Stealthchanger Software Programing

By: Traxman25
With parts and assistance from:
Thessien
ComputerMedic
Klipper3d.org
Viesturz – Tap Changer project
Frix-x – Shake Tune

You will need the following programs to complete this install:
Raspberry Pi Imager (https://www.raspberrypi.com/software/)
Notepad++ (https://notepad-plus-plus.org/downloads/)
WinSCP (https://winscp.net/eng/download.php)
Putty (https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)

This guide is written for a Voron 2.4 350 Stealth Changer project with 6 tool heads. The hardware on this build is:
Raspberry Pi4B
Big Tree Tech Octopus v1.1 for the main drive motors
EBB36 tool head boards running either CanBus or USB communication
For CanBus I used Big Tree Tech U2C
HDMI Screen
Big Tree Tech Filament Run Out Sensor V2.0
Orbiter V2.0 Extruders

The printer uses sensor less homing for X, and a Y switch moved into the A motor mount using this mod: https://github.com/VoronDesign/VoronUsers/tree/master/printer_mods/hartk1213/Voron2.4_Y_Endstop_Relocation

Flashing Raspberry Pi firmware:

Raspberry Pi Imager (https://www.raspberrypi.com/software/)

Select your Raspberry Pi Device (Raspberry Pi 4)

Select the Operating System -> “Other Specific-purpose OS” -> 3D Printing -> Mainsail OS -> Mainsail OS

Choose Storage -> [choose the storage device you want to flash to]

Click Next - > Yes -> Yes. The installer will now write the Raspberry Pi firmware to the SD card.

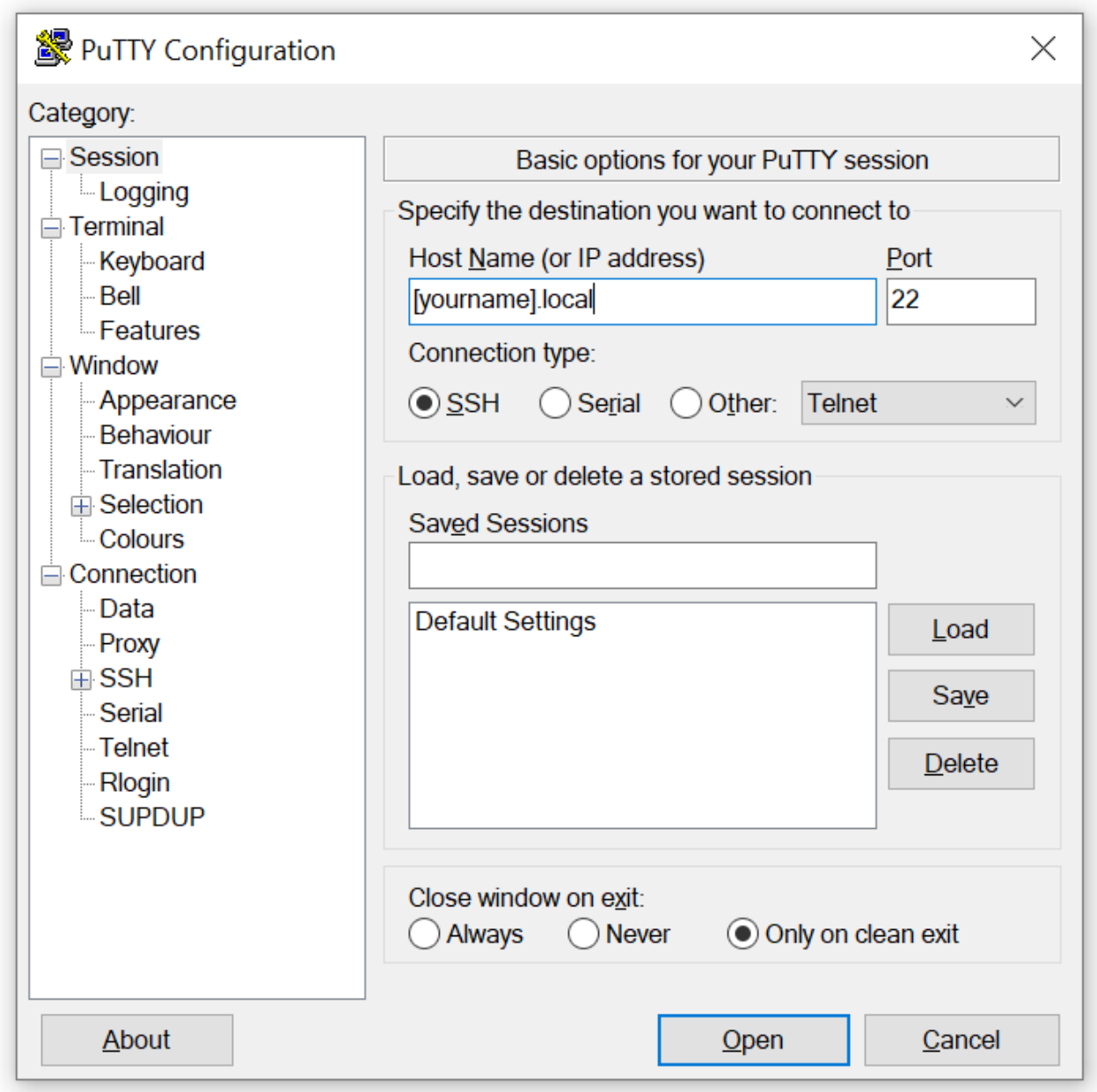
Activating WiFi – On the flash drive will be a sample file called “wpa_supplicant.conf.example” Open this file in Notepad++ (You can not use Windows Notepad to generate this file) and follow the prompts for how to setup your specific wifi information.

Disconnect the flash drive

Insert the flash drive into the Raspberry Pi and power it on.

