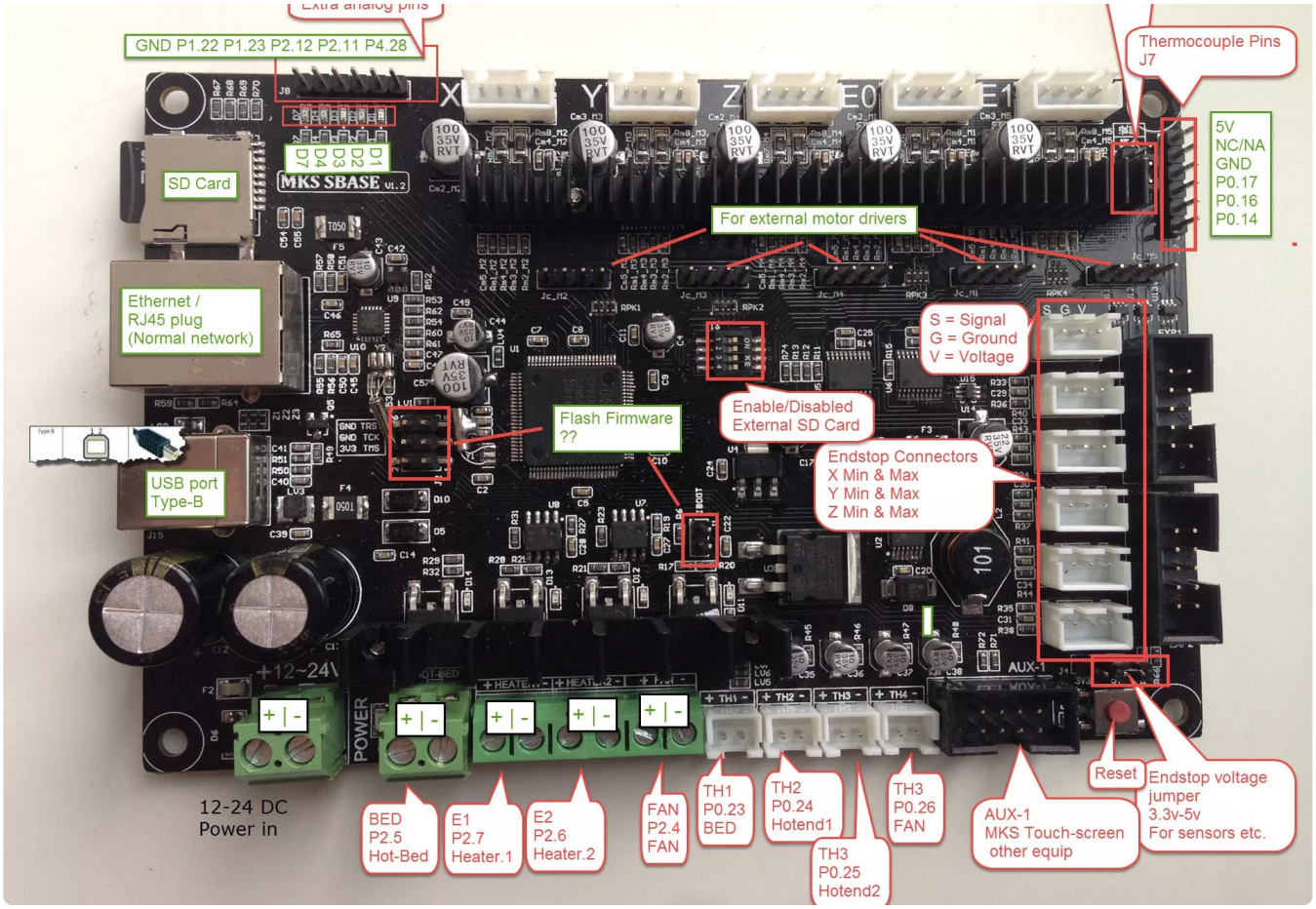
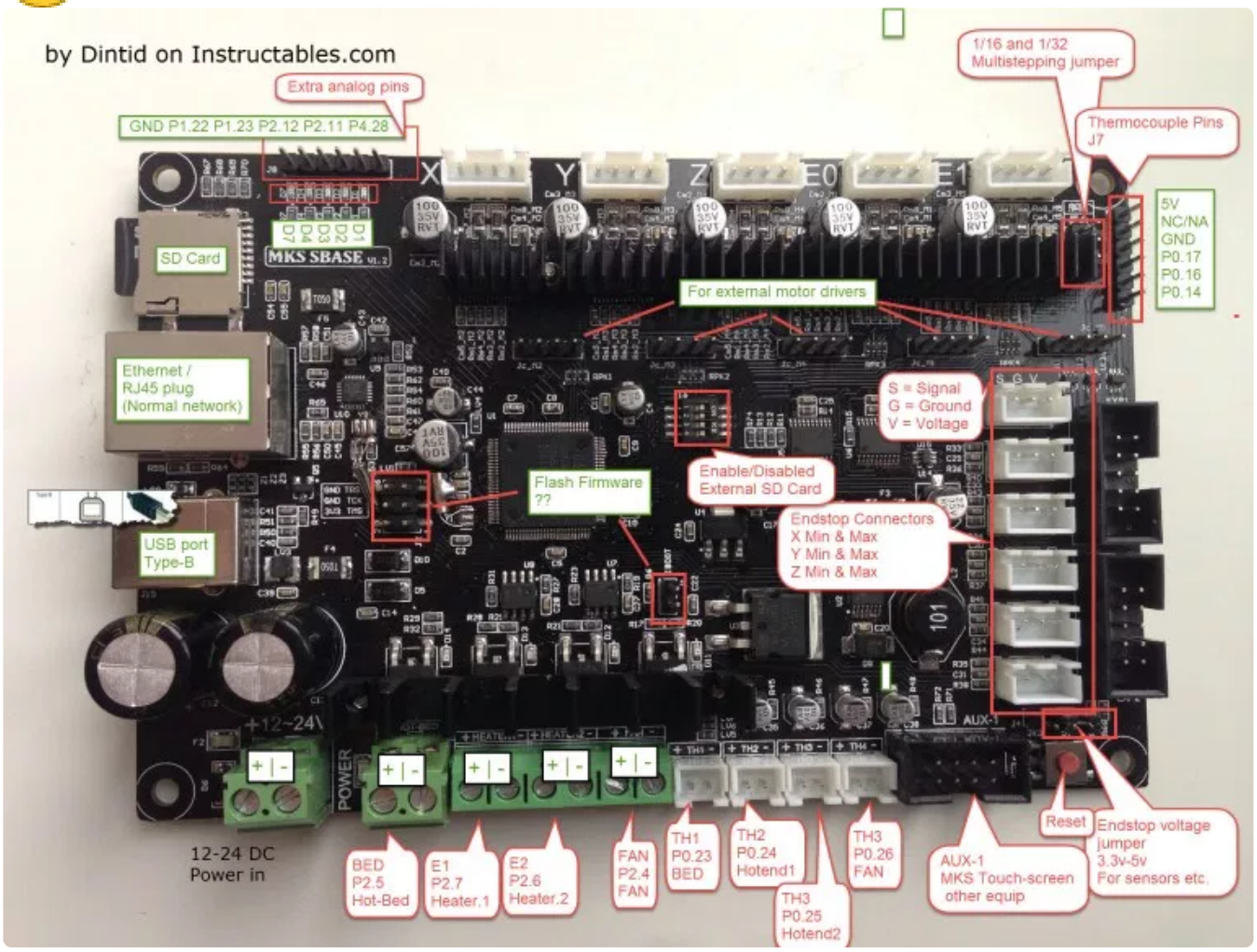
Posted on **26. May 2017** by **BeTrue3D** — **2 Comments**

Smoothieware – MKS Sbase – Controller Basics and Intro



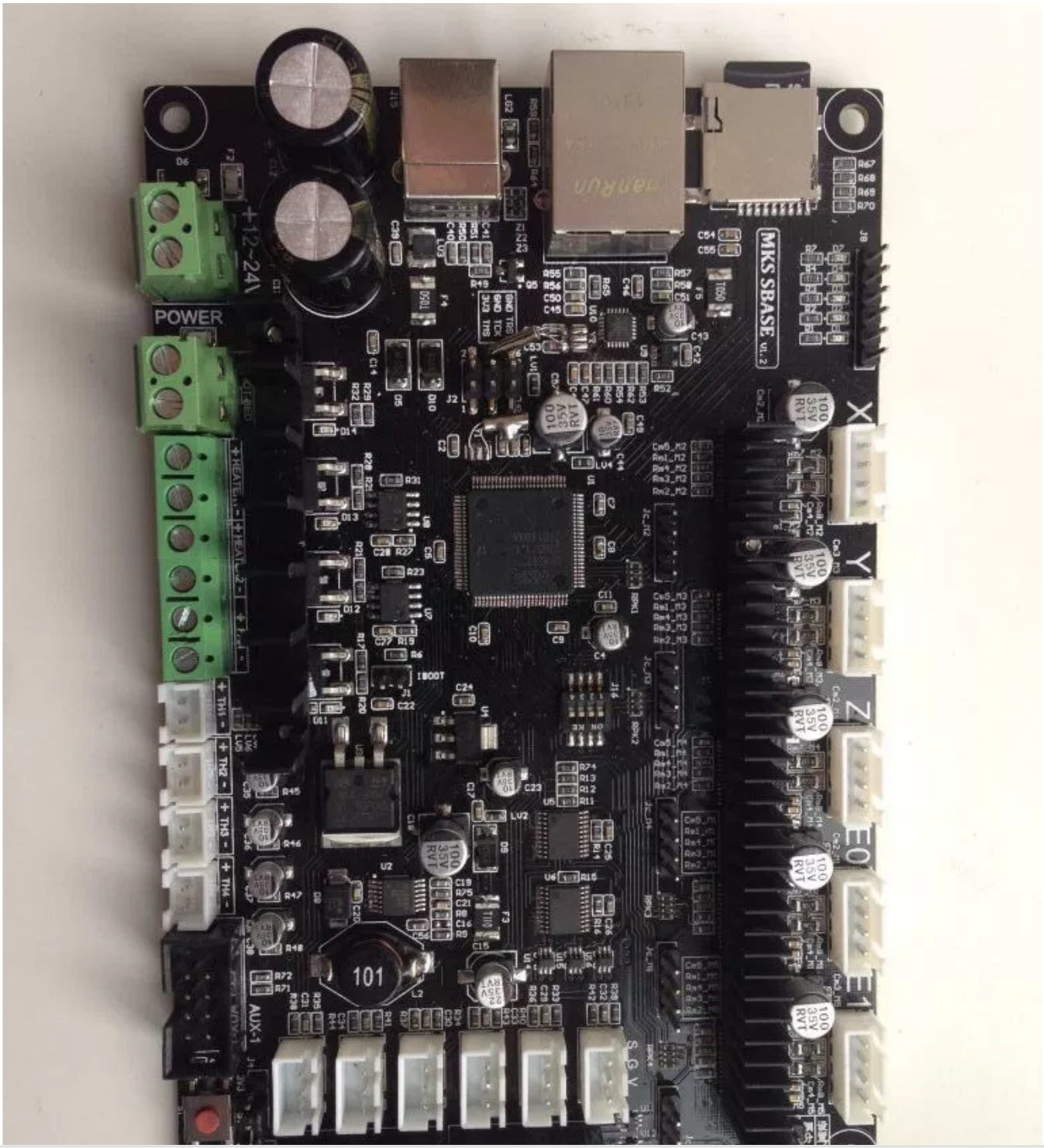
Configuring MKS Sbase V1.x 32-bit Controller Basics and Intro to Smoothieware.

