkzeuge wie `git bisect` und `git revert` zuverlässig funktionieren.

- Whitespace-Änderungen sollten nicht mit funktionalen Änderungen vermischt werden. Generell werden grundlose Whitespace-Änderungen nicht akzeptiert, es sei denn, sie stammen von dem etablierten "Besitzer" des zu ändernden Codes.

Klipper implementiert keinen strikten "Coding Style Guide", aber Änderungen an bestehendem Code sollten dem High-Level Code Flow, dem Einrückungsstil und dem Format des bestehenden Codes folgen. Einreichungen von neuen Modulen und Systemen haben mehr Flexibilität im Codierungsstil, aber es ist vorzuziehen, dass der neue Code einem intern konsistenten Stil folgt und allgemein den branchenweiten Codierungsnormen folgt.

Es ist nicht das Ziel einer Überprüfung, "bessere Implementierungen" zu diskutieren. Wenn ein Gutachter jedoch Schwierigkeiten hat, die Implementierung eines Beitrags zu verstehen, kann er um Änderungen bitten, um die Implementierung transparenter zu machen. Insbesondere dann, wenn sich die Prüfer nicht davon überzeugen können, dass ein Beitrag frei von Mängeln ist, können Änderungen erforderlich sein.

Als Teil einer Überprüfung kann ein Prüfer einen alternativen Pull Request für ein Thema erstellen. Dies kann geschehen, um ein übermäßiges "Hin und Her" bei kleineren Verfahrensfragen zu vermeiden und so den Einreichungsprozess zu straffen. Es kann auch geschehen, weil die Diskussion einen Prüfer dazu inspiriert, eine alternative Implementierung zu erstellen. Beide Situationen sind ein normales Ergebnis einer Überprüfung und sollten nicht als Kritik an der ursprünglichen Einreichung angesehen werden.

Hilfe bei Überprüfungen

Wir freuen uns über Hilfe bei Überprüfungen! Es ist nicht notwendig, ein gelisteter Reviewer listed reviewer zu sein, um einen Review durchzuführen. Einreicher von GitHub Pull Requests werden ebenfalls ermutigt, ihre eigenen Einreichungen zu überprüfen.

Um bei einer Überprüfung zu helfen, folgen Sie den Schritten, die unter "Was Sie bei einer Überprüfung erwarten können" what to expect in a review beschrieben sind, um die Einreichung zu überprüfen.
Fügen Sie nach Abschluss der Überprüfung einen Kommentar zum GitHub Pull Request mit Ihren Ergebnissen hinzu. Wenn die Einreichung die Überprüfung bestanden hat, geben Sie das bitte ausdrücklich in dem Kommentar an - zum Beispiel so etwas wie "Ich habe diese Änderung gemäß den Schritten im CONTRIBUTING-Dokument überprüft und alles sieht für mich gut aus". Wenn Sie nicht in der Lage waren, einige Schritte der Überprüfung durchzuführen, geben Sie bitte ausdrücklich an, welche Schritte überprüft wurden und welche nicht - zum Beispiel so etwas wie "Ich habe den Code nicht auf Fehler überprüft, aber ich habe alles andere im CONTRIBUTING-Dokument überprüft und es sieht gut aus".

Wir schätzen auch das Testen von Einreichungen. Wenn der Code getestet wurde, fügen Sie dem GitHub Pull Request bitte einen Kommentar mit den Ergebnissen Ihres Tests hinzu - Erfolg oder Misserfolg. Bitte geben Sie explizit an, dass der Code getestet wurde, und die Ergebnisse - zum Beispiel so etwas wie "Ich habe diesen Code auf meinem Acme900Z-Drucker mit einem Vasen-Druck getestet und die Ergebnisse waren gut".

Prüfer

Die Klipper "Gutachter" sind:

Name	GitHub Id	Areas of interest
Dmitry Butyugin	@dmbutyugin	Input shaping, resonance testing, kinematics
Eric Callahan	@Arksine	Bed leveling, MCU flashing
Kevin O'Connor	@KevinOConnor	Core motion system, Micro-controller code
Paul McGowan	@mental405	Configuration files, documentation

Bitte "pingen" Sie keinen der Gutachter an und richten Sie keine Beiträge an sie. Alle Reviewer beobachten die Foren und PRs und werden Reviews übernehmen, wenn sie Zeit dazu haben.

Die Klipper "Betreuer" sind:

Name	GitHub name
Kevin O'Connor	@KevinOConnor

Format der Commit-Nachrichten

Jeder Commit sollte eine Commit-Nachricht haben, die ähnlich wie die folgende formatiert ist:

```
Modul: Groß geschriebene, kurze (50 Zeichen oder weniger) Zusammenfassung

Ausführlicherer erläuternder Text, falls nötig.  Umfassen Sie ihn auf etwa 75
Zeichen oder so.  In manchen Kontexten wird die erste Zeile als
```

```
Betreff einer E-Mail und der Rest des Textes als Textkörper behandelt.  Die leere
Zeile, die die Zusammenfassung vom Textkörper trennt, ist wichtig (es sei denn, Sie lassen
den Textkörper ganz weglassen); Tools wie rebase können durcheinander kommen, wenn Sie die
zwei zusammen.

Weitere Absätze kommen nach Leerzeilen...

Abgezeichnet von: Mein Name <myemail@example.org>
```

Im obigen Beispiel sollte module der Name einer Datei oder eines Verzeichnisses im Repository sein (ohne Dateierweiterung). Zum Beispiel, `clocksync: Korrektur eines Tippfehlers im pause()-Aufruf zur connect time`. Der Zweck der Angabe eines Modulnamens in der Commit-Nachricht ist es, den Kontext für die Commit-Kommentare zu liefern.

Es ist wichtig, eine "Signed-off-by"-Zeile auf jedem Commit zu haben - sie bescheinigt, dass Sie mit dem Ursprungszertifikat des Entwicklers developer certificate of origin einverstanden sind. Sie muss Ihren echten Namen (keine Pseudonyme oder anonyme Beiträge) und eine aktuelle E-Mail-Adresse enthalten.

Zu Klipper Translations beitragen

Das Klipper-translations Project ist ein Projekt, das sich der Übersetzung von Klipper in verschiedene Sprachen widmet. Weblate hostet alle Gettext-Strings zum Übersetzen und Überprüfen. Lokalisierungen können auf klipper3d.org angezeigt werden, wenn sie die folgenden Anforderungen erfüllen:

75% Gesamtabdeckung
Alle Titel (H1) sind übersetzt
Eine aktualisierte Navigationshierarchie PR in klipper-translations.

Um die Frustration beim Übersetzen von domänenspezifischen Begriffen zu verringern und um auf die laufenden Übersetzungen aufmerksam zu machen, können Sie einen PR einreichen, der die `readme.md` des Klipper-translations Project modifiziert. Sobald eine Übersetzung fertig ist, kann die entsprechende Änderung am Klipper-Projekt vorgenommen werden.

Wenn eine Übersetzung bereits im Klipper-Repository existiert und die oben genannte Checkliste nicht mehr erfüllt, wird sie nach einem Monat ohne Aktualisierung als veraltet markiert.

Sobald die Anforderungen erfüllt sind, müssen Sie:

1. klipper-tranlations repository active_translations aktualisieren
2. Optional: fügen Sie eine manual-index.md Datei in den Ordner `docs\locals\<lang>` des klipper-translations Repositorys ein, um die sprachspezifische index.md zu ersetzen (die generierte index.md wird nicht korrekt dargestellt).

Bekannte Probleme:

1. Derzeit gibt es keine Methode, um Bilder in der Dokumentation korrekt zu übersetzen
2. Es ist nicht möglich, Titel in mkdocs.yml zu übersetzen.

Klipper verpacken

Klipper ist so etwas wie eine Paketierungsanomalie unter den Python-Programmen, da es keine setuptools zum Bauen und Installieren verwendet. Einige Hinweise, wie man es am besten verpackt, sind im Folgenden aufgeführt:

C-Module

Klipper benutzt ein C-Modul, um einige kinematische Berechnungen schneller durchzuführen. Dieses Modul muss zur Paketierungszeit kompiliert werden, um eine Laufzeitabhängigkeit von einem Compiler zu vermeiden. Um das C-Modul zu kompilieren, führen Sie `python2 klippy/chelper/__init__.py` aus.

Kompilieren von python code

Viele Distributionen kompilieren den gesamten Python-Code vor dem Paketieren, um die Startzeit zu verbessern. Sie können dies tun, indem Sie `python2 -m compileall klippy` ausführen.

Versionierung

Wenn du ein Klipper-Paket aus git erstellst, ist es üblich, kein .git-Verzeichnis mitzuliefern, so dass die Versionierung ohne git durchgeführt werden muss. Um dies zu tun, verwenden Sie das Skript in `scripts/make_version.py`, das wie folgt ausgeführt werden sollte: `python2 scripts/make_version.py YOURDISTRONAME > klippy/.version`.

Beispiel-Paketierungsskript

klipper-git ist für Arch Linux gepackt und hat ein PKGBUILD (Paketbau-Skript), das im Arch User Repository verfügbar ist.

Gerätespezifische Dokumente

Beispielkonfigurationen

Dieses Dokument enthält Richtlinien für das Einbringen einer Klipper-Beispielkonfiguration in das Klipper-Github-Repository (config directory).

Beachte, dass der Klipper Community Discourse server auch eine nützliche Ressource ist, um Konfigurationsdateien zu finden und zu teilen.

