Orbitool O2 v2.72

Author: Dr. Róbert Lőrincz

Should a tool-board *magic smoke* come out because of user error? Why?

Should a tool-board be destroyed by a fault in your 3D printer? Why?

My answer is NO! And this is what this tool-board is all about!

We are hobbists and professionals, and let's face it, we make mistakes, and when the magic smoke shows up, everybody is disappointed, even if we know it's a user's fault. In this board, I promise you the *magic smoke* is kept inside extremely tight!

Smokeless Features

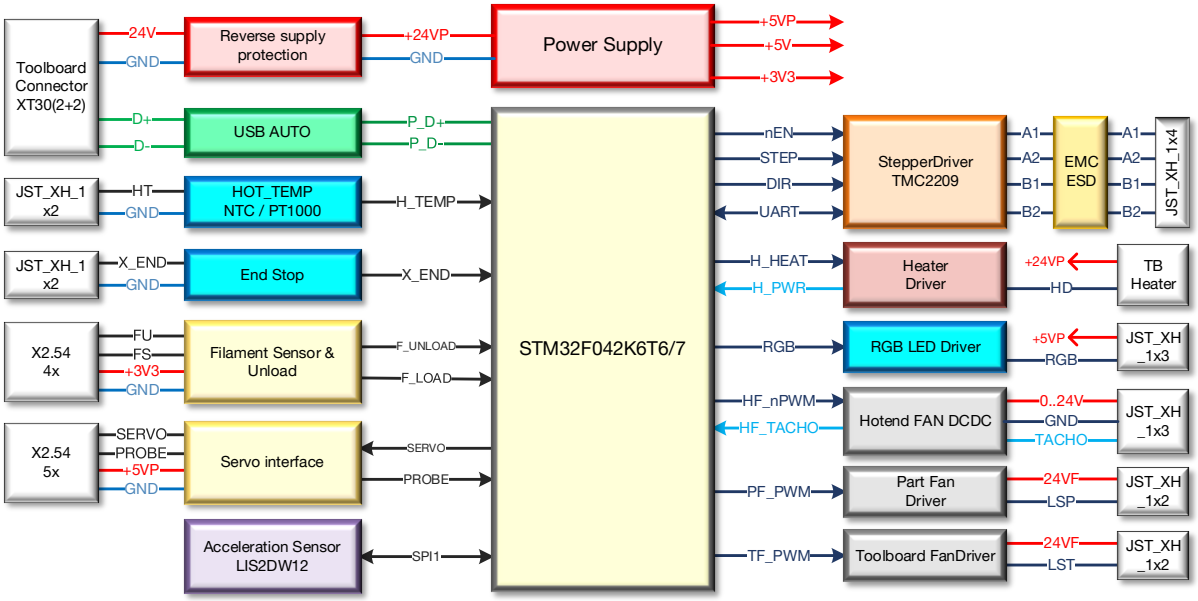
- Optimized for Orbiter v2 shape and features
- STM32F042 microcontroller running at 48MHz
- Automotive USB communication with a Raspberry Pi
- Onboard LIS2DW12 accelerometer
- TMC2209 extruder stepper driver
- Direct connection to the Orbiter v2 sensor
- 2x PWM controlled fan outputs
- DC-DC converter based hot-end fan driver, compatible with 12/24V fan types with RPM speed input signal
- Hot-end temperature sensor input compatible with standard NTC or PT1000 temperature sensor types
- I/O for bed level sensing
- X-Stop sensor input
- RGB LED driver output
- Onboard temperature sensor
- Advanced thermal management system