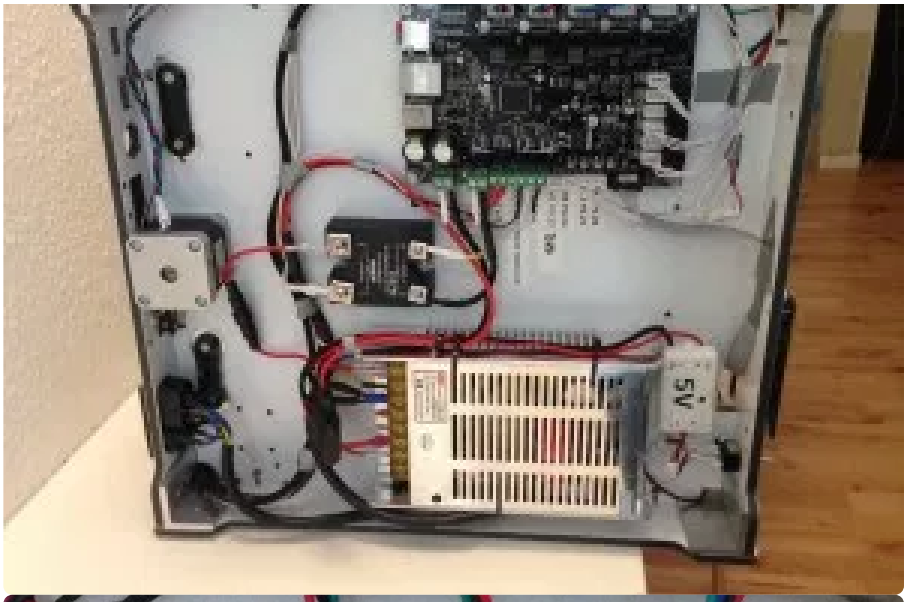
I originally posted this article on Instructables.com under my

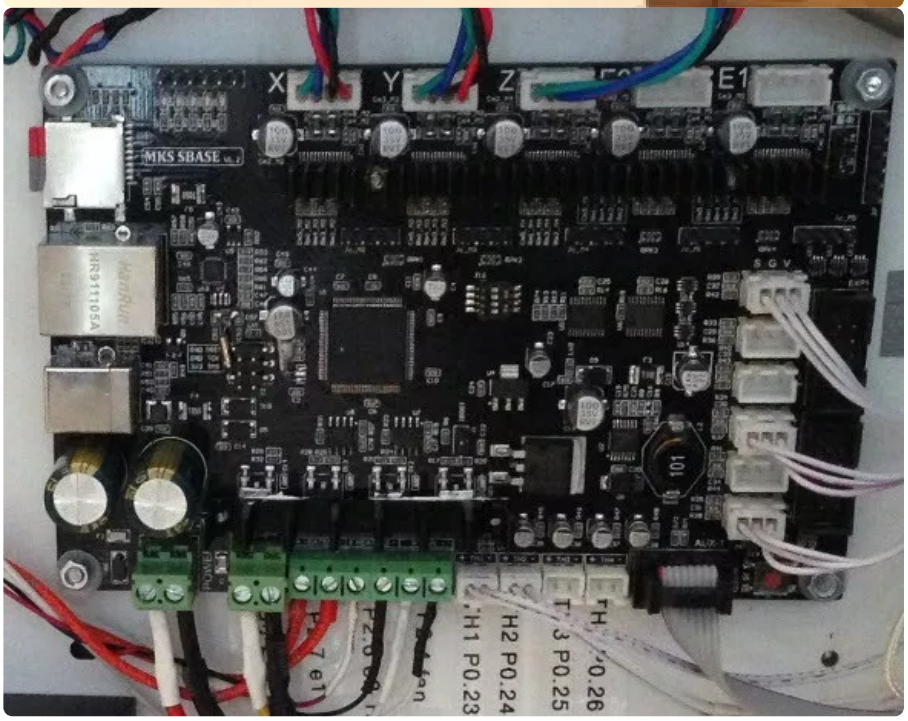


Prologue

I'm really pleased to see how this basic intro into Smoothieware and the MKS Sbase 1.x controller have gotten so much attention, and really hope it's going to help people get started or decide not to







What this is about:

Ultimaker 2 clone with a single extruder.

What this is not going to be:

I am not going to go into the hardware of the board, aside from the need to connect relevant cables.

I am not going to do anything advanced in the firmware/configuration

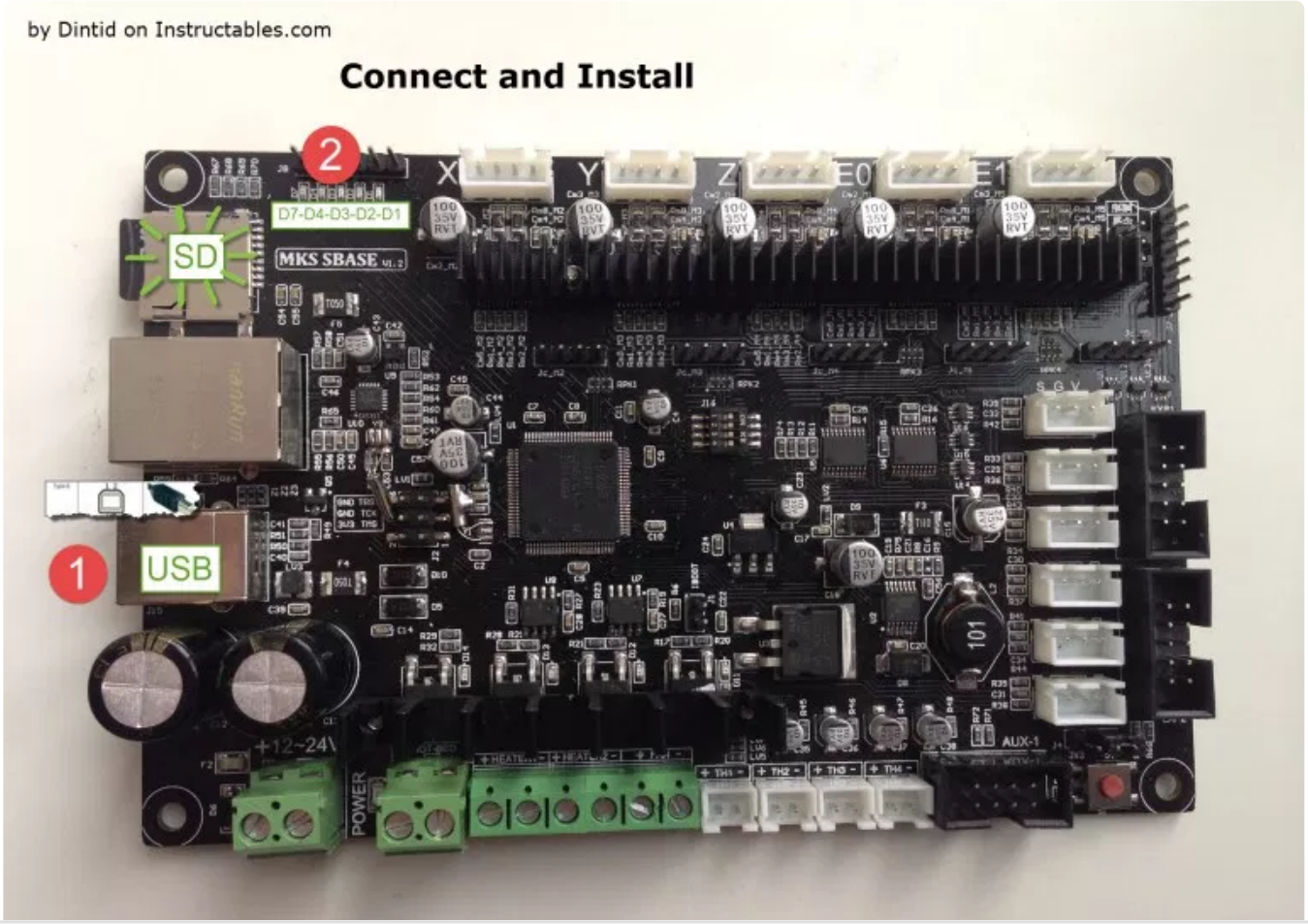
Table Of Contents:

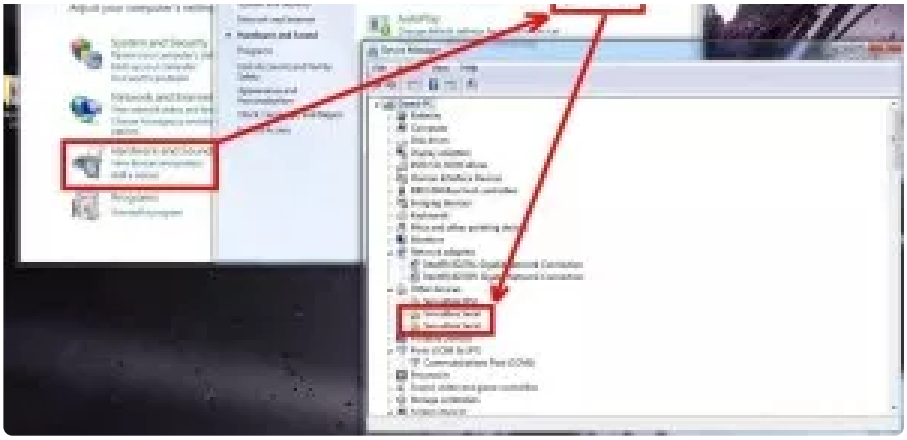
1. Connect and install
    1. Connecting USB
    2. Installing firmware
2. Configuring Smoothieware compared to Marlin
3. Obvious difference from Marlin
    1. Less obvious differneces
    2. Firmware and Config file(s)

3. External Motor Controllers
5. Firmware Step 2: Cartesian axis speed limits, pins and current
    1. Pins
    2. Current
6. Firmware Step 3: LCD, SD and Extruder
    1. LCD and SD
    2. Extruder Setup
    3. Delta driver current
7. Firmware Step 4: Hotend temperature control configuration
    1. Thermistor Type
    2. PID Tuning
    3. Hotend Thermistor – Physical Layout
    4. Hotend Heater Pins – Physical Layout
8. Firmware Step 5: Heated bed temperature control configuration
    1. Thermistor Type
    2. Temperature Control Bed BANG-BANG
    3. Heated bed thermistor – Physical Layout
    4. Heated bed heater pins – Physical Layout

10. Firmware step 7: Network Settings
11. Step 8: What's next?
    1. More advanced setup?
    2. Using Switches

Step 1: Connect and Install





In order to install the drivers for MKS Sbase v1.x (opens shop on Aliexpress) board on our computer we first need to go and download the necessary drivers from the makerbase makerbase-mks on Github.com

Note: *You do not want to use MKS firmware as it is redacted. Always use firmware from Smoothieware. If confused about the name "Repository", then just look at it as a folder of files.*

Click the file and click the *Download* in Github and save it somewhere on your computer, then extract it to a folder.

Connecting USB

while **D2** and **D3** blinks rapidly.

Open device manager and update drivers for the new *Smoothie/Seriel USB device* with the signed drivers you just downloaded from Github located in the easy to find folder: *\MKS-SBASE\MKS-SBASE\Driver\smoothieware-windows-signeddriver-v1.0*

Mine installed fine on Windows 10 x64 pro.

Installing firmware

1. **Firmware** from MKS are just and old copy of Smoothieware, so we always just want to use the newest version from Smoothieware.
    1. In other words: **always use firmware from Smoothieware**.
    2. If you can't find *config.txt* file from original smoothieware,

working fine, you can use the **Nightly** version instead. It is the newest version, but not fully tested yet, and considered as **beta/test**.

3. Insert the SD Card into the MKS Sbase board and plug in the USB Cable. Your SD Card is now going to show in your file-Explorer.
    1. If not, you need to make sure you have the Drivers installed.
4. If you can't find the ***config.txt*** file from Smoothieware, just locate the ***config.txt*** in *\MKS Sbase\MKS-SBASE-master\MKSSBase-firmware* and copy it to your SD Card.
    1. Smoothieware suggest disabling auto-Mount on the SD when connecting to USB. Especially when using a MAC, as MAC OSX tends to do funky stuff at strange times.
    2. I have changed nothing on my win10. And nothing bad has happened the past year.
5. The D7 lights up, shortly after D1 follows. D2, D3, D4 blinks and then D4 turns steady while D2 and D3 continues blinking.
    1. At this point, the **firmware.bin**changed had changed to

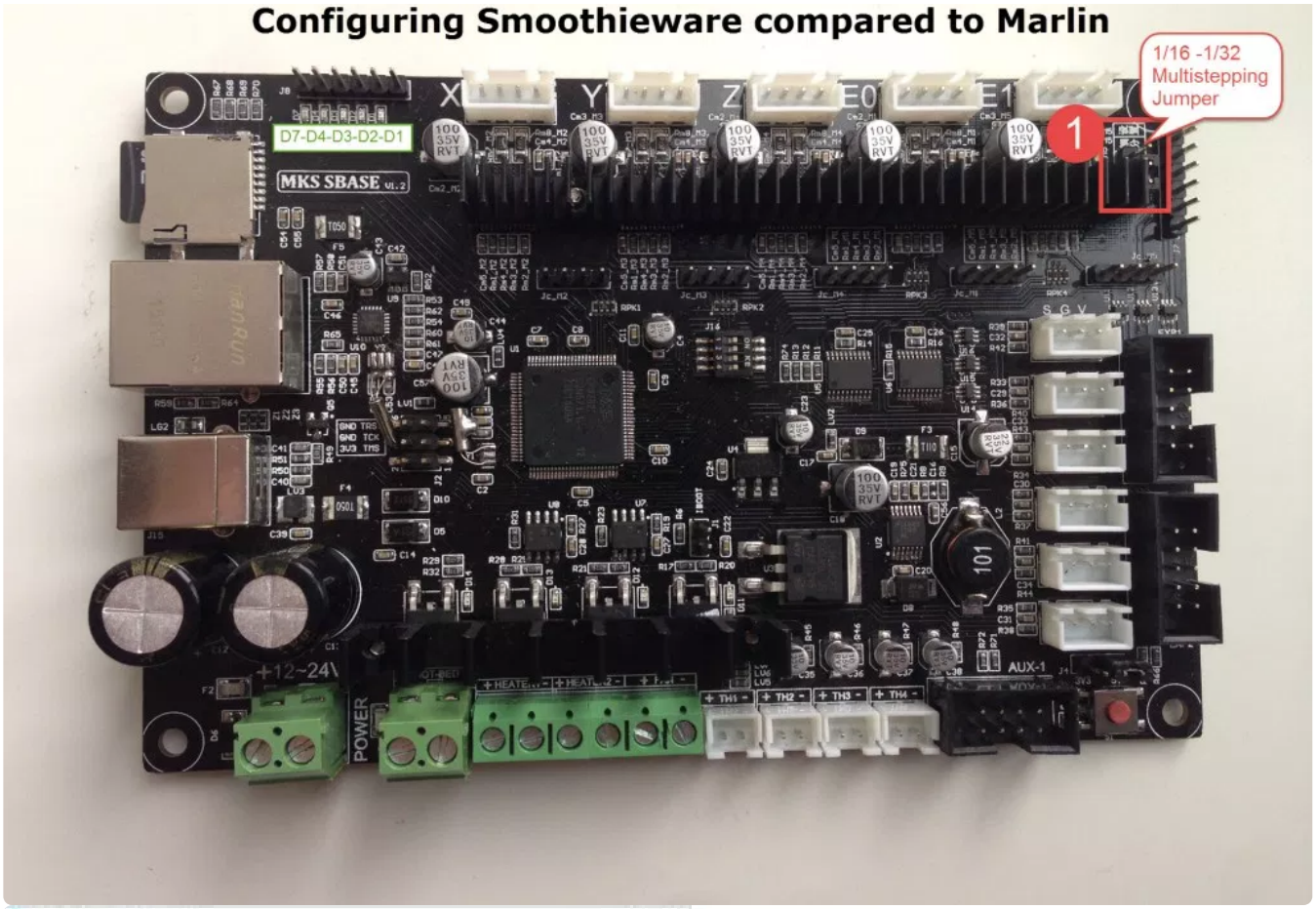
Changing or Updating Firmware

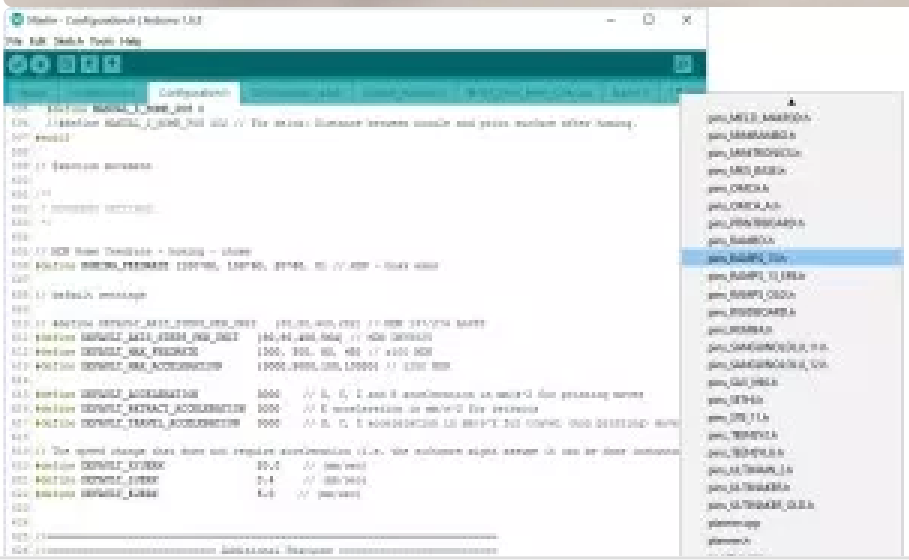
Just delete the **firmware.cur** file from your SD, or rename to *firmware.cur.old* or similar, and copy on the new **firmware.bin** file to your SD. Powercycle your printer (also unplug USB) and you can see your new firmware.cur file on your SD card.