1. Richtlinien
   - Wähle den passenden Präfix für den Dateinamen der Konfiguration:
   - Das Präfix `printer` wird für Standarddrucker verwendet, die von einem großen Hersteller verkauft werden.
   - Das `generic` Präfix wird für eine 3D-Druckerplatine verwendet, die in vielen verschiedenen Druckertypen verwendet werden kann.
   - Das Kit-Präfix wird für 3D-Drucker verwendet, die nach einer weit verbreiteten Spezifikation zusammengebaut werden. Diese "Kit"-Drucker unterscheiden sich im Allgemeinen von normalen "Druckern" dadurch, dass sie nicht von einem Hersteller verkauft werden.
   - Das s`ample`-Präfix wird für Konfigurations-"Schnipsel" verwendet, die man in die Hauptkonfigurationsdatei kopieren und einfügen kann.
   - Das Präfix `example` wird verwendet, um die Kinematik des Druckers zu beschreiben. Diese Art von Konfiguration wird normalerweise nur zusammen mit dem Code für eine neue Art von Druckerkinematik hinzugefügt.
2. Alle Konfigurationsdateien müssen mit dem Suffix `.cfg` enden. Die Druckerkonfigurationsdateien müssen mit einer Jahreszahl gefolgt von `.cfg` enden (z. B. `-2019.cfg`). In diesem Fall ist die Jahreszahl das ungefähre Jahr, in dem der jeweilige Drucker verkauft wurde.
3. Verwenden Sie keine Leerzeichen oder Sonderzeichen im Dateinamen der Konfigurationsdatei. Der Dateiname sollte nur die Zeichen `A-Z, a-z, 0-9, -,` und `.` enthalten.
4. Klipper muss in der Lage sein, die Konfigurationsdatei des Druckers, des generischen Druckers und des Kit-Beispiels ohne Fehler zu starten. Diese Konfigurationsdateien sollten zu dem Regressionstestfall test/klippy/printers.test hinzugefügt werden. Fügen Sie neue Konfigurationsdateien zu diesem Testfall im entsprechenden Abschnitt und in alphabetischer Reihenfolge innerhalb dieses Abschnitts hinzu.
5. Die Beispielkonfiguration sollte für die "Standard"-Konfiguration des Druckers gelten. (Es gibt zu viele "angepasste" Konfigurationen, um sie im Haupt-Repository von Klipper zu erfassen). Ebenso fügen wir nur Beispielkonfigurationsdateien für Drucker, Kits und Boards hinzu, die sich großer Beliebtheit erfreuen (z.B. sollten mindestens 100 Stück davon im aktiven Einsatz sein). Ziehe in Erwägung, den Klipper Community Discourse server für andere Konfigurationen zu verwenden.
6. Geben Sie nur die Geräte an, die auf dem jeweiligen Drucker oder Board vorhanden sind. Geben Sie keine Einstellungen an, die spezifisch für Ihr spezielles Setup sind.
   - Für `generic` Konfigurationsdateien sollten nur die Geräte auf dem Mainboard beschrieben werden. Es wäre z.B. nicht sinnvoll, einer "generischen" Config einen Abschnitt für die Display-Konfiguration hinzuzufügen, da es keine Möglichkeit gibt, zu wissen, ob das Board an diesen Typ von Display angeschlossen wird. Wenn das Board einen speziellen Hardware-Port hat, um ein optionales Peripheriegerät zu ermöglichen (z.B. einen Bltouch-Port), dann kann man einen "auskommentierten" Konfigurationsabschnitt für das jeweilige Gerät hinzufügen.
   - Geben Sie in einer Beispielkonfiguration nicht pressure_advance an, da dieser Wert spezifisch für das Filament ist, nicht für die Druckerhardware. Geben Sie auch keine max_extrude_only_velocity oder max_extrude_only_accel Einstellungen an.
   - Geben Sie keinen Konfigurationsabschnitt an, der einen Host-Pfad oder eine Host-Hardware enthält. Geben Sie zum Beispiel keine [virtual_sdcard] oder [temperature_host] Konfigurationsabschnitte an.
   - Definieren Sie nur Makros, die Funktionen nutzen, die für den jeweiligen Drucker spezifisch sind, oder um g-Codes zu definieren, die üblicherweise von Slicern ausgegeben werden, die für den jeweiligen Drucker konfiguriert sind.
7. Wenn möglich, ist es am besten, den gleichen Wortlaut, die gleiche Formulierung, die gleiche Einrückung und die gleiche Reihenfolge der Abschnitte wie in den bestehenden Konfigurationsdateien zu verwenden.
   - Am Anfang jeder Konfigurationsdatei sollte der Typ des Mikrocontrollers aufgeführt sein, den der Benutzer während "make menuconfig" auswählen sollte. Sie sollte auch einen Verweis auf "docs/Config_Reference.md" enthalten.
   - Kopieren Sie die Felddokumentation nicht in die Beispielkonfigurationsdateien. (Dies führt zu einem hohen Wartungsaufwand, da eine Aktualisierung der Dokumentation dann an vielen Stellen geändert werden müsste).
   - Beispielkonfigurationsdateien sollten keinen Abschnitt "SAVE_CONFIG" enthalten. Kopieren Sie bei Bedarf die relevanten Felder aus dem Abschnitt SAVE_CONFIG in den entsprechenden Abschnitt im Hauptkonfigurationsbereich.
   - Verwenden Sie die Syntax `field: value` anstelle von `field=value`.
   - Wenn Sie einen Extruder `rotation_distance` hinzufügen, ist es besser, ein `gear_ratio` anzugeben, wenn der Extruder ein Getriebe hat. Wir gehen davon aus, dass der Rotationsabstand in den Beispielkonfigurationen mit dem Umfang des Wälzfräsers im Extruder korreliert - er liegt normalerweise im Bereich von 20 bis 35 mm. Bei der Angabe eines gear_ratio ist es besser, die tatsächlichen Zahnräder des Mechanismus anzugeben (z.B. `gear_ratio: 80:20` gegenüber `gear_ratio: 4:1`). Weitere Informationen finden Sie im Dokument rotation distance document.
   - Vermeiden Sie es, Feldwerte zu definieren, die auf ihren Standardwert gesetzt sind. Zum Beispiel sollte man nicht `min_extrude_temp: 170` angeben, da dies bereits der Standardwert ist.
   - Wenn möglich, sollten die Zeilen nicht länger als 80 Spalten sein.
   - Vermeiden Sie das Hinzufügen von Attributions- oder Revisionsmeldungen
   - (Vermeiden Sie zum Beispiel Zeilen wie "Diese Datei wurde erstellt von ...".) Fügen Sie die Attribution und den Änderungsverlauf in die Git-Commit-Nachricht ein.
8. Verwenden Sie keine veralteten Funktionen in der Beispielkonfigurationsdatei.
9. Deaktivieren Sie kein Standard-Sicherheitssystem in einer Beispiel-Konfigurationsdatei. Zum Beispiel sollte eine Konfigurationsdatei keine benutzerdefinierte `max_extrude_cross_section` angeben. Aktivieren Sie keine Debugging-Funktionen. Zum Beispiel sollte es keine force_move Konfigurationssektion geben.
10. Alle bekannten Boards, die Klipper unterstützt, können die standardmäßige serielle Baudrate von 250000 verwenden. Empfehlen Sie keine andere Baudrate in einer Beispielkonfigurationsdatei.

Beispielkonfigurationsdateien werden durch das Erstellen eines Github "pull request" eingereicht. Bitte befolgen Sie auch die Anweisungen in dem Dokument, das den Beitrag liefert contributing document.

SDCard-Updates

Viele der heute gängigen Controller-Boards werden mit einem Bootloader ausgeliefert, der die Aktualisierung der Firmware über eine SD-Karte ermöglicht. Obwohl dies in vielen Fällen praktisch ist, bieten diese Bootloader in der Regel keine andere Möglichkeit zur Aktualisierung der Firmware. Dies kann lästig sein, wenn Ihr Board an einem schwer zugänglichen Ort montiert ist oder wenn Sie die Firmware häufig aktualisieren müssen. Nachdem Klipper auf einen Controller geflasht wurde, ist es möglich, die neue Firmware auf die SD-Karte zu übertragen und das Flashen über ssh zu starten.

Typisches Upgrade-Verfahren

Das Verfahren zur Aktualisierung der MCU-Firmware über die SD-Karte ist ähnlich wie bei anderen Methoden. Anstatt `make flash` zu verwenden, muss ein Hilfsskript, `flash-sdcard.sh`, ausgeführt werden. Die Aktualisierung eines BigTreeTech SKR 1.3 könnte wie folgt aussehen:

```
sudo service klipper stop
cd ~/klipper
git pull
make clean
make menuconfig
make
./scripts/flash-sdcard.sh /dev/ttyACM0 btt-skr-v1.3
sudo service klipper start
```

Es ist Sache des Benutzers, den Speicherort des Geräts und den Namen der Karte zu bestimmen. Wenn ein Benutzer mehrere Boards flashen muss, sollte `flash-sdcard.sh` (oder `make flash`, falls erforderlich) für jedes Board ausgeführt werden, bevor der Klipper-Dienst neu gestartet wird.

Die unterstützten Karten können mit dem folgenden Befehl aufgelistet werden:

```
./scripts/flash-sdcard.sh -l
```

Wenn dein Board nicht aufgelistet ist, kann es notwendig sein, eine neue Boarddefinition wie unten described below beschrieben hinzuzufügen.