Protection Features

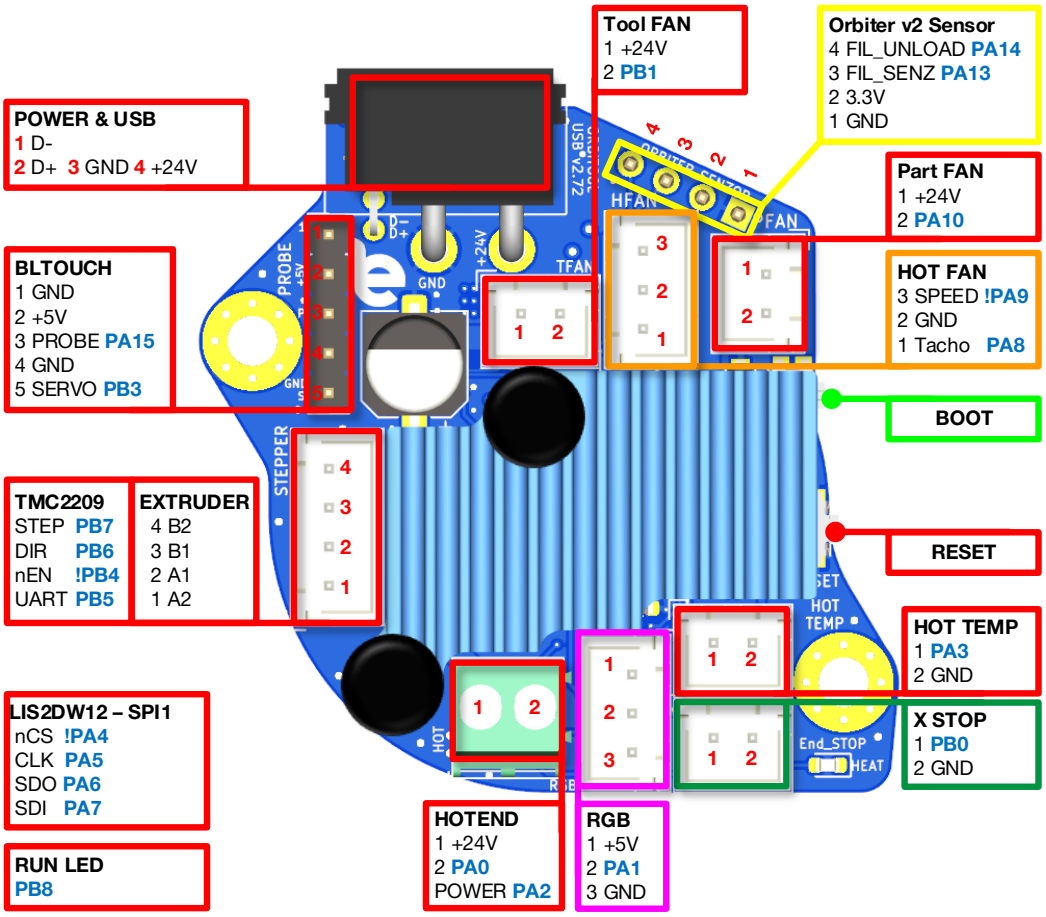
- Active short circuit protected Hot-end output
- Active short circuit protected fan driver outputs
- Active Protection circuit against reverse power supply connection
- Protection against loss of GND supply
- Analog and digital inputs protected against short to +24V supply voltage
- USB data lines are protected against short circuits to GND and +24V
- RGB LED power supply pin protected against short to GND
- EMI interference and ESD protection for all inputs and outputs
- Heater thermal runaway protection in case of a short between heater and sense thermistor wires

1 Block Diagram

Block diagram for Orbitool 02 v2.72



2 Pinout definition



3 Features description

3.1 Introduction

For me, the electrical robustness of a design is part of the requirements. One of my main design goals was to integrate protection against common user errors and 3D printer defects.

In the end, nothing is more frustrating than powering up your new board and letting the *magic smoke* out. Even if it is due to user fault, the disappointment will still be present.

This design is based on automotive technology, where protection from all kinds of hazards is a state-of-the-art a long time ago.

This document describes the main specific features of the tool-board.

3.2 Power supply concept

The board is supplied via an XT30 (2+2) connector, which includes +24V, GND, USB_DATA+ and USB_DATA- signals.

From the connector, the board is supplied via a reverse power supply protection circuit. This prevents tool-board damage. In the event of an accidental reversed supply by user error, the board will not start up and will behave like it's not powered.