Note: If you have issues connecting the Pi via WiFi, connect via wired connection. Once the Klipper install is completed WiFi network details can be entered via the system settings in the Klipper Screen interface.

Once the Raspberry Pi is connected to your network log into it using Putty. (Host Name [name you gave the pi].local, or the IP address of the pi)

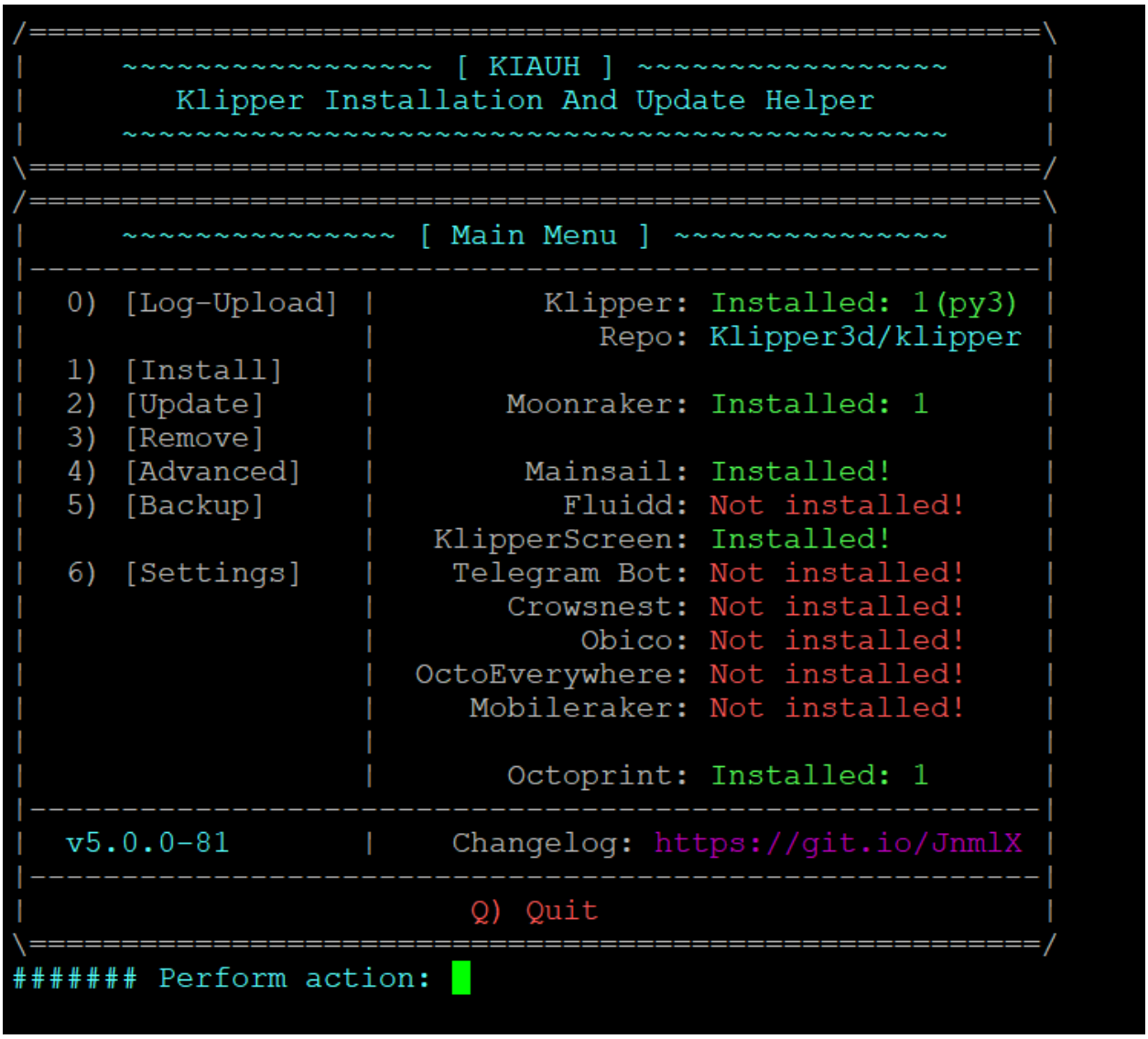


Enter the username and password you gave it while flashing the Pi OS.

