CHANGE FROM MARLIN TO OPEN SOURCE KLIPPER FOR A SOVOL SV06

Gary Dyrkacz February, 2025

INTRODUCTION

This document serves several purposes:

1. A running, step-by-step history of the conversion of a stock SV06 printer, circa 2023, from the original Marlin interface, to the open source Klipper interface, using a BigTreeTech Pad 7 with CB1 module as the communication and control mcu. Problems, confusions, and fixes are included, exposing my novice background in 3D printing. The first part of the document serves as a personal reference archive, if I ever redo, or change Klipper functionality, or do updates. Theoretically, it should help me avoid or bypass previous pitfalls.
2. Although based on the SV06 and the Pad 7, there is a hope that this document might help another non expert avoid or deal with similar issues that I ran into, regardless of the mcu being used. As a personal running history of development, good grammar was sacrificed in favor of quickly describing actions to get the task done. Therefore, in the first section especially, expect stilted language.
3. During an unexpected, way lengthier calibration process than I anticipated, many issues were encountered, raising questions and suspicions of how Klipper calibrations interact with each other. Several running discussions on strange, unexpected calibration behavior are documented here. Especially, I encountered issues with bed mesh calibration that I was not completely able to understand, or overcome automatically. In the end, I found a path forward, but it was not trivial. In several places, I offer possible alternate methods of calibration.
4.

Background: I am not an expert on 3D printing or printers. I used my SV06 for over a year, before deciding to "upgrade" to Klipper. I was happy with the way Marlin was working. However, when I was anticipating purchasing the SOVOL SV06, I read several glowing reviews of Klipper, and its future in 3D printing, and the BigTreeTech Pad 7. I made an uncharacteristic kneejerk decision and bought the Pad 7 at the same time I bought the printer. A year later, I decided it was time to finally use the Pad 7 and Klipper.

Initially, I stuck with the stock SV06 Marlin interface, to learn and understand 3D printing. I am glad I did. I spent a lot of time calibrating and fine tuning the SV06, because I intended to use the printer not just for artsy printing, but engineered items for house repairs of every sort. Some repairs demand knowing and calibrating the printer's limits to produce dimensionally accurate objects to fit together. For example, I needed to design and build a casement window lock mechanism to replace a broken lock. The result saved me a couple thousand dollars to replace a Pella window, because Pella no longer stocked what was probably a $30 item.

Despite the tone in some sections of this document, and unresolved issues, would I go back to Marlin? No! I will I stick with Klipper. I appreciate the open-source nature of Klipper. I appreciate the efforts of many 3d printing experts to improve Klipper. I appreciate the fine or manual control I have over many operations, such as bed mesh calibration. The ability to directly interact with the Pad 7/SV06 through the mainsail browser interface is a gamechanger for me, as is the ability to quickly send models to the printer directly from Cura or PrusaSlicer. In addition, there does seem to be a trend of printer manufacturers to move to Klipper, even if some abuse the open-source nature, by using proprietary Klipper versions. As far as the Pad 7, I am happy with my purchase, I like the large, colored screen, and the response. That does not mean that other mcu choices are worse.

Now for the nitty-gritty

I had a lot of help from the internet, through videos, forum posts, and blogs, as well as the surprising well-written Klipper documents. I certainly not listing every reference I looked at or used, but I try and tag those that helped me most. Note that I rarely believe, or accept only one reference or review. In some cases, I have directly copied what was written or said, and tried to provide the supporting reference. Some people may

recognize the format for the references. I use Microsoft OneNote exclusively for documenting any research I do. This entire document started life as a running OneNote page of my efforts.

Also, be forewarned that the calculations I discuss were done in Excel. I do include an Excel workbook of some of my methods, but the worksheets were not designed to be plug in templates to simply used by the casual user. There is some explanation provided, but mostly the user is on their own in understanding some of the details.

Watch or read these articles first. They are very honest, and I found to be sage appraisals of switching to Klipper.
BTT Pad 7 - FULL Klipper But It's Not Easy!
From <https://www.youtube.com/watch?v=AWaB4uA8aqk>

SV06 - will Klipper make it "easier?"
From <https://www.reddit.com/r/Sovol/comments/13w31v5/sv06_will_klipper_make_it_easier/>

If all fails and Klipper conversion becomes overwhelming, to go back to Marlin do this:
How to flash the firmware | Sovol SV06
From <https://www.youtube.com/watch?v=p6l253OJa34>
How To Return To Marlin After Klipper - Klipper 3D Printer Series Part 10
From <https://www.youtube.com/watch?v=h7OaJNlri14>

“Klipper Firmware: The Heart of the System

Klipper firmware is the core of the Klipper system. It runs on the microcontroller that is part of the 3D printer's mainboard. It handles the low-level operations of the printer, such as reading G-code commands, translating them into motor movements, and controlling the extruder and other components. Klipper firmware is highly configurable and can be tailored to specific printer models and requirements. If you are still debating whether or not to add Klipper to your 3D printer, check out our article about what Klipper is, and why you would want it.

Moonraker API: Bridging Firmware and Web Interfaces

The Moonraker API acts as a bridge between Klipper firmware and the web interfaces. It provides a set of commands and endpoints that web interfaces use to communicate with the firmware. The Moonraker API allows the web interfaces to send commands to the printer, retrieve printer status and information, and receive real-time updates during printing.

Web Server/Web Interface: The User's Control Panel

The web server and interface are key parts of Klipper, giving you a dashboard you can access through your web browser. This is where you control your 3D printer. You can start and stop print jobs, check on your printer's status, adjust temperatures, handle your filament, and do a lot more from here. It's like a remote control panel for all your 3D printing needs.”
From <https://kingroon.com/blogs/3d-print-101/klipper-interfaces-mainsail-fluidd-octoprint?srsltid=AfmBOooTaz2RzH-G7Nb4RLjd51OSHGJuh1TdwyPE22hLvaNqj1LLgYNk>

For reference, this is the Pad 7 I purchased:
BIGTREETECH Pad 7 Klipper Touch Screen 7 Inch 3D Printing Smart Pad Open-Source Klipper-preloaded with BTT CB1 for Ender-3 Voron Vzbot 3D Printer Accessories Compatible with CM4
From <https://www.amazon.com/dp/B0C3GTNPGN?smid=A1LCDIR1SM8KTJ&ref_=chk_typ_imgToDp&th=1>

If totally unfamiliar with Klipper or Pad 7, I emphatically believe you should spend the extra time to read the following collection of sites and videos, before starting. The steps I describe were directly derived from multiple perusing of these references to resolve differences between them, and to finally make the change to Klipper a bit less intimidating. That said, I still ran into plenty of trouble.

This was my initial go-to site for the BigTreeTech Pad 7 from AuroraTech, but later was superseded by others, because the addition of multiple printers got confusing:

Multiple instances Klipper: BigTreeTech Pad 7 / CB1, low-cost Klipper solution for multiple printers
From <https://www.youtube.com/watch?v=8vkzpvia4mg&t=4s>

This person talked fast, and the discussion was not super detailed, but used KIAUH to update to latest Klipper files, just like the AuroraTech video:
Full Klipper & KlipperScreen Experience With BTT Pad 7
From <https://www.youtube.com/watch?v=WeVLi7xTDo8>

Some good useful information:
How I Installed and Calibrated Klipper on My Sovol SV06 3D Printer
From <https://www.youtube.com/watch?v=rIqg9bFykw0&t=4s>

Very good 4-part sequence, but the first part is for generic raspberry Pi, not the Pad 7 CB1
2024 Easy How to Guide on Installing Klipper with MainSailOS and KIAUH for your 3D Printer - EP1
From <https://www.youtube.com/watch?v=nI8o6yQRxpY>

Good series, but not Pad 7 specific.
How To Install Klipper The Easy Way Using KIAUH - Klipper 3D Printer Series Part 2
From <https://www.youtube.com/watch?v=WVmGChyroTo>
How To Find And Use A Printer Configuration File - Klipper 3D Printer Series Part 3 - Printer.cfg
From <https://www.youtube.com/watch?v=-YB8wyt_uno>
How To Compile Klipper Firmware For Your 3D Printer - Klipper 3D Printer Series Part 4
From <https://www.youtube.com/watch?v=GM_MztrjqmE>

The below is quite good, although I found it too late for my Klipper installation.
How I Installed and Calibrated Klipper on My Sovol SV06 3D Printer
From <https://www.youtube.com/watch?v=rIqg9bFykw0>

Fairly straightforward how-to video. Was key for realizing the issue with mainsail include and especially finding the right port stuff. Does not use KIAUH though.
PAD 7 from BigTreeTech Unboxing & Setup | Transform Your 3D Printer with Klipper!
From <https://www.youtube.com/watch?v=pBk2D4ZiHMw&t=731s>

Not bad. Mentions multiple printers, but offers a clean path to setup. In written style.
Bigtreetech Pad 7: Guide to the best all-in-one for Klipper
From <https://3dwork.io/en/bigtreetech-Pad-7-klipper/#strong-strong-strong-configuration-files-strong-strong-strong>

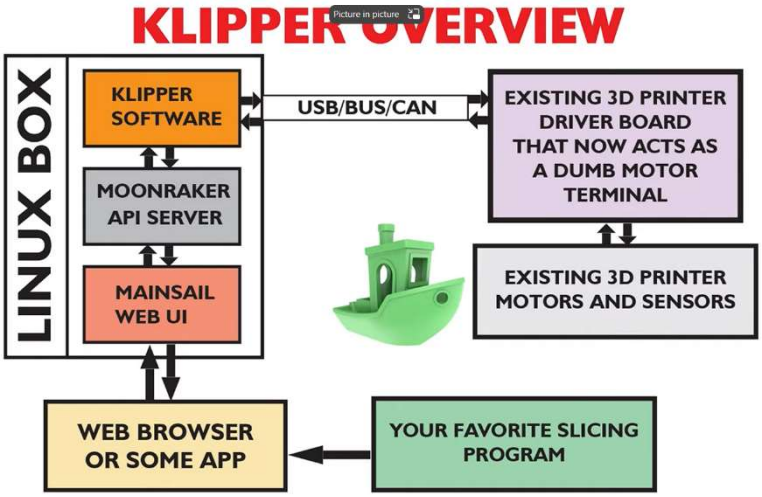
Excellent reference, almost directly applicable for SV06, In German, but can use CC for video translation.
SOVOL SV06 Plus & Bigtreetech Pad 7 | KLIPPER für ALLE! (inkl. Tutorial)
From <https://www.youtube.com/watch?v=zQZPUjVn3Vo>

Good written version of setting up Pad 7, uses KIAUH
Bigtreetech Pad 7: Guide to the best all-in-one for Klipper
From <https://3dwork.io/en/bigtreetech-Pad-7-klipper/>

Okay for some stuff, but goes low level instead of using KIAUH app.
BTT Pad 7 – Getting Started!
From <https://www.youtube.com/watch?v=91NJ8EQlB_o>

Specific for Pad 7: Note that the CB1 comes with the Pad 7 installed, and generally is considered better.
Big Tree Tech Klipper Pad7 Setup and Review
From <https://www.youtube.com/watch?v=D0JSJGgU4Xg>

The image below is taken from the above reference.



In the above case. Web browser can be computer. Linux box is in my case is a Pad 7 with CB1 mcu
KLIPPER IS THE PRINTER FIRMWARE FOR PRINTER, AKIN TO MARLIN
MAINSAIL IS A RESPONSIVE BROWSER INTERFACE FOR MANAGING AND MONITORING PRINTER OPERATION.
MOONRAKER IS A BACKEND SERVICE FOR COMMUNICATION BETWEEN KLIPPER AND MAINSAIL
KLIPPERSCREEN IS A SEPARATE CONTROL FOR SCREEN DEVICES SO NO NEED FOR A BROWSER.
CROWSNEST HANDLES CAMERA FUNCTIONALLITY FOR MAINSAIL.
SONAR IS A WATCHDOG FOR SOME PROCESSES
TIMELAPSE IS USED TO OBTAIN TIMELAPSES OF 3D PRINTS USING THE SLICER AND A CAMERA. PROVIDES A VIDEO DOCUMENT AS PRINTING UNFOLDS. LIKELY OF LIMITED USE FOR MANY, BUT VALUABLE FOR TEACHING OR SHOWING PROBLEMS.

CB1 is an alternate mcu to a Raspberry Pi, which would be a CM4 type. CB1 Is faster and supports better screens. The Pad7 does not provide the CM4 board, but does give the standoffs, screws and thermal Pad to mount a CM4.

THE KLIPPER CONVERSION PROCESS
There are several references that are specific to my computer or other hardware. This was primarily written if I ever need to do it all again, so is specific for me. Hopefully, the personal references will be clear.

1. I downloaded the Pad 7 instructions to a folder I created on one of my computer's drives to store all my files relevant to klipper: D:\Sovol\BigTreeTech\. Of course, this is personal reference.
2. To start:
3. Go to the Klipper github site, and locate the SV06 printer……..cfg file. Open the file and copy the info from the top comment section, which describes the communication protocols for the printer.
4. FIRST PHASE OF THE CHANGE FROM MARLIN TO KLIPPER IS SETTING UP THE BigTreeTech 7 CB-1 with latest OS, and Klipper.
5. The OS should already be on the supplied 32 GB micro card for the 7. It was. No need to load Debian OS.
6. Insert the SD card that comes with 7 into slot, turn on, it will show a *cfg* error. Ignore.
7. As soon as I plugged the Pad 7 in, it booted up and showed main screen.
8. SET UP WIFI
   Press Menu|NETWORK. The Pad 7 immediately found the NETGEAR router. Enter wifi with SSID and pswd. 7 only uses 2.4GHz.
   Tap the Refresh icon, which will check for updates.
   Clicked Restart. Mainsail update naturally killed WIFI, but it did remember info.
   Went to Menu|Network on the Pad 7 to get the IP address to use for browser; found 192.168.1.17 or use url : http://btt-Pad7.local/ True name was BTT-Pad7.

Go to the System icon and should see the IP address assigned to 7.

9. Once WIFI established, can proceed two ways: from browser or 7. First way, open browser (mainsail) on computer and put in: http://btt-Pad7.local/ or the IP address. Well, the url did not work initially, but the IP did (The version on the 7, only supports one printer.) Back out to main menu; Click System icon to get back to info screen.
10. Next update Klipper, moonraker, sonar, crowsnest; Several videos use KIAUH to update the files. However, the update command on the Pad 7 can do this. KIAUH will reflect the files are the latest, when it is loaded. Nevertheless, still need KIAUH to create the klipper.bin file. One thing mentioned in the BTT Pad 7 manual is that you want to make sure that on updates that moonraker.config points to change the origin of KlipperScreen to https://github.com/jordanruthe/KlipperScreen.git. It was already changed.
11. A very useful guide specific for the SV06 is: GitHub - bassamanator/Sovol-SV06-firmware: A comprehensive Klipper configuration for a variety of 3D printers Still needs to be used cautiously.
12. Clicked Full Update on Pad 7. This should have updated all files. Well, that would not work for whatever reason. There was some complaint about mainsail. Instead, I updated using the mainsail browser interface. Did single updates, first to mainsail, then moonraker. This also can be done using KIAUH. Most people recommend using KIAUH to update the Klipper and the helper apps. However, they are usually not using a Pad 7, or just don't want to use the Pad 7.
13. Browser will show same area as the Pad 7...sort of. This is the Mainsail browser interface. An error still shows. The left side allows printer control. The “Machine” icon opens up the /config/ folder on the Pad 7 to observe and manipulate the config files. There will be at least a "missing file" error because there is no printerxxxxx.cfg where xxxxx is specific printer from the github klipper/config. Don't worry about this yet.
14. The bassamanator site referenced above can help with some of the following.
15. Install PuTTY, or another terminal program on computer.
16. Install or update git so we can install KIAUH. Using the PuTTY terminal program, login to the Pad 7 mcu with “biqu” for name and pswd. I found git was already installed on the Pad 7, but needed to be updated. PuTTY was used extensively for all this. Clearly any time a sudo reference is displayed, or a simple typical Linux command, this refers to using PuTTY.
    To update and install git on the Pad 7, type: *sudo apt-get update && sudo apt-get install git -y*. This gave me a lot of trouble; had to look up an issue with "*error: dpkg was interrupted…run sudo dpkg --configure -a*". Did that and it put *“a >”* on the next line as if something was running. Ended up doing Ctrl C to get out of whatever this was doing. Looked up issue and the advice was to run *"sudo apt-get install -f*". Did that, and it was fine. However, when I ran the method to update git again, it found some issue and said to run *"sudo apt --fix-broken install*". Did that, and finally installed git without issues. Made sure by running the git install again, and it came back up to date.
17. Type: *cd ~ && git clone https://github.com/dw-0/KIAUH.git* (This is two cmds munged together cd~ changes to the root directory and the 2nd installs KIAUH.) This worked cleanly. The command *"ls”* showed: *crowsnest klipper klippy-env mainsail-config moonraker-env print_area_bed_mesh sonar KIAUH KlipperScreen mainsail moonraker moonraker-timelapse printer_data* (still in PuTTY)
18. Next, went to mainsail browser and tried to update System files, which was erroring out before. This time it worked to update the system files. The issue likely was the git stuff. Klipperscreen was showing invalid, but on a hard reboot, the system update seemed stuck, but then continued when I reset the browser, and finally everything was happy, including Klipperscreen. The only error showing was related to the printer.cfg file, which was anticipated. I had to reload the browser interface a couple time, so I rebooted the system cold a couple of time from the 7. Klipper was still showing an issue, but because Klipper already installed as we want for just one printer, just click 2 to update. If prompted to install macros do so. Alternatively, can do this from the browser using Update Manager in mainsail, which should be visible. Just click the round arrow icon. There will still be a printer error, but that should be the only error. However, Klipper was showing an "invalid" label.