The microcontroller is supplied from a 3.3V LDO supplied from the onboard 5V DC-DC buck converter for the highest power supply efficiency and lowest possible power consumption.

The RGB LED and the bed level sensor outputs are supplied from a short-to-GND protected power supply.

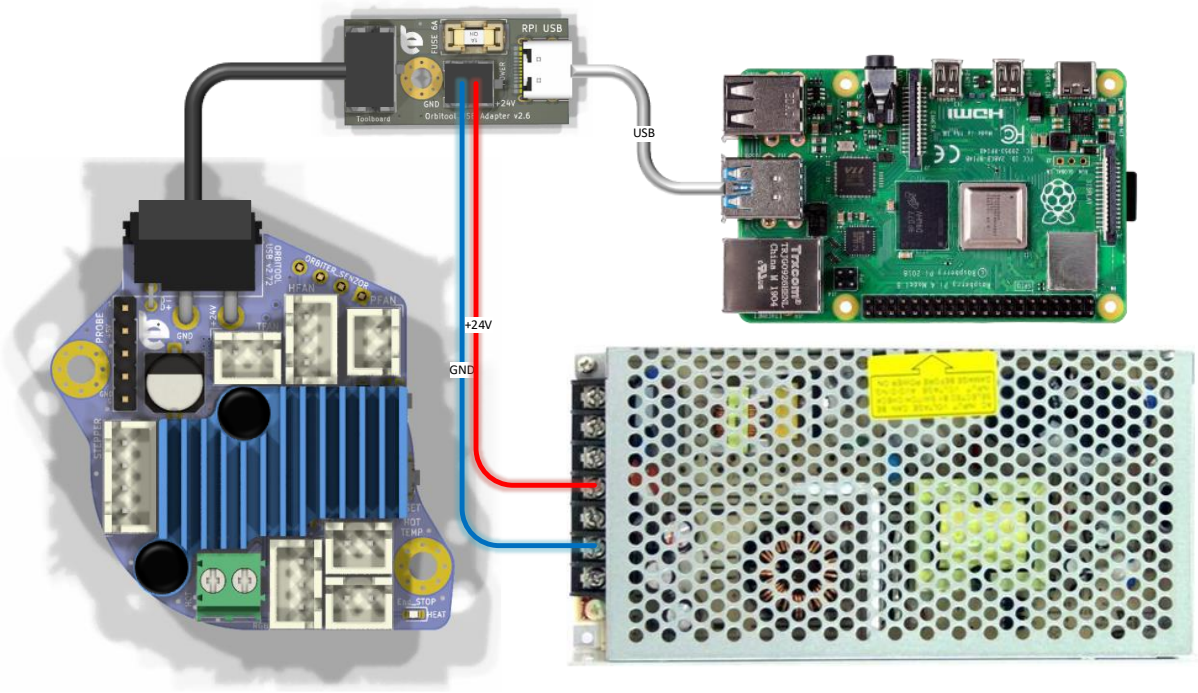
3.3 USB communication interface

The tool-board is equipped with an automotive grade USB interface. The main difference is the higher electrical robustness of the automotive USB, still maintaining highest possible communication speed. Therefore, if the USB data lines are short-to-GND or up to +28V, it will not cause any destruction and will work properly after the short circuit is removed.

The same circuit offers protection in case of GND connection loss, which will lead to biasing all the communication signals to the supply positive line (in our case, +24V).

The USB interface of the Raspberry Pi is not robust against these electrical hazards that may occur in a 3D printer. Therefore, the tool-board shall be supplied and connected to the Raspberry Pi using the supplied adapter board, which has onboard protection circuits to protect the Raspberry Pi USB port.

The next picture presents the tool-board wiring connection principle.



