

Voron Design

Klipper Installation

Written By: insurgus



INTRODUCTION

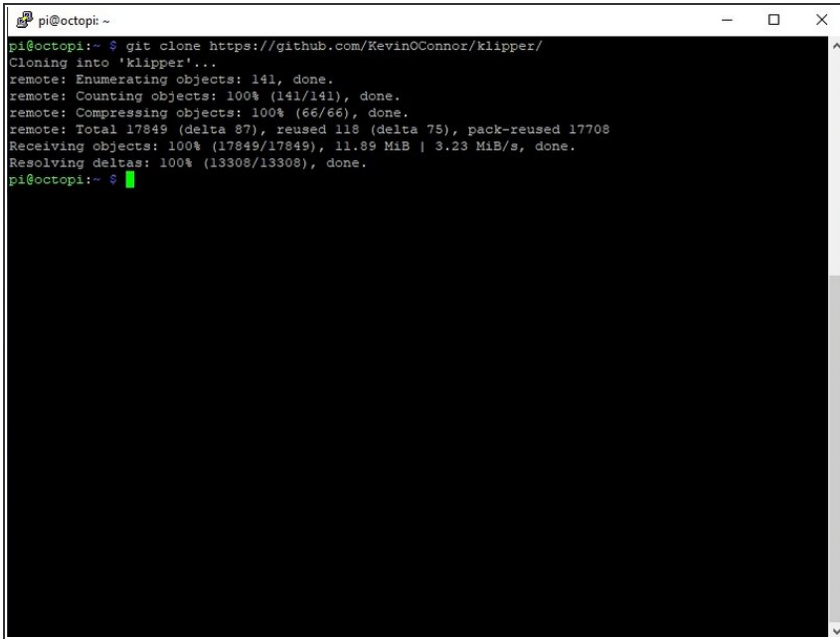
These instructions assume the software will run on a Raspberry Pi computer in conjunction with OctoPrint. It is recommended that a Raspberry Pi 2 or Raspberry Pi 3 computer be used as the host machine



PARTS:

- [Raspberry Pi 3 Model B +](#) (1)
 - [Micro SD Card](#) (1)
-

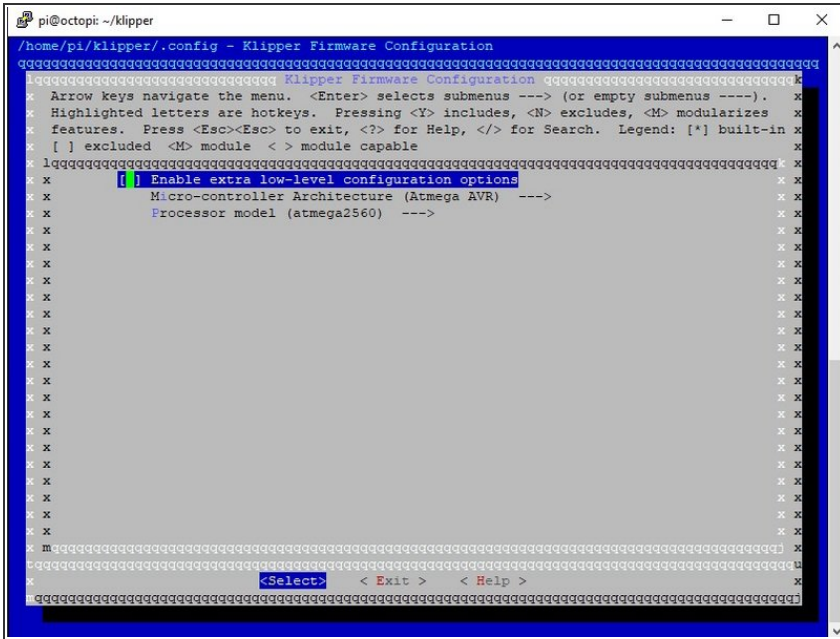
Step 1 — Prepping an OS image

A terminal window titled 'pi@octopi: ~' showing the execution of a git clone command. The output shows progress for cloning the repository 'klipper' from GitHub. The progress bars for enumerating, counting, and compressing objects are all at 100%. The total size of the repository is 17849 bytes, with 118 bytes reused from a previous clone. The download speed is 3.23 MiB/s.

```
pi@octopi:~$ git clone https://github.com/KevinOConnor/klipper/
Cloning into 'klipper'...
remote: Enumerating objects: 141, done.
remote: Counting objects: 100% (141/141), done.
remote: Compressing objects: 100% (66/66), done.
remote: Total 17849 (delta 87), reused 118 (delta 75), pack-reused 17708
Receiving objects: 100% (17849/17849), 11.89 MiB | 3.23 MiB/s, done.
Resolving deltas: 100% (13308/13308), done.
pi@octopi:~$
```