19. -Type ./KIAUH/KIAUH.sh to run KIAUH; the prompt should indicate ~$ as the root or home dir. This is what the KIAUH doc shows to do as well
20. Do *ls -l* to show the files/folders. This worked as indicated. KIAUH found all the files and registered them as current. Phew!
21. Turn Pad off and back on, or type *sudo reboot* in PuTTY.
22. Log back in to PuTTY and do *ls -l* to see folders/files;
23. One of the folders is */boot/*. *Do ls /boot* to see if there is a *system.cfg* file. Whether it is there or not, create/open it. The file already existed.
24. Set up *system.cfg* by doing *cd /boot/* and *sudo nano system.cfg* or typing, *sudo nano /boot/system.cfg*.
25. Can change parameters such as hostname, screen rotation, vibration. Screen touch sensitivity; The *system.cfg* file showed hostname=*"BTT-Pad7"* which I left alone, *TimeZone* was commented out. I changed the Chinese zone to "*America/Chicago*" for my region. Do a search for how Linux handles time zones, to find yours. There was some, but not all the possible stuff to change for HDMI KlipperScreen, all commented out. I left that alone. The manual discusses several other parameters that can be changed. There was also several of other settings in the file related to router and connections, which I left alone. Save file from nano CTRL + O (save without exit); to exit and/or save nano type CTRL-X answ Y or N)
26. Type: *sudo reboot* for changes to take effect.
27. Now go to browser and reconnect to the 7 with old name, or if changed host name in the *system.cfg* file use that with *.local suffix*.
28. The Pad 7 manual discusses the *BoardEnv.txt* file. This is also found in */boot/*. I checked the file and all the parameters were set to what they recommend.
29. Do *sudo reboot*.
30. The Pad 7 and mainsail browser interface still showed the error *" Include file '/home/biqu/printer_data/config/generic-bigtreetech-xxx.cfg' does not exist".* The include is in the original printer.cfg BTT supplied; this will be fixed later.
31. ---
32. THIS COMPLETES GETTING THE 7 SET UP.
33. Well…almost: TWO MORE THINGS TO DO: 1.TRANSFER KLIPPER.BIN TO PRINTER. 2. CONFIGURE PRINTER.CFG
34. Good idea to set up a static IP address on router using the MAC address. I would assume though that only pertains if using the IP address instead of the name URL.
    I have a NETGEAR router connected to my cable modem (which is in bridge mode.) See To reserve an IP address on your NETGEAR router: Obviously specific to my router.
35. Once the CB1 is connected, the MAC address can be found in the DHCP connected devices section of the router, or to check the MAC address of a Linux device, you can open a PUTTY terminal window and type the command "*ifconfig*" at the command prompt:
36. Despite many online references, the MAC address was not on Hwadr. What I saw was:
    inet 192.168.1.17 netmask 255.255.255.0 broadcast 192.168.1.255
    net6 fe80::375:9591:eba2:8fb4 prefixlen 64 scopeid 0x20<link>
    ether 48:8f:4c:62:1d:36 txqueuelen 1000 (Ethernet)
37. Confused, I Googled finding the MAC address in Linux and came up with the alternate "*ip addr*". In the output of that command, I found a line "*link/ether 48:8f:4c:62:1d:36"*, which also was vague. Turns out *ifconfig* is old and may not work on updated Linux. It did, but MAC address, turns out to be in the "ether" line or if using "*ip addr*", the MAC address is in *link/ether* line. From there, can now go to router and setup a permanent ip address.
38. Well, despite the above, I got lazy, and decided not to set up a permanent IP address. Instead, in PuTTY I set up and saved a session *"btt-pad7"* which I can load and immediately interface to the Pad 7. With regards to mainsail, I opened the browser, I wanted to use with the SV06 and put in: http://btt.cb1.local/. (Could of course used IP address as well. By getting from the Pad 7 Menu|Network

icon.) Once I established contact, I just saved the *.local* url to my browser Favorites Bar for quick access any time.

39. In PuTTY, once again went to *system.cfg* by typing *sudo nano /boot/system.cfg* to make sure nothing changed. Parameters, such as *screen rotation, vibration, Screen touch sensitivity*, and *timezone* can be changed. Finally, if any changes, do Ctrl-O to save the file and Ctrl-X to exit
40. Type *"system reboot"* for changes to take effect.
41. Now go to browser and reconnect or if changed host name, use that with *.local*
42. There was no indication of the printer's name in the browser. Open Settings from top right gear icon of mainsail. Brought up a bunch of categories. Under *General*, the first setting for printer name was blank. I added *"Sovol SV06"* and Saved. Top label in browser now showed the printer name.
43. Back in PuTTY console at home directory, type *sudo cat>KlipperScreen.conf*. The file was empty. It is not clear if I should follow the lines up to 46. I did not add them, with only one printer. Instead jumped to 46
44. Do *sudo nano KlipperScreen.conf*
45. At top level put in

    [main]
    default_printer: the printer name you gave printer
    moonraker_host: localhost
    moonraker_port: 7125
    Ctrl-X to save
    Type sudo service KlipperScreen restart
46. PART 2: NEED TO GET OR CREATE the specific klipper/Sovol SV06 klipper.bin file. This file is used to flash klipper onto the printer's mcu. It is NOT the printer.cfg file. THERE ARE TWO WAYS TO UPDATE: ONE IS USING KIAUH TO BUILD THE BIN FILE. WE NEED THE SPECS IN THE PRINTER.CFG HEADER FILE TO DO THIS OR, SECOND, USE THE .BIN VERSION BASSAOMETER PROVIDES ON HIS GITHUB SITE. I decided to generate my own file, so that I am not dependent on delayed updates, if something really hot for klipper appears.
47. Because I intended to edit especially, the printer.cfg file or any additional cfg files using my favorite editor, NotePad++, on my computer, I set up a folder *"D:\Sovol\BigTreeTech7\"* to hold all Pad 7, mainsail and klipper related files.
48. Before starting the next stages, also decide how files will be transferred. There are several ways to get files between the Pad7 and a computer. One of the most convenient, used by a couple of the references I suggested, is the WINSCP app. WINSCPs main function is file transfer between a local and a remote computer. Another method is through the Pad7 directly. For the latter, while in the mainsail Machine menu, click the upload key on the left top side browser window. It then opens file explorer and you can pick file to upload to the Pad7. A file can be downloaded from Pad7 file to the computer, by clicking the action box next to the file to download. Note that the file is automatically bundled into a *zip* file. Opening the *zip* file will show the desired downloaded file. PUTTY can also be used using Linux commands. However, I like the convenience of WINSCP and highly recommend it.
49. First, copy the existing printer.cfg file from the Pad 7 to the computer. From the browser input click on Machine, and should be in the /config folder. Click the Upload file button, and send it to D:Sovol/BigTreeTech/, Rename this printer file to something else, like printer_old.cfg, I downloaded or scraped several Sovol related *printer.cfg* files to *"D:\Sovol\BigTreeTech7\"* directory : 1. The latest Sovol SV06 file from the klipper github site, https://github.com/Klipper3d/klipper/blob/master/config/printer-sovol-sv06-2022.cfg , on 12/17/24 to the computer. 2. Downloaded the Bassamanator version. 3. Happened to also find, (well more like stumbled on), the printer.cfg file for the SOVOL SVO6 Klipper Screen upgrade, https://github.com/Sovol3d/SOVOL_KLIPAD50_SYSTEM/blob/main/klipper_configuration/SV06/printer.cfg

50. If going to use the Bassamanator klipper.bin file that is already downloaded, skip to line 58. Otherwise, the following lines use KIAUH to create the .bin file. (I did not bring the printer…cfg file into the Pad 7 yet, because needed to confirm and edit the cfg file.)
51. NOW WE CONFIGURE THE SOVOL TO USE KLIPPER instead of Marlin.
52. The KIAUH *"make"* command creates a firmware file *klipper.bin* which is stored in the folder */home/biqu/klipper/out*. This file is used to flash Klipper on onto the printer's mcu. This is a two-stage process. Installing *klipper.bin* on the printer's mcu, and updating the 7's *printer.cfg* file. There likely may be a *klipper.bin* file on the Pad 7, but it is not correct for the Sovol.
53. Need to use PuTTY again to log into and talk to the Pad 7
54. Type, (do not copy from here) *./KIAUH/KIAUH.sh* This will start KIAUH. To generate the *klipper.bin* file, click 4 Advanced|2 Build Only,

    A screen will open to fill in pin mappings and communication protocols klipper needs. The parameters are in the header of the *printer…..cfg* file downloaded. Here it is for the sv06:

    #This file contains pin mappings for the stock Sovol SV06
    # To use this config, during "make menuconfig" select the
    # STM32F103 with a "28KiB bootloader" and serial (on USART1 PA10/PA9) communication.
    # Also, since it is using the GD32F103, please select Disable SWD at startup
    #
    # Flash this firmware by copying "out/klipper.bin" to a SD card and
    # turning on the printer with the card inserted. The firmware
    # filename must end in ".bin" and must not match the last filename
    # that was flashed.

    One issue is that it say nothing about the clock reference which is set at 8 MHz.
55. ---
56. Once all the proper info is added, hit ESC or type Quit and KIAUH will generate the .bin file.
57. Close the ssh window that is open to use KIAUH.
58. Now copy the *klipper.bin* file from wherever it is, to the SDcard:

    Launch WinSCP on the Windows machine.
    Enter connection details:

    Host name: Paste the CB1 address ip or btt-Pad7.
    Protocol: Select "SFTP".
    Username: Enter the username you use to access your CB1, typicall biqu
    Password: Enter the corresponding password.
    Connect: Click "Login" to establish the connection.
    Transfer files: Once connected, navigate through the directories on the Pad 7 (or folder on computer where *printer.cfg* is) and Windows computer using the file explorer-like interface within WinSCP and drag and drop files to transfer them between the two systems. Once I had the folders set on the computer side (for me: *"D:\Sovol\BigTreeTech7\"* and "*home/biqu/klipper/out"* on the Pad 7 side, I just dragged *klipper.bin* to computer "D: …." folder. Worked beautifully. Closed WINSCP.
59. Prepare the microSD Card for Flashing with These Parameters:

    Size: 16GB maximum.
    File system: FAT32.
    Allocation unit size: 4096 bytes.
    Must not contain any files except the firmware file.
60. Flashing the Printer Procedure:

    Disconnect any USB cables that might be connected to the motherboard.
    Using File Explorer, copy *klipper.bin* from the computer folder to the microSD card.
    Make sure the printer is off.

Remove the micro SDcard from the computer and insert into printer's micro SDcard slot.
Turn on the printer and wait a minute (usually takes 10 seconds). Other than the old screen going blank, and staying that way, there will be no indication if the mcu has been updated. Wait a minute to make sure.
Turn off the printer and remove the microSD.

61. Load PuTTY and start a session. Remember username and pswd are both "biqu" to get into root of 7.
62. Type: *ls -l* (that is "L"). Not being a dedicated Linux user, I am somewhat confused about folder set up in Pad7. Apparently, it starts in the home directory, because the tilde sends it there. However doing *"cd .."* , moves from *biqu@Btt-Pad7* back up to *biqu@Btt-Pad7/home* and an "*ls*" command prints one folder *"biqu"*. Oh well, I am not a Linux expert. The home directory showed:
    crowsnest KlipperScreen mainsail-config moonraker-timelapse sonar KIAUH klippy-env moonraker print_area_bed_mesh, klipper, mainsail, moonraker-env, printer_data
63. SETTING UP THE PRINTER.CFG FILE
64. Most Klipper settings are determined by a "printer configuration file" that will be stored on the Raspberry Pi, or in my case the CB1. In addition, there are example printer files on the Pad 7. It is not clear, if these files are old, or whether they are a downloaded up-to-date mirror of the official Klipper *printerxxxxx.cfg* files. The printer file is in *config*\ folder. To open it, click the mainsail Machine icon on left side menu.
65. Connect Pad 7 to printer now. The bottom or top back Pad7 port may be more convenient to use than the side port. DECIDE AND SET NOW. I used the bottom back USB port.
66. The old Marlin printer screen, will still be blank at this stage, until the error representing a broken *cfg* file fixed.
67. The BigtreeTech SV06 *printer.cfg* will be in */home/biqu/printer_data/config* folder; it only has two lines *[include mainsail.cfg] and [include generic-bigtreetech-xxx.cfg].* The latter "include" line leads to an error on the Pad 7. The *[include mainsail.cfg]* line is needed.
68. The best choice for starting to edit the *printer.cfg*, is the klipper github reference file. I use NotePad++ for editing files from my computer. I created the *printer.cfg* in my BigTreeTech folder by saving the github Klipper file as a new *printer.cfg* file*,* from its initial long name version on github. Note the Klipper version did not contain the *mainsail.cfg* reference. I then opened all three *.cfg* files in NotePad++ to compare them. (For the more advanced people, of course, Microsoft Visual Studio Code works fine too.)
69. Although most of the hardware parameters were fine, there were quite a few differences between the github klipper *.cfg* and the Sovol version. Particularly, the Sovol version included an include to *plr.cfg*. This file and associated settings in Sovol's Klipper Screen file, raised my beginner concerns of many complications. The plr file, once I searched online and finally found it, and associated settings in the *printer.cfg* file are used to restart a print in the event of a power failure. I decided to just not deal with the plr stuff, and ignored it all.
    I will not go through the grief of detailed comparisons and judgements between the various printer files, and checking with Klipper documentation when confused. Likely, just using the standard github Klipper version, with the addition of the mainsail include, might be enough, but clearly some of the changes represent speeding up the max allowed settings of the SV06. Below are the changes I did make to the "stock" downloaded github Klipper *printer.cfg* file using the Notepad++ editor:

70. Added
    [i*nclude mainsail.cfg]*
    [Printer]
    max_acceleration 8000 from 3000 ; this can be left at 3000 though. It will be determined later
    max_z_velocity: 15 from 5
    max_z_accel: 150 from 100
    Added *square_corner_velocity:3.0* the default is normally 5, its tied to acceleration and similar to Marlin jerk; Sovol may have been put in because of high max_accel.

[stepper x]
homing_speed: 40 ;see why below
Added *step_pulse_duration:0.000002* ;normal default is 2 us.

There were large changes in *[tmc2209 stepper X]*. I left those alone, may be Sovol Klipper Screen specific

[Stepper y]
Homing speed 40 see y below
Added *step_pulse_duration:0.000002*

[Stepper z]
position_max: 260 from 261
homing_speed: 6 from 4

[tmc extruder]
Interpolate: True from False
stealthchop_threshold: 400 ;was not moved to *printer.cfg*. Appears to be a way to quiet steppers, but according to Klipper documentation, is not generally recommended.