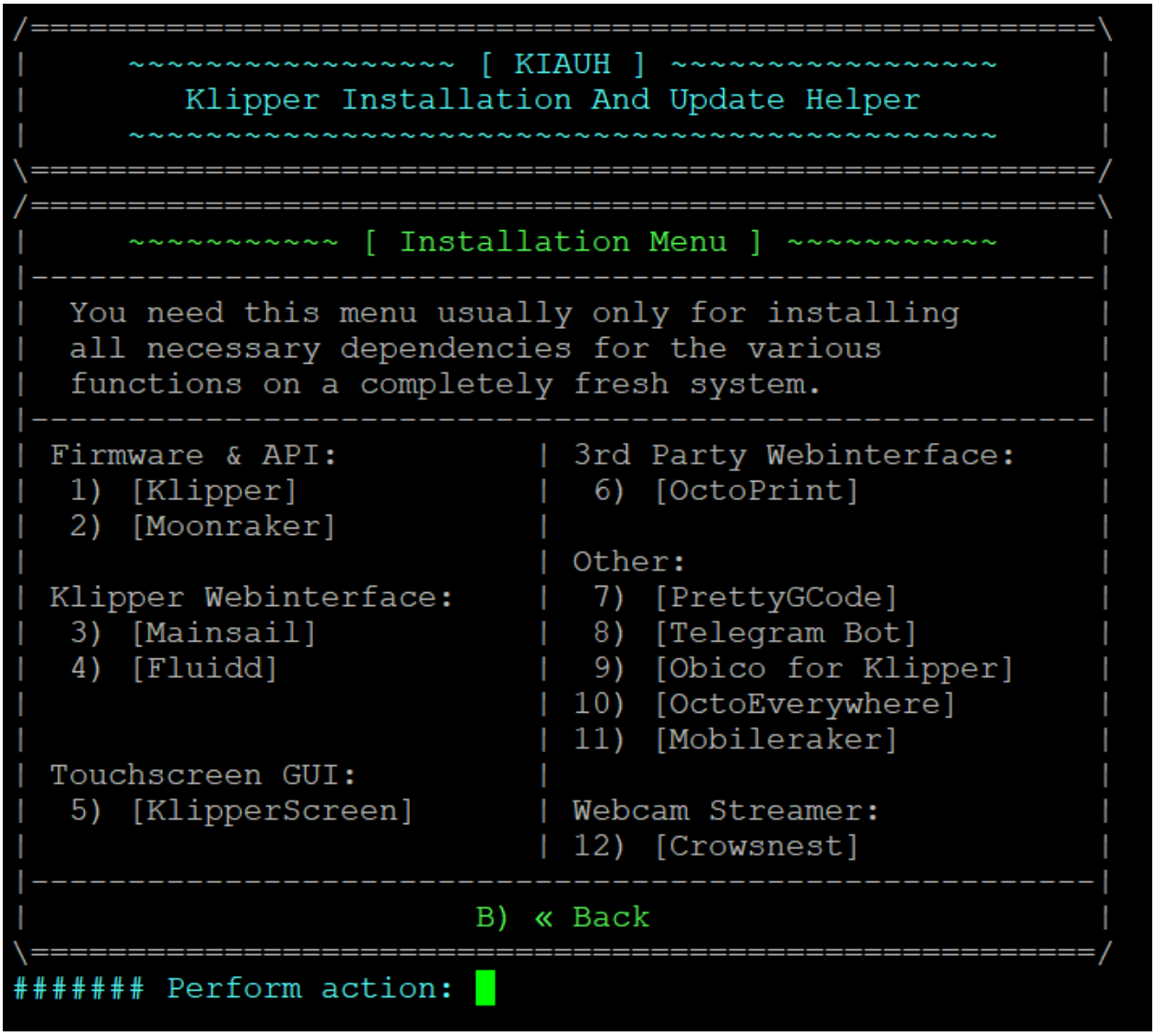
Install git using the command:
`sudo apt-get update && sudo apt-get install git`

Install Kiauh using the command:
`cd ~ && git clone https://github.com/dw-0/kiauh.git`

Run Kiauh using the command:
`./kiauh/kiauh.sh`



Press 1 enter to enter the Install menu



Install Klipper by entering 1 and following the prompts.

Complete the install by installing Moonraker, Mainsail or Fluid, KlipperScreen if you're running a touch screen and any of the other optional applications that you'd like.

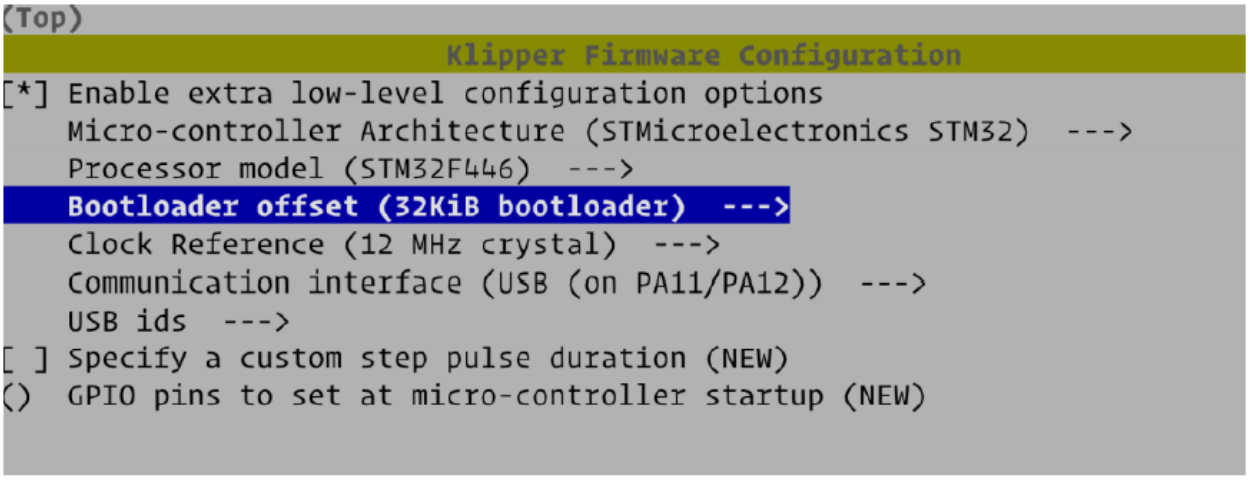
Once you're done press B to go back.

To flash the Firmware for the Octopus

Press 4 for Advanced

Press 2 for Build Only

Carefully setup the settings in the following menu according to the instructions for your control card:



Press Q to Quit.

Press Y to write the Klipper.Bin file to the SD card.

Press B, then Q to quit.

Log into the pi with WinSCP.

Navigate to /home/(pi name)/klipper/out folder.

Either move the Klipper.Bin to the root folder of the flash drive, or download it to a folder you'll remember.

Shut down the pi with the command sudo shutdown now

Remove the SD Card from the pi.

If you downloaded the Klipper.Bin file to your computer, connect the SD card to the computer and put the Klipper.Bin file onto the root directory of the SC card.

If you moved it using WinSCP you can put the flash drive directly into the control board.