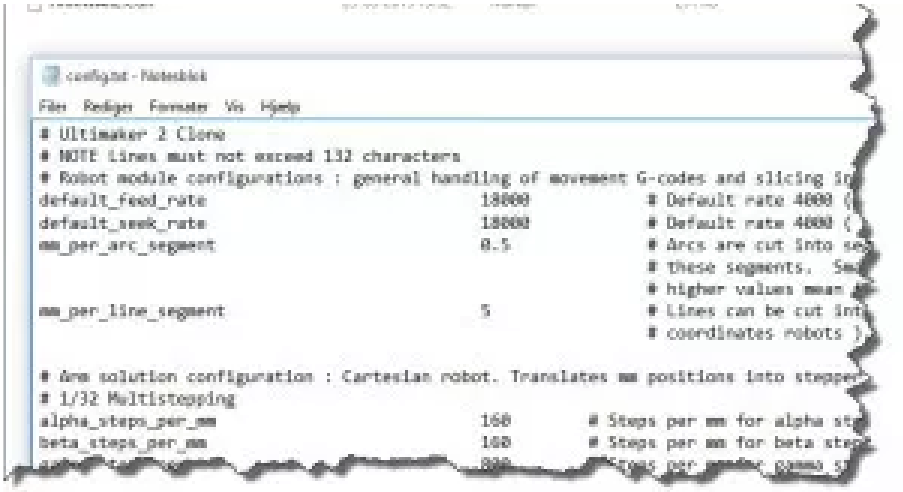
Note:*After making any changes to your config.txt file in the future, you need to power-cycle your controller, meaning disconnect both power (if in use) and USB. You can send a reset command, but only through true terminal use and not through Printrun/Pronterface or similar.*

You can use the *edge firmware* which is eseentially a realease candidate, which has more features, but also might contain some errors. This goes beyound this Instructables.

Step 2: Configuring Smoothieware Compared to Marlin







I've been using Marlin firmware on Arduino+Ramps for a long while, same as the majority of 3D people, so I'll use Marlin as comparative source for configuring firmware.

Obvious difference from Marlin

Configuring *Smoothieware* is done in a single ***config.txt*** file which is saved directly to the SD card, using notepad instead of a big program like Arduino IDE. The firmware settings are updated by power-cycling the printer or sending a reset command to it. This compared to Marlin with a very large amount of sketches (files) we

There are some less obvious differences between Marlin and Smoothieware, where the most notable are:

1. Speeds are rather confusingly listed as a scattering of mm/*sec* and mm/*min* in Smoothieware and Marlin, so have to be aware of this. No clear pattern is in place.
2. The axes in Marlin are listed as XYZ and Extruder while the axes in Smoothieware is listed as Alpha, Beta, Gamma and Delta/E0 and E1

Firmware and Config file(s)

As you might have gathered, the Smoothieware firmware is based on 2 files. A **firmware.bin** and a **config.txt** file. There is a *base config.txt* file and a bunch of preconfigured configuration files present in the repository from MKS, located in a subfolder. The Config file can also be named without the .txt extension.

Configuring firmware

First make a backup of your config.txt file, then open *config.txt* using notepad. Disable "*Word wrap*" as it might confuse you some.

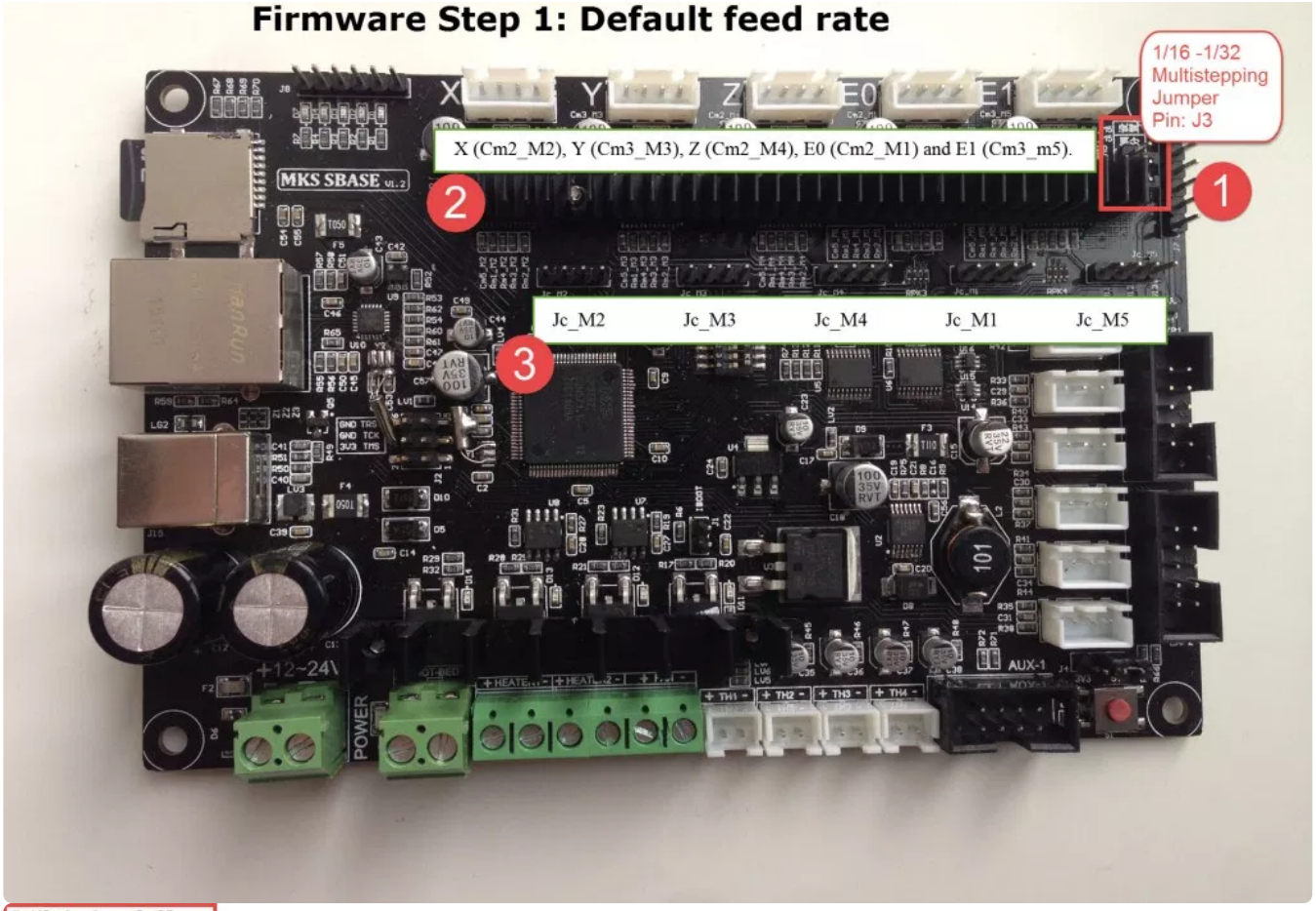
Note: *Smoothieware warns against Notepad++.*

Also remember to take your microstepping multiplier into account. My setup here is based on 1/32 multistepping to take advantage of the build in DRV8825 drivers. Multistepping jumper is placed in upper right corner. (1)

Comments

You can make your own comments by writing # at the start of your line and then your text. This is very usefull to remember what something is about.

I'm adding a line to the start of the file: # Ultimaker 2




```
# Ultimaker 2 Clone
# NOTE Lines must not exceed 132 characters
# Robot module configurations : general handling of movement G-codes and slicing into moves
default_feed_rate                          18000          # Default rate 4000 ( mm/minute ) for G1/G2/G3 moves
default_seek_rate                          18000          # Default rate 4000 ( mm/minute ) for G0 moves
mm_per_arc_segment                         0.5            # Arcs are cut into segments ( lines ), this is the length for
                                                          # these segments.  Smaller values mean more resolution,
                                                          # higher values mean faster computation
mm_per_line_segment                        5              # Lines can be cut into segments ( not usefull with cartesian
                                                          # coordinates robots ).

# Arm solution configuration : Cartesian robot. Translates mm positions into stepper positions
# 1/32 Multistepping
alpha_steps_per_mm                         160        # Steps per mm for alpha stepper
beta_steps_per_mm                          160        # Steps per mm for beta stepper
gamma_steps_per_mm                         800        # Steps per mm for gamma stepper

# Planner module configuration : Look-ahead and acceleration configuration
planner_queue_size                         32             # DO NOT CHANGE THIS UNLESS YOU KNOW EXACTLY WHAT YOU ARE DOING
acceleration                               300 # 600       # Acceleration in mm/second/second.
```

I have 300mm/sec in Marlin, which translates to 18000mm/min, so

changed to 160 while gamme_steps_per_mm is changed to 800.

acceleration is changed to 300 and z_acceleration to 80

1/32 multistepping

All of the above are 1/32 multistepping. In order to use 1/32 you also need to set the correct jumper on your Controller. It is located almost at the upper right corner with **16|32** next to it. The pins are named **J3**.