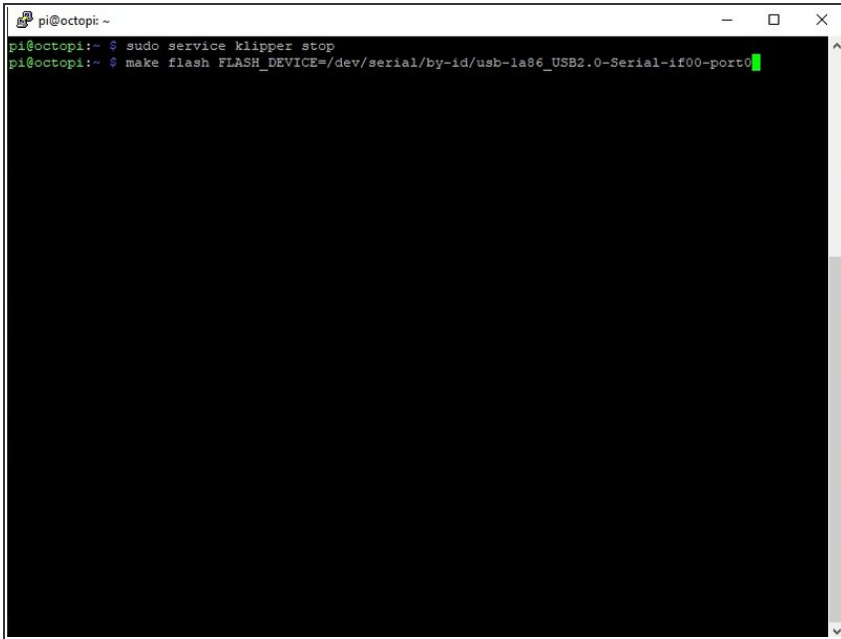
- Start by installing OctoPi on the Raspberry Pi computer. Use OctoPi v0.16.0 or later - see the octopi releases for release information. One should verify that OctoPi boots and that the OctoPrint web server works. After connecting to the OctoPrint web page, follow the prompt to upgrade OctoPrint to v1.3.12 or later.
- After installing OctoPi and upgrading OctoPrint, it will be necessary to ssh into the target machine to run a handful of system commands. There are free ssh clients available for other desktops (eg, PuTTY). Use the ssh utility to connect to the Raspberry Pi (ssh pi@octopi -- password is "raspberrry") and run the following commands:
 - git clone
<https://github.com/KevinOConnor/klipper/>
 - ./klipper/scripts/install-octopi.sh

Step 2 — Building the micro-controller



- To compile the micro-controller code, start by running these commands on the Raspberry Pi:
 - **cd ~/klipper/**
 - **make menuconfig**
- Select the appropriate micro-controller and review any other options provided. Once configured, run:
 - **make**
- It is necessary to determine the serial port connected to the micro-controller. For micro-controllers that connect via USB, run the following:
 - **ls /dev/serial/by-id/***
- It should report something similar to the following: **/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0**