Erweiterte Verwendung

Die obigen Befehle gehen davon aus, dass Ihre MCU mit der Standard-Baudrate von 250000 verbunden ist und die Firmware unter `~/klipper/out/klipper.bin` zu finden ist. Das Skript flash-sdcard.sh bietet Optionen zum Ändern dieser Voreinstellungen. Alle Optionen können über den Hilfebildschirm eingesehen werden:

```
./scripts/flash-sdcard.sh -h
SD Card upload utility for Klipper

usage: flash_sdcard.sh [-h] [-l] [-b <baud>] [-f <firmware>]
                       <device> <board>

positional arguments:
  <device>        device serial port
  <board>         board type

optional arguments:
  -h              show this message
  -l              list available boards
  -b <baud>       serial baud rate (default is 250000)
  -f <firmware>   path to klipper.bin
```

Wenn Ihr Board mit einer Firmware geflasht ist, die sich mit einer benutzerdefinierten Baudrate verbindet, ist es möglich, durch Angabe der Option -b ein Upgrade durchzuführen:

```
./scripts/flash-sdcard.sh -b 115200 /dev/ttyAMA0 btt-skr-v1.3
```

Wenn Sie einen Klipper-Build flashen möchten, der sich an einem anderen Ort als dem Standard-Speicherort befindet, können Sie dies durch Angabe der Option -f tun:

```
./scripts/flash-sdcard.sh -f ~/downloads/klipper.bin /dev/ttyAMA0 btt-skr-v1.3
```

Beachten Sie, dass es beim Upgrade eines MKS Robin E3 nicht notwendig ist, `update_mks_robin.py` manuell auszuführen und die resultierende Binärdatei an `flash-sdcard.sh` zu übergeben. Diese Prozedur wird während des Upload-Prozesses automatisiert.

Vorsichtsmaßnahmen

Wie in der Einleitung erwähnt, funktioniert diese Methode nur für die Aktualisierung der Firmware. Das anfängliche Flashen muss manuell gemäß den für Ihr Controller-Board geltenden Anweisungen durchgeführt werden.

Es ist zwar möglich, einen Build zu flashen, der die serielle Baudrate oder die Verbindungsschnittstelle ändert (z. B. von USB zu UART), aber die Verifizierung wird immer fehlschlagen, da das Skript nicht in der Lage ist, sich erneut mit der MCU zu verbinden, um die aktuelle Version zu verifizieren.

Es werden nur Boards unterstützt, die SPI für die SD-Kartenkommunikation verwenden. Boards, die SDIO verwenden, wie z.B. das Flymaker Flyboard und MKS Robin Nano V1/V2, werden nicht funktionieren.

Board-Definitionen
Die meisten gängigen Boards sollten verfügbar sein, es ist jedoch möglich, bei Bedarf eine neue Boarddefinition hinzuzufügen. Die Boarddefinitionen befinden sich in ~/klipper/scripts/spi_flash/board_defs.py. Die Definitionen werden in einem Wörterbuch gespeichert, zum Beispiel:

```
BOARD_DEFS = {
    'generic-lpc1768': {
        'mcu': 'lpc1768',
        'spi_bus': "ssp1",
        "cs_pin": "P0.6"
    },
    ...<weitere Definitionen>
}
```

Die folgenden Felder können angegeben werden:

- `mcu:` Der mcu-Typ. Dieser kann nach dem Konfigurieren des Builds über make menuconfig durch Ausführen von `cat .config | grep CONFIG_MCU` wiedergefunden werden. Dieses Feld ist erforderlich.
- `spi_bus:` Der SPI-Bus, der mit der SD-Karte verbunden ist. Dieser sollte aus dem Schaltplan des Boards entnommen werden. Dieses Feld ist erforderlich.
- `cs_pin:` Der Chip Select Pin, der mit der SD-Karte verbunden ist. Dies sollte aus dem Schaltplan der Karte entnommen werden. Dieses Feld ist erforderlich.
- `firmware_pfad:` Der Pfad auf der SD-Karte, in den die Firmware übertragen werden soll. Die Vorgabe ist `firmware.bin`.
- `current_firmware_path` Der Pfad auf der SD-Karte, in dem sich die umbenannte Firmware-Datei nach einem erfolgreichen Flash befindet. Die Vorgabe ist `firmware.cur`.

Wenn Software-SPI erforderlich ist, sollte das Feld `spi_bus` auf `swspi` gesetzt werden und das folgende zusätzliche Feld angegeben werden:

- `spi_pins:` Dies sollten 3 durch Komma getrennte Pins sein, die mit der SD-Karte verbunden sind, und zwar im Format `miso,mosi,sclk`.

Es sollte äußerst selten vorkommen, dass Software-SPI notwendig ist, typischerweise werden nur Boards mit Designfehlern dies erfordern. Die `btt-skr-pro-`Boarddefinition ist ein Beispiel dafür.

Vor der Erstellung einer neuen Tafeldefinition sollte geprüft werden, ob eine bestehende Tafeldefinition die für die neue Tafel erforderlichen Kriterien erfüllt. Wenn dies der Fall ist, kann ein `BOARD_ALIAS` angegeben werden. Zum Beispiel kann der folgende Alias hinzugefügt werden, um `my-new-board` als Alias für `generic-lpc1768` anzugeben:

```
BOARD_ALIASES = {
  ...<vorherige Aliasnamen>,
  'my-new-board': BOARD_DEFS['generic-lpc1768'],
}
```

Wenn Sie eine neue Boarddefinition benötigen und sich mit der oben beschriebenen Vorgehensweise nicht wohlfühlen, empfehlen wir Ihnen, eine solche im Klipper Community Discord anzufordern.

RPi-Mikrocontroller

Dieses Dokument beschreibt den Prozess, Klipper auf einem RPi laufen zu lassen und denselben RPi als sekundäre MCU zu verwenden.

Warum RPi als sekundäre MCU verwenden?

Oft haben die MCUs, die für die Steuerung von 3D-Druckern bestimmt sind, eine begrenzte und vorkonfigurierte Anzahl von freiliegenden Pins, um die wichtigsten Druckfunktionen zu steuern (Thermowiderstände, Extruder, Stepper ...). Die Verwendung des RPi, auf dem Klipper als sekundäre MCU installiert ist, bietet die Möglichkeit, die GPIOs und die Busse (i2c, spi) des RPi innerhalb von Klipper direkt zu nutzen, ohne Octoprint-Plugins (falls verwendet) oder externe Programme zu verwenden, was die Möglichkeit bietet, alles innerhalb des Druck-GCODE zu steuern.

Warnung! Wenn deine Plattform ein Beaglebone ist und du die Installationsschritte korrekt befolgt hast, ist die Linux-MCU bereits installiert und für dein System konfiguriert.

Installiere das rc-Skript

Wenn du den Host als sekundäre MCU verwenden willst, muss der klipper_mcu Prozess vor dem klippy Prozess laufen.

Nach der Installation von Klipper installiere das Skript. run:

```
cd ~/klipper/
sudo cp "./scripts/klipper-mcu-start.sh" /etc/init.d/klipper_mcu
sudo update-rc.d klipper_mcu defaults
```

Kompilieren des Mikrocontroller-Codes

Um den Klipper-Mikrocontroller-Code zu kompilieren, müssen Sie ihn zunächst für den "Linux-Prozess" konfigurieren:

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

Setze im Menü "Microcontroller Architecture" auf "Linux process", speichere und beende.

Um den neuen Mikrocontroller-Code zu bauen und zu installieren, führe aus:

```
sudo service klipper stop
make flash
sudo service klipper start
```

Wenn klippy.log beim Versuch, eine Verbindung zu /tmp/klipper_host_mcu herzustellen, den Fehler "Permission denied" meldet, müssen Sie Ihren Benutzer zur tty-Gruppe hinzufügen. Mit dem folgenden Befehl wird der Benutzer "pi" zur tty-Gruppe hinzugefügt:

```
sudo usermod -a -G tty pi
```

Verbleibende Konfiguration

Vervollständige die Installation, indem du die sekundäre Klipper MCU konfigurierst, indem du den Anweisungen in RaspberryPi sample config und Multi MCU sample config folgst.

Optional: Aktivieren von SPI

Vergewissern Sie sich, dass der Linux SPI-Treiber aktiviert ist, indem Sie `sudo raspi-config` ausführen und SPI im Menü "Interfacing options" aktivieren.

Optional: Identifizieren Sie den richtigen gpiochip

Auf dem Raspberry Pi und vielen Klonen gehören die Pins, die auf dem GPIO freigelegt sind, zum ersten gpiochip. Sie können daher auf klipper verwendet werden, indem man sie einfach mit dem Namen gpio0..n bezeichnet. Es gibt jedoch Fälle, in denen die freiliegenden Pins zu anderen gpiochips als dem ersten gehören. Zum Beispiel im Falle einiger OrangePi-Modelle oder wenn ein Port Expander verwendet wird. In diesen Fällen ist es nützlich, die Befehle für den Zugriff auf das *Linux-GPIO-Zeichengerät* zu verwenden, um die Konfiguration zu überprüfen.

Um das Linux GPIO character device - binary auf einer debian-basierten Distribution wie octopi zu installieren, führen Sie aus:

```
sudo apt-get install gpiod
```

Um den verfügbaren gpiochip zu überprüfen, führen Sie aus:

```
gpiodetect
```

Um die Pin-Nummer und die Pin-Verfügbarkeit zu prüfen, tun:

```
gpioinfo
```

Der gewählte Pin kann dann in der Konfiguration als `gpiochip<n>/gpio<o>` verwendet werden, wobei **n** die Chipnummer ist, die der Befehl `gpiodetect` angibt, und **o** die Zeilennummer, die der Befehl `gpioinfo` angibt.

Achtung: nur als `unused` markierte gpio können verwendet werden. Es ist nicht möglich, dass eine *Zeile* von mehreren Prozessen gleichzeitig benutzt wird.

