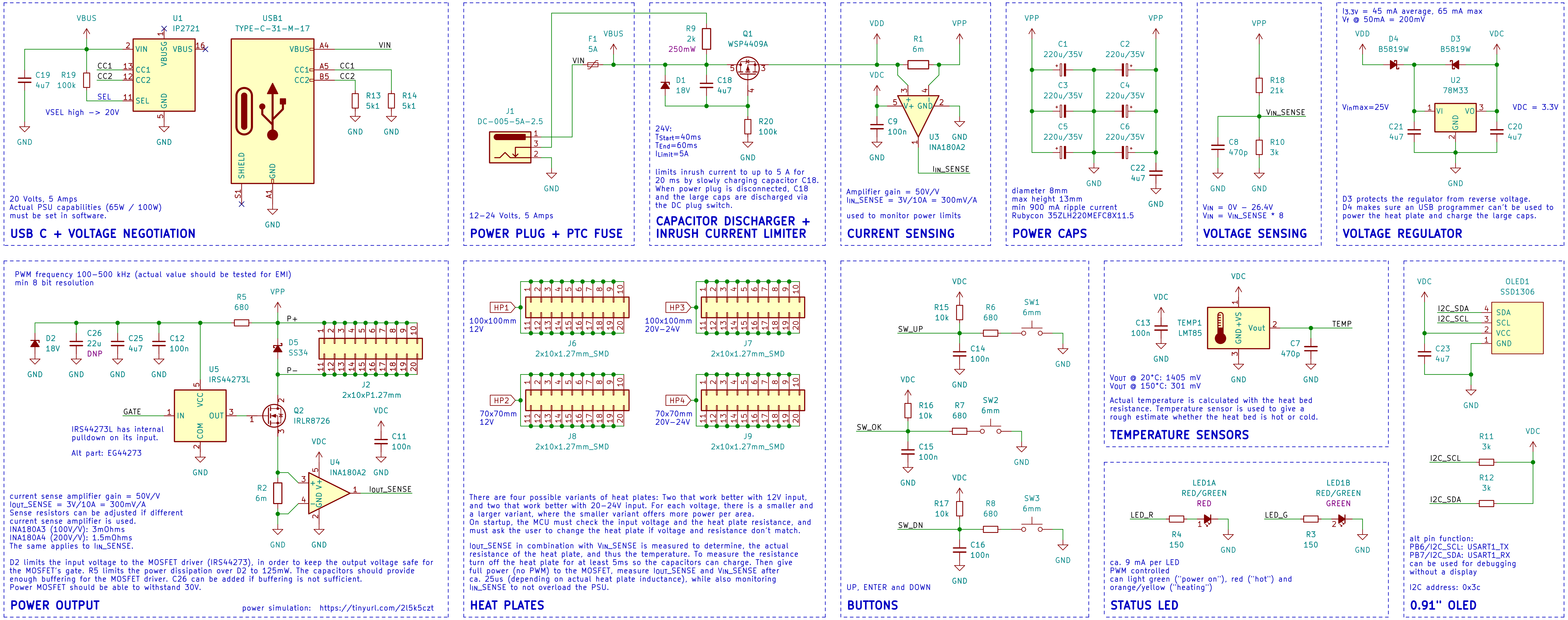


A few words regarding firmware:

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, unsufficient power, not OK
19.5V−20.5V: USB−C, sufficient power, promt 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 resitance 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).