Without the USB cable hooked to the pi, power up the control board. It will flash the firmware to the card in a few seconds.

Wait about a minute than power down the control card. To confirm the firmware was successfully flashed you can load the SD card and check that Klipper.Bin file name has been changed to Klipper.CUR

Reinsert the SD card into the Pi.

Hook up a USB cable from the USB A output on the Pi to the USB C input on the control card.

Boot up the pi and the control card.

ADXL RPi MCU install (Reference: https://www.klipper3d.org/RPi_microcontroller.html)

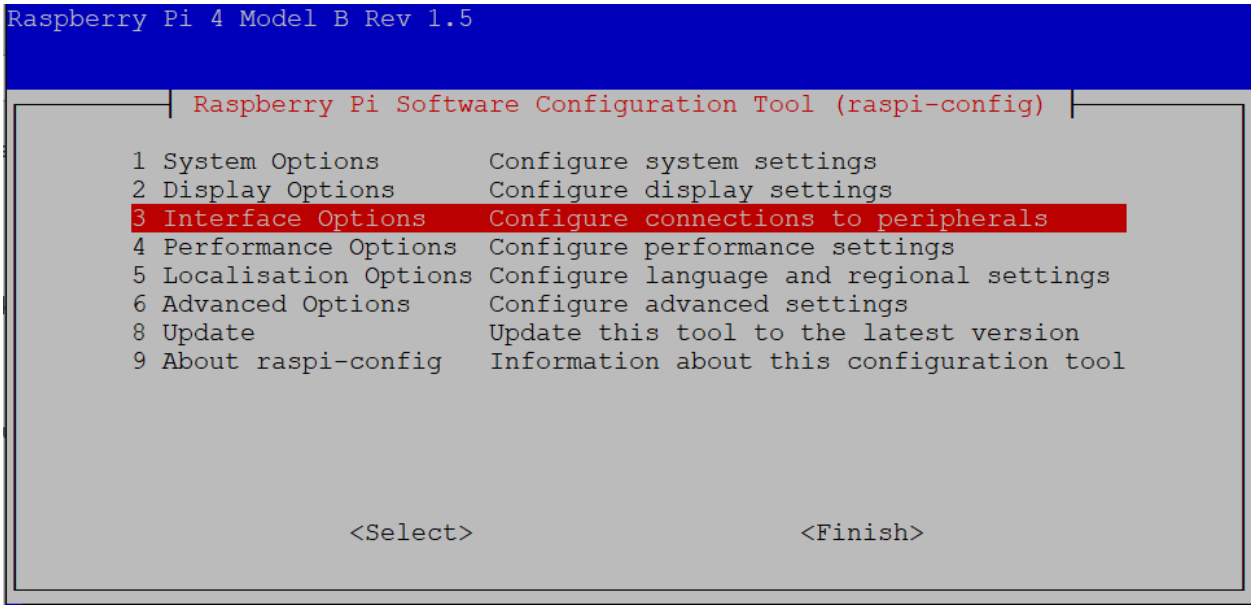
Install Raspberri Pi klipper MCU by entering the following command prompts
cd ~/klipper/
sudo cp ./scripts/klipper-mcu.service /etc/systemd/system/
sudo systemctl enable klipper-mcu.service

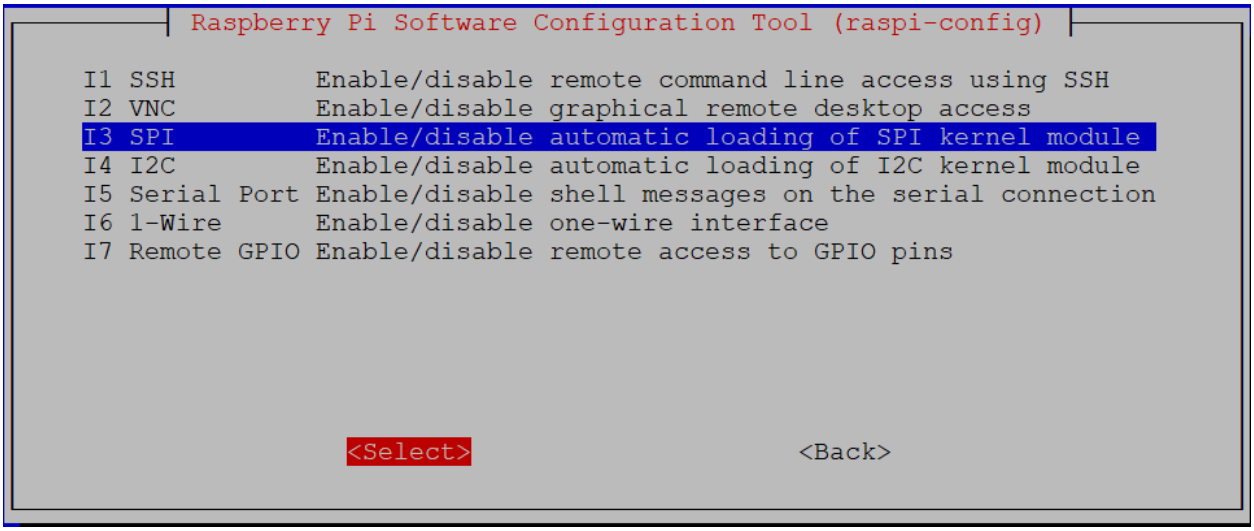
Make microcontroller code:
cd ~/klipper/
make menuconfig
Set “Microcontroller Architecture” to “Linux process” by pressing the down arrow to highlight Microcontroller Architecture, press the right arrow, down arrow down to Linnux process, press enter.
Press Q
Press Y to save