Zum Beispiel auf einem RPi 3B+, wo klipper den GPIO20 für einen Schalter verwendet:

```
$ gpiodetect
gpiochip0 [pinctrl-bcm2835] (54 Zeilen)
gpiochip1 [raspberrypi-exp-gpio] (8 Zeilen)

$ gpioinfo
gpiochip0 - 54 lines:

$ gpiodetect
gpiochip0 [pinctrl-bcm2835] (54 lines)
gpiochip1 [raspberrypi-exp-gpio] (8 lines)

$ gpioinfo
gpiochip0 - 54 lines:
        line   0:      unnamed       unused   input  active-high
        line   1:      unnamed       unused   input  active-high
        line   2:      unnamed       unused   input  active-high
        line   3:      unnamed       unused   input  active-high
        line   4:      unnamed       unused   input  active-high
        line   5:      unnamed       unused   input  active-high
        line   6:      unnamed       unused   input  active-high
        line   7:      unnamed       unused   input  active-high
        line   8:      unnamed       unused   input  active-high
        line   9:      unnamed       unused   input  active-high
        line  10:      unnamed       unused   input  active-high
        line  11:      unnamed       unused   input  active-high
        line  12:      unnamed       unused   input  active-high
        line  13:      unnamed       unused   input  active-high
        line  14:      unnamed       unused   input  active-high
        line  15:      unnamed       unused   input  active-high
        line  16:      unnamed       unused   input  active-high
        line  17:      unnamed       unused   input  active-high
        line  18:      unnamed       unused   input  active-high
        line  19:      unnamed       unused   input  active-high
        line  20:      unnamed    "klipper"  output  active-high [used]
        line  21:      unnamed       unused   input  active-high
        line  22:      unnamed       unused   input  active-high
        line  23:      unnamed       unused   input  active-high
        line  24:      unnamed       unused   input  active-high
        line  25:      unnamed       unused   input  active-high
        line  26:      unnamed       unused   input  active-high
        line  27:      unnamed       unused   input  active-high
        line  28:      unnamed       unused   input  active-high
        line  29:      unnamed       "led0"  output  active-high [used]
        line  30:      unnamed       unused   input  active-high
        line  31:      unnamed       unused   input  active-high
        line  32:      unnamed       unused   input  active-high
        line  33:      unnamed       unused   input  active-high
        line  34:      unnamed       unused   input  active-high
        line  35:      unnamed       unused   input  active-high
        line  36:      unnamed       unused   input  active-high
        line  37:      unnamed       unused   input  active-high
        line  38:      unnamed       unused   input  active-high
        line  39:      unnamed       unused   input  active-high
        line  40:      unnamed       unused   input  active-high
        line  41:      unnamed       unused   input  active-high
        line  42:      unnamed       unused   input  active-high
        line  43:      unnamed       unused   input  active-high
        line  44:      unnamed       unused   input  active-high
        line  45:      unnamed       unused   input  active-high
        line  46:      unnamed       unused   input  active-high
        line  47:      unnamed       unused   input  active-high
        line  48:      unnamed       unused   input  active-high
        line  49:      unnamed       unused   input  active-high
        line  50:      unnamed       unused   input  active-high
        line  51:      unnamed       unused   input  active-high
        line  52:      unnamed       unused   input  active-high
```

```
        line  53:      unnamed       unused   input  active-high
gpiochip1 - 8 lines:
        line   0:      unnamed       unused   input  active-high
        line   1:      unnamed       unused   input  active-high
        line   2:      unnamed       "led1"  output   active-low [used]
        line   3:      unnamed       unused   input  active-high
        line   4:      unnamed       unused   input  active-high
        line   5:      unnamed       unused   input  active-high
        line   6:      unnamed       unused   input  active-high
        line   7:      unnamed       unused   input  active-high
```

Optional: Hardware-PWM

Raspberry Pi's haben zwei PWM-Kanäle (PWM0 und PWM1), die auf dem Header freigelegt sind oder, falls nicht, auf vorhandene gpio-Pins geroutet werden können. Der Linux mcu-Daemon verwendet die pwmchip sysfs-Schnittstelle, um Hardware-PWM-Geräte auf Linux-Hosts zu steuern. Die pwm sysfs-Schnittstelle ist auf einem Raspberry standardmäßig nicht verfügbar und kann durch Hinzufügen einer Zeile in `/boot/config.txt` aktiviert werden:

```
# Enable pwmchip sysfs interface
dtoverlay=pwm,pin=12,func=4
```

Dieses Beispiel aktiviert nur PWM0 und leitet es an gpio12. Wenn beide PWM-Kanäle aktiviert werden sollen, können Sie `pwm-2chan` verwenden.

Das Overlay gibt die pwm-Zeile im sysfs beim Booten nicht frei und muss durch echo'ing der Nummer des pwm-Kanals nach `/sys/class/pwm/pwmchip0/export` exportiert werden:

```
echo 0 > /sys/class/pwm/pwmchip0/export
```

Dadurch wird das `Gerät /sys/class/pwm/pwmchip0/pwm0` im Dateisystem angelegt. Am einfachsten geht das, indem Sie dies in `/etc/rc.local` vor der Zeile `exit 0` einfügen.

Wenn Sie das sysfs eingerichtet haben, können Sie den/die pwm-Kanal/Kanäle verwenden, indem Sie die folgende Konfiguration zu Ihrer `printer.cfg` hinzufügen:

```
[output_pin caselight]
pin: host:pwmchip0/pwm0
pwm: True
hardware_pwm: True
cycle_time: 0.000001
```

Dies fügt die Hardware-PWM-Steuerung zu gpio12 auf dem Pi hinzu (weil das Overlay so konfiguriert wurde, dass pwm0 zu pin=12 geleitet wird).

PWM0 kann auf gpio12 und gpio18 geroutet werden, PWM1 kann auf gpio13 und gpio19 geroutet werden:
PWM gpio PIN Func

PWM	gpio PIN	Func
0	12	4
0	18	2
1	13	4
1	19	2

Beaglebone

Dieses Dokument beschreibt den Prozess, wie Klipper auf einer Beaglebone PRU läuft.

Erstellen eines Betriebssystem-Images

Beginnen Sie mit der Installation des Debian 9.9 2019-08-03 4GB SD IoT Images. Man kann das Image entweder von einer micro-SD-Karte oder von der eingebauten eMMC ausführen. Wenn Sie die eMMC verwenden, installieren Sie es jetzt auf die eMMC, indem Sie den Anweisungen unter dem obigen Link folgen.

Dann loggt man sich per ssh auf dem Beaglebone ein (`ssh debian@beaglebone` -- Passwort ist `temppwd`) und installiert Klipper mit den folgenden Befehlen:

```
git clone https://github.com/Klipper3d/klipper
./klipper/scripts/install-beaglebone.sh
```

Octoprint installieren

Anschließend kann man Octoprint installieren:

```
git clone https://github.com/foosel/OctoPrint.git
cd OctoPrint/
virtualenv venv
./venv/bin/python setup.py install
```

Und richten Sie OctoPrint so ein, dass er beim Hochfahren startet:

```
sudo cp ~/OctoPrint/scripts/octoprint.init /etc/init.d/octoprint
sudo chmod +x /etc/init.d/octoprint
sudo cp ~/OctoPrint/scripts/octoprint.default /etc/default/octoprint
sudo update-rc.d octoprint defaults
```

Es ist notwendig, die OctoPrint-Konfigurationsdatei /etc/default/octoprint zu ändern. Man muss den Benutzer `OCTOPRINT_USER` in `debian` ändern, `NICELEVEL` auf `0` setzen, die Einstellungen `BASEDIR, CONFIGFILE` und `DAEMON` auskommentieren und die Verweise von `/home/pi/ auf /home/debian/` ändern:

```
sudo nano /etc/default/octoprint

Starten Sie dann den Octoprint-Dienst:

sudo systemctl start octoprint
```

Vergewissern Sie sich, dass der OctoPrint-Webserver erreichbar ist - er sollte sich unter http://beaglebone:5000/.

Erstellen des Mikrocontroller-Codes

Um den Klipper-Mikrocontroller-Code zu kompilieren, muss er zunächst für die "Beaglebone PRU" konfiguriert werden:

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

Um den neuen Mikrocontroller-Code zu bauen und zu installieren, führe aus:

```
sudo service klipper stop
make flash
sudo service klipper start
```

Es ist auch notwendig, den Mikrocontroller-Code für einen Linux-Host-Prozess zu kompilieren und zu installieren. Konfigurieren Sie ihn ein zweites Mal für einen "Linux-Prozess":

```
make menuconfig
```

Installieren Sie dann auch diesen Mikrocontroller-Code:

```
sudo service klipper stop
make flash
sudo dienst klipper start
```

Verbleibende Konfiguration

Schließen Sie die Installation ab, indem Sie Klipper und Octoprint gemäß den Anweisungen im Hauptinstallationsdokument Installation konfigurieren.

Drucken auf dem Beaglebone

Leider kann der Beaglebone-Prozessor manchmal Schwierigkeiten haben, OctoPrint gut auszuführen. Es ist bekannt, dass es bei komplexen Ausdrucken zu einem Druckstopp kommt (der Drucker bewegt sich möglicherweise schneller, als OctoPrint Bewegungsbefehle senden kann). In diesem Fall sollten Sie die Funktion "virtual_sdcard" (siehe Config Reference für Details) verwenden, um direkt von Klipper aus zu drucken.

Bootloader

Dieses Dokument enthält Informationen über gängige Bootloader auf Mikrocontrollern, die von Klipper unterstützt werden.

Der Bootloader ist eine Software eines Drittanbieters, die auf dem Mikrocontroller läuft, wenn dieser das erste Mal eingeschaltet wird. Er wird üblicherweise verwendet, um eine neue Anwendung (z.B. Klipper) auf den Mikrocontroller zu flashen, ohne dass spezielle Hardware benötigt wird.

Leider gibt es weder einen industrieweiten Standard für das Flashen eines Mikrocontrollers, noch einen Standard-Bootloader, der für alle Mikrocontroller funktioniert. Schlimmer noch, jeder Bootloader erfordert in der Regel eine andere Anzahl von Schritten, um eine Anwendung zu flashen.