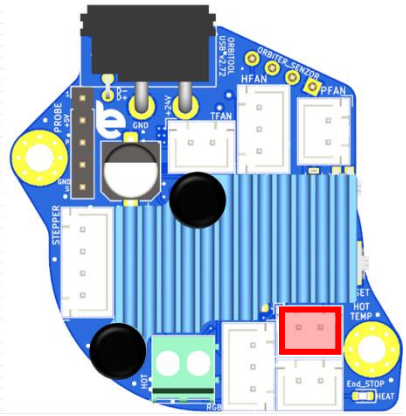
3.4Hot-end temperature sensor

The hot-end input circuit accepts two types of thermistors, 100k NTC type similar to the ATC Semitec 104GT-2 and PT1000.

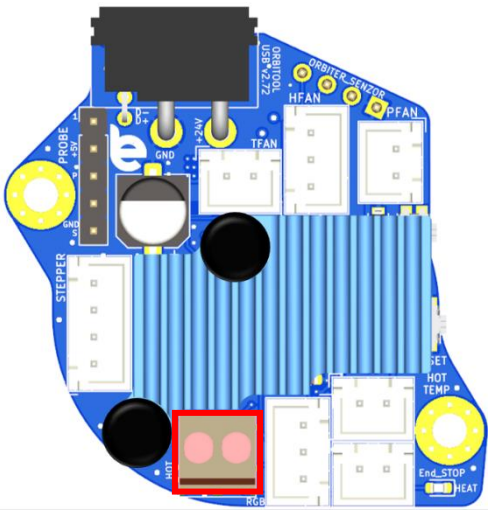
Pull-up resistor value: 2200Ω.

The senor input is protected against short circuits to GND and supply voltage of 24V.

The GND connection of the sensor is current limited to avoid thermal runaway of the hotend in the event of a short circuit between the sensor and heater wires.



3.5 Driver circuit for hot-end heater



The hot-end heater driver circuit is implemented with active protection against short circuits.

The circuit measures the heater current and in case the current rises over a predefined threshold i_{OC_HOT}, it switches off the heater MOSFET until the next PWM cycle to prevent damage.

The heater max power shall be limited to 99.5%, meaning max_power: 0.995 to avoid driver stuck in protection mode due to an unlikely event of a fake error detection.

```
max_power: 0.995
```

The heater current is fed to Klipper, which can calculate the actual power consumption of the heater element. This can be used to detect failures of the heater element, like a short circuit or a loss of heating power.

You may say yes; other boards also have protection with onboard fuses. Well, that is simply not enough to protect MOSFETs from being damaged because fuses have a long reaction time, about 1-2s range. Their mission is to protect the circuit from catching fire in case of a defect, but it cannot protect the heater driver MOSFET from getting destroyed. This active protection reacts within 10ms, ensuring the driver stage is switched off before it gets damaged due to a short circuit or overload event.

To display the actual power of the heater, include the following section:

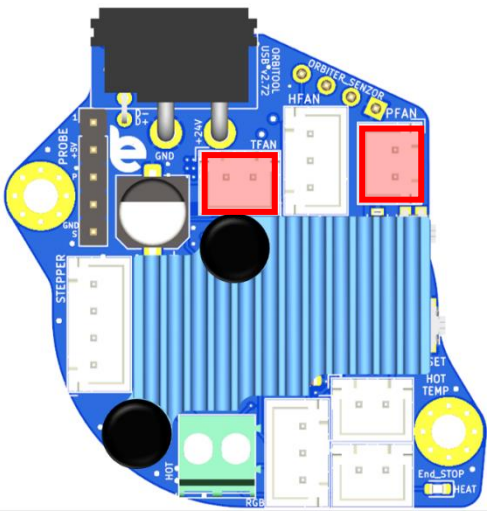
```
[adc_temperature HOT_P]
#temperature1:5 # value in Ampere
temperature1:119 # value in Watts
#voltage1:1.32
voltage1:1.91
#temperature2:10 # value in Ampere
temperature2:239 # value in Watts
voltage2:3.82
#voltage2:2.64

[temperature_sensor Hotend_power]
sensor_pin: orbitoolO2:PA2
sensor_type: HOT_P
```

The hot-end power or current can be displayed based on preference. The power is calculated considering a 24V power supply.

