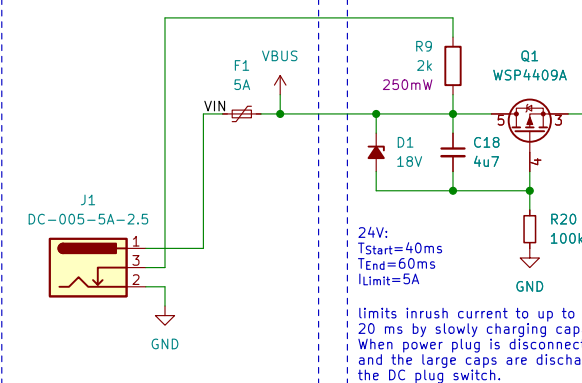
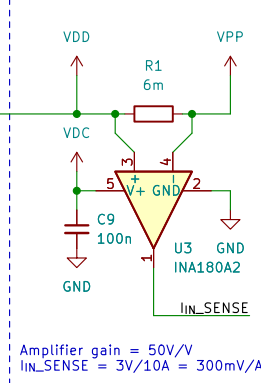


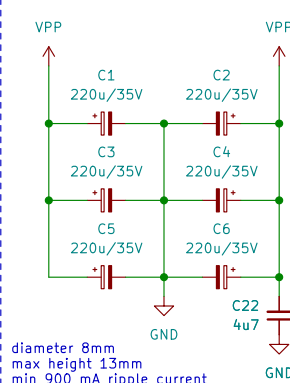
### USB C + VOLTAGE NEGOTIATION



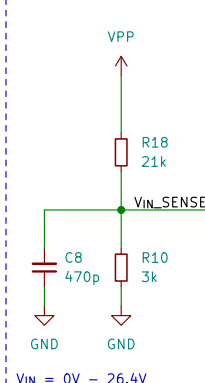
### POWER PLUG + PTC FUSE



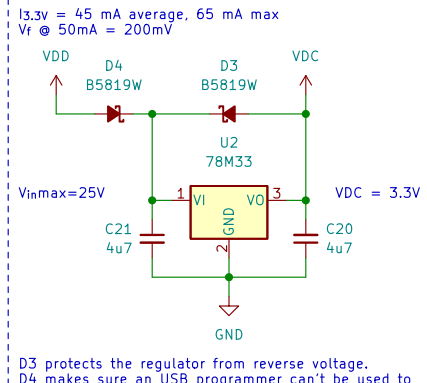
### CURRENT SENSING



### POWER CAPS

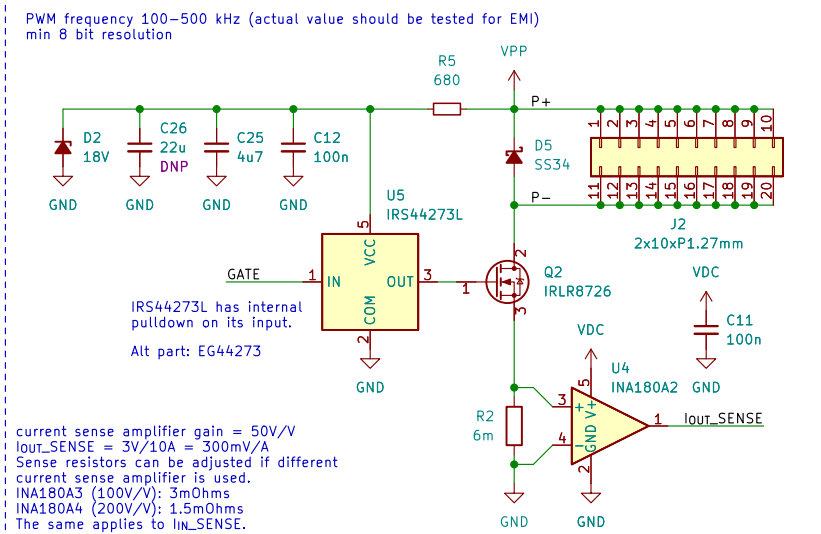


### VOLTAGE SENSING



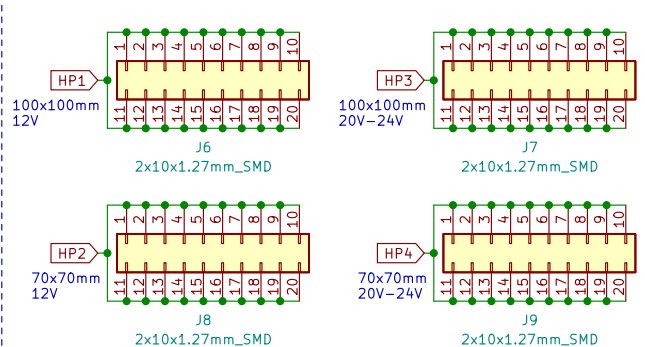
D3 protects the regulator from reverse voltage. D4 makes sure an USB programmer can't be used to power the heat plate and charge the large caps.

### VOLTAGE REGULATOR



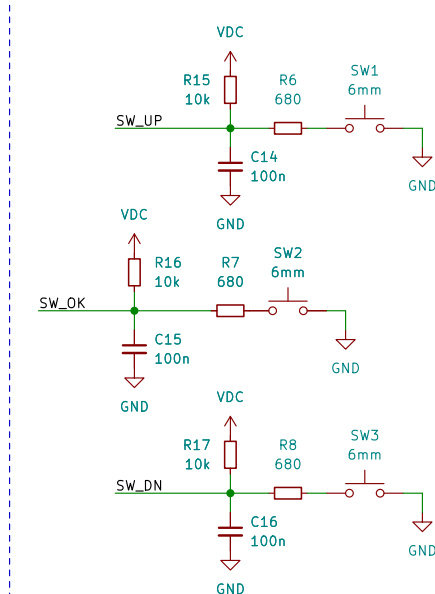
### POWER OUTPUT

power simulation: <https://tinyurl.com/2l5k5czt>

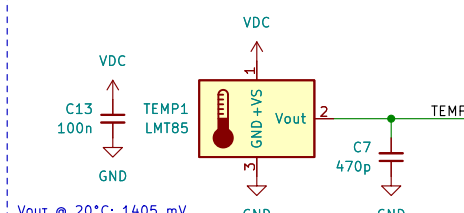


I<sub>out\_SENSE</sub> in combination with V<sub>in\_SENSE</sub> is measured to determine, the actual resistance of the heat plate, and thus the temperature. To measure the resistance turn off the heat plate for at least 5ms so the capacitors can charge. Then give full power (no PWM) to the MOSFET, measure I<sub>out\_SENSE</sub> and V<sub>in\_SENSE</sub> after ca. 25us (depending on actual heat plate inductance), while also monitoring I<sub>in\_SENSE</sub> to not overload the PSU.

### HEAT PLATES

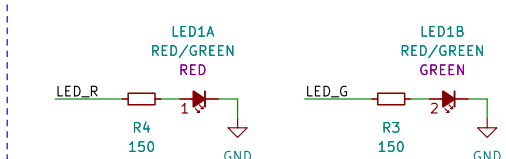


### BUTTONS

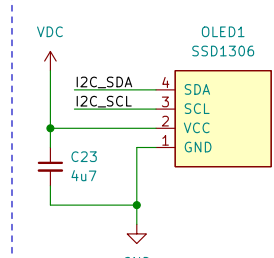


### TEMPERATURE SENSORS

Actual temperature is calculated with the heat bed resistance. Temperature sensor is used to give a rough estimate whether the heat bed is hot or cold.



### STATUS LED



### 0.91" OLED

### A few words regarding firmware:

After initialisation, the MCU should check the input voltage:  
<11.5V: Not OK  
11.4V-12.6V: Barrel plug, OK  
14.5V-15.5V: USB-C, insufficient power, not OK  
19.5V-20.5V: USB-C, sufficient power, prompt user for PSU wattage, OK  
23.0V-24.5V: Barrel plug: OK  
All other unspecified voltages: not OK

Next, the MCU should check if the device is at ambient temperature by probing the analog temperature sensor. If the device is at ambient temperature, MCU can measure the heat bed resistance in short pulses to determine the cold heat bed resistance. MCU then writes cold resistance into EEPROM. If device is not at ambient temperature, MCU reads the last cold resistance from EEPROM, measures warm heat bed resistance, and then checks if the combination of estimated heat bed temperature from analog sensor and heat bed temperature derived from actual resistance and last stored cold resistance is somewhat plausible. If it is, firmware assumes the heatbed was not changed after saving the cold resistance the last time. If not, user must wait until heat plate is cooled down (or they can override the heatbed resistance).