[bed_mesh]
speed:200 ;from 120
Probe_count 10,10 ;from 5,5; turns out just 10 is equivalent to 10,10.

Added entire *[RESPOND]*

[tmc2209 stepper_x]
driver_SGTHRS: 90 ;see below

[tmc2209 stepper_y]
*driver_SGTHRS: 82 ;*see below

[virtual_sdcard]
This was not clear. Saw comment that it already is required by mainsail. Found in mainsail.cfg:
[virtual_sdcard]
path: ~/printer_data/gcodes
on_error_gcode: CANCEL_PRINT

[bed_mesh] changed or added following.
mesh_pps:4,4
move_check_distance: 3
Later added cubic_tension=0.5

Added:
[save_variables]
filename =/home/biqu/printer_data/config/saved_variables.cfg This worked, it added file to /config/

Very important is START_PRINT. Added entire *[START_PRINT]* from Sovol Klipper Screen version

Added entire *[END_PRINT]* from Sovol Klipper Screen version, but change G1 Z1 to G1 Z10. Note the default temperatures were for PLA, which work fine for my SV06.

[gcode_macro START_PRINT]
gcode:
 # Parameters
{% set bedtemp = params.BED|default(60)|int %}
{% set extrudertemp = params.HOTEND|default(195)|int %}
RESPOND TYPE=echo MSG="START_PRINT CALLED"
 # set and wait for bet T
M190 S{bedtemp}
#set and wait for extruder T
M109 S{extrudertemp}
M84 E ; Disable E Motor for probe accuracy on direct drive systems
G90 ; Absolute positioning
G92 E0 ; Reset Extruder distance to 0
G1 E-1 ; Retracts filament to prevent blobs during probing
G92 E0 ; Reset Extruder distance to 0
G28 ; home all axes
BED_MESH_PROFILE LOAD=default ; Loads the mesh - not automatically loaded anymore
G92 E0 ; reset extruder
G1 Z1.0 F3000 ; move z up little to prevent scratching of surface
G1 X0.1 Y20 Z0.3 F5000.0 ; move to start-line position
G1 X0.1 Y200.0 Z0.3 F1500.0 E15 ; draw 1st line
G1 X0.4 Y200.0 Z0.3 F5000.0 ; move to side a little
G1 X0.4 Y20 Z0.3 F1500.0 E30 ; draw 2nd line
G92 E0 ; reset extruder
G1 Z2.0 F3000 ; move z up little to prevent scratching of surface

In Sovol's *printer.cfg* see line path*: home/mks/printer_data/gcodes*. The mks is clearly sort of like in BTT where home seems to be the same as biqu, e.g. *home/biqu/printer_data*. I decided to use same command as mainsail has and added:
[save_variables]
filename =/home/biqu/printer_data/config/saved_variables.cfg
From <https://gist.github.com/EzraByn/1268fdd6a3403d78c8f36bed750f9b49>
Or can also do
[save_variables]
filename: ~/printer_data/config/variables.cfg
from <https://github.com/rootiest/zippy_guides/blob/main/guides/variables.md>
And use example as:
variable_test: 0
gcode:
{% set my_test = params.SAMPLES|default(0)|int %}
SAVE_VARIABLE VARIABLE=test VALUE={my_test}
From <https://github.com/rootiest/zippy_guides/blob/main/guides/variables.md>

Based on step 70 add serial: to mcu
[mcu]
serial: /dev/serial/by-id/usb-1a86_USB_Serial-if00-port0

Some of the changes made really end up as placeholders that will be changed during the calibration process.

71. Most important is changing the port so klipper on the printer can talk to the mcu. To get the serial Id the Pad 7, must connect to the printer's USB port. I did not have *klipper.bin* loaded at this point.
    The *printer.cfg* file did contain a port0 command under [mcu]. It will not be the right one. Using PuTTY, while the printer is on, and connected with a USB cable to the Pad 7, do: "*ls -l /dev/serial/by-id/**" What returned was *"/dev/serial/by-id/usb-1a86_USB_Serial-if00-port0 -> ../../ttyUSB0"*
    Find the *[mcu]* section in *printer.cfg* and add the entire line without the quotes.
72. DOWNLOAD THE ADDITIONAL MACROS FOR THE SV06 FROM BASSAMANTOR GITHUB SITE. See the German video mentioned at the beginning of this document for some details. However, he directly downloaded the zip file and unpacked it into the Pad 7 config folder. I had already had the *printer.cfg* file changed, so all I did was add the additional macros. The macro files were downloaded individually from Bassamanator site, but it might have been better just to download the zip file and extract only the *cfg* folder. Some of the macro files need editing, because they indicate rpi instead of cb1, which was also changed in *printer.cfg* file, and I changed *"/pi/"* to *"/biqu/"* in any file reference that needed it. To transfer the files, used the mainsail interface, with *Machine|config*. Click the add folder link in mainsail. Create a new folder, */cfg/*, under the */config/* folder. To get back to */config/*, just click the current path */config/ crumbs* in the header of the file list. Mainsail could be used to upload the files, but I chose to use WINSCP.
73. Copy all the altered macros.
74. Next is the final update to the mcu *printer.cfg*. Using mainsail go to MACHINE and click open the *printer.cfg* file that was originally on the Pad 7 mcu. Remove the two include lines so the file is blank. Leave the file open
75. Open or go to the new modified *printer.cfg* file created on the computer. Using whatever editor was used to create the new *printer.cfg*, described in steps 69 and 70, select and copy the entire contents. Go back to mainsail, right click and paste the entire copied lines into the open *printer.cfg* file. (Could probably use WINSCP also to transfer, but would have to delete the *.cfg* file on the Pad 7 first.)
76. Add to *printer.cfg*:
    [include ./cfg/misc-macros.cfg]
    [include ./cfg/PARKING.cfg]
    [include ./cfg/MECANICAL_GANTRY_CALIBRATION.cfg]
    [include ./cfg/adxl-direct.cfg]
    Initially, I commented out all of the above, to reduce any complications of finding errors within the *printer.cfg* file itself. For instance, I failed to comment out the adxl include and kept getting an error about unknown pin 'cb1'. After much confusion, I looked at the *klipper.log* file, and saw where the boot up process went south. It was in the *adxl-direct.cfg* file.
77. If the accelerometer is not yet connected, will get a rpi mcu not connected error. Make sure to also comment out all the accelerometer lines (under rpi) or cb1 in my case.
78. Click the Save icon in mainsail.
79. If no errors show up, the printer should respond; More errors did show up for me. Mainly because of typing errors. Klipper is quite good about indicating most simple errors.
80. As for more macros, see jschuh/klipper-macros <https://github.com/jschuh/klipper-macros> . Apparently, has a good selection.
81. The following reference shows roughly how to add the macros: *How I Installed and Calibrated Klipper on My Sovol SV06 3D Printer* From <https://www.youtube.com/watch?v=rIqg9bFykw0&t=4s>
82. Save and again restart.

Before worrying about accelerometer, it is best to do some initial calibrations.
The Pad7 should now be updated and connected to printer. The test procedure of Bassamanator can be used, and some calibrations from the Pad 7.

TUNING AND CALIBRATION
PID (temperature) tune the bed
Ideally, all PID tuning should occur at the temperatures that you print most at. See:
Klipper PID Tuning - How to Guide

From <https://www.obico.io/blog/klipper-pid-tuning/>
Nice description of what happens.
Check the bed heating:
Do PID_CALIBRATE HEATER=heater_bed TARGET=70, followed by SAVE_CONFIG
This takes a while, ~15 minutes, then returns the best values, and indicates doing a SAVE_CONFIG.
My PID parameters: pid_Kp=68.574 pid_Ki=1.398 pid_Kd=840.888. Will be different for each printer.
Check the extruder heating:
PID_CALIBRATE HEATER=extruder TARGET=200
Same process for extruder
Final parameters: PID parameters: pid_Kp=27.259 pid_Ki=2.300 pid_Kd=80.755
A SAVE_CONFIG section is added to the *printer.cfg* with these parameters when saved.

ADJUST Z_OFFSET
Make sure your nozzle very clean. Do the Paper test. Well, start with the paper test. However, I always have problems with this. My final probe measurement is always lower than indicated by the paper test to get a good first layer. This was true with Klipper as well.
To start Type: PROBE_CALIBRATE
Follow z_offset setup in Mainsail/Fluidd.
SAVE_CONFIG (once completed)
From <https://github.com/bassamanator/Sovol-SV06-firmware>
Instead of the above process dfrom Bassamanator site, I used the Pad 7. First do a Home All. Note that this might cause really violent vibration against the gantry for X and the frame for Y. The problem and how to fix it will be discussed below.

Additional likely setup
From <https://github.com/bassamanator/Sovol-SV06-firmware>
"You will be pasting/typing these commands into the Mainsail/Fluidd console.
Check to see if X and Y max positions can be reached, and adjust position_max, if necessary. You might be able to go further, which is great, but I recommend leaving a 2mm gap for safety." Do
G28
G90
G1 X223 F3000
G1 Y223 F3000"

I did not do the last two position commands, because, though G28 Auto Home did work…the extruder vibrated violently against the printer frame compared to what I had originally found and fixed with the Marlin software. The problem has to do with more subtle manipulations needed for sensorless homing, which seem very rarely discussed, and probably only really occur, if going to push accelration.

Real sensorless homing
Some finer resetting of the sensorless homing parameters is necessary, than to just stop violent crashing into the gantry supports. Later when I ran into issues trying to automatically find the max_speed and max_acceleration using the AUTO_SPEED macro, I ended up learning even more than I ever wanted to know about homing.
Sensorless Homing - Klipper - Chris's Basement - 2023
From <https://www.youtube.com/watch?v=TJoeJxH5itM>
When doing these tests, MUST MAKE SURE EXTRUDER IS LEFT OF CENTER TO AVOID ANY CRASHING OF THE EXTRUDER INTO THE BED.

Examining several example printer.cfg files for the SV06 online, I found the rotation distance in [stepper_x] and [stepper_y] was the same as recommended, 40 with 16 microsteps, but the [homing_speed]:[driver_SGTHRS] (hs:ds) values were quite different. Bassamanator's printer.cfg uses homing speeds hs:ds for x; 40:81 and y, 40:81; Sovol's were x 90:115 and y: 60:85, and the github Klipper config had x 90:81 and y 60:82. In generally setting up hs and ds, the Klipper documentation suggested using half of the rotation distance as starting for hs, e.g., 20. The max sensitivity for the TMC2209 steppers is 255.

The differences in parameters are not hard to understand, because it depends on the specific mechanical and electronic responses of the printer. Bad bearings, badly lubricated bearings, bad lubrication of the gantry rods, corroded or nicked gantry rods, loose screws, loose belts, all will affect homing. As mentioned, initial attempt to use the Bassamanator homing_speed, and driver_SGTHRS values failed miserably with G28, with the extruder cycling violently against the gantry support for at least a few seconds, indicating that the sensitivity was set to low. After a few bad guesses, I found two settings for the x motor that sort of worked, 70:90 and 50:89. A setting of 70:88 ended up with extensive vibrating against the gantry support. Therefore, the working setting was right on the edge of lower sensitivity. I initially settled on 50:89, as what I thought seemed to work well. It did…until…. it didn't.

In Ellis' TEST_SPEED macro, he gets the mcu position and other position using GET_POSITION. If I did two successive G28 homing commands and ask for position with GET_POSITION between them the mcu values differed greatly from the expected 16.
SET_TMC_FIELD_STEPPER=stepper_x FIELD=SGTHRS VALUE=89

See the Klipper docs *Sensorless* Homing Section from <https://www.klipper3d.org/TMC_Drivers.html#sensorless-homing> Several parameters control the behavior of sensorless homing, especially the homing_speed and driver_SGTHRS. (SGTHRS stands for stallgaurd threshold) These two parameters work together to ensure a reliable extruder homing in the XY plane. For convenience, I will refer to them as pairs [homing_speed]:[driver_SGTHRS], [hs:ds]. I am not going to pretend to understand the underlying electronics that lead to a ds value. Driver_SGTHRS is related to the stepper motor sensitivity to changes in voltage and amperage, when stressed as in hitting a gantry post. The basis for recognizing a stopping point is therefore not dependent on a microswitch, or some other mechanical means of indicating a limit. Instead, motor driver boards have chips that monitor the voltage and amperage to detect when there is a significant jump in the values. The ds is somehow related to the changes in the voltage and amperage. If a certain threshold, is reached, a signal is sent to the software. The ds is a sensitivity parameter, which can have values ranging from 0-255, with higher values increasing sensitivity.

Even though I thought I had found a good set of hs:ds pair, after dealing with other calibrations, when I decided to find my maximum speed and acceleration, using the AUTO_SPEED macro (see further on). Well gee, I ended up with a max missed error, which prevented the AUTO_SPEED macro from running. I tried doing repetitive G28 homings, and used GET_POSITION to read the last stored results. The [mcu] was reporting variances in the stepper_x values much higher than the expected 16 microsteps. My guess was the problem had to do with the hs:ds settings.

So once again I was forced to research how hs and ds work. From my naïve perspective, the two parameters end up defining a range between the extruder never getting to the left gantry post, or alternatively slamming in to the left gantry post. In the former case, because the ds setting is too sensitive, even extremely minor mechanical/electronic perturbations stop the extruder before ever reaching the left gantry post. In contrast, if the lower ds sensitivity is set too low, the extruder slams repeatedly into gantry post. Somewhere in between these two cases lies a sweet spot or region. However, the spot may not be at the point, just above the violent vibration point. The best goal appears to find the point where on homing from any point, when the x and y axis are homed, the error is less than the microsteps, which is 16, for the SV06. Based on what will be discussed later for finding the max_speed and max_acceleration limits, the ds value was changed and homing was done and recorded several times to see if a certain ds value resulted in a stable steps value under 16 steps. I used the following procedure to find a sweet spot.

TRY THIS SEQUENCE OF COMMANDS TO START:

1. Find a hs:ds position that taps, but does not violently cycle against the left gantry post. A value of 50:90 might be a good starting position.
2. Set the desired homing_speed for the motor to be tested in printer.cfg, for instance stepper_x.
3. Home the extruder.
4. In mainsail console type: *SET_TMC_FIELD_STEPPER=stepper_x FIELD=SGTHRS VALUE=89* . In this case, the device_SGTHR is set to 89 as an example.
5. Either click *Home All* in the mainsail toolhead section or type G28 in the console
6. Type GET_POSITION in the mainsail console, copy and paste to record the first line related to the [mcu]
7. Repeat steps iv and v at least 3 times, and see if the stepper_x values are 16 or less.
8. If the microsteps are greater than 16, repeat steps 1 to 7, changing the TMC_FIELD_STEPPER value for stepper_x

Once established repeat the same for stepper_y.
Finally, open the *printer.cfg* file and change the device_SGTHRS x and y stepper motor values to what was found.

The process can be sped up a bit, but with a greater risk of problems by only homing the x stepper motor:

1. Do M84, or ensure the extruder is near the center of the build plate, and the gantry is near the bed.
2. SET_TMC_FIELD_STEPPER=stepper_x FIELD=SGTHRS VALUE=89
3. Do G28 X
4. GET_POSITION and record the position
5. Do M84 and slowly move the extruder back to center of bed.
6. Repeat 2 to 5 as needed until the [mcu] microsteps variation are under 16.

Following the procedure above, I was able to get microstep variations of +/- 2, with consistent extruder travel and no violent cycling of the extruder against the left post. Note that before starting, I made sure the x and y rods had a thin coating of grease. When I initially set up my printer. I had removed all my bearings and packed them with grease, which likely gave me somewhat smoother (and quieter) operation, and possibly a broader sweet region to more easily define a good hs:ds. I have not discussed modifying the initial homing speed, because I shied away from dealing with it. I just felt that the Klipper recommendation of 20 was too small, and saw no reason to need to do superfast homing; 50 was my intuitive compromise, partly based on the range from the other *.cfg* files I mentioned previously.

Z_TRAMMING
DO NOT USE THE Z TILT IN THE PAD 7; IT ONLY WORKS FOR DUAL DRIVER STEPPERS, NOT SINGLE DRIVER DUAL STEPPERS.