Wenn man einen Bootloader in einen Mikrocontroller flashen kann, dann kann man diesen Mechanismus im Allgemeinen auch zum Flashen einer Anwendung verwenden, aber dabei ist Vorsicht geboten, da man den Bootloader versehentlich entfernen könnte. Im Gegensatz dazu erlaubt ein Bootloader dem Benutzer im Allgemeinen nur das Flashen einer Anwendung. Es wird daher empfohlen, zum Flashen einer Anwendung nach Möglichkeit einen Bootloader zu verwenden.

In diesem Dokument wird versucht, gängige Bootloader, die zum Flashen eines Bootloaders erforderlichen Schritte und die zum Flashen einer Anwendung erforderlichen Schritte zu beschreiben. Dieses Dokument ist keine verbindliche Referenz; es ist als eine Sammlung nützlicher Informationen gedacht, die die Klipper-Entwickler gesammelt haben.

AVR-Mikrocontroller

Im Allgemeinen ist das Arduino-Projekt eine gute Referenz für Bootloader und Flash-Verfahren auf den 8-Bit Atmel Atmega-Mikrocontrollern. Insbesondere die Datei https://github.com/arduino/Arduino/blob/1.8.5/hardware/arduino/avr/boards.txt ist eine nützliche Referenz.

Um einen Bootloader selbst zu flashen, benötigen die AVR-Chips ein externes Hardware-Flashing-Tool (das mit dem Chip über SPI kommuniziert). Dieses Tool ist käuflich zu erwerben (z. B. durch eine Websuche nach "avr isp", "arduino isp" oder "usb tiny isp"). Es ist auch möglich, einen anderen Arduino oder Raspberry Pi zu verwenden, um einen AVR-Bootloader zu flashen (führen Sie z. B. eine Websuche nach "program an avr using raspberry pi" durch). Die folgenden Beispiele gehen davon aus, dass ein Gerät vom Typ "AVR ISP Mk2" verwendet wird.

Das Programm "avrdude" ist das gebräuchlichste Tool zum Flashen von Atmega-Chips (sowohl zum Flashen des Bootloaders als auch zum Flashen von Anwendungen).

Atmega2560

Dieser Chip ist typischerweise im "Arduino Mega" zu finden und wird häufig in 3D-Drucker-Boards verwendet.

Um den Bootloader selbst zu flashen, verwenden Sie etwas wie:

```
wget
'https://github.com/arduino/Arduino/raw/1.8.5/hardware/arduino/avr/bootloaders/stk500v2/stk500boot_v
2_mega2560.hex'

avrdude -cavrispv2 -patmega2560 -P/dev/ttyACM0 -b115200 -e -u -U lock:w:0x3F:m -U efuse:w:0xFD:m -U
hfuse:w:0xD8:m -U lfuse:w:0xFF:m
avrdude -cavrispv2 -patmega2560 -P/dev/ttyACM0 -b115200 -U flash:w:stk500boot_v2_mega2560.hex
avrdude -cavrispv2 -patmega2560 -P/dev/ttyACM0 -b115200 -U lock:w:0x0F:m
```

Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
avrdude -cwiring -patmega2560 -P/dev/ttyACM0 -b115200 -D -Uflash:w:out/klipper.elf.hex:i
```

Atmega1280

Dieser Chip ist typischerweise in früheren Versionen des "Arduino Mega" zu finden.

Um den Bootloader selbst zu flashen, benutze etwas wie:

```
wget
'https://github.com/arduino/Arduino/raw/1.8.5/hardware/arduino/avr/bootloaders/atmega/ATmegaBOOT_168
_atmega1280.hex'

avrdude -cavrispv2 -patmega1280 -P/dev/ttyACM0 -b115200 -e -u -U lock:w:0x3F:m -U efuse:w:0xF5:m -U
hfuse:w:0xDA:m -U lfuse:w:0xFF:m
avrdude -cavrispv2 -patmega1280 -P/dev/ttyACM0 -b115200 -U flash:w:ATmegaBOOT_168_atmega1280.hex
avrdude -cavrispv2 -patmega1280 -P/dev/ttyACM0 -b115200 -U lock:w:0x0F:m
```

Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
avrdude -carduino -patmega1280 -P/dev/ttyACM0 -b57600 -D -Uflash:w:out/klipper.elf.hex:i
```

Atmega1284p

Dieser Chip wird häufig in 3d-Drucker-Boards im "Melzi"-Stil verwendet.

Um den Bootloader selbst zu flashen, benutze etwas wie:

```
wget 'https://github.com/Lauszus/Sanguino/raw/1.0.2/bootloaders/optiboot/optiboot_atmega1284p.hex'
```

```
avrdude -cavrispv2 -patmega1284p -P/dev/ttyACM0 -b115200 -e -u -U lock:w:0x3F:m -U efuse:w:0xFD:m -U
hfuse:w:0xDE:m -U lfuse:w:0xFF:m
avrdude -cavrispv2 -patmega1284p -P/dev/ttyACM0 -b115200 -U flash:w:optiboot_atmega1284p.hex
avrdude -cavrispv2 -patmega1284p -P/dev/ttyACM0 -b115200 -U lock:w:0x0F:m
```

Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
avrdude -carduino -patmega1284p -P/dev/ttyACM0 -b115200 -D -Uflash:w:out/klipper.elf.hex:i
```

Beachten Sie, dass eine Reihe von "Melzi"-Boards mit einem Bootloader vorinstalliert sind, der eine Baudrate von 57600 verwendet. Verwenden Sie in diesem Fall zum Flashen einer Anwendung stattdessen etwas wie dieses:

```
avrdude -carduino -patmega1284p -P/dev/ttyACM0 -b57600 -D -Uflash:w:out/klipper.elf.hex:i
```

At90usb1286

Dieses Dokument behandelt weder die Methode zum Flashen eines Bootloaders auf den At90usb1286 noch das allgemeine Flashen von Anwendungen auf dieses Gerät.

Das Teensy++ Gerät von pjrc.com kommt mit einem proprietären Bootloader. Er erfordert ein benutzerdefiniertes Flashtool von https://github.com/PaulStoffregen/teensy_loader_cli. Man kann eine Anwendung damit flashen, indem man etwas wie:

```
teensy_loader_cli --mcu=at90usb1286 out/klipper.elf.hex -v
```

Atmega168

Der atmega168 hat nur begrenzten Flash-Speicherplatz. Wenn Sie einen Bootloader verwenden, wird empfohlen, den Optiboot-Bootloader zu benutzen. Um diesen Bootloader zu flashen, verwenden Sie etwas wie:

```
wget
'https://github.com/arduino/Arduino/raw/1.8.5/hardware/arduino/avr/bootloaders/optiboot/optiboot_atm
ega168.hex'

avrdude -cavrispv2 -patmega168 -P/dev/ttyACM0 -b115200 -e -u -U lock:w:0x3F:m -U efuse:w:0x04:m -U
hfuse:w:0xDD:m -U lfuse:w:0xFF:m
avrdude -cavrispv2 -patmega168 -P/dev/ttyACM0 -b115200 -U flash:w:optiboot_atmega168.hex
avrdude -cavrispv2 -patmega168 -P/dev/ttyACM0 -b115200 -U lock:w:0x0F:m
```

Um eine Anwendung über den Optiboot-Bootloader zu flashen, verwenden Sie etwas wie:

```
avrdude -carduino -patmega168 -P/dev/ttyACM0 -b115200 -D -Uflash:w:out/klipper.elf.hex:i
```

SAM3-Mikrocontroller (Arduino Due)

Es ist nicht üblich, einen Bootloader mit dem SAM3-Mikrocontroller zu verwenden. Der Chip selbst verfügt über ein ROM, das es ermöglicht, das Flash über den seriellen 3,3-V-Anschluss oder über USB zu programmieren.

Um das ROM zu aktivieren, wird der "Erase"-Pin während eines Resets auf High gehalten, was den Flash-Inhalt löscht und das ROM zum Laufen bringt. Bei einem Arduino Due kann diese Sequenz durch Einstellen einer Baudrate von 1200 am "programming usb port" (der USB-Anschluss, der der Stromversorgung am nächsten liegt) erreicht werden.

Der Code auf https://github.com/shumatech/BOSSA kann zur Programmierung des SAM3 verwendet werden. Es wird empfohlen, die Version 1.9 oder höher zu verwenden.

Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
bossac -U -p /dev/ttyACM0 -a -e -w out/klipper.bin -v -b
bossac -U -p /dev/ttyACM0 -R
```

SAM4-Mikrocontroller (Duet Wifi)

Es ist nicht üblich, einen Bootloader mit dem SAM4-Mikrocontroller zu verwenden. Der Chip selbst verfügt über ein ROM, das es ermöglicht, das Flash über den seriellen 3,3-V-Anschluss oder über USB zu programmieren.

Um das ROM zu aktivieren, wird der "Erase"-Pin während eines Resets auf High gehalten, was den Flash-Inhalt löscht und das ROM zum Laufen bringt.

Der Code auf https://github.com/shumatech/BOSSA kann zur Programmierung des SAM4 verwendet werden. Es ist notwendig, die Version 1.8.0 oder höher zu verwenden.

Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
bossac --port=/dev/ttyACM0 -b -U -e -w -v -R out/klipper.bin
```

SAMD21-Mikrocontroller (Arduino Zero)

Der SAMD21-Bootloader wird über die ARM Serial Wire Debug (SWD) Schnittstelle geflasht. Dies wird üblicherweise mit einem speziellen SWD-Hardware-Dongle durchgeführt. Alternativ kann man auch einen Raspberry Pi with OpenOCD verwenden.

Um einen Bootloader mit OpenOCD zu flashen, verwenden Sie die folgende Chipkonfiguration:

```
source [find target/at91samdXX.cfg]
```

Besorgen Sie sich einen Bootloader - zum Beispiel:

```
wget 'https://github.com/arduino/ArduinoCore-samd/raw/1.8.3/bootloaders/zero/samd21_sam_ba.bin'
```

Flashen Sie mit OpenOCD Befehle ähnlich wie:

```
at91samd bootloader 0
Programm samd21_sam_ba.bin verify
```

Der gebräuchlichste Bootloader für den SAMD21 ist der des "Arduino Zero". Er verwendet einen 8KiB-Bootloader (die Anwendung muss mit einer Startadresse von 8KiB kompiliert werden). Man kann diesen Bootloader durch einen Doppelklick auf den Reset-Knopf aufrufen. Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
bossac -U -p /dev/ttyACM0 --offset=0x2000 -w out/klipper.bin -v -b -R
```

Im Gegensatz dazu verwendet der "Arduino M0" einen 16KiB Bootloader (die Anwendung muss mit einer Startadresse von 16KiB kompiliert werden). Um eine Anwendung auf diesen Bootloader zu flashen, setzen Sie den Mikrocontroller zurück und führen den Flash-Befehl innerhalb der ersten Sekunden nach dem Booten aus - etwa so:

```
avrdude -c stk500v2 -p atmega2560 -P /dev/ttyACM0 -u -Uflash:w:out/klipper.elf.hex:i
```

SAMD51 Mikrocontroller (Adafruit Metro-M4 und ähnliche)

Wie der SAMD21 wird der SAMD51 Bootloader über die ARM Serial Wire Debug (SWD) Schnittstelle geflasht. Um einen Bootloader mit OpenOCD on a Raspberry Pi zu flashen, verwenden Sie die folgende Chipkonfiguration:

```
source [find target/atsame5x.cfg]
```

Besorgen Sie sich einen Bootloader - mehrere Bootloader sind unter https://github.com/adafruit/uf2-samdx1/releases/latest erhältlich. Zum Beispiel:

```
wget 'https://github.com/adafruit/uf2-samdx1/releases/download/v3.7.0/bootloader-itsybitsy_m4-
v3.7.0.bin'
```

Flashen Sie mit OpenOCD-Befehlen ähnlich wie:

```
at91samd bootloader 0
program bootloader-itsybitsy_m4-v3.7.0.bin verify
at91samd bootloader 16384
```

Der SAMD51 verwendet einen 16KiB-Bootloader (die Anwendung muss mit einer Startadresse von 16KiB kompiliert werden). Um eine Anwendung zu flashen, verwenden Sie etwas wie:

```
bossac -U -p /dev/ttyACM0 --offset=0x4000 -w out/klipper.bin -v -b -R
```

STM32F103-Mikrocontroller (Blue Pill-Bausteine)

Die STM32F103-Bausteine verfügen über ein ROM, das einen Bootloader oder eine Anwendung über eine serielle 3,3-V-Schnittstelle flashen kann. Normalerweise würde man die Pins PA10 (MCU Rx) und PA9 (MCU Tx) mit einem 3,3-V-UART-Adapter verbinden. Um auf das ROM zuzugreifen, sollte man den "boot 0"-Pin auf high und den "boot 1"-Pin auf low schalten und dann das Gerät zurücksetzen. Das "stm32flash"-Paket kann dann verwendet werden, um das Gerät zu flashen, z.B. mit

```
stm32flash -w out/klipper.bin -v -g 0 /dev/ttyAMA0
```

Beachten Sie, dass das stm32flash-Protokoll, wenn Sie einen Raspberry Pi für die serielle 3,3V-Verbindung verwenden, einen seriellen Paritätsmodus verwendet, den der "Mini UART" des Raspberry Pi nicht unterstützt. Siehe

https://www.raspberrypi.com/documentation/computers/configuration.html#configuring-uarts für Details zur Aktivierung des vollen UART an den GPIO-Pins des Raspberry Pi.

Setzen Sie nach dem Flashen sowohl "boot 0" als auch "boot 1" wieder auf low, damit zukünftige Resets vom Flash booten.

STM32F103 mit stm32duino-Bootloader

Das "stm32duino"-Projekt hat einen USB-fähigen Bootloader - siehe: https://github.com/rogerclarkmelbourne/STM32duino-bootloader

Dieser Bootloader kann über 3,3V seriell geflasht werden mit etwas wie:

```
wget 'https://github.com/rogerclarkmelbourne/STM32duino-
bootloader/raw/master/binaries/generic_boot20_pc13.bin'

stm32flash -w generic_boot20_pc13.bin -v -g 0 /dev/ttyAMA0
```

Dieser Bootloader benötigt 8KiB Flash-Speicher (die Anwendung muss mit einer Startadresse von 8KiB kompiliert werden). Flashen Sie eine Anwendung mit etwas wie:

```
dfu-util -d 1eaf:0003 -a 2 -R -D out/klipper.bin
```

Der Bootloader läuft in der Regel nur für eine kurze Zeit nach dem Booten. Es kann notwendig sein, den obigen Befehl so zu timen, dass er ausgeführt wird, während der Bootloader noch aktiv ist (der Bootloader lässt eine Board-LED blinken, während er läuft). Alternativ kann man den "boot 0"-Pin auf low und den "boot 1"-Pin auf high setzen, um nach einem Reset im Bootloader zu bleiben.

STM32F103 mit HID-Bootloader

Der HID bootloader ist ein kompakter, treiberloser Bootloader, der über USB flashen kann. Es gibt auch einen fork with builds specific to the SKR Mini E3 1.2.

Für generische STM32F103-Boards wie die blaue Pille ist es möglich, den Bootloader über 3,3 V seriell mit stm32flash zu flashen, wie im obigen Abschnitt über stm32duino beschrieben, wobei der Dateiname durch die gewünschte HID-Bootloader-Binärdatei zu ersetzen ist (z. B.: hid_generic_pc13.bin für die blaue Pille).

Es ist nicht möglich, stm32flash für den SKR Mini E3 zu verwenden, da der boot0-Pin direkt an Masse gebunden ist und nicht über Header-Pins herausgebrochen wird. Es wird empfohlen, einen STLink V2 mit STM32Cubeprogrammer zu verwenden, um den Bootloader zu flashen. Wenn Sie keinen Zugang zu einem STLink haben, können Sie auch einen Raspberry Pi and OpenOCD mit der folgenden Chipkonfiguration verwenden:

```
source [find target/stm32f1x.cfg]
```

Wenn Sie möchten, können Sie mit dem folgenden Befehl ein Backup des aktuellen Flash erstellen. Beachten Sie, dass dies einige Zeit in Anspruch nehmen kann:

```
flash read_bank 0 btt_skr_mini_e3_backup.bin
```

Schließlich können Sie mit Befehlen wie dem folgenden flashen:

```
stm32f1x mass_erase 0
program hid_btt_skr_mini_e3.bin verify 0x08000000
```

HINWEISE:

- Das obige Beispiel löscht den Chip und programmiert dann den Bootloader. Unabhängig von der gewählten Methode zum Flashen wird empfohlen, den Chip vor dem Flashen zu löschen.
- Bevor Sie den SKR Mini E3 mit diesem Bootloader flashen, sollten Sie beachten, dass Sie die Firmware nicht mehr über die SD-Karte aktualisieren können.
- Eventuell müssen Sie den Reset-Knopf auf dem Board gedrückt halten, während Sie OpenOCD starten. Es sollte etwas angezeigt werden wie:
  ```
  Open On-Chip Debugger 0.10.0+dev-01204-gc60252ac-dirty (2020-04-27-16:00) Lizenziert unter GNU GPL v2 Für Fehlerberichte, lesen Sie http://openocd.org/doc/doxygen/bugs.html DEPRECATED! use 'adapter speed' not 'adapter_khz' Info : BCM2835 GPIO JTAG/SWD bitbang driver Info : JTAG and SWD modes enabled Info : clock speed 40 kHz Info : SWD DPIDR 0x1ba01477 Info : stm32f1x. cpu: hardware has 6 breakpoints, 4 watchpoints Info : stm32f1x.cpu: external reset detected Info : starting gdb server for stm32f1x.cpu on 3333 Info : Listening on port 3333 for gdb connections
  ```
  Danach können Sie den Reset-Knopf loslassen.

Dieser Bootloader benötigt 2KiB Flash-Speicherplatz (die Anwendung muss mit einer Startadresse von 2KiB kompiliert werden).

Das Programm hid-flash wird verwendet, um ein Binary in den Bootloader zu laden. Sie können diese Software mit den folgenden Befehlen installieren:

```
sudo apt install libusb-1.0
cd ~/klipper/lib/hidflash
make
```

Wenn der Bootloader läuft, kann man mit folgendem Befehl flashen:

```
~/klipper/lib/hidflash/hid-flash ~/klipper/out/klipper.bin
```

alternativ kann man auch make flash verwenden, um klipper direkt zu flashen:

```
make flash FLASH_DEVICE=1209:BEBA
```

ODER wenn klipper bereits geflasht wurde:

```
make flash FLASH_DEVICE=/dev/ttyACM0
```

Es kann notwendig sein, den Bootloader manuell zu starten, indem man "boot 0" niedrig und "boot 1" hoch setzt. Auf dem SKR Mini E3 ist "Boot 1" nicht verfügbar, daher kann dies durch Setzen von Pin PA2 auf low geschehen, wenn Sie "hid_btt_skr_mini_e3.bin" geflasht haben. Dieser Pin ist auf dem TFT-Header im "PIN"-Dokument des SKR Mini E3 mit "TX0" bezeichnet. Neben PA2 befindet sich ein Masse-Pin, mit dem Sie PA2 auf low ziehen können.