Note:*Do not change any jumper with power on. Including connected USB cable*

Connecting Motors

There is really not much to it. You need to have some 4 pin XH 2.54

If you need to hook up some external motor controller, you can do it using the pins just below the big heatsink over the drivers.

The pin-outs are labeled *Jc_M2*, *Jc_M3*, *Jc_M4*, *Jc_M1* and *Jc_M5* listed in order of above motor conenctors.

Step 4: Firmware Step 2: Cartesian Axis Speed Limits, Pins and Current

```
# Cartesian axis speed limits
x_axis_max_speed                        18000     # default 10000 mm/min
y_axis_max_speed                        18000     # default 10000 mm/min
z_axis_max_speed                        4800      # default 200 mm/min

# Stepper module pins ( ports, and pin numbers, appending "!" to the number will invert a pin )
# My AlphaBetaGamme - XYZ motors are all rated at 1.7amp
# Defaults are 1.2amp for XY and 1.5amp for Z
alpha_step_pin                          2.0       # Pin for alpha stepper step signal
alpha_dir_pin                           0.5       # Pin for alpha stepper direction
alpha_en_pin                            0.4       # Pin for alpha enable pin
alpha_current                           1         # X stepper motor current
alpha_max_rate                          18000     # Default is 10000.0 mm/min

beta_step_pin                           2.1       # Pin for beta stepper step signal
beta_dir_pin                            0.11      # Pin for beta stepper direction
beta_en_pin                             0.10      # Pin for beta enable
beta_current                            1         # Y stepper motor current
```

setting. This is a good idea in case (when) you need to make changes.

I have 300mm/sec on my Ultimaker 2 though, so I'll put in 18000 for XY and 4800 for my Z axis.

Pins

I'm not changing any pins at the moment, but we have to type in the **_max_rate** for each axis, which on a cartesian printer i the same as **_axis_max_speed**. This is not the case for Delta printers.

So for **alpha_max_rate** and **beta_max_rate** I enter *18000* and for **gamma_max_rate** I enter *4800*.

Take note of the **!** listed after each **_dir_pin** – this denotes the direction of an axis, which means you can remove or add **!** if your axes is going the wrong way.

My XYZ Nema motors are rated at 3.2 voltage and 1.7amp.

If I wanted to feed my motors max current, I'll set the **_current** to *1.7* amp, but the motors would run rather hot then, so I'll try out using the 1 instead which should be plenty.

I have set my Z-motor to *0.8* amp as it really isn't doing much Work, translating to: **gamma_current 0.8**

Step 5: Firmware Step 3: LCD, SD and Extruder

```
# Extruder module configuration
extruder.hotend.enable                  true    # Whether to activate the extruder module at all. All configuration is
extruder.hotend.steps_per_mm            564     # Default is 90 steps per mm for extruder stepper
extruder.hotend.default_feed_rate       2700    # Default is 600 ( mm/minute ) for moves where only the extruder moves
extruder.hotend.acceleration            1000    # Default is 500 # Acceleration for the stepper motor, as of 0.6, arbi
extruder.hotend.max_speed               45      # Default is 50 mm/s

extruder.hotend.step_pin                2.3     # Pin for extruder step signal
extruder.hotend.dir_pin                 0.22!   # Pin for extruder dir signal
extruder.hotend.en_pin                  0.21    # Pin for extruder enable signal

# extruder offset
#extruder.hotend.x_offset                0       # x offset from origin in mm
#extruder.hotend.y_offset                0       # y offset from origin in mm
#extruder.hotend.z_offset                0       # z offset from origin in mm

# firmware retract settings when using G10/G11, these are the defaults if not defined, must be defined for each extruder if nc
extruder.hotend.retract_length          4.15    # Default is 3. Retract length in mm
extruder.hotend.retract_feedrate        45      # retract feedrate in mm/sec
```

Extruder setup

I am not using the original extruder, so my numbers will be different here!.

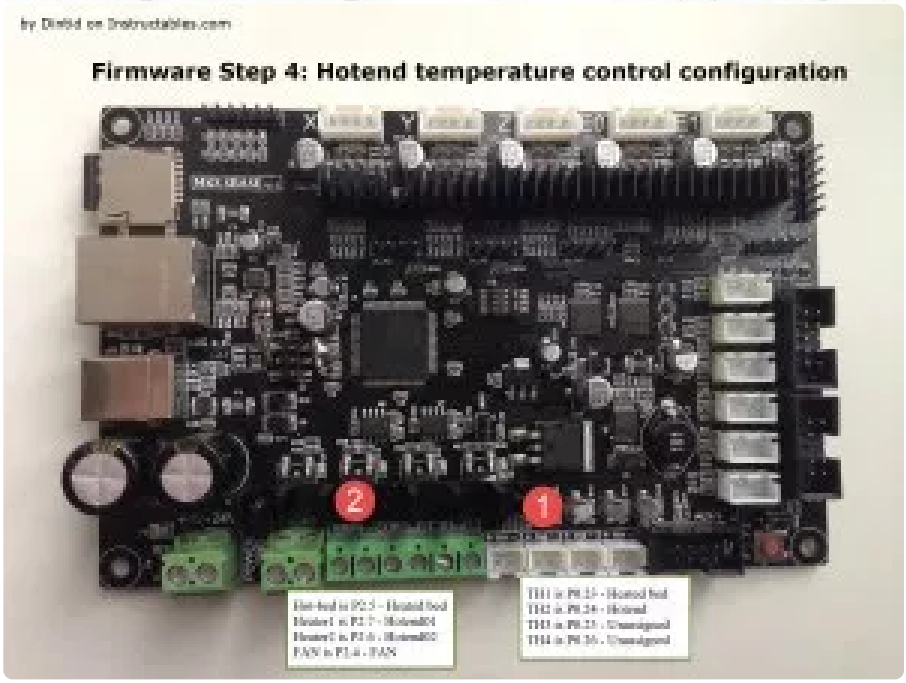
I have uncommented two settings here: **extruder.hotend.retract_length** and **extruder.hotend.retract_feedrate** but I must admit I do not know wheter this is necessary or not, since we configure this in our slizer in any case.

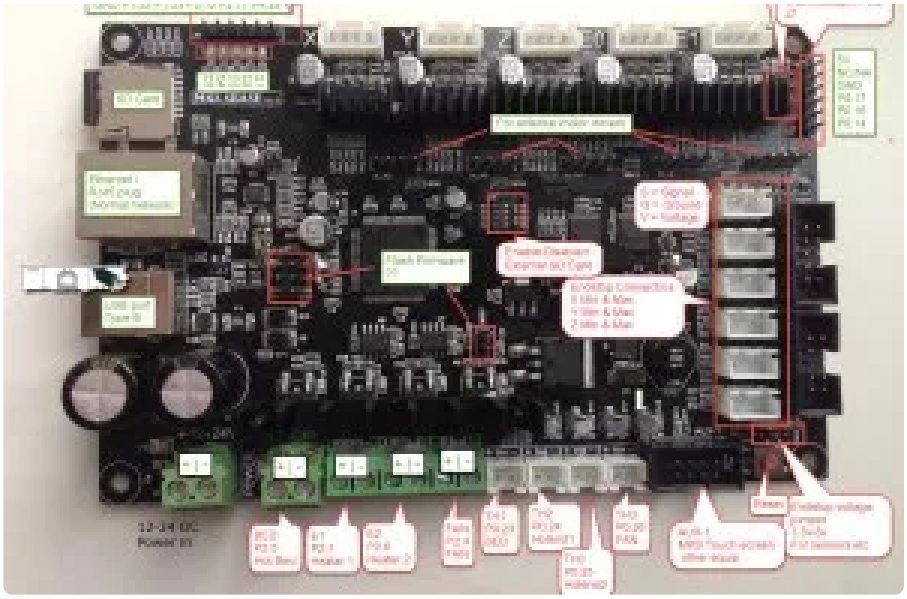
Delta driver current

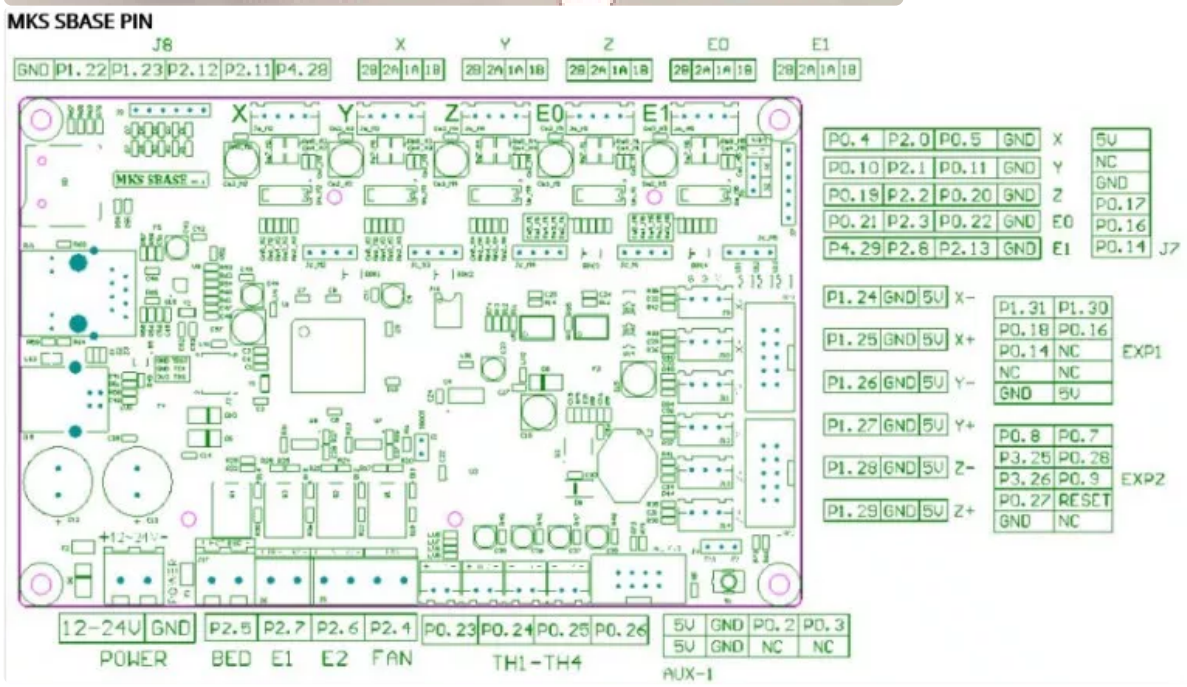
One thing of note here is the current for our **delta driver**, which translates to our Extruder 01 motor.