Why the enhanced warning? Experience, ending up in serious problem. The Z tilt procedure in the Pad 7 crashed the extruder into plate at 220, 220 mm position. After attempts to use Z-Tilt on the Pad 7 ended up with the two Z steppers way out of alignment. The only way to reset was to manually turn the screws to move the gantry. The gantry was so skewed, I ended up using the soup can method to initially adjust. However, the soup can method depends on "feeling" when the two sides are parallel to the bed, which left me a bit uneasy. Researching, I found several semi-automated Z-Tramming macros exist for setting the gantry parallel to the bed. These were the three that I looked into extensively.

Below are three sites used to help understand how to adjust the z bed tilt:
Z-markers to keep Z-motors aligned on Sovol SV06, SV06Plus + How to Z-align using inductive probe
From <https://www.youtube.com/watch?v=VI7icch4VEM>
GatoMiopia/Klipper_Z_Tramming
From <https://github.com/GatoMiopia/Klipper_Z_Tramming/blob/main/Z_TRAMMING.cfg>
klipper-z-tramming
From <https://github.com/Jomik/klipper-z-tramming/blob/main/README.md>
I decided to use GatoMiopia Z tramming method. However, the Bassamanator version may allow the standard Z crashing into the frame. I did not look into the details.

First, check all screws everywhere to make sure they are tight/ We are going deep into calibrations now. I found I had lost one extrude screw, and two were loose after a year. I was able to replace the lost screw with a 3x6 mm. The real screw is supposed to be 3x5 flat head countersunk type. (Hooray! Found the original under the printer two days later.)

Skip this paragraph, unless you feel the need to understand a few details on how macros work. I examined the macros to have a cursory understanding of what was being done. This was for my future reference in macro writing, should I get industrious or fed up with current processes. For instance, there is a printer object always available; “config” is apparently an object representing the printer.cfg file information. Also, there is a configfile object available, which represents the last version read into klipper to use. Further, there is a settings object under configfile Thus,
printer.configfile.settings.stepper_x grabs the [stepper_x] from printer.cfg. As an example.
{% set stepper_x = printer.configfile.settings.stepper_x %}
{% set x_min = stepper_x.position_min | float %}
Grabs the minimum defined position from the printer.cfg file.
Also, learned that when accessing an object that has space, use printer[], e.g., {% set tolerance = printer["gcode_macro Z_TRAMMING"].tolerance %}. For reason that are not clear, the Y position of the procedure was set at 55 mm, but for the

SV06 the typical probe center is at 85 mm. The value was changed in the macro. Also found, if too far out, the macro may not be able to settle on a value. In addition, finally learned that variables are defined by using the prefix, variable_, followed by the variable name. When calling the variable, we simply use the name itself.

In the second macro, the settings file defines the left as 5 and right as 169.5. This puts the true probe measuring points at 32 and 196.5 This is not quite symmetrical, but it is close the bed mounting screws. Why these values? The plate x is 235 mm wide. First y grid line to last y grid line is 170 mm. From the edge to the first full y grid line is 34 mm. The right Y grid line to right edge of plate is only 32 mm.

By manually adjusting X, just before it hits the left gantry support, the probe is about 32 mm in from the left edge of the plate. Klipper reads X:0. The nozzle is ~5 mm from the edge of the plate at this point. However, the probe is actually very near the first y direction grid line, which is ~5+27 = 32 mm from edge of plate.

On the right side, the corresponding symmetrical position at the last y direction grid line klipper shows X:170 mm. this is why both of the Z_tramming methods use 169.5. It is the probe position that is important for Z_tramming, not the nozzle position. The problem with this is that in my case, the bed z height dropped off very quickly from flat. The screws on X were close to 33 mm and 197 measured from the edge of the bed, not absolute coordinates, which would be 28 and 192 mm.
One other measurement will be important for later discussion. The center of the left gantry screw is 45 mm from where the nozzle sits for the x coordinate reading of 0 mm. When the nozzle is at 0, the probe is at 27 mm or 27+45 = 72 from the left screw. The distance between the screws is 325 mm. (See *D:\Sovol\Calibration_files\z_tramming_Calc.xlsm* for results of some analysis; internal reference use only). For public see *Z_tramming_Calc.xlsm* file in github.

I downloaded the Z-Tramming macro to my computer and using WINSCP created the folder */config/cfg/* and uploaded the file to the mcu. Added *[include ./cfg/Z-tramming.cfg]* to *printer.cfg* using mainsail. Saved and restarted. No errors.
To use, in console do: G28, then Z_TRAMMING. Two points will be probed, and the macro will print out how far to turn the z axis screws in hours and minutes. The macro also exists in the macro section of the Pad 7, if downloaded from Bassam. Took about 4 tries to get within the set 0.04 tolerance. However, I noted subsequently that initial values where way off.

Now begins a truly gruesome process that ended up requiring a several weeks work, and tens of hours.
The next phases of setting up the hardware amounted to an iterative process of Z-tramming calibration, Z Offset calibration, and bed mesh leveling and 3x3 first layer print tests.

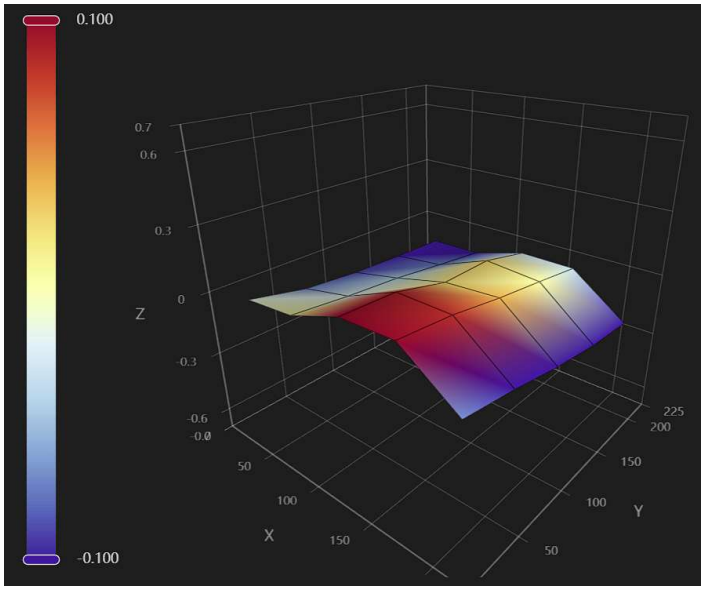
Z Offset Calibration
Go to Z-calibrate on the Pad 7. This made more sense than using the mainsail interface, because my desktop computer is too far away from the printer. The process is straightforward. This is under the Configuration menu on the Pad 7. I ended up with -1.77 for Z_Offset, or so I initially thought.

Create a bed mesh
This turned out to be the most gruesome aspect of the entire change to Klipper. To some extent, it is due to my own desire to produce a very uniform first layer, which is the test for how well all sorts of corrections are compensating for a non flat bed.
Unless the bed is perfectly flat (by a long shot, it was not) a bed_mesh offset matrix needed to be set up.
First, go to MACHINE and click on the *printer.cfg* file. Check to make sure *[probe]x_offset* and *[bed_mesh]mesh_min* for x are the same number. (I messed up at some point by mixing from different .cfg files and got an error indicating a negative x value.
Type AUTO_HOME in mainsail, or in Pad 7 do *Home|Home All*.
On Pad 7 click *CONFIGURATION|BED MESH|CALIBRATE*, or in mainsail browser, click *HeightMap* in the *Menu*. Click Calibrate, or in Console, type *BED_MESH_CALIBRATE*
When the profile is done, it is necessary to save the profile, if you intend to use it later. In Pad 7 click "Continue". In mainsail, do SAVE_CONFIG in console. The bed mesh will be saved as default in *printer.cfg*. At the end of the *printer.cfg* file will be a special section named “SAVE_CONFIG” added, with the new mesh information.

Depending on when you wish to see the heatmap of the bed_mesh, it may or may not be necessary to load the mesh. A sometime useful command is:
BED_MESH_PROFILE SAVE=<name> LOAD=<name> REMOVE=<name> or, as example, BED_MESH_PROFILE LOAD=default
From <https://www.klipper3d.org/Bed_Mesh.html?h=adap>

BED_MESH_OUTPUT PGP=1 prints the raw values to console, but this turned out not to be much help. The "Tool Adjusted" points refer to the nozzle location for each point, and the "Probe" points refer to the probe location. Note that when manually probing the "Probe" points will refer to both the tool and nozzle locations.
From <https://www.klipper3d.org/Bed_Mesh.html#output>
What I wanted were the corrected values. These are located in the last section of printer.cfg, in the SAVE_CONFIG section.

Below was the Initial bed mesh heatmap, using only a 5x5 bed mesh probe matrix.



First Layer Test and Update Cura to Use KlipperD
Obviously, the bed was not flat. I expected that doing a bed mesh calibration would compensate for most, if not all the curvature. That what I believed, it was supposed to do. Of course, the test for how well the bed mesh calibration is working, or how flat the bed really is, is to do a first layer test. Therefore, it was time to connect the slicer I use, Cura, to the mcu. Other slicers will have similar ways to connect Klipper and Cura to transfer files via WIFI, or Ethernet.

These references were very helpful in setting up Cura to work with Klipper:
Cura & Klipper: How to Make Them Work Together
From <https://all3dp.com/2/cura-klipper-tutorial/>
How to Connect CURA to Klipper via Moonraker
From <https://www.youtube.com/watch?v=kJIQitB2pvs>

First reference is an excellent must read. Some highlights: For one, do not enable Coasting in Cura Experimental, this conflicts with Pressure Advance plugin. I suspect it must disable the Linear Advance in Cura, and use the Klipper version. Acceleration Control and Jerk Control in the Speed settings tab are also best left disabled (but latter will be ignored in Klipper). In contrast, it's okay to use Cura's retraction settings, even after Pressure Advance has been tuned. A small distance (e.g. 0.75 mm) can help especially with filaments that tend to stick to the nozzle.
From <https://all3dp.com/2/cura-klipper-tutorial/>
Bottom line: IN CURA DISABLE: COASTING, JERK, ACCELERATION CONTROL
Go to marketplace. Use the search field of Cura's Marketplace (top right of Cura's main page) to find and install "Moonraker Connection".
When installation is complete, restart Cura.
Go to "Settings > Printer > Manage Printers" and select "Connect Moonraker".

Now configure the URL for your Klipper-Moonraker instance using the IP address or url for your CB1 or in my case http://btt-Pad7.local
Not added: Optionally, add the identifiers of any powered devices you've configured in Klipper (e.g. LED lights), and add a camera URL if installed.
Done: Choose to upload G-code or UFP files. The UFP with Thumbnail option will display a thumbnail of your model next to the file name. This makes it easy to quickly recognize your 3D file without needing to look at the name or load it into a G-code viewer. You can keep the rest of the settings as default. Pad 7 allows UFP.
Also invoked hide dialog after 30 s.
The configuration settings are saved when this window is closed.
After slicing a model, the usual file save options will include the prompt "Upload to <Printername>" that pops up a window where you can change file names and set other details. It really was that simple!
From <https://all3dp.com/2/cura-klipper-tutorial/>
I chose UFP.
Less simple was the necessity of changing the START_PRINT and END_PRINT g code settings on Cura. Researching online left me confused, because different printers/slicers require different protocols. This is what I found:
In Cura, go to *Settings|Printers|Manage Printers|MACHINE SETTINGS*. Copy the existing START PRINT and END PRINT gcodes to new files somewhere before deleting from Cura. (I used NotePad++.) Remove the Cura START PRINT gcodes in the MACHINE SETTINGS, text box. In the text box, copy and Ctrl-V to paste on one line:

START_PRINT BED={material_bed_temperature_layer_0} HOTEND={material_print_temperature_layer_0}

This code calls the START_PRINT macro, we added to printer.cfg, and specifies 2 parameters. Klipper apparently grabs the 2 parameters we create and dumps then into a params collection, from which we access them in the START_PRINT macro. After much grief trying to understand where those two parameters come from, I learned they are parameters within the Cura API; they start the first layer heating for the bed and extruder.

The START_PRINT section was previously loaded *printer.cfg.* Check it against the Cura start print code that was copied to the editor. The header stuff is not used. The first lines should read:
[gcode_macro START_PRINT]
gcode:
{% set BED_TEMPERATURE = params.BED|default(60)|float %}
{% set EXTRUDER_TEMPERATURE = params.HOTEND|default(195)|float %}
etc…
From <https://all3dp.com/2/cura-klipper-tutorial/>

Back in Cura MACHINE SETTINGS, remove the END_PRINT stuff and just add the single line "END_PRINT", no quotes.
To the End Print code that was copied from MACHINE SETTINGS to the editor file, add the lines
[*gcode_macro END_PRINT]*
gcode:
to the top of the code. Copy the entire code, go to mainsail, open *printer.cfg* if not open and before the last SAVE_CONFIG and after the START_PRINT section copy and place both *[gcode_macro START_PRINT]* and *[gcode_macro END_PRINT]* macros. (Can really put these anywhere., except in the SAVE_CONFIG section; my choice was at the end.)

At some point after version 5.6, Cura and my late model Windows 11 computer, had a falling out, where Cura would randomly sometimes load, and hang on loading at other times. I switched to PrusaSlicer, with the following start and end code
settings:]

M104 S0 ; Stops PS/SS from sending temp waits separately
M140 S0
PRINT_START BED=[first_layer_bed_temperature] HOTEND=[first_layer_temperature[initial_extruder]]

I had finally reached the point to run a first layer test. I used Parametric Bed Level Test by slartibartfast42 https://www.thingiverse.com/thing:3614389 modified for the SV06 and 220 mm bed. This is a 3x3 matrix of 20 mm

square prints for a total of 9 squares. Of course, there are many variants, but this was sufficient for me. The testing was done with PLA and a 0.4 mm nozzle. The bed is the standard SV06 textured PEI coated bed, so some of the light scattering is likely from the dimpled polymer coated surface. From my previous years use of Marlin and Cura, I knew that the Cura generic PLA settings and Sovol SV06 machine settings were a good starting point, with the only changes being a bed temperature of 60 °C and extruder hot end of 195 °C. With Marlin, I had previously had a z offset of -1.61. However, because of changes to the Z tilt, it was not clear the z would be the same.

After several attempts to adjust the Z tilt and find an appropriate z offset, the first layer results were clearly not great. There were profound differences in the first layer squares and the reasons why were very confusing. I realized there was going to be a lot more to this calibration.

SHIMMING THE BED

I found initially that there was too much difference in the prints over the bed, even to the point of not filling in between lines. Considering the rather large curvature in the bed, I decided to try and shim the bed to see if I could improve the overall character of the first layer printing. This will raise a question of why should I need to shim at all, if I set the z tilt and load a bed mesh. I still do not have a good answer to this.

When I originally set up the machine with Marlin, I used the excellent Pronterface app to get and display the bed. Of course, Klipper superseded the need to use Pronterface. Although I had adjusted the bed height when I bought the SV06, I did not realize just how far from level the bed still was, or I felt that the Marlin software bed mesh calculation would adequately compensate for any issues. I only made a half-hearted attempt to level the bed when the printer was first set up.

Reading more about bicubic and mesh accuracy, and several online discussions that seemed to make sense to me, I decided to change several bed mesh parameters. I changed the *[bed mesh]probe_count* to 10, 10 and *[PROBE]samples*=5 from 2. (Later the 5 counts were considered overkill, and I locked in at 3; Klipper automatically averages the values. In addition, after a bit of reading, changed *[BED_MESH]bicubic_tension to 0.5*. Of course, BED_MESH_CALIBRATE took a lot longer to run. In my case, this was around 25 minutes. The SV06 probe is an induction sensor

The SV06 bed is mounted to the Y carriage with five screws. Some SV06 users purchase silicone shims to replace the existing metal shims, then generally cut and squish them as needed to level the bed. Personally, I was not convinced that a soft pliable material would not distort over time, and require periodic readjustment. I therefore settled on the more tedious method of using metal shims.

The shimming process is not complicated, but depending on how uneven the bed is, can take quite a bit of time; it did in my case. The process required several weeks, for reason that will become clear. However, that also included time investigating what constituted a good first layer, and much further research on leveling methods.