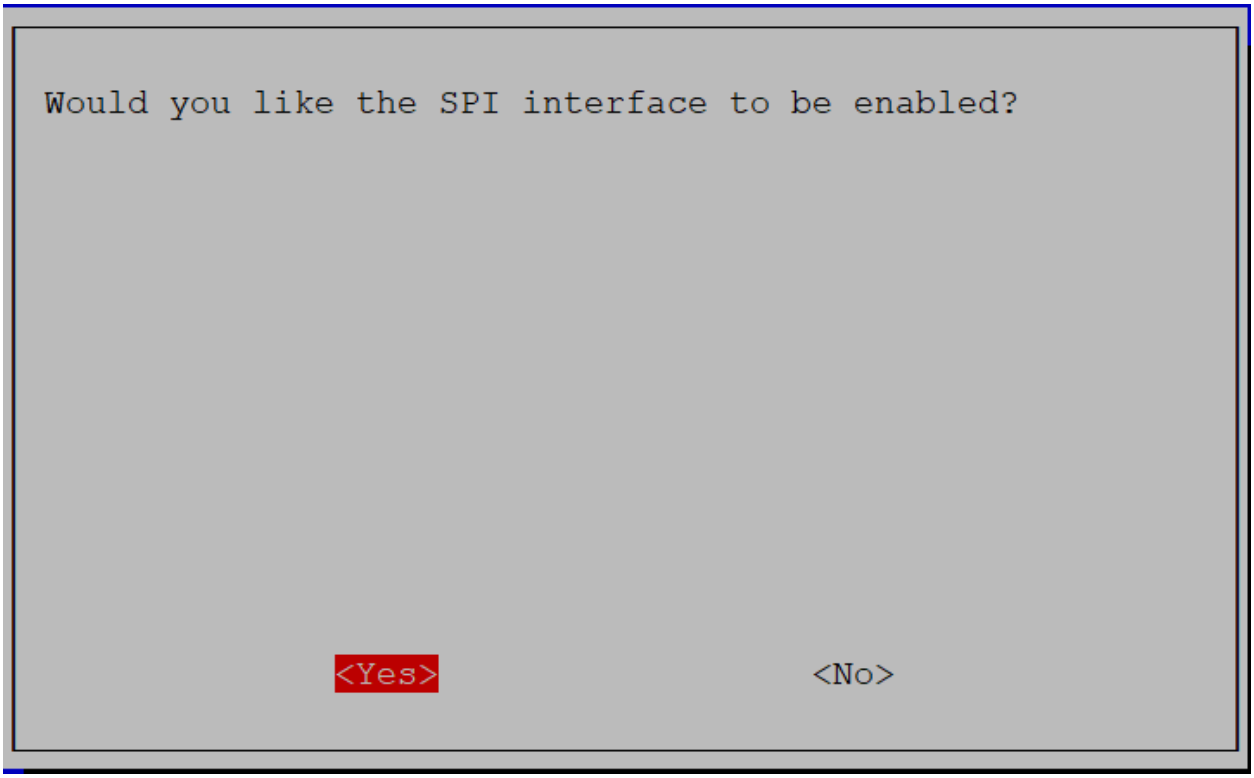
Enter command prompt:

sudo service klipper stop
make flash
sudo service klipper start

Make sure the Linux SPI driver is enabled by running:
sudo raspi-config
and enabling SPI under the "Interfacing options" menu. (arrow down to option 3, right arrow to highlight <Select> hit Enter.







Also Enable I2C while you're in this menu by going back to Interface Options and Enabling I2C.

You'll also need to configure the I2C by setting the baud rate to 400000. With the SD Card in your PC go to the boot folder, open config.txt file. Make sure the following line is uncommented or add it if it is not there:
dtparam=i2c_arm=on
and
i2c_arm_baudrate=400000

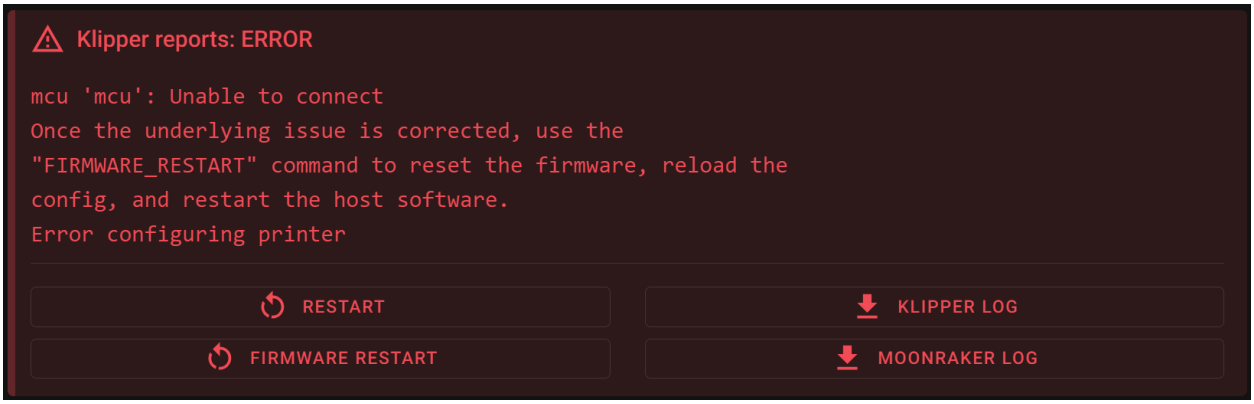
Install Shaketune Input Shaper and Tuning Module (Reference: https://github.com/Frix-x/klippain-shaketune)

In putty enter command prompt:
`wget -O - https://raw.githubusercontent.com/Frix-x/klippain-shaketune/main/install.sh | bash`

(There is an additional step we will come back to later)