It is highly recommended to use ferules to connect the heater wires to the tool-board screw terminal block.

3.6Fan driver outputs



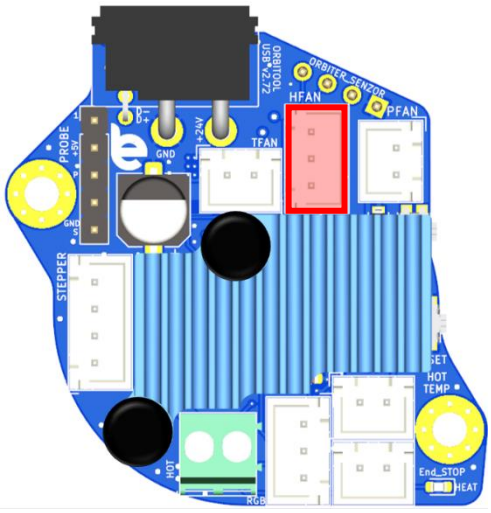
The fan drivers (part fan and aux / tool-board fan) have a similar protection circuit to the hot-end driver; in case of overcurrent, the fan driver is switched off until the next PWM cycle.

The PWM duty cycle of the fan driver shall be limited to 99.5%, meaning max_power: 0.995, to avoid the fan driver being stuck off due to an unlikely event of a fake error detection.

Configuration of the fan driver:

```
[fan]
pin: orbitoolO2:PA10
max_power: 0.995
shutdown_speed: 0.0
cycle_time: 0.02
kick_start_time: 0.2
hardware_pwm: False
```

3.7 Hot-end fan output



The hot-end fan circuit is implemented using a microcontroller-controlled DC-DC converter.

PWM control of three wire fans will destroy the speed signal. PWM (Pulse Width Modulation) control means switching on and off the fan power supply several times every second. This means there are moments when the fan is actually not supplied with power. In those moments, the speed sensing electronic circuit is not supplied, therefore it cannot generate a reliable speed signal. Practically, the PWM signal will be superimposed over the speed signal.

Another advantage of this control circuit is the possibility of limiting the maximum voltage according to the fan type the user has.

This output supports 12V or 24V fan types configured by the max power value in the fan configuration section:

- **12V** fan – max_power: 0.5
- **24V** fan – max_power: 1.0

Since the fan driver is based on a DC-DC converter, the PWM frequency required for correct operation is defined by the converter design, not by fan properties. Fan PWM speed control frequency shall be set to 10 Khz, cycle time = 0.0001 – do not change this to a lower frequency.

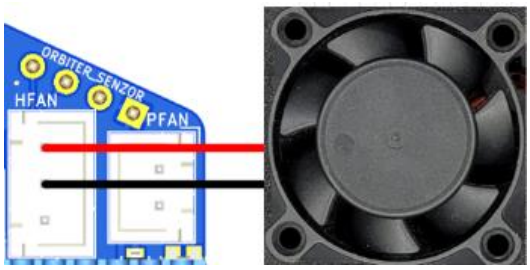
Based on a specific printer design, the maximum power can be limited to reduce fan noise. Increase this to a higher level in case you experience hot end clogs.

The connection of a **three-wire** fan should respect the following wiring:

Klipper configuration example:

```
[heater_fan hotend_Fan]
pin: !orbitoolO2:PA9
tachometer_pin: orbitoolO2:PA8
tachometer_ppr: 2
tachometer_poll_interval: 0.0005
heater: extruder
cycle_time: 0.0001 #10KHz PWM frecvency
heater_temp: 75
fan_speed: 0.4
hardware_pwm: false
shutdown_speed: 0.0
max_power: 0.7 #setup specific
```

The connection of a **two-wire** fan should respect the following wiring:



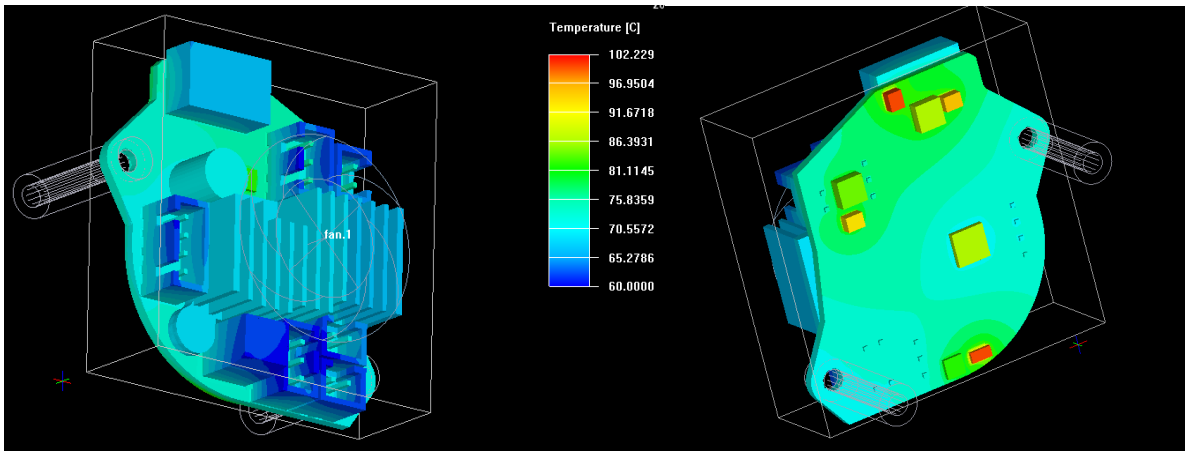
In the Klipper configuration, the tachometer related definitions shall be commented out as follows:

```
[heater_fan hotend_Fan]
pin: !orbitoolO2:PA9
#tachometer_pin: orbitoolO2:PA8
#tachometer_ppr: 2
#tachometer_poll_interval: 0.0005
heater: extruder
cycle_time: 0.0001 #10KHz PWM frecvency
heater_temp: 75
fan_speed: 0.4
hardware_pwm: false
shutdown_speed: 0.0
max_power: 0.7 #setup specific
```

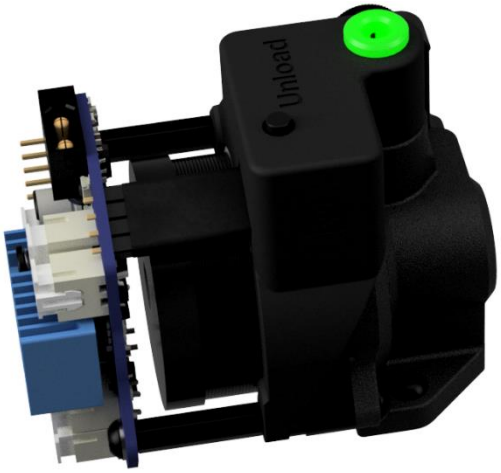
3.8 Thermal considerations

Closed printer chamber requires tool-board operation in high ambient temperature conditions. In addition, this tool-board is mounted on the backside of the stepper motor, which transfers a considerable amount of heat. Therefore, the circuits of this tool-board are designed to have low temperature drift and efficient thermal management to make sure none of the components heat up over their maximum allowed operation temperature.

I performed lots of thermal simulations using finite element simulations and real-life measurements using a Flir thermal camera.



The simulation result show that in a 60°C chamber temperature environment, the tool-board components do not heat over their maximum junction temperature rating.



To reduce the heat transfer from the stepper towards the tool-board a minimum distance of 4-5 mm shall be kept. Orbiter V2 equipped with the LDO-36STH20-1004AHG steper and the Orbiter sensor, standoffs of 18 mm length are recommended. This way, the distance between the toolboard PCB and the stepper back side is about 5 mm.

Without the Orbiter sensor, the standoff length shall be minimum 20 mm.

The toolboard is designed for passive cooling, but even with the most optimized thermal design, stepper current derating over temperature still applies. This is mainly because there is simply not enough space for a bigger heatsink. Even if the heatsink size were doubled, it would still not be enough.