STM32F103/STM32F072 mit MSC-Bootloader

Der MSC bootloader ist ein treiberloser Bootloader, der über USB flashen kann.

Es ist möglich, den Bootloader über 3,3 V seriell mit stm32flash zu flashen, wie im obigen Abschnitt über stm32duino beschrieben, wobei der Dateiname durch die gewünschte MSC-Bootloader-Binärdatei zu ersetzen ist (z. B.: MSCboot-Bluepill.bin für die blaue Pille).

Bei STM32F072-Boards ist es auch möglich, den Bootloader über USB zu flashen (via DFU), z.B. mit:

```
dfu-util -d 0483:df11 -a 0 -R -D MSCboot-STM32F072.bin -s0x08000000:leave
```

Dieser Bootloader verwendet 8KiB oder 16KiB Flash-Speicherplatz, siehe Beschreibung des Bootloaders (die Anwendung muss mit der entsprechenden Startadresse kompiliert werden).

Der Bootloader kann durch zweimaliges Drücken des Reset-Knopfes auf dem Board aktiviert werden. Sobald der Bootloader aktiviert ist, erscheint das Board als USB-Stick, auf den die Datei klipper.bin kopiert werden kann.

STM32F103/STM32F0x2 mit CanBoot-Bootloader

Der CanBoot Bootloader bietet die Möglichkeit, Klipper-Firmware über den CANBUS hochzuladen. Der Bootloader selbst ist vom Klipper-Quellcode abgeleitet. Derzeit unterstützt CanBoot die Modelle STM32F103, STM32F042 und STM32F072.

Es wird empfohlen, einen ST-Link Programmer zum Flashen von CanBoot zu verwenden, es sollte jedoch möglich sein, `stm32flash` auf STM32F103-Bausteinen und `dfu-util` auf STM32F042/STM32F072-Bausteinen zu verwenden. Siehe die vorherigen Abschnitte in diesem Dokument für Anweisungen zu diesen Flash-Methoden, wobei der Dateiname gegebenenfalls durch `canboot.bin` ersetzt wird. Das oben verlinkte CanBoot-Repositorium enthält Anweisungen zur Erstellung des Bootloaders.

Wenn CanBoot zum ersten Mal geflasht wird, sollte es feststellen, dass keine Anwendung vorhanden ist und den Bootloader aufrufen. Sollte dies nicht der Fall sein, kann der Bootloader durch zweimaliges Drücken der Reset-Taste aufgerufen werden.

Das Dienstprogramm `flash_can.py`, das im Ordner `lib/canboot` enthalten ist, kann zum Hochladen der Klipper-Firmware verwendet werden. Zum Flashen ist die UUID des Geräts erforderlich. Wenn Sie keine UUID haben, ist es möglich, Knoten abzufragen, auf denen der Bootloader gerade läuft:

```
python3 flash_can.py -q
```

Dies gibt die UUIDs aller angeschlossenen Knoten zurück, denen derzeit keine UUID zugewiesen ist. Dies sollte alle Knoten umfassen, die sich derzeit im Bootloader befinden.

Sobald Sie eine UUID haben, können Sie die Firmware mit dem folgenden Befehl hochladen:

```
python3 flash_can.py -i can0 -f ~/klipper/out/klipper.bin -u aabbccddeeff
```

Wobei `aabbccddeeff` durch Ihre UUID ersetzt wird. Beachten Sie, dass die Optionen `-i` und `-f` weggelassen werden können, sie sind standardmäßig auf `can0` bzw. `~/klipper/out/klipper.bin` eingestellt.

Wenn Sie Klipper für die Verwendung mit CanBoot bauen, wählen Sie die Option 8 KiB Bootloader.

STM32F4-Mikrocontroller (SKR Pro 1.1)

STM32F4-Mikrocontroller sind mit einem eingebauten System-Bootloader ausgestattet, der über USB (via DFU), 3,3V Serial und verschiedene andere Methoden flashen kann (siehe STM-Dokument AN2606 für weitere Informationen). Einige STM32F4 Boards, wie z.B. das SKR Pro 1.1, sind nicht in der Lage, den DFU-Bootloader aufzurufen. Der HID-Bootloader ist für STM32F405/407-basierte Boards verfügbar, falls der Benutzer das Flashen über USB der Verwendung der SD-Karte vorzieht. Beachten Sie, dass Sie möglicherweise eine für Ihr Board spezifische Version konfigurieren und erstellen müssen. Ein build for the SKR Pro 1.1 is available here verfügbar.

Wenn Ihr Board nicht DFU-fähig ist, ist die am besten zugängliche Methode zum Flashen wahrscheinlich die serielle 3,3V-Verbindung, die dem gleichen Verfahren folgt wie das flashing the STM32F103 using stm32flash. Zum Beispiel:

```
wget https://github.com/Arksine/STM32_HID_Bootloader/releases/download/v0.5-
beta/hid_bootloader_SKR_PRO.bin

stm32flash -w hid_bootloader_SKR_PRO.bin -v -g 0 /dev/ttyAMA0
```

Dieser Bootloader benötigt 16Kib Flash-Speicherplatz auf dem STM32F4 (die Anwendung muss mit einer Startadresse von 16KiB kompiliert werden).

Wie beim STM32F1 verwendet der STM32F4 das Tool hid-flash, um Binärdateien auf die MCU zu laden. Details zur Erstellung und Verwendung von hid-flash finden Sie in der obigen Anleitung.

Es kann notwendig sein, den Bootloader manuell zu starten, indem Sie "boot 0" niedrig und "boot 1" hoch einstellen und das Gerät anschließen. Nach Abschluss der Programmierung das Gerät abstecken und "boot 1" wieder auf low setzen, damit die Anwendung geladen wird.

LPC176x-Mikrocontroller (Smoothieboards)

Dieses Dokument beschreibt nicht die Methode zum Flashen eines Bootloaders selbst - weitere Informationen zu diesem Thema finden Sie unter: http://smoothieware.org/flashing-the-bootloader.

Es ist üblich, dass Smoothieboards mit einem Bootloader von https://github.com/triffid/LPC17xx-DFU-Bootloader geliefert werden. Bei Verwendung dieses Bootloaders muss die Anwendung mit einer Startadresse von 16KiB kompiliert werden. Der einfachste Weg, eine Anwendung mit diesem Bootloader zu flashen, ist, die Anwendungsdatei (z.B. `out/klipper.bin`) in eine Datei namens `firmware.bin` auf einer SD-Karte zu kopieren und dann den Mikrocontroller mit dieser SD-Karte neu zu starten.

Ausführen von OpenOCD auf dem Raspberry PI

OpenOCD ist ein Softwarepaket, das Low-Level-Chip-Flashing und Debugging durchführen kann. Es kann die GPIO-Pins eines Raspberry Pi nutzen, um mit einer Vielzahl von ARM-Chips zu kommunizieren.

Dieser Abschnitt beschreibt, wie man OpenOCD installieren und starten kann. Er ist abgeleitet von den Anweisungen unter: https://learn.adafruit.com/programming-microcontrollers-using-openocd-on-raspberry-pi

Beginnen Sie mit dem Herunterladen und Kompilieren der Software (jeder Schritt kann mehrere Minuten dauern und der "make"-Schritt kann mehr als 30 Minuten in Anspruch nehmen):

```
sudo apt-get update
sudo apt-get install autoconf libtool telnet
mkdir ~/openocd
cd ~/openocd/
git clone http://openocd.zylin.com/openocd
cd openocd
./bootstrap
./configure --enable-sysfsgpio --enable-bcm2835gpio --prefix=/home/pi/openocd/install
make
make install
```

OpenOCD konfigurieren

Erstellen Sie eine OpenOCD-Konfigurationsdatei:

```
nano ~/openocd/openocd.cfg
```

Verwenden Sie eine Konfiguration ähnlich der folgenden:

```
# Uses RPi pins: GPIO25 for SWDCLK, GPIO24 for SWDIO, GPIO18 for nRST
source [find interface/raspberrypi2-native.cfg]
bcm2835gpio_swd_nums 25 24
bcm2835gpio_srst_num 18
```

```
transport select swd

# Use hardware reset wire for chip resets
reset_config srst_only
adapter_nsrst_delay 100
adapter_nsrst_assert_width 100

# Specify the chip type
source [find target/atsame5x.cfg]

# Set the adapter speed
adapter_khz 40

# Connect to chip
init
targets
reset halt
```

Verdrahten Sie den Raspberry Pi mit dem Zielchip
Schalten Sie sowohl den Raspberry Pi als auch den Zielchip vor der Verkabelung aus! Vergewissern Sie sich, dass der Zielchip 3,3 V verwendet, bevor Sie ihn mit dem Raspberry Pi verbinden!

Verbinden Sie GND, SWDCLK, SWDIO und RST auf dem Zielchip mit GND, GPIO25, GPIO24 bzw. GPIO18 auf dem Raspberry Pi.

Schalten Sie dann den Raspberry Pi ein und versorgen Sie den Zielchip mit Strom.

Starten Sie OpenOCD

Starten Sie OpenOCD:

```
cd ~/openocd/
sudo ~/openocd/install/bin/openocd -f ~/openocd/openocd.cfg
```

Dies sollte dazu führen, dass OpenOCD einige Textmeldungen ausgibt und dann wartet (es sollte nicht sofort zum Unix-Shell-Prompt zurückkehren). Wenn OpenOCD sich von selbst beendet oder weiterhin Textnachrichten ausgibt, überprüfen Sie die Verkabelung.