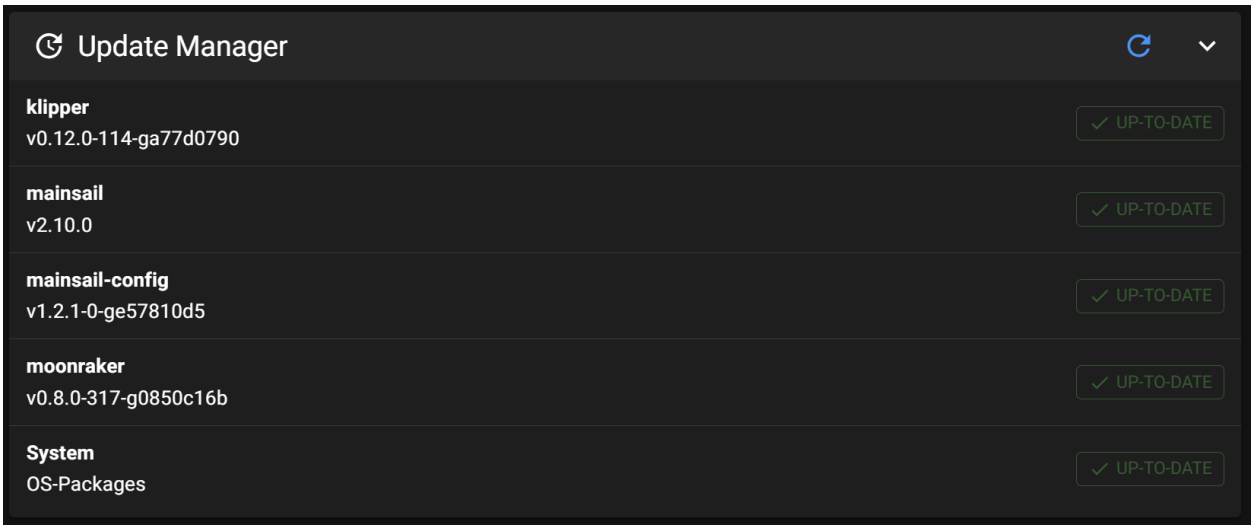
You should now have Klipper Screen working and be able to connect to the printer via a standard web interface. To log into the printer via a web browser you simple enter [printername].local or the IP address into your web browser address bar.

You will get an error message when connecting via the web browser, and on Klipper Screen. Ignore this for now.



If you wish to connect via WiFi and had issues with that before, you can now go to Menu -> Network and select the network you wish to connect and enter the password.

On the Machine Tab, it is recommended to ensure all applications are up to date



ToolChanger Klipper Extension: https://github.com/viesturz/klipper-toolchanger

Run the following command in Putty:

```
wget -O - https://raw.githubusercontent.com/viesturz/klipper-toolchanger/main/install.sh | bash
```

Next to add toolchanger to the update section of Moonraker open moonraker.conf and add the following to the end of the file.

```
[update_manager klipper-toolchanger]
type: git_repo
channel: dev
path: ~/klipper-toolchanger
origin: https://github.com/viesturz/klipper-toolchanger.git
managed_services: klipper
primary_branch: main
install_script: install.sh
```

Also be sure to add the macros.cfg to your printer from the ToolChanger repo.

Setting up printer.cfg

Starting with a stock Voron printer.cfg profile for your particular printer control board you'll need to find the printer's MCU ID. With the main printer control card plugged into the control computer, SSH into the controller and enter the command `ls /dev/serial/by-id`

```
printy@V24-350-SC2:~ $ ls /dev/serial/by-id
usb-Klipper_stm32f446xx_32001200
```

This is your [mcu] serial address. Enter it after serial: /dev/serial/by-id/ as shown below.
Next is to include all of the sub configuration files. This allows settings and configurations to be located in separate files to help keep things manageable.

Below is a list of the files needed for a 6 tool changer. Homing.cfg is the custom homing override configuration. T0 through T5 is the configs for each individual tool head. Tool_detection.cfg is Also included is the config folder for ShakeTune.

```
[mcu]
##  Obtain definition by "ls -l /dev/serial/by-id/" then unplug to verify
##--------------------------------------------------------------------
serial: /dev/serial/by-id/usb-Klipper_stm32f446xx_#########
restart_method: command
##--------------------------------------------------------------------

[include homing.cfg]
[include T0-DB-Rapido-Orbiter.cfg]
[include T1-DB-Rapido-Orbiter.cfg]
[include T2-DB-Rapido-Orbiter.cfg]
[include T3-DB-Rapido-Orbiter.cfg]
[include T4-DB-Rapido-Orbiter.cfg]
[include T5-DB-Rapido-Orbiter.cfg]

[include tool_detection.cfg]
[include toolchanger.cfg]
[include macros.cfg]
[include calibrate-offsets.cfg]
[include K-ShakeTune/*.cfg]
```

Add the [gcode_arcs], [rounded_path], and [excluding_object] sections as shown below.

```
63  [gcode_arcs]
64  resolution: 0.2
65
66  [rounded_path]
67  resolution: 0.2 # the length of a circle approximation segments.
68  replace_g0: False # Use at your own risk
69
70  [exclude_object]
```

If you're using sensorless homing on x change the x endstop_pin to *tmc2209_stepper_x:virtual_endstop*

```
89   endstop_pin: tmc2209_stepper_x:virtual_endstop
```

Also add *driver_SGTHRS: 75 # 255 is most sensitive value, 0 is least sensitive* below the stepper x section

```
[tmc2209 stepper_x]
uart_pin: PC4
diag_pin: ^PG6
interpolate: false
run_current: 0.8
sense_resistor: 0.110
stealthchop_threshold: 0
driver_SGTHRS: 75 # 255 is most sensitive value, 0 is least sensitive
```