Firmware then checks whether the heat bed resistance and the input voltage are compatible (resistance must be low enough to allow enough current, but high enough that a hot heat bed is below the 5A current limit, in case of PWM failure).

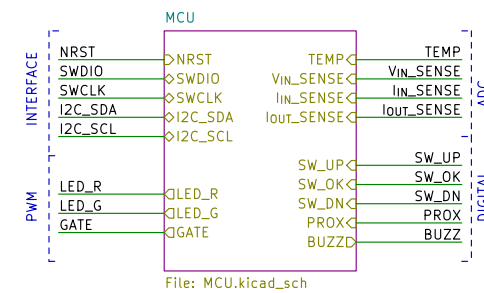
If all checks (input voltage, known cold heat bed resistance, heat bed compatibility) are OK, user is able to enter the main menu.

The main menu items would probable be: "SOLDER TEMP", "SOLDER PROFILE", "START", "SETTINGS". Settings could include additional power settings (Input current limit, USB C power limit, heating time out etc.), and settings for the proximity sensor and the acoustic alarm.

The LED should be controlled via PWM, it should light up green for a cold heat bed, orange when heating is in progress, and red when the heat bed is hot.

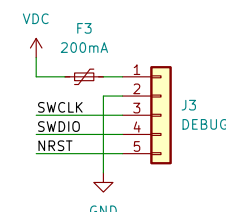
MCU should always measure the input current during heating, and adjust the PWM duty cycle accordingly.

A few things to check on an actual prototype:  
- How is the connection between heat bed and base? Does the header work as intended?  
- Does the proximity sensor work? Sensitivity can be adjusted by changing R21.  
- How much does R9 heat up when the device is powered by USB C?

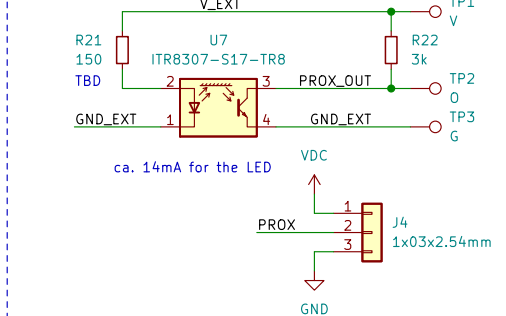


MCU could be replaced if parts are not available. Route buttons to interrupts.

### MCU

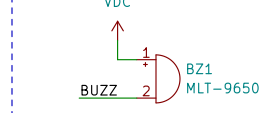


### DEBUG + PROGRAMMER

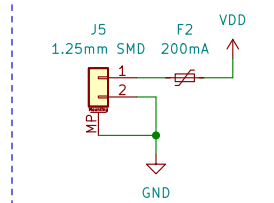


### PROXIMITY SENSOR (optional)

U4 is a proximity sensor. It is placed under the center of the heat plate and is connected to the main PCB via cable. It is shielded from the heat plate heat and detects if anything is placed on the center of the heat plate. This makes it possible to turn the heat plate off when nothing is placed on it, or automatically turn the heatbed on when something is placed on it. This should make working through batches of PCBs easier and faster, for example. R23 is responsible for the IR LED brightness. The LED should be bright enough to trigger the sensor when something is placed over it, but not so bright that it triggers all the time from the heat plate PCB.