Sobald OpenOCD läuft und stabil ist, kann man ihm Befehle über Telnet schicken. Öffnen Sie eine weitere ssh-Sitzung und führen Sie Folgendes aus:

```
telnet 127.0.0.1 4444
```

(Man kann telnet beenden, indem man ctrl+] drückt und dann den Befehl "quit" ausführt).

OpenOCD und gdb

Es ist möglich, OpenOCD mit gdb zu benutzen, um Klipper zu debuggen. Die folgenden Befehle gehen davon aus, dass man gdb auf einer Maschine der Desktop-Klasse laufen lässt.

Füge das Folgende zur OpenOCD-Konfigurationsdatei hinzu:

```
bindto 0.0.0.0
gdb_port 44444
```

Starten Sie OpenOCD auf dem Raspberry Pi neu und führen Sie dann den folgenden Unix-Befehl auf dem Desktop-Rechner aus:

```
cd /pfad/zu/klipper/
gdb out/klipper.elf
```

Innerhalb von gdb ausführen:

```
target remote octopi:44444
```

(Ersetzen Sie "octopi" durch den Hostnamen des Raspberry Pi.) Sobald gdb läuft, können Sie Haltepunkte setzen und Register untersuchen.

CANBUS

Dieses Dokument beschreibt die CAN-Bus-Unterstützung von Klipper.

Geräte-Hardware

Klipper unterstützt derzeit nur CAN auf stm32 Chips. Außerdem muss der Mikrocontroller-Chip CAN unterstützen und er muss sich auf einem Board befinden, das einen CAN-Transceiver hat.

Um für CAN zu kompilieren, führen Sie make menuconfig aus und wählen Sie "CAN bus" als Kommunikationsschnittstelle. Schließlich kompilieren Sie den Mikrocontroller-Code und flashen ihn auf die Zielplatine.
Host-Hardware¶

Um einen CAN-Bus verwenden zu können, ist ein Host-Adapter erforderlich. Derzeit gibt es zwei gängige Optionen:

1. Verwenden Sie einen Waveshare Raspberry Pi CAN hat oder einen seiner vielen Klone.

2. Verwenden Sie einen USB-CAN-Adapter (zum Beispiel https://hacker-gadgets.com/product/cantact-usb-can-adapter/). Es gibt viele verschiedene USB-zu-CAN-Adapter - wenn Sie einen auswählen, empfehlen wir Ihnen zu überprüfen, ob er die candlelight firmware ausführen kann. (Leider haben wir festgestellt, dass einige USB-Adapter mit fehlerhafter Firmware laufen und gesperrt sind, daher sollten Sie sich vor dem Kauf vergewissern).

Es ist auch notwendig, das Host-Betriebssystem für die Verwendung des Adapters zu konfigurieren. Dies geschieht in der Regel durch Anlegen einer neuen Datei namens `/etc/network/interfaces.d/can0` mit dem folgenden Inhalt:

```
auto can0
iface can0 can statisch
    Bitrate 500000
    up ifconfig $IFACE txqueuelen 128
```

Beachten Sie, dass der "Raspberry Pi CAN hat" auch Änderungen an der changes to config.txt erfordert.

Abschlusswiderstände

Ein CAN-Bus sollte zwei 120-Ohm-Widerstände zwischen den CANH- und CANL-Leitungen haben. Idealerweise befindet sich jeweils ein Widerstand an den Enden des Busses.

Beachten Sie, dass einige Geräte einen eingebauten 120-Ohm-Widerstand haben (z. B. hat der "Waveshare Raspberry Pi CAN hat" einen angelöteten Widerstand, der nicht einfach entfernt werden kann). Einige Geräte enthalten überhaupt keinen Widerstand. Andere Geräte verfügen über einen Mechanismus zur Auswahl des Widerstands (in der Regel durch Verbinden eines "Pin-Jumpers"). Überprüfen Sie unbedingt die Schaltpläne aller Geräte am CAN-Bus, um sicherzustellen, dass sich zwei und nur zwei 120-Ohm-Widerstände auf dem Bus befinden.

Um zu prüfen, ob die Widerstände korrekt sind, kann man die Stromversorgung des Druckers unterbrechen und mit einem Multimeter den Widerstand zwischen den CANH- und CANL-Drähten messen - bei einem korrekt verdrahteten CAN-Bus sollte er ~60 Ohm anzeigen.

Ermitteln der canbus_uuid für neue Mikrocontroller

Jedem Mikrocontroller auf dem CAN-Bus wird eine eindeutige ID zugewiesen, die auf der werkseitigen Chip-Kennung basiert, die in jedem Mikrocontroller kodiert ist. Um die Geräte-ID jedes Mikrocontrollers zu finden, stellen Sie sicher, dass die Hardware mit Strom versorgt und korrekt verdrahtet ist, und führen Sie dann aus:

```
~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0
```

Wenn nicht initialisierte CAN-Geräte gefunden werden, wird der obige Befehl Zeilen wie die folgenden ausgeben:

```
found canbus_uuid=11aa22bb33cc
```

Jedes Gerät hat einen eindeutigen Bezeichner. Im obigen Beispiel ist `11aa22bb33cc` die "canbus_uuid" des Mikrocontrollers.

Beachte, dass das Werkzeug `canbus_query.py` nur uninitialisierte Geräte meldet - wenn Klipper (oder ein ähnliches Werkzeug) das Gerät konfiguriert, erscheint es nicht mehr in der Liste.

Klipper konfigurieren

Aktualisiere die Klipper mcu-Konfiguration, um den CAN-Bus zur Kommunikation mit dem Gerät zu verwenden - zum Beispiel:

```
[mcu my_can_mcu]
canbus_uuid: 11aa22bb33cc
```

TSL1401CL Filamentbreitensensor

Dieses Dokument beschreibt das Host-Modul Filament Width Sensor. Die für die Entwicklung dieses Hostmoduls verwendete Hardware basiert auf dem linearen Sensorarray TSL1401CL, kann aber mit jedem Sensorarray mit Analogausgang arbeiten. Sie können Designs auf Thingiverse finden.

Um ein Sensorarray als Fadenbreitensensor zu verwenden, lesen Sie die Config Referenceund die G-Code documentation.

Wie funktioniert es?

Der Sensor erzeugt einen analogen Ausgang basierend auf der berechneten Fadenbreite. Die Ausgangsspannung entspricht immer der ermittelten Filamentbreite (z.B. 1.65v, 1.70v, 3.0v). Das Host-Modul überwacht die Spannungsänderungen und passt den Extrusionsmultiplikator an.

Hinweis:

Der Sensor misst standardmäßig in 10 mm Abständen. Falls erforderlich, können Sie diese Einstellung durch Bearbeiten des Parameters MEASUREMENT_INTERVAL_MM in der Datei **filament_width_sensor.py** ändern.

Hall-Filamentbreitensensor

Dieses Dokument beschreibt das Host-Modul Filament Width Sensor. Die für die Entwicklung dieses Host-Moduls verwendete Hardware basiert auf zwei linearen Hall-Sensoren (z. B. ss49e). Die Sensoren im Gehäuse befinden sich auf gegenüberliegenden Seiten. Funktionsprinzip: zwei Hallsensoren arbeiten im Differenzmodus, Temperaturdrift für alle Sensoren gleich. Eine spezielle Temperaturkompensation ist nicht erforderlich.

Entwürfe finden Sie auf Thingiverse, ein Montagevideo ist auch auf Youtube verfügbar

Um den Hall-Filament-Breitensensor zu verwenden, lesen Sie die Config Reference und die G-Code documentation.

Wie funktioniert es?

Der Sensor erzeugt zwei analoge Ausgänge basierend auf der berechneten Filamentbreite. Die Summe der Ausgangsspannung ist immer gleich der ermittelten Filamentbreite. Das Host-Modul überwacht Spannungsänderungen und passt den Extrusionsmultiplikator an. Ich verwende den aux2-Anschluss auf der rampenähnlichen Platine mit den Pins analog11 und analog12. Sie können auch andere Pins und andere Platinen verwenden.

Vorlage für Menüvariablen

```
[menu __main __filament __width_current]
type: command
enable: {'hall_filament_width_sensor' in printer}
name: Dia: {'%.2F' % printer.hall_filament_width_sensor.Diameter}
index: 0

[menu __main __filament __raw_width_current]
type: command
enable: {'hall_filament_width_sensor' in printer}
name: Raw: {'%4.0F' % printer.hall_filament_width_sensor.Raw}
index: 1
```

Kalibrierungsverfahren

Um den Rohsensorwert zu erhalten, können Sie den Menüpunkt oder den Befehl QUERY_RAW_FILAMENT_WIDTH im Terminal verwenden.

Einsetzen des ersten Kalibrierstabs (1,5 mm), um den ersten Rohsensorwert zu erhalten

Zweiten Kalibrierstab einführen (Größe 2,0 mm), um den zweiten Rohsensorwert zu erhalten

Rohsensorwerte in den Konfigurationsparametern Raw_dia1 und Raw_dia2 speichern

So aktivieren Sie den Sensor

Standardmäßig ist der Sensor beim Einschalten deaktiviert.

Um den Sensor zu aktivieren, geben Sie den Befehl ENABLE_FILAMENT_WIDTH_SENSOR ein oder setzen Sie den Parameter `enable` auf `true`.

Protokollierung

Standardmäßig ist die Durchmesserprotokollierung beim Einschalten deaktiviert.

Geben Sie den Befehl **ENABLE_FILAMENT_WIDTH_LOG** ein, um die Protokollierung zu starten, und geben Sie den Befehl **DISABLE_FILAMENT_WIDTH_LOG** ein, um die Protokollierung zu beenden. Um die Protokollierung beim Einschalten zu aktivieren, setzen Sie den Parameter logging auf true.

Der Fadendurchmesser wird bei jedem Messintervall protokolliert (standardmäßig 10 mm).