Temperature Control Configuration

```
# HOTEND temperature control configuration
temperature_control.hotend.enable               true    # Whether to activate this ( "hotend" ) module at all.
                                                        # All configuration is ignored if false.
temperature_control.hotend.thermistor_pin       0.24    # TH2     # Pin for the thermistor to read
temperature_control.hotend.heater_pin           2.7     # Heater1 # Pin that controls the heater, set to nc if a readonly th
temperature_control.hotend.thermistor           Semitec # Default is RRRF100K - I use E3Ds - see http://smoothieware.org/tem
#temperature_control.hotend.beta                3960    # or set the beta value
temperature_control.hotend.set_m_code           104     #
temperature_control.hotend.set_and_wait_m_code  109     #
temperature_control.hotend.designator           T       # For use in temperatureswitch.hotend.designator

# PID configure 24v heater
# M303 E0 S60
# M303 E0 S200
#temperature_control.hotend.p_factor            57.7    # Default 13.7 - permanently set the PID values after an auto pid
#temperature_control.hotend.i_factor            6789    # Default 0.097
#temperature_control.hotend.d_factor            123     # Default 24

#temperature_control.hotend.max_pwm         255         # max pwm, 64 is a good value if driving a 12v resistor with 24v.
```







I'm not changing the pins or connector we use, but stick to the default.

Thermistor type

http://smoothieware.org/temperaturecontrol#toc5

PID tuning

We do need to configure this, but we cant really do it at this stage since we need everything setup first.

When you are ready to do it, the command is slightly different than with Marlin firmware.

In Smoothiware the command is **M303 E0 S60** to test at 60c. I did a test at 60 and then Again at 200 with new values.

Hotend thermistor – physical layout

In order to actually connect our thermistor we take a look at our pins image to figure out where we connect it.

3. TH3 is P0.25 – Unassigned
4. TH4 is P0.26 – Unassigned

I have made a small comment in the **config.txt** file to remind me that 0.24 is TH2

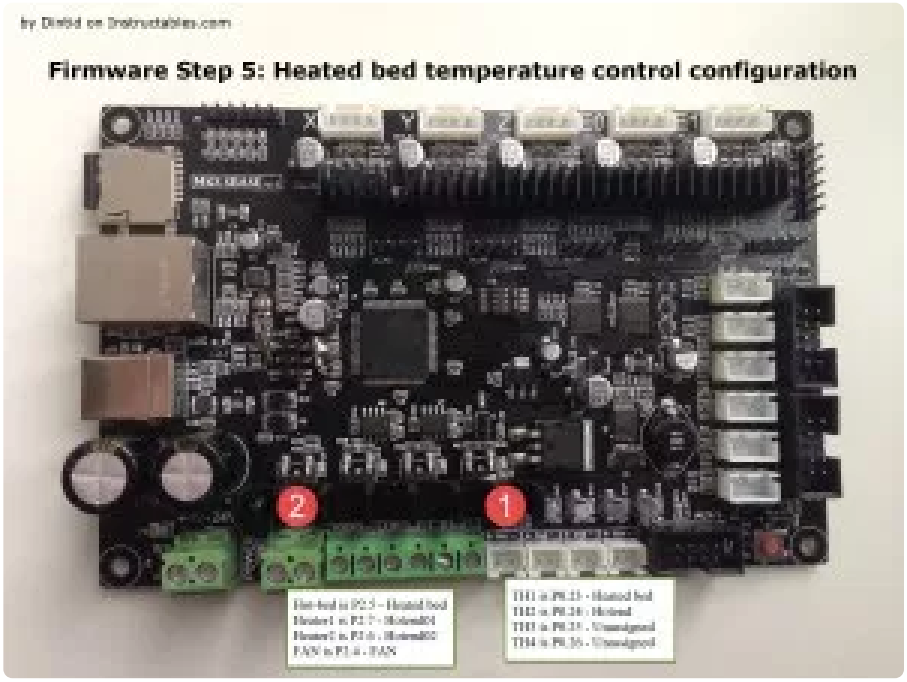
Hotend heater pins – physical layout

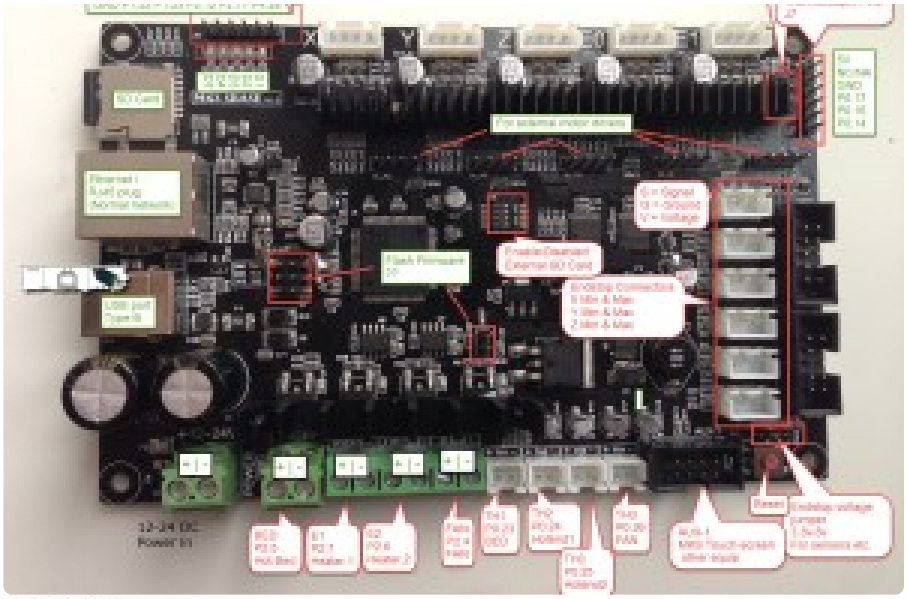
Control.hotend.heater.pin 2.7 is the screw-terminals labeled **Heater1** on the board

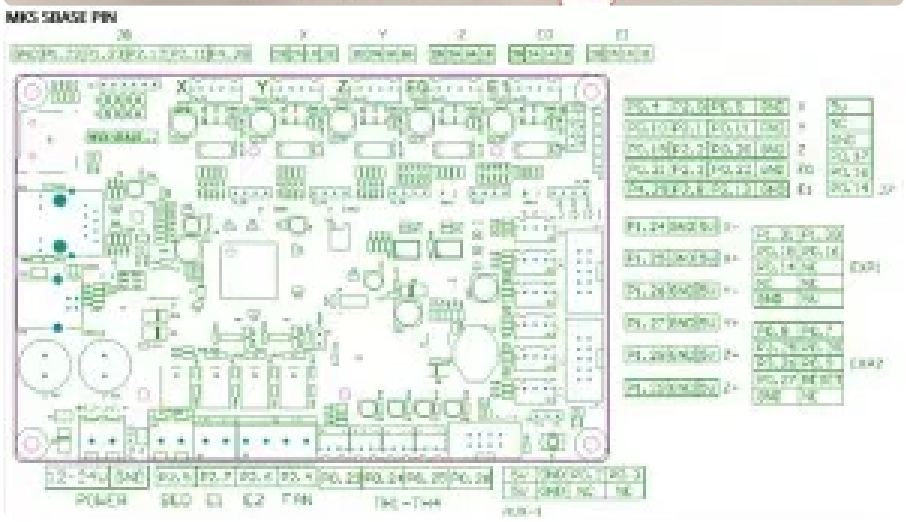
Below chart looks rather unorderly, but is listed as left to right on the controller-board. It is listed on the board itself, which makes it not as bad as it might seem.

1. Hot-bed is P2.5 – Heated bed
2. Heater1 is P2.7 – Hotend01
3. Heater2 is P2.6 – Hotend02
4. FAN is P2.4 – FAN

```
# Heated bed temperature control configuration
temperature_control.bed.enable               true       #
temperature_control.bed.thermistor_pin       0.23       # TH1
temperature_control.bed.heater_pin           2.5        # Hot-bed
temperature_control.bed.thermistor           RRRF100K   # see http://smoothieware.org/temperaturecontrol#toc5
#temperature_control.bed.beta                3960       # or set the beta value

temperature_control.bed.set_m_code           140        #
temperature_control.bed.set_and_wait_m_code  190        #
temperature_control.bed.designator           B          #

temperature_control.bed.bang_bang            true       # set to true to use bang bang control rather than PID
temperature_control.bed.hysteresis           2.0        # set to the temperature in degrees C to use as hysteresis
                                                        # when using bang bang
```







There was no comment to clearly state the start of the heated bed section, so I made a start comment **#Heated bed temperature control configuration** to match the start of hotend section.

at:http://smoothieware.org/temperaturecontrol#toc5

Temperature Control Bed BANG-BANG

I am running my heated bed through a SSR (Solid State Relay) which is not very good at handling PID, which uses higher PWM signals. I could do it by turning signals down to 20 (according to smoothiewares site), but I honestly don't know how to do a PID tuning of my Heated-bed?