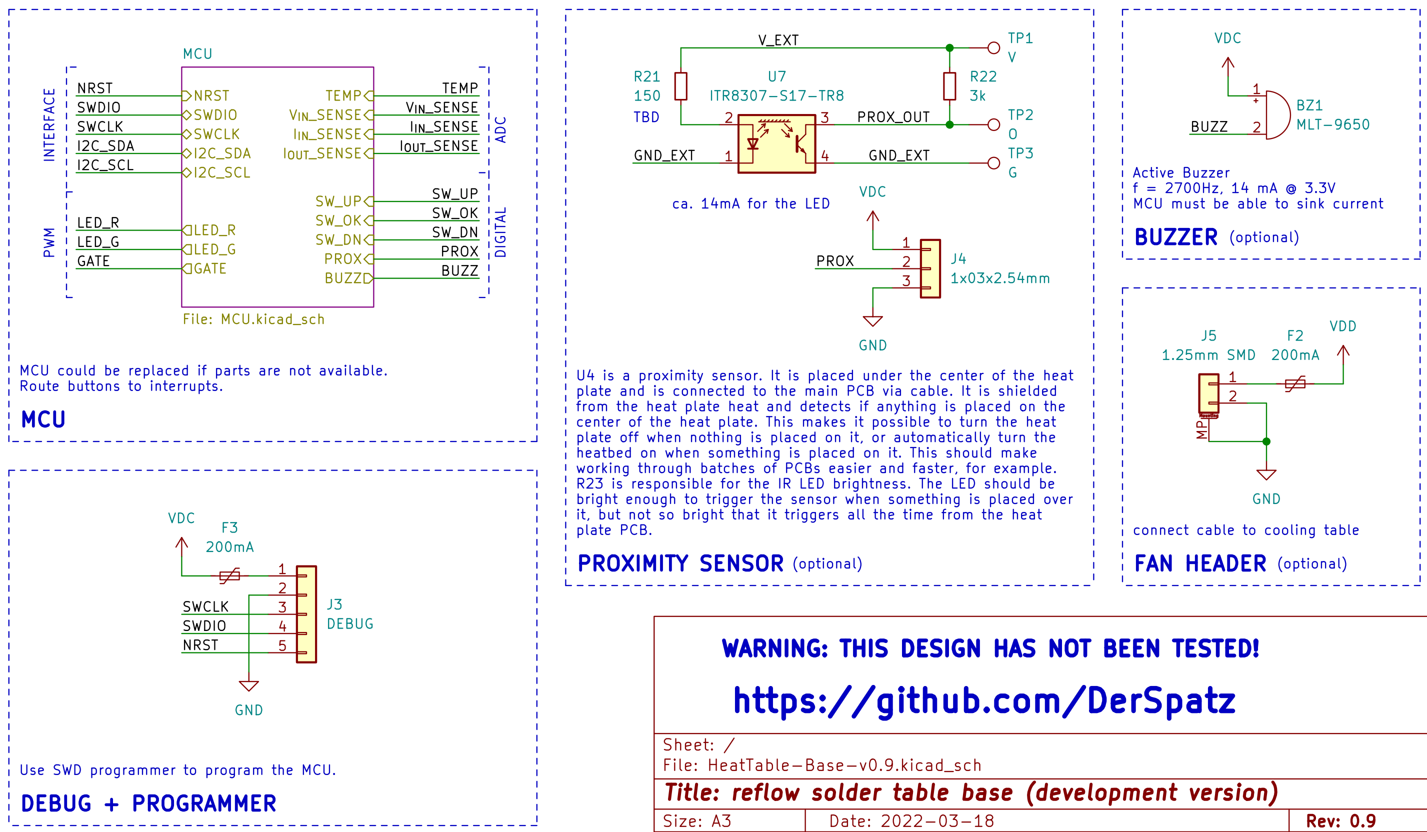
If all checks (input voltage, known cold heat bed resitance, heat bed compatibility) are OK, user is able to enter the main menu.

The main menu items would be problable 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?



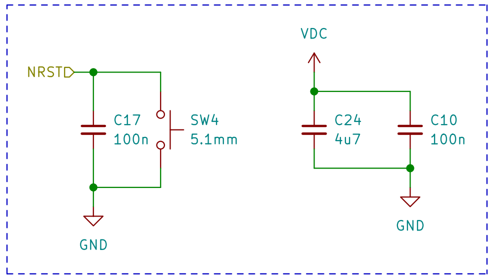
WARNING: THIS DESIGN HAS NOT BEEN TESTED!

https://github.com/DerSpatz

Sheet: /
File: HeatTable−Base−v0.9.kicad_sch

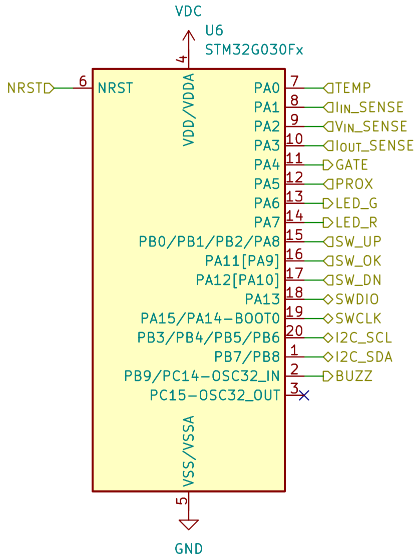
Title: reflow solder table base (development version)

Size: A3	Date: 2022−03−18	Rev: 0.9
KiCad E.D.A. kicad (6.0.7)		Id: 1/2



STM32G030
I2C programmer? UART programmer?
use internal temp sensor for safety?

possible pin compatible MCUs:
STM32G031
STM32G041



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 resitance 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:

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

$PWM_{max} = 4.75A / I_{cold}$ (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 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 accetable 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
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