Change Z endstop_pin to probe:z_virtual_endstop

```
157   endstop_pin: probe:z_virtual_endstop
```

Add the ADXL section shown below.

```
375  ########################################################################
376  #   ADXL
377  ########################################################################
378
379  [mcu rpi]
380  serial: /tmp/klipper_host_mcu
381
382  [adxl345]
383  cs_pin: rpi:None
384  axes_map: x, y, z
385
386  [resonance_tester]
387  accel_chip: adxl345
388  probe_points: 175, 175, 25
389
390  [input_shaper]
391
```

Add the [respond] and [force_move] sections as shown below

```
397  [respond]
398  default_type: echo
399  #   Sets the default prefix of the "M118" and "RESPOND" output to one of the following:
400  #       echo: "echo: " #(This is the default)
401  #       command: "// "
402  #       error: "!! "
403  #default_prefix: echo:
404  #   Directly sets the default prefix. If present, this value will override the "default_type".
405
406  [force_move]
407  enable_force_move: True
```

Toolhead configs

Note: Tool head configs and pin addresses work a little differently with the tool changer. Tools are numbered starting a 0, so a 6 head tool changer would have tools 0 through 5.

Each tool head board is labeled by the mcu name that is given to it in the first line of the tool head board config file. Example shown below is tool 0 labeled et0 and tool 1 labeled et1. We'll need this label as we build the pin callouts for each tool.

T0-DB-Rapido-Orbiter.cfg

```
1  [mcu et0]
```

T1-DB-Rapido-Orbiter.cfg

```
1  [mcu et1]
```

Each [function] section heading will need the tool head number after the label, except for on tool 0. Shown in the example below tool 0 extruder section is [extruder], tool 1 is [extruder1], etc.

Also each pin call out requires the *tool head board name*:pin# also shown in the example below. Example tool zero step_pin: et0:(pin#), tool 1 step_pin: et1:(pin#)

```
13  [extruder]
14  step_pin: et0:PD0
```

```
13  [extruder1]
14  step_pin: et1:PD0
```

You'll likely need to look up the pin numbers on the pin layout for the particular tool head board you're using.

With the controller powered down hook up your tool head boards, boot up the controller, SSH into it and use ls /dev/serial/by-id to obtain all of the MCU IDs. Record the IDs that aren't the main control board.

Download the toolhead sample config that matches your setup the closest from Viesturz tapchanger repo here: https://github.com/viesturz/tapchanger/tree/main/Klipper/config-example

If using Canbus for communication your MCU will be `canbus_uuid: "ID number"`
If using USB you will use the `serial: "ID"` style.

```
1  [mcu et0]
2  #canbus_uuid: 296c6b0da245
3  serial: /dev/serial/by-id/usb-Klipper_stm32g0b1x
```

You'll need to look up and enter your thermistor type.

It is easiest while setting up the printer to place the PID tune settings into the tool head config files rather than in the Save_Config in the main printer.cfg. This in case a tool head needs to be commented out then the Save_Config section does not need to be edited.

```
13  [extruder]
14  step_pin: et0:PD0
15  dir_pin: et0:PD1
16  enable_pin: !et0:PD2
17  rotation_distance: 4.715
18  microsteps: 16
19  full_steps_per_rotation: 200    #200 for 1.
20  nozzle_diameter: 0.400
21  filament_diameter: 1.75
22  heater_pin: et0:PB13
23  ## Check what thermistor type you have. See
24  ## Use "Generic 3950" for NTC 100k 3950 the
25  sensor_type: ATC Semitec 104NT-4-R025H42G
26  sensor_pin: et0:PA3
27  min_temp: -250
28  max_temp: 300
29  max_power: 1.0
30  min_extrude_temp: 170
31  max_extrude_only_distance: 150.0
32  control = pid
33  pid_kp = your-value
34  pid_ki = your-value
35  pid_kd = your-value
```

If you're running headlight LEDs you'll want to add the LED Control section into each tool head config
Sequin LEDs usually use color_order: GRBW and have the initial_white: section

LED Sequins usually use color_order: GRB and do not use initial_white: section

```
#######################
#   LED Control
#######################
[neopixel headlight]
pin: et0:PD3
chain_count: 3
color_order: GRBW
initial_RED: 0.0
initial_GREEN: 0.0
initial_BLUE: 1.0
initial_WHITE: 0.0
```

The [tool T#] section contains the tool offset and park parameters for that tool.
Tool zero will have 0 for all gcode_offset values.

Params-type is the dock path type that is used during docking moves. We will address this a bit later

params_park_x y and z are where you will enter the physical location of that tool's dock. As you're setting up the config it is a good idea to enter an approximate location of the dock. Example your first dock will usually be X=25, Z close to the max z height of your printer. The next tools will be roughly +60mm in Y. We will fine tune these values later after initializing the printer and tuning z-offset.

Params_safe_y is the Y distance that is safe to for Z move up to the level of the docks.

Params_close_y is the Y distance that is save for x moves while moving from tool to tool, after dropping off a tool in it's dock.