If I wanted to specify my frequency for the bed, it would be like this:

```
temperature_control.bed.pwm_frequency
```

I uncomment and change the setting for **temperature_control.bed.bang_bang** to **true** and also

Heated bed thermistor – Physical Layout

Here we once Again take a look at our pins image to figure out where we connect the thermistor.

Default connector for heated bed thermistor is the one named **TH1** on the board.

- TH1 is P0.23 – Hot-bed
- TH2 is P0.24 – Hotend
- TH3 is P0.25 – Unassigned
- TH4 is P0.26 – Unassigned

I have made a small comment in the *config.txt* file to remind me that 0.23 is TH1

To find the correct thermistor, we look at: http://smoothieware.org/temperaturecontrol#toc5

Heated bed thermistor – Physical Layout

the controller-board

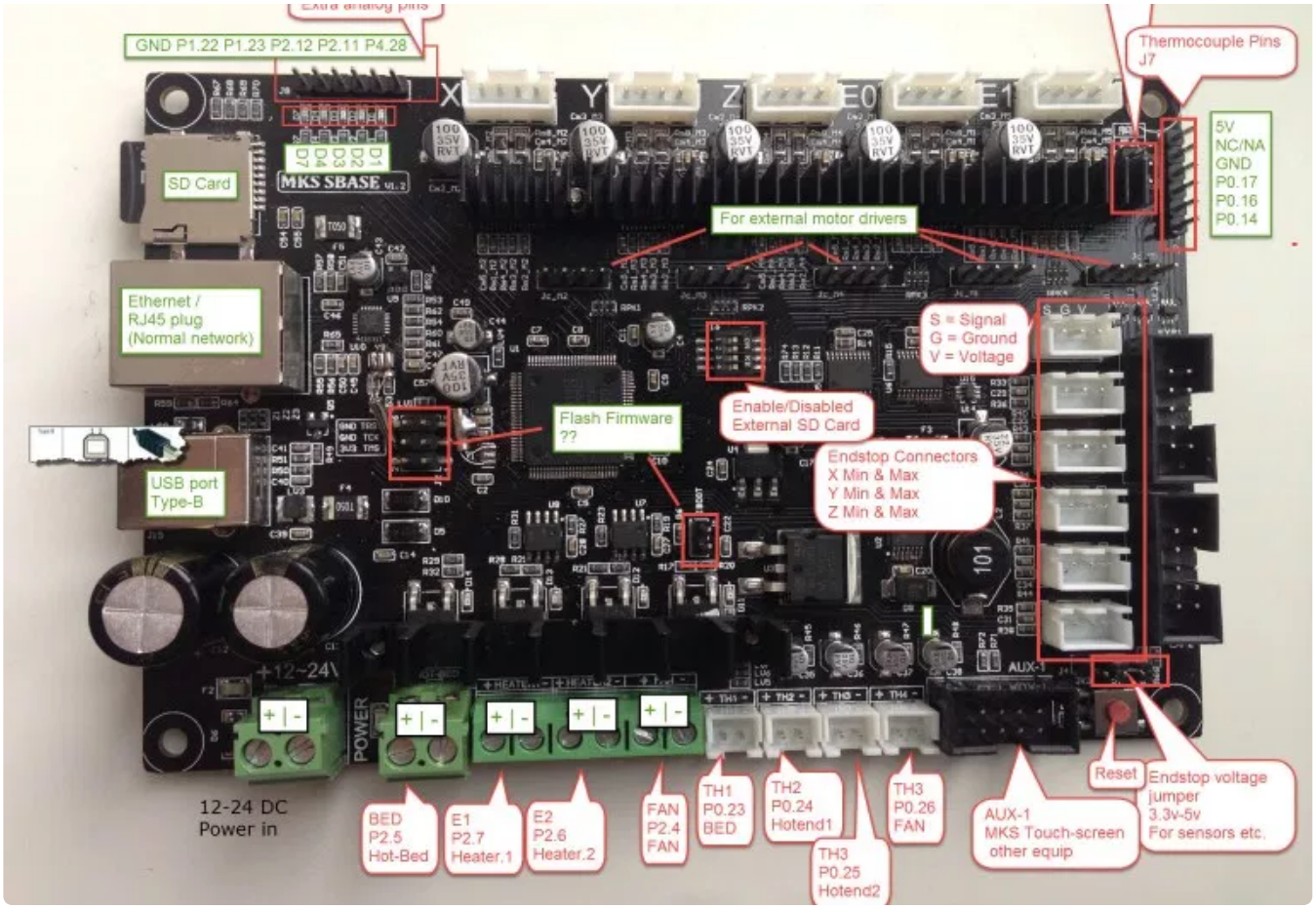
1. Hot-bed is P2.5 – Heated bed
2. Heater1 is P2.7 – Hotend01
3. Heater2 is P2.6 – Hotend02
4. FAN is P2.4 – FAN

Heated bed is default set as bang_bang which is fine for large areas like a heated bed.

Step 8: Firmware Step 6: Configuring Endstops

```
alpha_max_endstop               nc # 1.25^          # NOTE set to nc if this is not installed
alpha_homing_direction          home_to_min         # or set to home_to_max and set alpha_max
alpha_min                       0                   # this gets loaded after homing when home_to_min is set
alpha_max                       213 # 250           # this gets loaded after homing when home_to_max is set
beta_min_endstop                nc # 1.26^!         #
beta_max_endstop                1.27^               #
beta_homing_direction           home_to_max         # Default is home_to_min
beta_min                        0                   #
beta_max                        179 # 250          #
gamma_min_endstop               nc # 1.28^!         #
gamma_max_endstop               1.29^               #
gamma_homing_direction          home_to_max         # Default is home_to_min
gamma_min                       0                   #
gamma_max                       195.8  # 120        #

# optional order in which axis will home, default is they all home at the same time,
# if this is set it will force each axis to home one at a time in the specified order
#homing_order                    XYZ                 # x axis followed by y then z last

# optional enable limit switches, actions will stop if any enabled limit switch is triggered
#alpha_limit_enable              false               # set to true to enable X min and max limit switches
#beta_limit_enable               false               # set to true to enable Y min and max limit switches
#gamma_limit_enable              false               # set to true to enable Z min and max limit switches

alpha_fast_homing_rate_mm_s     100 # 50             # feedrates in mm/second
beta_fast_homing_rate_mm_s      100 # 50             # "
gamma_fast_homing_rate_mm_s     20 # 4              # "
alpha_slow_homing_rate_mm_s     40 # 25             # "
beta_slow_homing_rate_mm_s      40 # 25             # "
gamma_slow_homing_rate_mm_s     10 # 2              # "

alpha_homing_retract_mm         5                   # distance in mm
beta_homing_retract_mm          5                   # "
gamma_homing_retract_mm         1                   # "
```

```
Printer is now online.
SENDING:M119
min_x:1 max_y:0 max_z:0
SENDING:M119
min_x:0 max_y:0 max_z:0
```



This sections looks very confusing at first glance, but it really not bad, when you first figure out how it is ordered.

Smoothieware source: http://smoothieware.org/endstops

Homing direction

- gamma_homing_direction home_to_max

Disable unused endstops

We do not use X max, Y min and Z min endstops, so lets disable those, by typing nc, which translates to **N**ot **C**onnected.

- alpha_max_endstop nc
- beta_min_endstop nc
- gamma_min_endstop nc

Define axes size

We need to define the size of our printer. This translates to Software endstops in Marlin.

- alpha_max 213

Reverting endstop output

Test output by issuing a **M119** command to it. This looks different than when using Marlin, so instead of *triggered*, you get an *1* if the endstop is signalling a hit.

If you get a false positive you need to revert the endstop signal by adding an **!** after the pin number.

You can see how I removed the **!** from my **alpha_min_endstop** and how it changed the output when issuing **M119** command.

Hint: Just don't focus on 1.24, 1.27 or 1.29 which are the pin numbers, but just look at the !