Keep in mind that the bed mounting screws are not at the corners of the bed. They are very close to 33 mm and 200 mm measured from the left edge of the bed. The rapid drop off of the bed height from these points to the bed edges, meant that no amount of just placing metal shims at the screw mounts will lead to a flat bed. A three-stage shimming process was necessary. First, shimming under the bed mounting bolts, second, shimming the top of the plate with Kapton tape, and third Z tramming, because the bed angle was clearly biased across most of the X plane.

. The first stage of shimming at the screw mounts was based on this great idea of using aluminum pop can material as spacers.

Beer Can Mod For Prusa MK3/S/+

From <https://arcady.genkin.ca/2021/01/beer-can-mod-for-prusa-mk3-s/>

The second stage was using Kapton tape:

I Leveled My Printer's Warped Bed using a Mesh Visualizer | Sovol SV06

From <https://www.youtube.com/watch?v=QwBPS7IFBMk>

I had to purchase Kapton tape, and found a kit on Amazon containing 5 different widths of 0.060 micron tape. I only used the 10 mm and 20 mm wide tape.

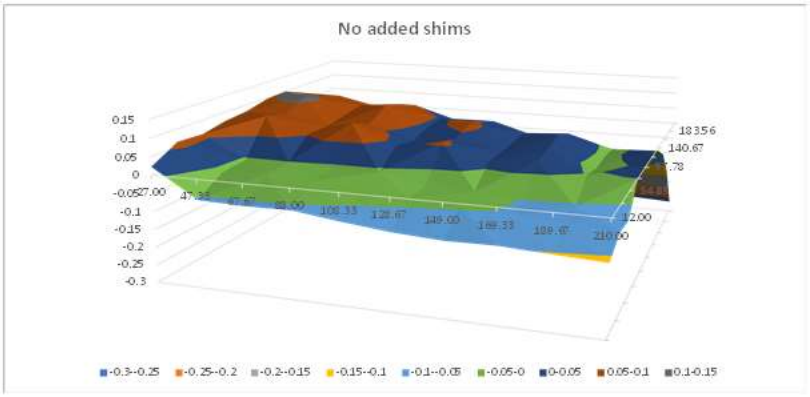
With regards to using pop can sheet material for the metal, I mostly followed the author's process of making the shims. I sandwiching the can material between wood blocks, but I had the advantage of having a table saw and an adjustable finger joint jig to accurately cut the slots for the screws. Yes, overkill, but I got nice clean cutouts. I measured the pop can sheet thickness with a micrometer to an get an average thickness of 0.102 mm.

Adding the shims is frustrating. Without upending the printer or turning it on its side, which I really wanted to avoid, adding shims was a bit tricky and frustrating. The idea is to loosen the screws, but not remove existing shims, so the bed can be raised slightly to add the shim. Fingers, tweezers, and a headlamp were the instruments of choice for placing the shims.

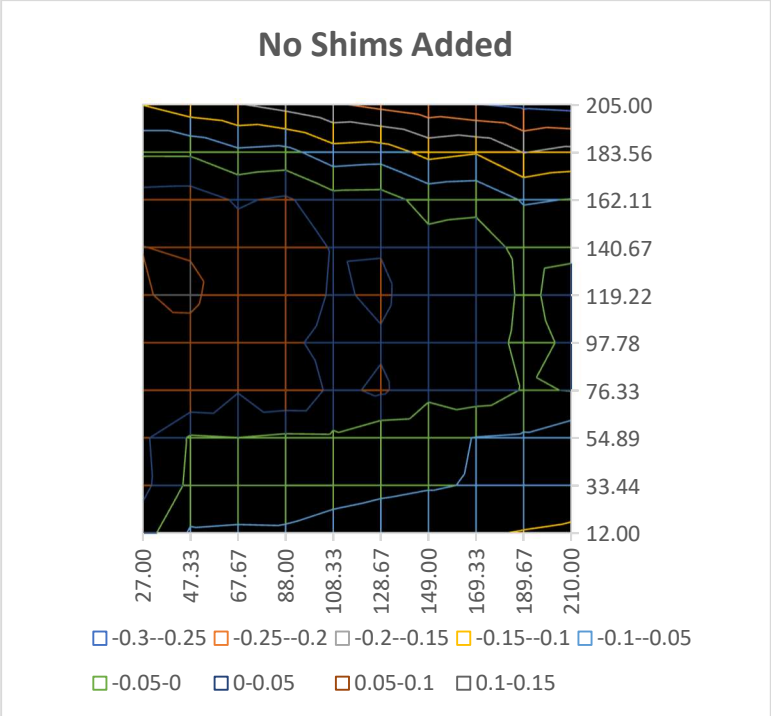
As the shimming process proceeded, I thought I needed at least 3 shims on some mounting screws. I decided to cut some shims from some thicker scrap aluminum roof flashing to make shimming easier. The flashing was quite uniform and was 0.381 mm thick. In the end, the flashing shims were not needed.

Fine tuning shimming with Kapton tape, was straightforward. The top sheet was removed, and based on the bed mesh data, Kapton tape was carefully applied to various areas.

Below is another view of what the Bed Mesh looked like before shimming. Excel is how I handle both simple and sophisticated data efficiently (other than R for some real high powered statistical stuff). These plots are surface plots generated in Excel; they are not smoothed across points in any way, so the appearance seems rough. Note that the binning intervals between layers is fairly large, which further increases the "roughness" texture.



Below is a contour view of the same data:



Below is a heatmap view of the data. Note that the values for the heatmap are outside the color range of the map. This is for easier comparison with the shimmed versions.

205.00	0.023	-0.06	-0.06	-0.06	-0.075	-0.089	-0.097	-0.095	-0.103	-0.108
183.56	0.064	-0.01	-0	-0.01	-0.025	-0.035	-0.045	-0.054	-0.063	-0.073
162.11	0.06	-0.01	-0	-0.01	-0.007	-0.033	-0.038	-0.051	-0.057	-0.08
140.67	0.074	0.098	0.053	0.094	0.039	0.058	0.013	0.027	-0.003	0.001
119.22	0.081	0.082	0.051	0.067	0.024	0.044	0.013	0.034	-0.016	0.008
97.78	0.097	0.111	0.063	0.099	0.041	0.06	0.016	0.046	-0.011	0.019
76.33	0.101	0.096	0.072	0.082	0.046	0.047	0.017	0.024	-0.015	-0.01
54.89	0.07	0.073	0.045	0.057	0.019	0.02	-0.018	-0.014	-0.055	-0.049
33.44	-0.01	-0.01	-0.04	-0.03	-0.08	-0.074	-0.115	-0.104	-0.152	-0.135
12.00	-0.1	-0.13	-0.15	-0.17	-0.194	-0.214	-0.233	-0.249	-0.261	-0.268
	27.00	**27.00**	**27.00**	**27.00**	**27.00**	**27.00**	**27.00**	**27.00**	**27.00**	**210.00**

I did not realize how complicated it would be to get to a reasonable level bed, represented by nice even surface squares from 3x3 first layer print. A part of the problem turned out to be me. I did initially appreciate whether I should level the bed heated or at ambient temperature. I had read several forum discussions and reputable sites addressing bed leveling. The views were mixed. At least one site that usually offered good advice on a variety of topics, was wrong when it comes to at least my SV06.

REVISITING Z TRAMMING

Z tramming is the action of adjusting the gantry to make it level with respect to the bed. Before discussing the details of leveling the bed, a broader discussion of Z tramming is necessary, because I used a modified procedure for adjusting the Z axis screws. The original tramming procedure measures the probe trigger height for two x positions on the bed. These probe positions heights are relative to the homing height position, which establishes the basis for all height measurements.

A sensorless probe measures the distance to the bed relative to the Homing position as the base reference point. A higher probe height on, say, the right side comparted to the left side, results in a negative difference when the right side height is subtracted from the left side height; the bed is too low on right side. The right side distance needs to decrease so distance between probe and bed match the height on the left side. The bed on the right side must be raised. However, adjusting

the gantry screws, raises or lower the extruder gantry, not the bed. The height distance is therefore decreased by turning the right gantry screw CCW to move the gantry toward the bed. If the difference between left and right sides is positive, the bed screw must be turned CW to raise the gantry to increase the gap (height) between probe and bed. This process is a bit confusing at first.

The bed mesh is the difference between z offset probe distance to the bed and probe distance to bed. In terms of tramming the bed, the bed mesh values are just the probe height values at various positions minus the height the probe found at the Homing position. Except for this frame of reference difference, the effect and adjustments to z_tramming is the same. Assuming that z offset is relatively constant with respect to small changes in the screw changes, The bed mesh differences between various positions still reflect differences in height.

Manually adjusting the gantry screws must be coincident with manually shimming an uneven bed, however, not at the same time. A reasonably flat, even bed that shows a nearly monotonic height gradient in the X direction, is indicative of tilted bed, which is easily fixed by manually adjusting the Z screws, i.e, z tramming. Unfortunately, the less flat the bed, especially near the edges, the entire bed angle will be thrown off. Some intuition is needed on tramming the bed, based on what the bed mesh pattern looks like. Of course, the process can be aided and refined by some math.

There is no simple process for automatically figuring out what shimming to use on the bed and what changes to make to level the gantry relative to the bed. With enough thought and programming, I see the potential for a semi-automated process that would suggest a combination of both bed shims and gantry screw adjustments. This would be fairly complex, because it has to take into account how the bed is mounted to the Y gantry. Even the idea of an automated process raises a question of what is the most appropriate points to pick for tramming.

There are many ways to consider how to make tramming adjustments based on the bed mesh: 1. The original two-point slope method. 2. Calculating the slope of all the X values across the Y mid-point. 3. Calculating a slope from a matrix selection of centrally located points, or XY cross relationships. To a large extent, what will work best depends on how badly twisted the bed is. There is little doubt that the first phase of gross adjustment should be to check, and if necessary, level the bed by judicious use of some sort of shims, whether solid, semi-solid, or screws.

The basis of all of these methods is calculation of the slope across X bed probe measurements. That is easy to implement for any of the tramming ideas above. What is the impact or differences of these ideas on a practical level? I decided to invest a bit of time into looking into these methods.

The original tramming macro found on the internet was straightforward; Get the difference at the X positions corresponding to 5 mm and 169.5 mm probe positions at some Y position of the bed. Divide that by the pitch to get the distance needed to move a Z screw and convert that to hours and minutes or revolution of the screw/coupling. As an example, the hours are given by the lowest integer value of

$$\frac{(x2 - x1)}{p}$$

Where *p* is the pitch of the threaded shaft, which in the case of the SV06 is 4. This value is also found in the *printer.cfg* file as *the [stepper_z]rotation_distance*. It is the millimeters per rotation. X2 and x1 are the value of the probe z output at two positions of the probe in the X direction. The minutes to rotate the screw shaft are given by,

$$\frac{60*(x2 - x1)}{p}$$

Note that these calculations do not explicitly calculate the slope as a function of distance. For this two-point method, the [change in height]/mm Is not needed. However, for comparison of different methods of defining Z tilt, the numerator must be replaced by the calculated slope. For cases, with more than two points, the general equation for slope is used:

$$\frac{N\sum XY - \sum X \sum Y}{N\sum X^2 - (\sum X)^2}$$

N in the number of points. In Excel (or any spreadsheet, or even R), there are several ways to calculate the slope. Here is one of them with X position in column A, and Z values in column *B: =((COUNT(A:A)*(SUMPRODUCT(A:A,B:B)) - (SUM(A:A)*SUM(B:B))))/ ((COUNT(A:A)*SUMPRODUCT(A:A,A:A)) - (SUM(A:A)^2))*. (Google Sheets or OpenOffice could also be used.)

The original tramming equation based on two points (55, 169.5) uses a difference of 0.04 mm as a break point for when to rotate a Z screw. I found that my probe with 5 measurements at each position has an overall standard deviation of 0.0021 mm. At 2σ, accounting for 95% of the variation this would be 0.004 mm. However, the average error in my probe measurements was 0.012 mm. A value of 0.04 mm seems reasonable, from a practical standpoint.

One aspect that the two available tramming macro does not address, was the frame of reference of the slope. The range of 5-169.5 means that the z screws are always being adjusted from the same two positions so there is no need to worry about explicitly calculating the slopes. However, because several different z tramming methods of calculating a slope are going to be used and compared, the distance between probe positions must be included. However, the height distance to manage for tramming must depend on the positions we choose as X position reference points. Particularly, I decided to use as the measurement basis the distance between the screws, which was 325 mm. Of course, however we calculate the slope, it must be off the bed plate and then extrapolated to the distance between the screws. Basing the distance to adjust the gantry screw on 114.5 against the broader x range of 325 mm, means any distance difference at the screw position and hence the rotational change needed, will be greater than represented by using the former distance.

As a basis for comparison, a difference of +0.04 mm with the 2-point macro amounts to a 0 h 0.6 min CW rotation, which is negligible. The equivalent of a 0.04 mm change based on the distance between the screws is 1.7 min CW. The translates to the Z_tramming macro being conservative in its estimation of the rotational change, which can lead to requiring multiple tramming measurements to dial in the gantry parallel to the bed. Extrapolating the slope to the distance between the screw, means that fewer tramming cycles might be needed. This does not imply the original tramming macro is wrong. Whatever the basis is used for the slope, the final position ultimately will arrive at the same result. The remaining issue is just what points should be included in the calculation.

First, we need to really understand the relationships between bed, nozzle, probe and Z gantry screw positions in the X direction. The following Table will help for the SV06.

Hardware Measurements			
		Mm	
Left screw center to nozzle tip at 0		52	
Probe X_offset		27	
tramming pts nozzle positions		5	169.5
tramming pts probe positions		32	196.5
Probe Y_offset		-20	
centers distance between Z screws		325	
X bed max		210	
Y bed max		205	
XY spread	range	Increment	
x points (15)	183.00	13.071429	
y points(15)	193.00	13.785714	

Some of these measurements are my own, others are standard for the SV06. The first row is self-explanatory. The second row tells us the offset between the nozzle and probe in the x direction. The third row are where the tramming macro

takes readings. These are absolute nozzle positions. The fourth row is where the probe is really measuring the differences. The last two rows were needed for calculations to correctly find the increment distance between successive points measured so the slopes of various points could be accurately calculated.

Examining the tramming macron itself, two "probe" points are used as indicated in the table. However, they are not where the probe is really measuring. The true probe position is dependent on the Probe X_offset, which is found in *printer.cfg*. For the SV06 X direction it is 27 mm. To confuse even more, the X "0" nozzle position is not the edge of the bed, but is about 11 mm in from the left edge in the X direction. The true probe points are measured at 32 and 196.5 respectively. Note that these positions are very close to the where the bed mounting screws are located. Thus, the Z-Tramming positions are at or close to symmetrically located on the bed.

The data displayed in the Table below represent the variation of the final bed mesh that will be discussed below, with different probe points or probed areas being used to determine the bed tilt from the derived slope points or areas. The math for each case is not difficult to work out, but can be tedious. However, the calculations are straightforward using a spreadsheet. The calculations can be seen in the included Excel sheet. Be aware, this sheet was not designed for the complete spreadsheet novice.

Z_Tramming Summary				
	2 pt original	Mid Y row only; all X pts slope	mid 4 Y rows; all X points averaged slopes	mid 4 Y; mid 4 X averaged slopes
calculated diff. at Right Screw; mm	0.0063	0.0079	0.0144	0.0272
minutes to move	0 h 0.1 min CW	0 h 0.1 min CW	0 h 0.2 min CW	0 h 0.4 min CW

The above table summarizes the various ways of calculating the magnitude of tilt. Keep in mind this is based on the bed mesh displayed below. The first data column is the simple two-point solution based near 55 and 169.5, the second column represents using all X row points in the bed mesh at near the mid-row Y value (119 mm) to calculate the slope, and find the height difference on the right screw. The third column averages the slopes of all X row points over the central Y rows, 4-7; the fourth column averages the slopes of the central 4-7 X row points over Y rows 4-7. This column represents a central 4x4 matrix of probe points with slopes of the 4 X points calculated for each Y row and the slopes averaged over the 4 Y rows to provide a single slope. Another option could have been to get the slope from X probe points 1,2,9,10, but the two point slope is sufficient for most work.

All the methods show exceptional z tilt compensation across the areas measured as indicated by the difference measurements. Once major shimming and z tilt adjustments had been made, to get to this level, required very small changes to the screw rotation; it was easy to over-compensate one way of another, and I suspect z axis screw backlash was a player in trying to bracket an acceptable value. Because the data represents what I originally thought would be my final mesh, it does not reflect the extensive effort to get to this stage. The range of values was definitely not always as tight as suggested.