### BUZZER (optional)

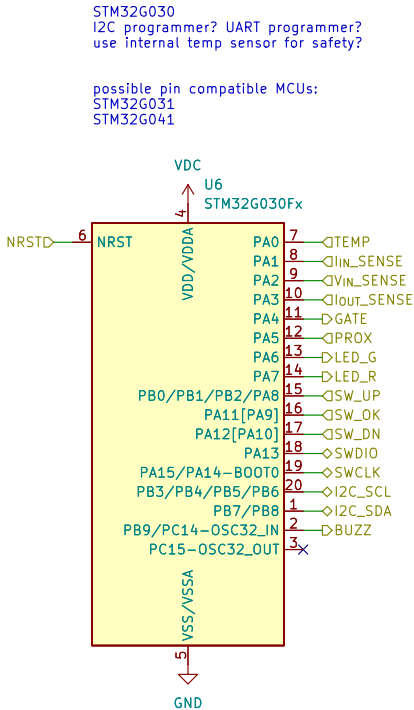
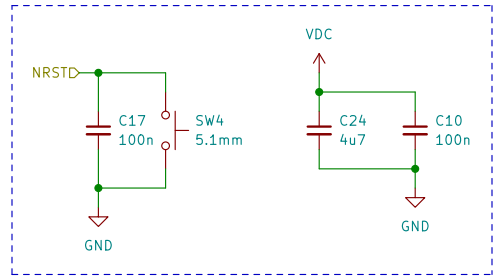


### FAN HEADER (optional)

**WARNING: THIS DESIGN HAS NOT BEEN TESTED!**

<https://github.com/DerSpatz>

Sheet: /	File: HeatTable-Base-v0.9.kicad_sch	
<b>Title: reflow solder table base (development version)</b>		
Size: A3	Date: 2022-03-18	Rev: 0.9
KiCad E.D.A. kicad (6.0.7)		Id: 1/2



Startup:  
slow pwm ramp up, to check too low heatbed resistance  
needs 1–5ms to adjust current to pwm  
check temp sensors for ambient temperature (18–22°C)  
if at ambient temp, check cold resistance by pulsing at 50% pwm  
correct cold resistance to 20°C  
enter updated cold resistance into eeprom  
if not at ambient temp,  
– look for earlier data in eeprom, ask user if heatplate was changed  
– or enter resistance manually  
– or ask user to wait for ambient temp

after changing duty cycle, wait 3–5 ms for the system to settle  
before measuring current and voltage

A few notes regarding power consumption:  
With a cold heat plate, the current consumption is probably too high for the DC plug and the power supply, which are both rated at 5A. To reduce the current consumption, a PWM signal MUST be applied to the MOSFET gate during the warmup phase. The PWM frequency must be at least 50 kHz for the filter capacitors to work properly.  
To calculate the allowed duty cycle, the user must enter the calculated or measured heat trace resistance before first use.

NOTE: with the newer board version, the MCU should be able to measure the heat plate resistance by itself.

With this information,  
the MCU can calculate the maximum current at 100% PWM, and then calculate the allowed duty cycle:

$$I_{cold} = V_{in} / R_{cold}$$

$$PWM_{max} = 4.75A / I_{cold} \text{ (0.25A as margin)}$$

As the temperature rises, the duty cycle can be set higher.  
The resistance multiplies by approximately 1.63 from 20°C to 180°C.

With the calculated heat plate resistance, the results are:

$$R_{cold} = 1.39 \text{ Ohms}$$
$$V_{in} = 12V$$
$$I_{cold} = 8.63A$$
$$PWM_{max} = 55\%$$

The resistance of the hot heat plate must reach at least 2.4 Ohms for the heat plate to not overload the PSU when at 100% (worst case condition).  
is 2.54 Ohms, so the current flowing through a hot heat plate is 4.72 A, which is not overloading the PSU or DC plug. This means that even leaving the hot heat plate at 100% power will not destroy the PSU. Setting a cold heat plate to 100% power will most likely blow the resettable fuse, and if not, the current consumption will get to an acceptable level as the heat plate heats up.

Behind this link you can simulate the power draw: <https://tinyurl.com/ycbas8x9>

As always, some margin should be added to all calculations, do not run this device at full power all the time if it is avoidable.

Sheet: /MCU/  
File: MCU.kicad\_sch

**Title:**

Size: A3

Date:

**Rev:**

KiCad E.D.A. kicad (6.0.7)

Id: 2/2