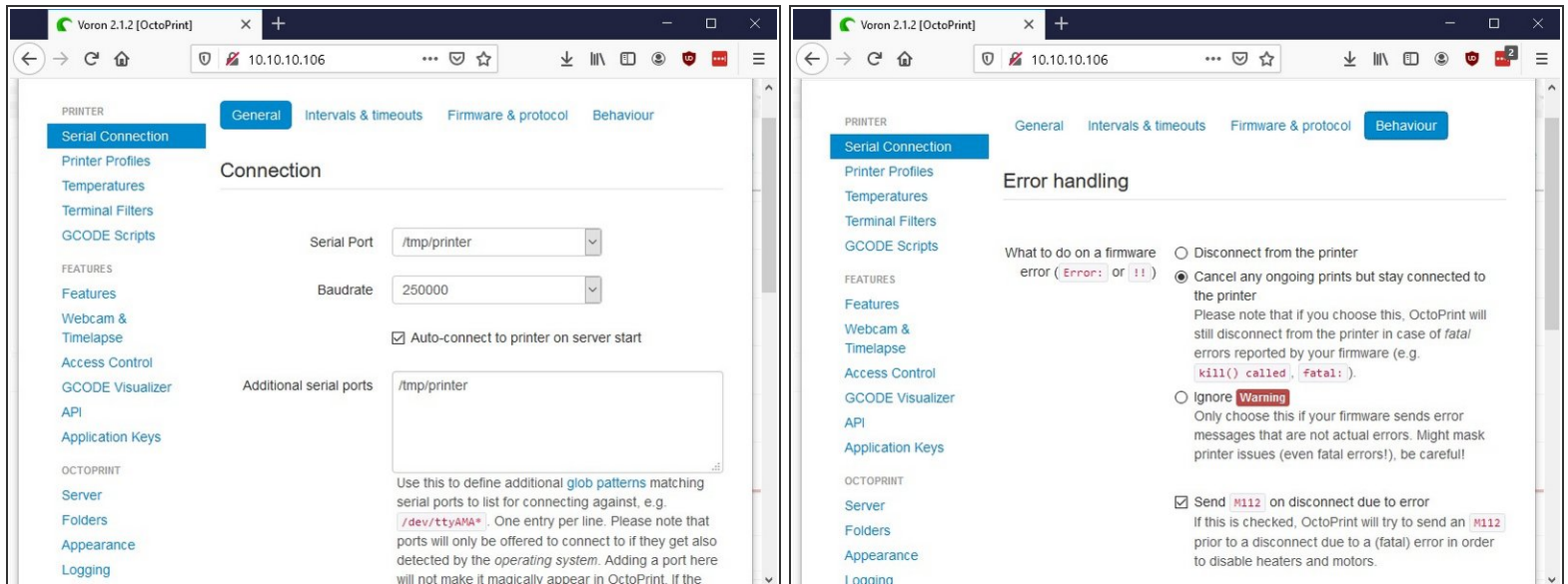
Step 3 — Flashing the micro-controller

A terminal window titled 'pi@octopi: ~' showing the execution of two commands. The first command is 'sudo service klipper stop' and the second is 'make flash FLASH_DEVICE=/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0'. The output of the first command is not visible, and the output of the second command is also not visible, suggesting the terminal output is being scrolled through. The cursor is at the end of the second command line.

```
pi@octopi: ~  
pi@octopi:~$ sudo service klipper stop  
pi@octopi:~$ make flash FLASH_DEVICE=/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
```

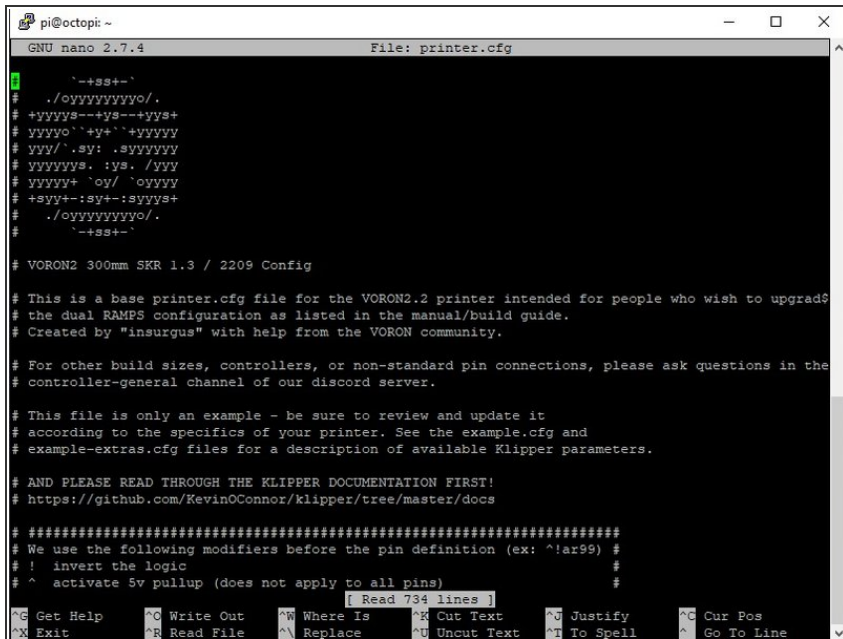
- It's common for each printer to have its own unique serial port name. This unique name will be used when flashing the micro-controller. It's possible there may be multiple lines in the above output - if so, choose the line corresponding to the micro-controller (see the FAQ for more information).
- For common micro-controllers, the code can be flashed with something similar to:
 - **sudo service klipper stop**
 - **make flash**
FLASH_DEVICE=/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0
 - **sudo service klipper start**
- Be sure to update the FLASH_DEVICE with the printer's unique serial port name. When flashing for the first time, make sure that OctoPrint is not connected directly to the printer (from the OctoPrint web page, under the "Connection" section, click "Disconnect").