LEVELING THE BED

The Klipper documentation recommends bed leveling with the bed heated. Moreover, the SV06 Marlin bed leveling process sets the bed temperature at 60 °C and extruder temperature at 120 °C. My assuming there would not be large changes in bed warp with temperature changes, resulted in days of frustration trying to get even near a level bed that produced first layer prints that were uniform, with proper infilling. Finally, I reread the documentation and decided to level with the Marlin/SV06 recommended temperatures. As an example of the differences, this is a heatmap of the difference between center and corner points for the SV06:

-0.111		-0.104
	-0.015	
-0.167		-0.212

The numbers represent the difference in bed height using the PROBE command on the heated bed at E120B60, and the room temperature bed (20.2 °C). The negative numbers translate to the corners of the bed being raised by the increased temperature. Also, the difference in offset between E195B60 and E120/B60 was not discernable within the error bounds of the probe measurements.

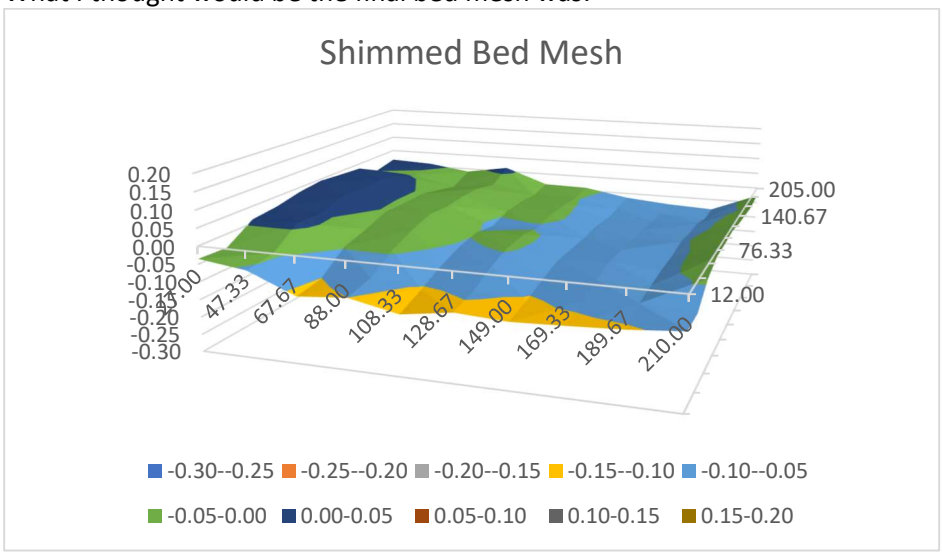
The final shimming result was a combination of aluminum shims and judiciously placed 10 mm and 20 mm Kapton tape strips, and adjusting the Z axis screws to achieve a reasonably uniform bed. The process was:

1. Copy the *printer.cfg* save_config section bed mesh data derived from a BED_MESH_CALIBRATE command to Excel and see what changes in shimming and/or Z tilt needed to be made.
2. Use g code M84 to disable steppers to be able to move bed for convenience, or adjust the z screws.
3. Add shims, or adjust the Z feed screws as directed by calculation (see section of revisiting tramming below).
4. Heat the printer 120E/60B (usually from mainsail). Wait a minute or so for temperatures to stabilize.
5. Home All (G28)
6. BED_MESH_CALIBRATE
7. Look at Heatmap.
8. Open printer.cfg file. Transfer data to Excel to create plots to see magnitude of shimming needed and to calculate z tilt for z axis tramming adjustment, and watch standard deviation and variance, for improvements (lower values).
9. Repeat above steps as needed.

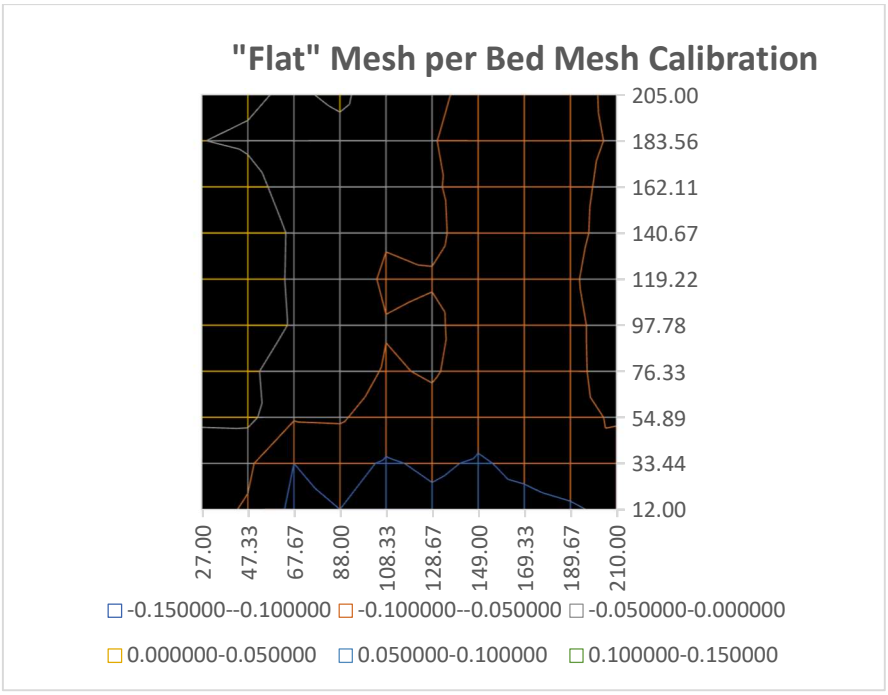
All data used bicubic interpolation with 10x10 mesh, probe count=3; pps: 4,4; tension: 0.5. All these runs were done at 120E/60B.

Rarely were shim changes and Z axis screw adjustments made simultaneously. Changing too many variables at once easily leads to just confusion, on what changes were really accomplishing, especially as the bed gets closer to a flat surface.

What I thought would be the final bed mesh was:



This is on the same scale as the original mesh above, to make the differences clearer.
Below is a narrower range contour plot of the same mesh with the same bin intervals:



Below is an Excel type heatmap of the flat bed mesh above.

	27.00	47.33	67.67	88.00	108.33	128.67	149.00	169.33	189.67	210.00
205.00	0.016	0.010	-0.010	0.013	-0.037	-0.040	-0.065	-0.078	-0.084	-0.027
183.56	0.001	-0.008	-0.039	-0.020	-0.048	-0.047	-0.073	-0.079	-0.086	-0.036
162.11	0.034	0.018	-0.023	-0.005	-0.042	-0.043	-0.076	-0.076	-0.068	-0.030
140.67	0.039	0.032	-0.007	-0.002	-0.043	-0.041	-0.068	-0.069	-0.067	-0.025
119.22	0.045	0.030	-0.008	-0.007	-0.061	-0.053	-0.073	-0.067	-0.059	-0.015
97.78	0.036	0.031	-0.005	-0.001	-0.047	-0.041	-0.072	-0.062	-0.067	-0.018
76.33	0.020	0.010	-0.028	-0.018	-0.055	-0.045	-0.070	-0.060	-0.068	-0.018
54.89	0.011	0.013	-0.046	-0.044	-0.076	-0.063	-0.085	-0.073	-0.075	-0.040
33.44	-0.039	-0.042	-0.100	-0.086	-0.104	-0.094	-0.104	-0.091	-0.081	-0.091
12.00	-0.035	-0.054	-0.112	-0.100	-0.130	-0.109	-0.117	-0.111	-0.104	-0.092

Blue represents negative values, white as 0.0 mm, and red as positive values. The range was set for all Excel generated heatmaps to -0.010 to 0010. This range will not always encompass the entire range of values, but allows easy comparison of patterns. As the heat map suggests, there are still probe points particularly at the bed periphery, with fairly large negative regions. The central area as expected based on using the average slopes to make changes to the z tilt.

A way of generally monitoring the overall flatness of the bed mesh is using the standard deviation and the variance. The original mesh had a standard deviation of 0.0854 mm and a variance of 0.00722 mm^2. This flat mesh had a standard deviation of 0.0383 mm and variance of 0.00146 mm^2. Definitely, significant improvement in flatness. The range of all the probe points was 0.114 mm. Unfortunately, the standard deviation and variance are more satisfying statistically, than practically. I was pleased with what I thought would be the final mesh, until I ran the first layer test

Despite all the effort to achieve a flat mesh with shimming, z tilt adjustments, and X Axis Twist Compensation, the first layer results were disappointing. The mesh produced some leveling, but consistently on multiple tests, the mesh calibration produced clearly uneven first layer prints. The left side square prints always showed more squishing, and in some iterations, evidence of the start of blobbing, while the right side of the prints exhibited gaps between some or all the extruded lines. Not only could the differences be seen visually and in micrographs, but even gently rubbing a fingertip or fingernail over the squares, the differences could be felt. The consistent result for many prints and minor tweaks of the

shimming was the left side of the bed presented a more squished first layer, and the right side presented too thin first layers.

Whatever bed mesh is supposed to do, it was not working as anticipated or expected with my SV06. One relatively new process that has been added to Klipper is X axis twist compensation. I though this might have something to do with my lack of attaining uniform first layer prints.

X AXIS TWIST COMPENSATION
I was somewhat skeptical of this process, although some users claimed great benefit for printers like the SV06. Despite what the Klipper docs say, the twist range is not based on the nozzle position, but the probe position. So X had to start at 27 and end at 193. Y at 20 and 200. I ran AXIS_TWIST_COMPENSATION_CALIBRATE and AXIS_TWIST_COMPENSATION_CALIBRATE AXIS=Y. Here are the results of my tests:

X axis: AXIS_TWIST_COMPENSATION_CALIBRATE: Calibration complete, offsets: [0.03555555555554646, 0.03430555555558179, -0.0698611111111278], mean z_offset: 1.567454
Y axis: AXIS_TWIST_COMPENSATION_CALIBRATE AXIS: Calibration complete, offsets: [0.02027777777777917, -0.005138888888883919, -0.015138888888895252], mean z_offset: 1.585407

Except in the case of the X axis, the variations were quite small. The bad news was the prints with and without this compensation were effectively the same as before, with no visible effect on the final first layer prints. I ended up commenting out the calibration in printer.cfg and manually removed the section from the SAVE_CONFIG section. (I also discovered that I could not simply comment out a section of the SAVE_CONFIG section. Some sort of error will show up.)

Honestly, I am not quite clear on just how effective is X axis compensation. I began to wonder whether it even makes sense, if the bed z-tilt is compensated using the bed mesh calibration values, instead of smashing the gantry into the top printer support. One might expect directly reading the bed mesh to adjust the bed tilt would automatically incorporate any twist compensation. In addition, if I understood enough of the axis compensation python code, what it does is calculate an average twist across the bed, in the x and y directions. If true, then by its nature this compensation is likely to smooth out some, but not all of the needed compensation, especially if the twist is nonlinear across the bed.

FINAL PROCESS TO PRODUCE MORE EVEN FIRST LAYER
Before I discuss the final push to get a flat bed, there are a couple of issues to address. First, the variations being discussed are quite small. The entire range of values falls within 0.3 mm, compared to the first layer thickness of 0.20 mm. However, that is the maximum range. The standard deviation of 0.04 mm for the entire bed suggests that 67% of the values will fall within 0.04 mm of the average value.

At this point, some advanced user or less discerning print makers, might be saying, “overkill” in any further effort to obtain a flatter bed. Basically, live with it. However, considering the supposed accuracy of the inductive probe, why should I not be able to achieve a more uniform bed? It bothered me.

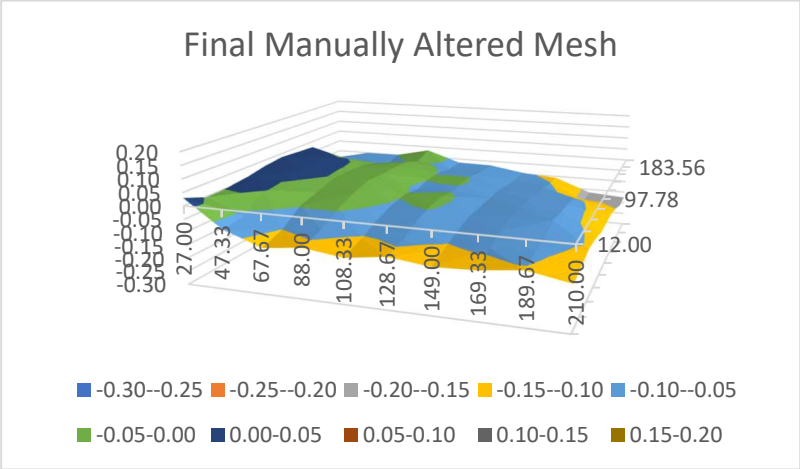
Second, be forewarned I am discussing direct changes to the section of the *printer.cfg* file where it is not recommended to make changes. Anyone considering doing what is done here, better be comfortable with spreadsheet manipulations, creating *.csv* files from spreadsheets, and thoroughly familiar to what the *SAVED_CONFIG* section of *printer.cfg* is all about. Considering by the end of several weeks work to change over to Klipper, I had developed a sufficient idea of how Klipper interacts with *printer.cfg*, I felt confident enough to manipulate the “DO NOT EDIT….” bed mesh section. But still no expert.

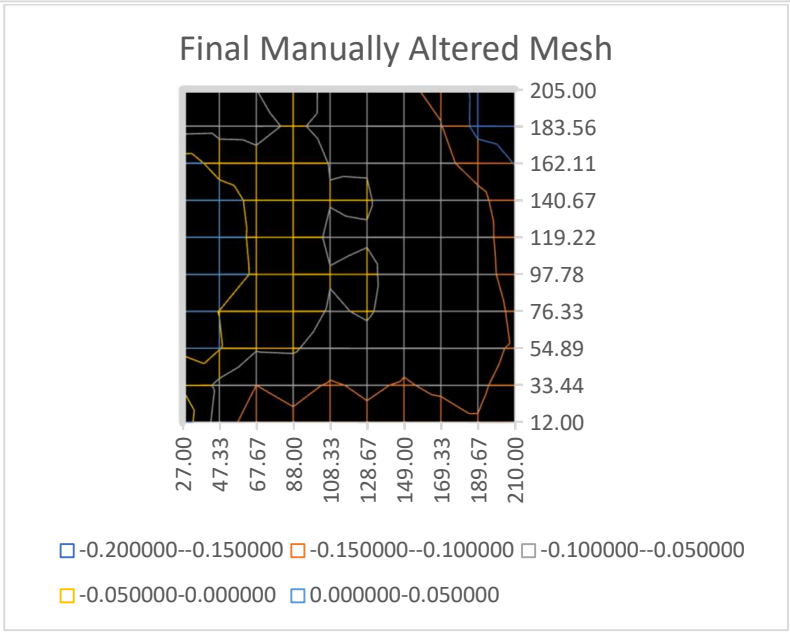
To see if it was even possible to achieve uniform first layering, I began to manually change the bed mesh values in *printer.cfg*. This was a long, time_consuming, repetitive process of adding small increments of 0.01-0.02 mm to select mesh probe positions the left side of the bed, and subtracting 0.01-0.02 mm increments on the right side probe positions, and running 3x3 first layer prints. The largest changes were to mesh X probe points 1 and 2 or 9 and 10, across the Y axis probe points. The central section of the bed mesh matrix required less drastic, or no changes. Many of the initial

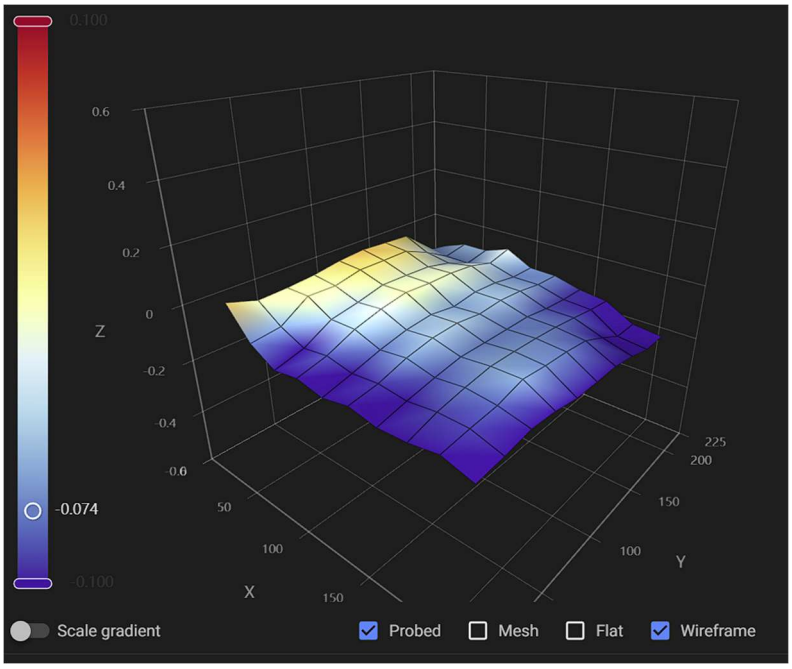
corrections were made across the entire Y axis, but some final tweaks were to individual probe points, especially at the bed corners.