[tool_probe T#] is where you can adjust your probing samples, speed, retract distance, etc.
Pin is the pin number for the probe sensor
z-offset is that tools z-offset
speed is the downward z speed used during probing moves
Lift-speed is the upward z speed used to retract after the probe is triggered
samples is the number of probe samples taken at any given point
sample_retract_distance is the z height that the probe will raise between samples at a given point
sample_tolerance is the allowable tolerance between probe measurements.
activate_code Heater=extruder(tool number unless zero) For tool zero leave the number off.

toolchanger.cfg

```
[toolchanger]
  t_command_restore_axis: Z
  params_safe_y: 125
  params_close_y: 40
  params_fast_speed: 20000 # Go as fast as we can ***30000***
  params_path_speed: 900 # 20mm/s for the actual change
  params_type = 'rods_mini_hookon'   # See available paths below
  # The dropoff path of each tool type, pickup path is the same in reverse.
  params_pads_mini_path: [{'y':9 ,'z':2}, {'y':8, 'z':0}, {'y':0, 'z':0, 'f
  params_pads_sb_path: [{'y':9.5 ,'z':8}, {'y':9.5, 'z':2}, {'y':5.5, 'z':0
  params_rods_mini_path: [{'z':0, 'y':4}, {'z':0, 'y':0, 'f':0.5}, {'z':-6,
  params_rods_sb_path: [{'y':9.5 ,'z':8}, {'y':9.5, 'z':2}, {'y':5.5, 'z':0
  params_rods_mini_hookon_path: [{'z':0, 'y':4}, {'z':0, 'y':0, 'f':0.5}, {
```

Params_safe_y is the y distance that a loaded tool will safely clear the back of the parked tools. This will be used as the tools approach the dock from below.

Params_close_y is the y distance that is safe for an unloaded carriage to clear the back of docked tools. This will be used to move from dock to dock while changing tools. ******NOTE: The current sample config has this set to 15, please change it to at least 40 to prevent tools crashing.*** This is a global coordinate location, not an incremental move. So if you don't change this the carriage will likely crash back into the tool after dropping it off.

Params_fast_speed is the rapid speed used during docking (mm/m). ****** Note: It is recommended to turn this speed down to 1000 while tuning the docking moves.*** You can turn it back up once all the docks are tuned and proven to work. This speed is in mm/minute, divide by 60 for mm/s.

Params_path_speed is the slow speed used during docking and pickup moves. This speed is in mm/minute, divide by 60 for mm/s.

Params_type = '###' Enter the tool path name to be used by your tool/dock combination. The path name is the params_(type)_path line below

Printer Startup

Voron Designs has an excellent writeup for all the steps to initialize your new printer, available here: https://docs.vorondesign.com/build/startup/

Follow that guide to initialize your printer until you get to the Z-offset portion of the setup.

To set Z-offset, have tool 1 loaded onto the tool carriage. Home, quad gantry level and rehome the machine. Then move the print head to the center of the print bed. With a piece of paper on the bed plate carefully move the z axis down using the move commands until the nozzle causes a slight amount of drag on the paper. You should be able to push the paper under the nozzle. Also Be careful to not use a command that moves too far.

Note the z axis position and enter this as your z-offset position in that tools' config file under the [tool_probe] - z_offset

```
[tool_probe T0]
pin: et0:PB6
tool: 0
z_offset: -0.825
```

Repeat these steps of hand loading each tool head you have to set the z-offset for each tool head. Be sure to run a quad gantry level and homing after swapping each tool head.

Once all of the z-offsets are complete use save and restart to restart Klipper.

Dock Parking Location Alignment

Ensure that approximate park locations are setup in each tool's config file.

Load Tool 0 onto the carriage, home, quad gantry level and rehome the printer.

Set the Speed factor to 50%

Run the Tool Align Start Macro – the tool head will move to the params_park_x and z values that are entered in that tool's config. The tool head also move to 100mm more positive than params_park_y.

Caution: Be sure to look from all directions to ensure the tool is not crashing during this step Use the move commands to carefully move the tool into the dock. This location needs to be very precise so take your time to get the tool in just the right place. Most docking paths consider the parking location where the tool slides onto the pin/bolt. Example the hook-on dock park location is when the x and z locations line the head of the bolt up with the open hole in the tool face, and y location with the tool face against the dock.

Record the x, y and z location.

Tool Align Done will move the tool +100 y.

Move the tool to the center of the bed at Z=25mm.

Enter the X, Y and Z locations for the tool you just aligned into the params_park_x y and z in the tool config for that tool.

Save and Restart the printer.

Home, Quad Gantry Level and rehome with T0.

Set the speed to 50% or less.

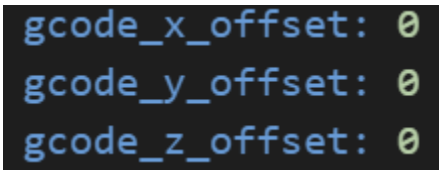
Run TEST_TOOL_DOCKING to test the T0's parking move path and parking location are correct.

Manually place the next tool onto the carriage.

Repeat these steps for all remaining tools.

Tool G-code offset alignment.

G-code offset alignment is to adjust for slight variations in each tool's assembly, aligning all tools to each other for clean prints. No matter the method T0 should almost always have gcode_offsets for x, y and z at 0.



With any of the 3 methods outlined below take note of the offset for each tool and enter those offsets in the tool's config file gcode_(axis)_offset setting.

Option 1: kTAMV - Klipper Tool Alignment (using) Machine Vision
kTAMV uses a small circuit board based web camera hooked up to your printer to look at each nozzle and measure the offset between T0 nozzle and the other tool's nozzle. Instructions for kTAMV are available here: https://github.com/TypQxQ/kTAMV

Follow the steps there to get the offset measurements and take note of those offsets.

Option 2: NozzleAlign
NozzleAlign is a mostly 3D printed nozzle probe ball with a script to measure the nozzle position. More information and instructions are available here: https://github.com/viesturz/NozzleAlign?tab=readme-ov-file

Option 3: Print alignment prints
This method is to print test prints with T0 and each other tool. Then make adjustments to the gcode_offset values based on those prints and repeat the process.

Two options for test prints are available on Printables here:
https://www.printables.com/model/109267-nozzle-alignment-assist
https://www.printables.com/model/201707-x-y-and-z-calibration-tool-for-idex-dual-extruder-

A test print of stacked cubes will help ensure all of the nozzles are aligned.