Fast and slow homing rates

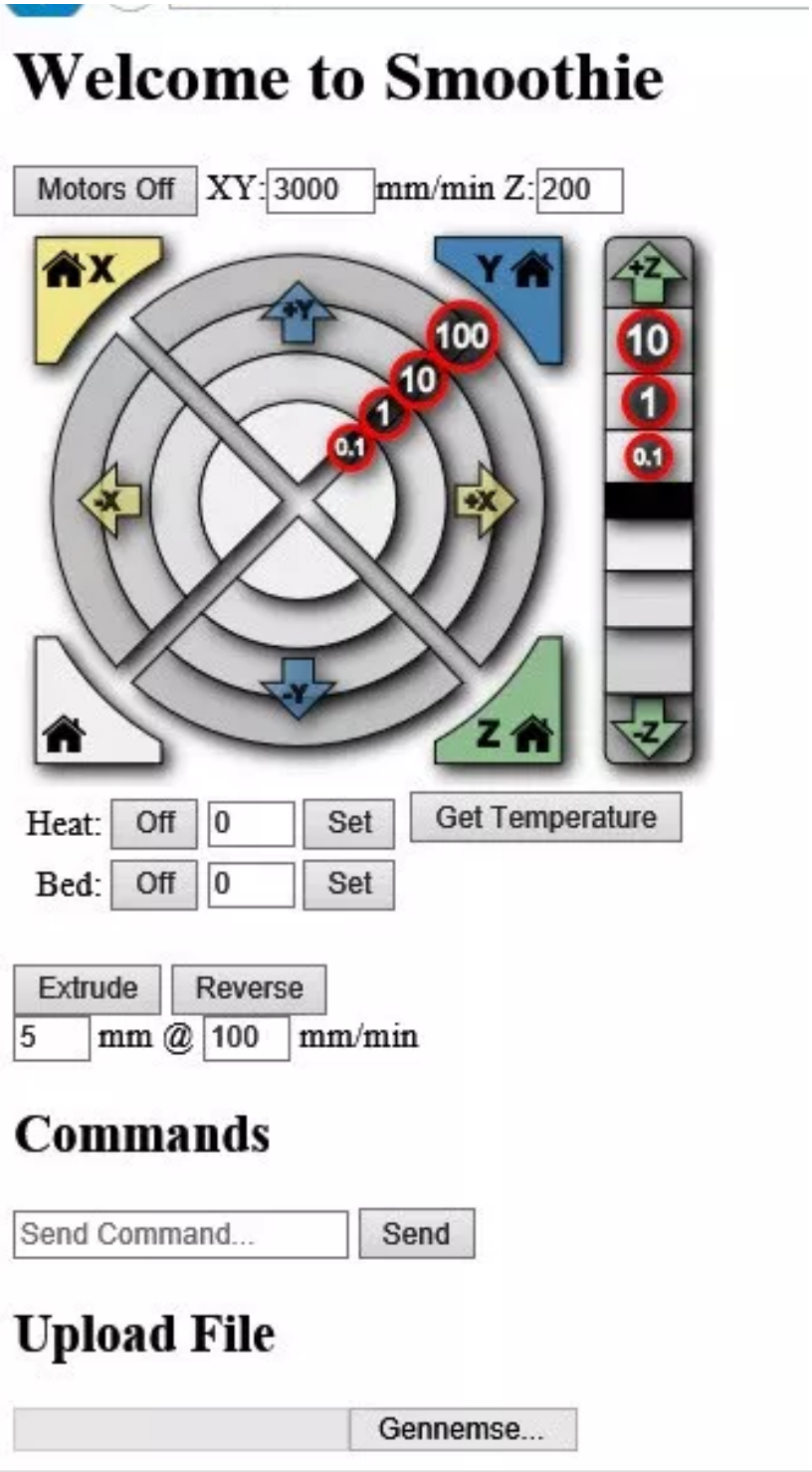
Homing rate transfers directly to move speed mm/sec.

before raising it, since I just changed my heated bed to something heavier.

When hitting the endstop the axes retracts a bit, where the distance i defined in **_homing_retract_mm**, and then moves slowly back. This is defined by **_slow_homing_rate** some. I upped these well, especially for my Z axes.

Step 9: Firmware Step 7: Network Settings

```
# network settings
network.enable                       true          # enable the ethernet network services
network.webserver.enable             true          # enable the webserver
network.telnet.enable                true          # enable the telnet server
#network.ip_address                  auto          # use dhcp to get ip address
# uncomment the 3 below to manually setup ip address
network.ip_address                   192.168.1.221 # the IP address
network.ip_mask                      255.255.255.0 # the ip mask
network.ip_gateway                   192.168.1.1   # the gateway address
#network.mac_override                 xx.xx.xx.xx.xx.xx  # override the mac address, only do t
```



automatic setup, where you enabled **DHCP** and your printer is given an IP from your router/DNS server.

If you want to configure your printer with a static address instead, you simply type in the IP address and subnet mask here.

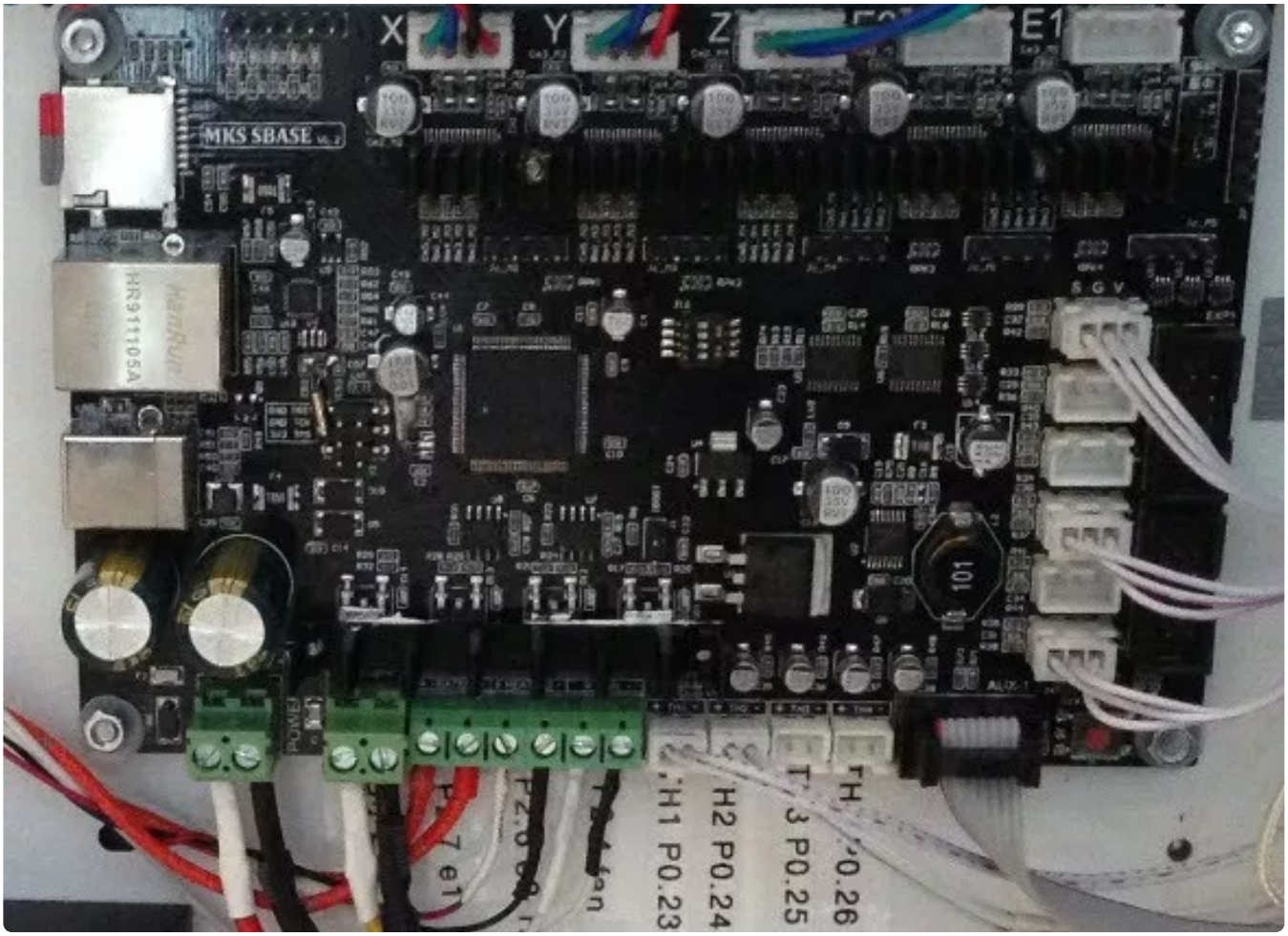
I have changed it to something fitting for my subnet. Yours might be looking different.

I can now connecto to the printer using both a standard internet browser or using Pronterface.

Using an Explorer we get a simplified Pronterface GUI for our use.

I have kept telnet enabled as I want to see what options it gives me.

Note: Do disable any networking service you do not use, aa it will open a security breach where people might take over your machine. ***Especially disable Telnet if you do not use it.***



make the leap for Smoothie-type boards and Smoothieware or stay away.

I believe I have fullfilled the purpose of this Instructables by getting started using the Sbase 1.2 controller and using Smoothieware.

We should by now have a functional printer, albit some tuning is stillneeded.

More advanced setup?

I allready did setup a Hotend Temperature and -fanswitch (2 switches), which I will cover in an upcoming Instructables.

Using switches

1) Fanswitch :: Be able to turn on/off my fan used to cool my hotend

Del dette:

Related

Duet WiFi - Generating firmware using Configurator
17. March 2017
In "Duet Wifi"

Duet WiFi - Intro & Explained
11. February 2017
In "Duet Wifi"

Duet WiFi/Eth - Recover from Erase + basic setup
25. September 2017
In "Duet Wifi"

MKS Sbase, Smoothieware

Configure Smoothieware, MKS Sbase, MKS Sbase Setup, Setup Smoothieware, Smoothieware

← Lets build an Ultimaker

Configuring Endstops on

2 thoughts on “Smoothieware – MKS Sbase – Controller Basics and Intro”

nguyen tai

15. October 2017

> Dear,
> Can you tell how to use mosfets at pint 1.22 and 1.23, there r no outputs (12/24v) as pint 2.4->2.7 with this board.

Reply

BeTrue3D

16. October 2017

You need to assign the pins to the function you want them to do.

If we want to use 1.22 as hotend we need to set the hotend.heater_pin to 1.22 from 2.7

temperature_control.hotend.heater_pin 1.22

This way it will activate this pin when it turns the heater on. I'm using Mosfet boards for these things. So using the signal from 1.22 to activate the mosfet, which opens for the voltage you have attached to the boards DC input.

I've written a small instructables on using a mosfet board

https://www.ebay.co.uk/sch/i.html?control+MOS+FET+switch+control+panel.TRS0&_nkw=DC+control+MOS+FET+switch+control+panel&_sacat=0 but there are other links in the above instructables as well.

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Name *

Email *

Website

CAPTCHA Code*

Post Comment

This site uses Akismet to reduce spam. Learn how your comment data is processed.

Categories

Select Category

Monthly overview

- February 2018 (1)
- January 2018 (2)
- December 2017 (2)
- November 2017 (2)
- October 2017 (2)
- September 2017 (3)
- August 2017 (3)
- July 2017 (2)
- June 2017 (7)
- May 2017 (6)
- April 2017 (2)
- March 2017 (5)
- February 2017 (4)