One important point is that with manual compensation, the mesh tweaks are reversed from what seems intuitive. Probe values must be positively adjusted to minimize overly squished lines, i.e., move the probe further from the nozzle; conversely, probe values must be negatively adjusted to move the probe toward the nozzle to ensure filling gaps between lines.
Because the central probe points were not altered by large amounts, I decided to leave the z offset at 1.500, which I had established under the "best" bed mesh calibration. I have always found the paper test for z offset to be just a starting point. Consistently, I end up moving to a lower value. For instance, I initially found -1.46 mm with the paper test, and even then, there was quite a bit of resistance against copier paper (which I later found had an average thickness of 0.092 mm).

Below is the final mesh, which could be also termed the point I just finally gave up:





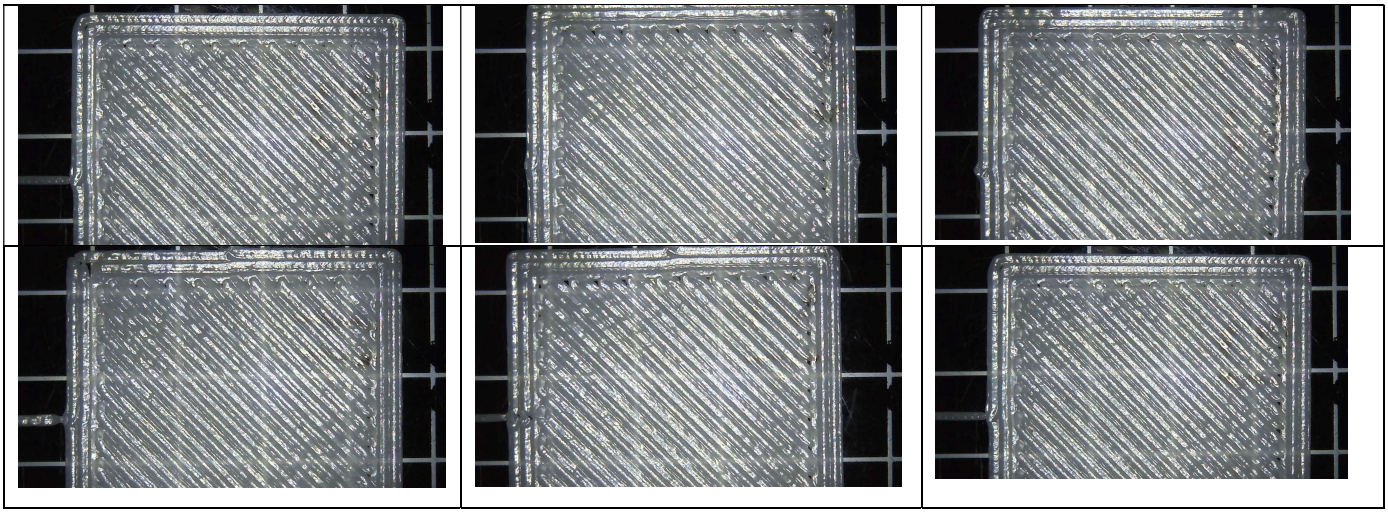


Below is the final Excel heatmap with the values.

205.00	-0.083	-0.050	-0.050	-0.018	-0.067	-0.070	-0.095	-0.108	-0.164	-0.175
183.56	-0.070	-0.068	-0.069	-0.040	-0.068	-0.067	-0.093	-0.099	-0.165	-0.176
162.11	0.024	-0.018	-0.033	-0.015	-0.052	-0.053	-0.086	-0.086	-0.122	-0.151
140.67	0.029	0.022	-0.012	-0.007	-0.048	-0.046	-0.073	-0.074	-0.085	-0.135
119.22	0.035	0.020	-0.008	-0.007	-0.061	-0.053	-0.073	-0.067	-0.074	-0.135
97.78	0.026	0.021	-0.005	-0.001	-0.047	-0.041	-0.072	-0.062	-0.068	-0.132
76.33	0.010	0.000	-0.028	-0.018	-0.055	-0.045	-0.070	-0.060	-0.058	-0.114
54.89	0.001	0.003	-0.046	-0.044	-0.076	-0.063	-0.085	-0.073	-0.083	-0.106
33.44	-0.004	-0.062	-0.100	-0.086	-0.104	-0.094	-0.104	-0.091	-0.091	-0.121
12.00	0.029	-0.074	-0.127	-0.110	-0.130	-0.109	-0.127	-0.121	-0.103	-0.131
	27.00	**47.33**	**67.67**	**88.00**	**108.33**	**128.67**	**149.00**	**169.33**	**189.67**	**210.00**

Comparing the original heatmap and the manually altered version above, note the large differences that had to be made to improve the first layer results, especially at the right side of the bed. Looking at the overall pattern, clearly this was not a z tilt issue.

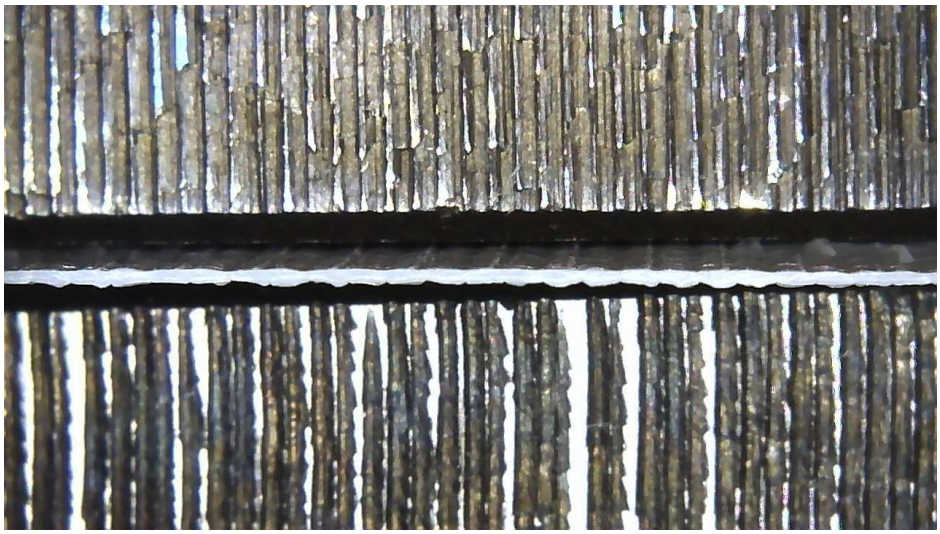
The photos below document the final first layer mesh. Photo 1,1 represents the left front corner of the bed and 3,3 the right back corner. Anly black marks visible on the surface are identification marks from pencil marks to identify the square.



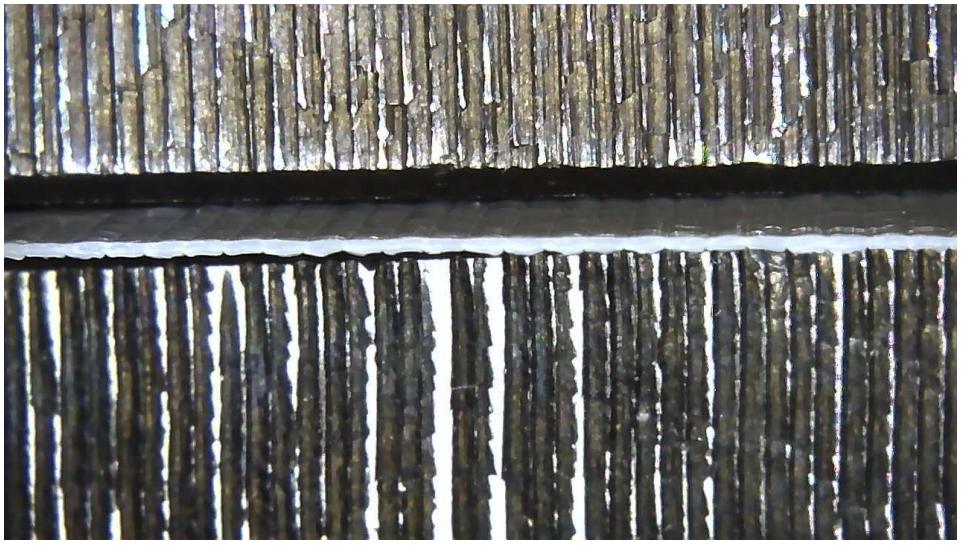
3x3 final, manually modified bed mesh loaded. Lower left is designated square 1,1.

The largest variation in line texture is between print 1,1 and 3,3. The differences are subtle, but they can be seen by the interplay of the light on and between the lines. As mentioned, this difference could even be felt by rubbing a fingertip gently across the square, normal to the line direction.

Below are two microphotographs, where the extruded lines were crosscut with scissors, and were held with the edge up, but slightly off the lens axis so the surface line patterns could be observed. The print is the lighter grey thin lines between the black area and vertical cut lines from the steel blocks holding the square piece. The upper face is toward the top. Keep in mind the lower face is against the textured PEI surface.



2,1 edge on view; top is at bottom

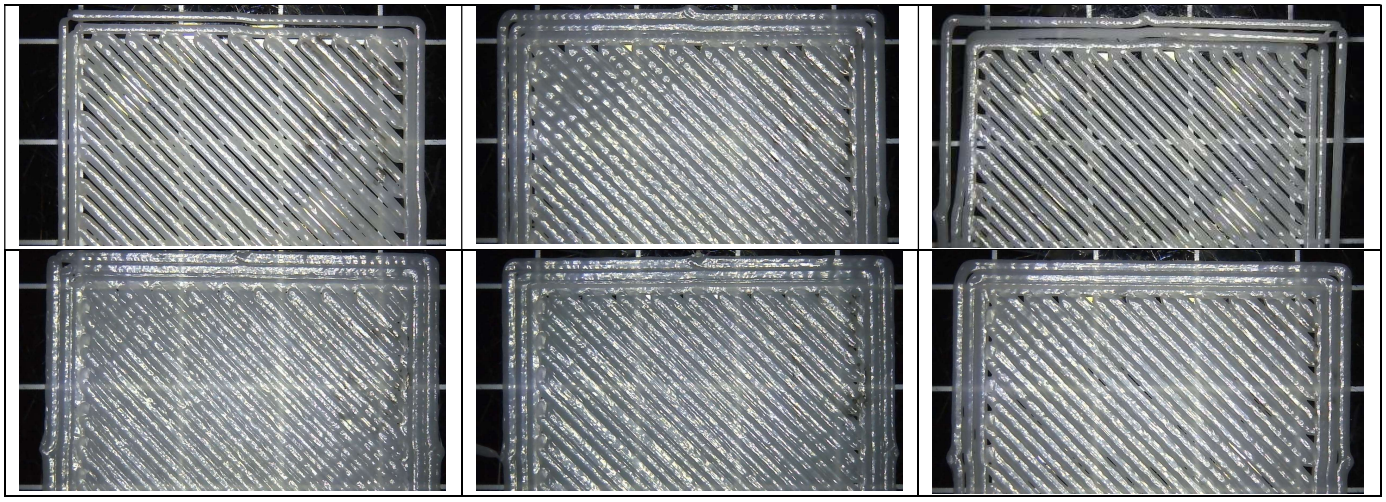


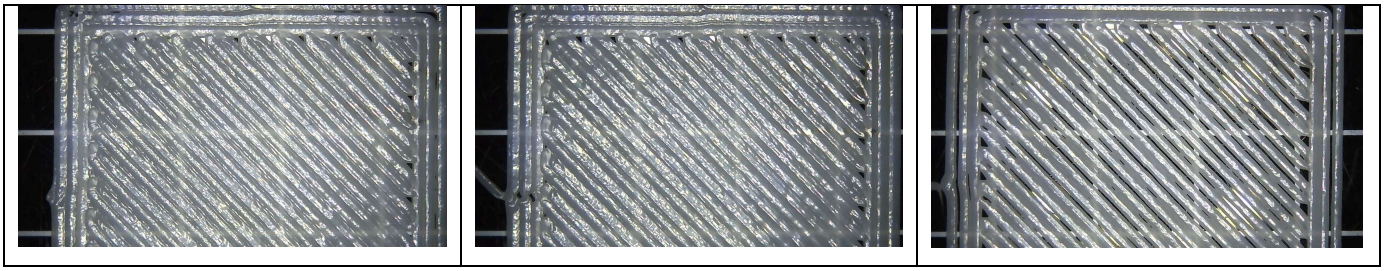
2,3 top surface of print is at the top

These are basically at the same magnification. There are subtle differences between the two edges, but the 2,3 section clearly has an overall flatter, more even appearance than the 2,1. Although I am not showing the mesh for the case where I thought the bed mesh should have accounted for any bed twist, the differences shown above are much less that was observed in that case.

This was my final bed mesh. For one, only rarely would I be printing to the edges of the bed. Moreover, this only for the first layer. Fade was set in the [bed_mesh] section. An important point is that in the manually unaltered case, as well as the final mesh case, all first layers were binding sufficiently to the bed.

As a more dramatic counterpoint, from an earlier run, I did first layer print without loading the bed mesh, but left the axis compensation on, to see what happened. The photos below show the impact





3x3 first layer; bed mesh not loaded. Lower left is designated square 1,1

The above was much worse that the simple automated bed mesh calibration procedure produced with the modified z tramming procedure (first layers results not shown). I did see improvement over the badly infilled and badly squished squares observed in this set of prints, especially in squares 1,3 and 3,3, but again, not completely. The bed mesh only calibration exhibited behavior between the extreme cases of no bed mesh loaded, and the manually adapted bed mesh.

Is Mathematically Altering the mesh Possible?

I played for far too long with ideas for a more mathematical method for adjusting the bed based on manual (paper probing various bed points. Of course, my first efforts hit the proverbial fan and died. One reason was that a limited simple linear string of probed points in x and y directions is not sufficient to represent all the changes across the bed, at least my bed. Clearly, from the final bed mesh shown above, the changes across the bed are not simply linear, nor can a single line represent the entire x or y dimension.

The way the mesh final mesh was assembled from first layer print tests was to start with a 10x10 matrix of zeros, add correction heights to this matrix and add the result the original bed mesh I presumed would be the final version, but was not.

This is the correction matrix that was applied to the bed mesh.

	27.00	**47.33**	**67.67**	**88.00**	**108.33**	**128.67**	**149.00**	**169.33**	**189.67**	**210.00**
12.00	0.064893	-0.020000	-0.015000	-0.010000	0.000000	0.000000	-0.010000	-0.010000	0.001404	-0.039144
33.44	0.035626	-0.020000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.010364	-0.030495
54.89	-0.010000	-0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.008382	-0.066012
76.33	-0.010000	-0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.010264	-0.096116
97.78	-0.010000	-0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.001259	-0.113890
119.22	-0.010000	-0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.014972	-0.120052
140.67	-0.010000	-0.010000	-0.005000	-0.005000	-0.005000	-0.005000	-0.005000	-0.005000	-0.018269	-0.110257
162.11	-0.010000	-0.035000	-0.010000	-0.010000	-0.010000	-0.010000	-0.010000	-0.010000	-0.053649	-0.120920
183.56	-0.070960	-0.060000	-0.030000	-0.020000	-0.020000	-0.020000	-0.020000	-0.020000	-0.078616	-0.140414
205.00	-0.098423	-0.060000	-0.040000	-0.030000	-0.030000	-0.030000	-0.030000	-0.030000	-0.079829	-0.148710

One aspect of this is that to make adjustments to be original bed mesh, turns out to be reversed from what is intuitively expected. To raise the bed height value, a more negative value is added, and to lower a height value a positive value is added

The question remains as to why the original bed mesh calibration procedure (BED_MESH_CALIBRATE) and axis compensation do not do a better job of compensating for an uneven bed structure, even when adjusted for temperature, considering how accurate a sensorless probe is. To not belabor an already overly long discussion on bed mesh calibration, a separate document in the same folder as this document provides more detail.