Step 4 — Configuring OctoPrint to use Klipper



- The OctoPrint web server needs to be configured to communicate with the Klipper host software. Using a web browser, login to the OctoPrint web page and then configure the following items:
- Navigate to the Settings tab (the wrench icon at the top of the page). Under "**Serial Connection**" in "**Additional serial ports**" add `/tmp/prINTER`. Then click "Save".
- Enter the Settings tab again and under "**Serial Connection**" change the "**Serial Port**" setting to `/tmp/prINTER`.
- In the **Settings** tab, navigate to the "**Behavior**" sub-tab and select the "**Cancel any ongoing prints but stay connected to the printer**" option. Click "Save".
- From the main page, under the "**Connection**" section (at the top left of the page) make sure the "**Serial Port**" is set to `/tmp/prINTER` and click "**Connect**". (If `/tmp/prINTER` is not an available selection then try reloading the page.)
- Once connected, navigate to the "**Terminal**" tab and type `status` (without the quotes) into the command entry box and click "**Send**". The terminal window will likely report there is an error opening the config file - that means OctoPrint is successfully communicating with Klipper. Proceed to the next section.

Step 5 — Configuring Klipper



```

pi@octopi: ~
GNU nano 2.7.4 File: printer.cfg

#+ss+-+
# ./oooooooooooo/
# +yyyyy-+ys-+yyy+
# yyyo`'+y+'+yyyy
# yyy/'.'sy: .syyyyy
# yyyyyy. :ys. /yy
# yyyyy+ `oy/ `oyyy
# +sy+-:sy+-:syys+
# ./oooooooooooo/
# `+ss+-+

# VORON2 300mm SKR 1.3 / 2209 Config

# This is a base printer.cfg file for the VORON2.2 printer intended for people who wish to upgrade
# the dual RAMPS configuration as listed in the manual/build guide.
# Created by "insurgus" with help from the VORON community.

# For other build sizes, controllers, or non-standard pin connections, please ask questions in the
# controller-general channel of our discord server.

# This file is only an example - be sure to review and update it
# according to the specifics of your printer. See the example.cfg and
# example-extras.cfg files for a description of available Klipper parameters.

# AND PLEASE READ THROUGH THE KLIPPER DOCUMENTATION FIRST!
# https://github.com/KevinOConnor/klipper/tree/master/docs

#####
# We use the following modifiers before the pin definition (ex: ^!ar99) #
# ! invert the logic #
# ^ activate 5v pullup (does not apply to all pins) #

^G Get Help      ^O Write Out    ^M Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos
^X Exit          ^R Read File    ^\ Replace      ^U Uncut Text   ^I To Spell     ^_ Go To Line
  
```

- The Klipper configuration is stored in a text file on the Raspberry Pi. Take a look at the example config files in the [config directory](#). The [example.cfg](#) file contains documentation on command parameters and it can also be used as an initial config file template. However, for most printers, one of the other config files may be a more concise starting point.
- Copy the Voron config file for your specific printer spec to the Raspberry Pi. You can use Nano or WinSCP. Whatever you feel more comfortable with.
- The location of the config file is **/home/pi/printer.cfg**

Step 6 — Errors



- Klipper reports error messages via the OctoPrint terminal tab. The "status" command can be used to re-report error messages. The default Klipper startup script also places a log in **/tmp/klippy.log** which provides more detailed information.
- In addition to common g-code commands, Klipper supports a few extended commands - "status" and "restart" are examples of these commands. Use the "help" command to get a list of other extended commands.
- After Klipper reports that the printer is ready go on to the [config check document](#) to perform some basic checks on the pin definitions in the config file.