PRESSURE ADVANCE CALIBRATION

Generally, PA should be done before input shaping. I did "pressure advance" called Linear Advance in the Marlin world. However, I found information which indicated the Marlin version and Klipper have different coding.

Klipper has a standard way to do PA using the square tower, but I decided not to do this. Ellis' uses a different calibration method, which I did not use this time, but did when the printer was first set up.

Ellis' Pressure Advance / Linear Advance Calibration Tool
From <https://ellis3dp.com/Pressure_Linear_Advance_Tool/> Pattern Method | Ellis' Print Tuning Guide
Instead, I used the simple linear pattern, which I also tried when the printer was initially set up.
Pressure Advance With Your Sonic Pad - Sonic Pad Intermediate Part 6.
From <https://www.youtube.com/watch?v=JDis2X-QNZU>
Pressure Advance Calibration for your 3d printer (Klipper)
From <https://www.youtube.com/watch?v=LVp15eGXwm0>
Good review.
Pressure Advance Calibration Pattern
From <https://realdeuce.github.io/Voron/PA/pressure_advance.html>

I created the gcode file from the file creator in the immediate reference above.
I had to completely remove the start code section and change to just START_PRINT
Based on my current Cura settings for the SV06 which were initially the Cura version, I had to change the parameters to:

Retraction to 0.5
Changed Layer Height to 0.2
Fan remain at 0, because only one layer.
Outer wall speed, i.e fast print speed changed to 25 mm/s
Acceleration for outside wall changed from 3000 to 8000. This was the default I set in printer.cfg, but see next section.
Extrusion multiplier was set to Cura value, but instead of 100, has to be 1.0
Based on my old linear advance calibration with Marlin, I set the K factor range from 0 to 0.4 with a 0.02 step.

As others have suggested, the differences might be very subtle in such a range; they were. At quick glance all the lines looked the same, but close scrutiny finely saw the 20/40/20 splits. The "smoothest" line was the 0.12 line. The value was added to the *printer.cfg* by creating under [extruder] *pressure_advance: 0.12*.

Optimizing Klipper Acceleration and Speed

Up to this point, I had been using the more generic 300 mm/s speed, and Sovol's 8000 m/s^2 as max speed and max acceleration, respectively. It was time to find out just how far I could push the limits. So begins what should have been a simple process that turned into another saga. Initially I watched this video:
Optimize Klipper for Speed: Calculation and Configuration Tips
From <https://www.youtube.com/watch?v=dl1MGiZwhB0>
The above uses the Ellis method, to determine at least a set of starting points to fined maximum acceleration and maximum speed for the printer. https://ellis3dp.com/Print-Tuning-Guide/
I downloaded Ellis' TEST_SPEED.cfg file, but found out, I had already added it to the /config/cfg folder earlier. However, I did have to do an include in the printer.cfg file. I used the recommended starting test: TEST_SPEED ACCEL=5000 ITERATIONS=2
From <https://ellis3dp.com/Print-Tuning-Guide/articles/determining_max_speeds_accels.html>
Well, at 300 mm/s, I was detecting strange variations in the [mcu]stepper_x value. The x and y stepper values were bouncing all over the place, even at ungodly low accelerations. I did persevere and go has high as 8500 acceleration, and still heard no strange noises from the printer. I decided something was amiss. Was it me or the macro? I decided to try another speed calibrator. I found a video discussing the klipper AUTO_SPEED macro
Optimize Klipper for Speed: Using Klipper Auto Speed
From <https://www.youtube.com/watch?v=8d49_QUmP9k>
Klipper Auto Speed
From <https://github.com/Anonoei/klipper_auto_speed#installation>
From the above I followed the instructions, and download and set up auto_speed
cd ~
git clone https://github.com/Anonoei/klipper_auto_speed.git
cd klipper_auto_speed
./install.sh
From <https://github.com/Anonoei/klipper_auto_speed#installation>
Ender 3 Klipper Max Acceleration Speed Limits
From <https://www.youtube.com/watch?v=RgTvi0QgXho>

In mainsail, I did a Homing All, and then AUTO_SPEED. The process did not make it past the endstop test, with error "*Please increase MAX_MISSED (currently 1.0), or tune your steppers/homing macro.*" Well, back to research; I found this:"Sensorless homing: If you're using sensorless homing MAX_MISSED=1.0 is probably too low."
From <https://github.com/Anonoei/klipper_auto_speed>

Well, heck, that suggested some sort of issue with my homing settngs, which I was blissfully naive about. I ended up learning that the process for setting up homing conditions is more complicated than just stopping violent cycling of the extruder against the left gantry post, especially if you expect to do more sophisticated calibrations. Go back and read the sensorless homing section in this document, if you did not heed my warnings there.

Having redone the sensorless homing tests to get stable endstop readings, I ran AUTO_SPEED again. To make sure it was not going to bomb, I did add max_missed=6 to [auto_speed]. After doing a homing and trying AUTO_START it bombed out anyway, complaining again about max_missed too low. At this point, I decided to set it at 20 and not 12. That worked. The resulting print out was:

AUTO SPEED found recommended acceleration and velocity after 152.94s
| X max: a78350 v690
| Y max: a75050 v669
Recommended accel: 75050
Recommended velocity: 669

The 75000 seemed REALLY too good to be true. Others have commented on values being highly theoretical and maybe not so practical. I therefore went back to the TEST_SPEED macro. TEST_SPEED ACCEL=10000 ITERATIONS = 2 as example. With my newly calibrated homing driver settings, this test worked very nicely. I saw very stable readings until I jumped from 13000 m/s^2 to 16000 m/s^2 . Ellis recommends at least a 15% derating of the acceleration, which would be 13000*0.85 = 11050.0 Decided that even that might be too close, and replaced the 8000 value with 10000 as the *printer.cfg max_accel* value.

Next, used Ellis theoretical graph to see if the AUTO_SPEED velocity value was reasonable. The graph indicated the 600 mm/s speed recommended by AUTO_SPEED was conservative and well within the bounds. I ran the first test TEST_SPEED SPEED=600 ITERATIONS = 2 and it was spot on for both x and y. Decided that was good enough for maximum acceleration and changed to 600. I did not run the torture tests.

The new acceleration values needed to be entered into Cura. This required downloading the Klipper settings add-in from the Marketplace, and invoking the viewing of the various parameters in Cura Settings|Preferences to be able to at least add the maximum speed and maximum acceleration for extruding and retracting. Although I lost the reference, go to normal printing and change acceleration to the high values in: normal wall, inner wall, bridge, sparse infill, and travel he left top surface alone and reduced first layer to 50000. saved as Auto Speed profile in Cura.

Input Shaping.

Found out by reading other users problems finding the input shaper icon on the Pad 7 that the manual neglected to mention that to get the input shaper icon and screen showing up, must add *[input_shaper]* to *printer.cfg*. Once I did that the icon Input shaper can be found in the Printer|Configuration Screen.

Klipper documentation has a Resonance Compensation section for manual shaping. However, the Pad 7 comes with an ADXL345 sensor to automate the process. Klipper supports the ADXL345 sensor.

Measuring Resonances
From <https://www.klipper3d.org/Measuring_Resonances.html?h=reson>

I decided to follow the Klipper recommended procedure, which required going into PUTTY to install more software.

sudo apt update
sudo apt install python3-numpy python3-matplotlib libatlas-base-dev libopenblas-dev

Then
~/klippy-env/bin/python -c 'import numpy;'
From <https://www.klipper3d.org/Measuring_Resonances.html?h=adxl>

First, needed to add the adxl345 section and resonance tester to *printer.cfg*. The info is partly in the Pad 7 manual, but also needed to add in the resonance section, which was not mentioned. I found in my initial research on Klipper the needed code for the ADXL345, and added to *printer.cfg*, The lines were commented out to not cause any issues until I needed it.

I connected the accelerometer cable to the back of the Pad 7. This plug has to be pushed in quite hard to really seat the cable. I uncommented the relevant adxl345 lines in printer.*cfg*, including the *[mcu CB1]* section, and restarted the Pad 7. There were a couple of errors that I had made. The first attempt failed, with an error about cb1 not found. I apparently had not capitalized "cb1", Fixed that and still found errors related to cs_pin: CB1:None. Confused about what adding a section [mcu CB1] meant, which clearly was connected to the cs_pin reference, I did some more reading.

From what I could rationalize as a novice, mulitple mcu's can be defined with specific communication protocols for specific purposes, like to read an adxl345 spi based output module, or probably for multiple machines. Of course, my impression could be wrong or misleading, but it gave me some basis for understanding. After resolving the naming issue, the next error that showed up was that klipper could not find *spidev 1_1*. Well, that was in a file that the manual said was */boot/BoardEnv.txt*. Usuing PuTTY, I found it, and sure enough, I had not uncommented the line. Restarted….and mainsail presented the following error message:

MCU Protocol error

This is frequently caused by running an older version of the
firmware on the MCU(s). Fix by recompiling and flashing the
firmware.

Your Klipper version is: v0.12.0-418-g0114d72a6-dirty
MCU(s) which should be updated:
mcu: Current version v0.12.0-405-g8a3d2afd7 the mcu is the printer processor.
CB1: Current version v0.11.0-122-ge6ef48cd
Up-to-date MCU(s):
<none>
Once the underlying issue is corrected, use the "RESTART"
command to reload the config and restart the host software.

mcu 'CB1': Command format mismatch: query_adxl345 oid=%c rest_ticks=%u vs query_adxl345 oid=%c clock=%u rest_ticks=%u

(Note that this issue only came up when I tried to invoke the adxl345 sections.) The error showed that the *klipper.bin* mcu file was not the same as on the Pad 7 version. (Of course, from my novice position, this was not as obvious to me as it is now.) At the same time, mainsail and Pad 7 reported that there were Klipper and moonraker updates available. I thought that was the issue, and I updated the files. It was not the problem. Next, I rationalized that I had to flash the new klipper.bin file to the printer mcu. This was not as straightforward to understand, because I saw conflicting information on the internet. Some mcus' do have a way to automatically update the .bin file through the host.

I went back through the *KIAUH* .bin file generation process using PuTTY, copied the *klipper.bin* file to the sdcard, removed the USB cable connection to the Pad 7, inserted the sdcard into the printer's port, and turned on the printer. After a minute or two, I shut off the printer, removed the sdcard, reinserted the micro USB cable and rebooted everything. The same error came up, when I tried to invoke the adxl345 sections. Worse, the host and printer's mcu were still not same version.

At this point, it was getting old fast to keep uncommenting and commenting out the *printer.cfg* lines for the adxl345 stuff. I then realized that I had originally downloaded the *adxl-driect.cfg* file from Bassamanator. I also had added *[include ./cfg/adxl345.cfg]* in *printer.cfg*. I fixed any stuff that did not match my *printer.cfg* lines related to the adxl345 for the Pad 7. All I had to do was then uncomment *[include ./cfg/adxl-direct.cfg]* to start the adxl stuff. Of course, this did not fix any problems, only reducing the time of commenting and uncommenting lines hassle. I tried a couple of more times to flash the Klipper file, even reformatting the micro sdcard and reflashing. Nothing updated the mcu. Back to the internet for searching the issues. To make several hours frustration short, there turned out to be two problems:

Fix 1: At one point, I caught a post to someone with similar problems suggesting the problem might be the mcu will not update, if the name is the same as original. I suddenly remembered reading somewhere that on Marlin updates for the SV06, the mcu would not update, if the new *.bin* file had the same name as the previous flash update. Sure enough, changed *klipper.bin* to *klipper1.bin* on the micro sdcard, flashed the printer, and the host and mcu now had the same version. But I still got the same error as before when I invoked the adxl stuff.

Fix 2 was more insidious to figure out. Luckily, and after much searching, I happened to run into the following video:
Klipper MCU version error fix (read description for more details)
From <https://www.youtube.com/watch?v=cnthbR3tS9M>
Apparently, some sort of Linux issue could be the problem. As instructed by the video, I used *KIAUH* to rebuild some part of Linux. At last, Klipper/mainsail was happy, even with the adxl345 code enabled.

I had to decide how I was going to mount the sensor to the bed and to the extruder. There are all sorts of printable mounts available for printers, including the SV06, but they require removing extruder screws to add the sensor clamp, and may require a separate mount for the bed. After reading,
ADXL345 Mount with CR-Touch?
From <https://www.reddit.com/r/Sovol/comments/14yq0js/adxl345_mount_with_crtouch/>
and some of the embedded references, creating a special mount not seem worth doing, particularly since resonance testing would not be done often.

I had on hand 0.4" wide, 3M VHB Double side Tape, a high strength tape, from another project. I mounted the sensor to the top of the extruder; I was able to mount it with the +x, +y and +z direction labeled on the sensor module in the same directions as the printer. In the *adxl-direct.cfg* file I changed, *axis map= x,y,z* . I used a piece of 18-gauge solid core copper wire over the top of the printer gantry, to hold the spi cable out of the way, making sure to provide plenty of slack to minimize wire stress on the sensor; this was overkill and not necessary, the extruder does no move that much. However, the cable to the adxl should not have any tension on it, which could bias the frequency measurements.

The test can be done from the Pad 7 or from mainsail. From the latter, after uncommenting the *printer.cfg* ADXL345 section I did the recommended test to see if Klipper found the adxl: *ACCELEROMETER_QUERY CHIP=adxl345*. It came back with the expected report.

After reviewing the Klipper documentation, I decided to just use SHAPER_CALIBATE AXIS=X, rather than do residence tests. The process took about five minutes to complete. A series of results were printed, with the final statement suggesting, *SAVE_CONFIG*, to save the results to the *printer.cfg* file. The data is then added to the special bottom section of *printer.cfg*.

The adxl module was then carefully removed from the extruder head (The VHB tape holds very tightly) and stuck to the bed, again being sure to orient the module with the proper x, y, z, relative to the printer. The command *SHAPER_CALIBATE AXIS=Y* was run. First run attempt came back with message that max_accel needed to be < 9800 m/s^2; In *printer.cfg* changed the current maximum I described earlier from 10000 to 9000. The calibration then ran without issues, and the data saved to the config file. Shaper calibration data written to */tmp/calibration_data_y_20250129_110959.csv* file
I then uncommented the procedure to measure the resonances and ran *TEST_RESONANCE AXIS=X* and *TEST_RESONANCES AXIS=Y*, attaching the sensor to the appropriate device.

Finally, removed the adxl from the printer, and disconnected from the Pad 7.

Real Printing

Time to print a real object. I chose the standard 20 mm xyz cube, and set the conditions as my typical parameters when under Marlin. The x, y and z dimensions where within experimental error. The only change that was needed was to the first layer. Originally, with Cura. I found the Initial First Layer Horizontal Expansion to be 0.22 mm. which required a correction of -0.11 mm. However, with Klipper, I found the bottom edge was 20.85 mm instead of 20.0 mm; this meant an elephant foot of 0.425. Because I already had a -0.11 dialed in, the first layer offset was changed to -0.54.

Because I do not believe that the single cube provides a reasonable test of the x and y step calibration. I ran a personal xy calibration file, xy_step_pyramid, which produces a series of steps in the x,y plane, running from a width of 10 mm to 100 mm. The x and y lengths of each step are measured, brought into Excel and linear regression statistics run to find whether the motor steps/mm need to be changed. None of my original steps/mm needed to be changed.

Final Thoughts

Although I expected some hiccups with the changeover process, I was surprised and ultimately dismayed by how much effort it took to finally be satisfied with the Klipper installation. As for speeding up printing, I did a test changing the speed form 60 mm/s to 150 mm/s (Cura limit). There print time was reduced by 1/3. However, more or less time will depend on the object being printed.