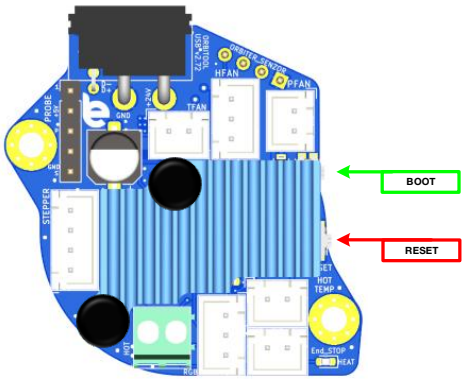
Therefore, we have two possible solutions:

1. Reduce stepper current for high ambient temperature operation. Example down to 0.55A for chamber temperature above 60°C.
2. Use active cooling to improve the thermal behavior. Adding a small 20x20mm or 25x25mm fan on the back of the toolboard that blows air toward the heatsink will improve a lot.

4 Microcontroller programming

1. Connect the board to the host Raspberry Pi via a USB adapter cable.
2. This step depends on whether your board has firmware on it or not:
   a) If your board is pre-flashed with Klipper, first must enter DFU mode as follows:
      - press the BOOT and RESET buttons,
      - release the RESET while keeping the BOOT button pressed,
      - release the BOOT button after about three seconds.

   b) If your board is a new, un-flashed board:
   - The MCU is already in DFU mode; confirm this at step 4



3. Connect to your host Raspberry Pi via SSH
4. Run `lsusb` from the command prompt
   - Make sure you see an STM32 in DFU mode listed
5. Run `dfu-util --list` from the command prompt
   - note the text inside the [xxxx:yyyy]
6. Run `cd ~/klipper` from the command line to enter the Klipper directory
7. Run `make menuconfig` settings should be:
   - Cristal oscillator – 8Mhz

```
(Top)
                    Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (STMicroelectronics STM32)  --->
    Processor model (STM32F042)  --->
    Bootloader offset (No bootloader)  --->
    Clock Reference (8 MHz crystal)  --->
    Communication interface (USB (on PA11/PA12))  --->
    USB ids  --->
    Optional features (to reduce code size)  --->
()  GPIO pins to set at micro-controller startup
```

- Set custom USB ID to *OrbitoolO2*

```
(Top) → USB ids
                        Klipper Firmware Configuration
(0x1d50) USB vendor ID
(0x614e) USB device ID
[ ] USB serial number from CHIPID
(SO3Tool) USB serial number
```

- Optional features (to reduce code size):

```
(Top) → Optional features (to reduce code size)
                        Klipper Firmware Configuration
[*] Support GPIO "bit-banging" devices
[ ] Support LCD devices
[*] Support external sensor devices
[*] Support lis2dw 3-axis accelerometer
[ ] Support software based I2C "bit-banging"
[ ] Support software based SPI "bit-banging"
```

- Hit Q to Exit and Save

8. Run make clean to clean up the make environment
9. Run make flash FLASH_DEVICE=xxxx:yyyy (using the xxxx:yyyy noted from step 5 usually 0483:df11)
   If you encounter some errors but still have the message of File downloaded successfully, you are good to proceed to the next step.

```
DFU mode device DFU version 011a
Device returned transfer size 2048
DfuSe interface name: "Internal Flash  "
Downloading to address = 0x08000000, size = 23004
Download        [=========================] 100%        23004 bytes
Download done.
File downloaded successfully
Transitioning to dfuMANIFEST state
dfu-util: can't detach
Resetting USB to switch back to runtime mode
roby@Machcube:~/klipper $
```

10. Press the RESET button to restart the MCU and enter normal operating mode
11. Run ls /dev/serial/by-id/* should return a device with
    /dev/serial/by-id/usb-Klipper_stm32f042x6_OrbitoolO2-if00
    Copy this virtual serial port name to the OrbitoolO2 config file in the MCU section

```
 9  [mcu orbitoolO2]
10  serial:/dev/serial/by-id/usb-Klipper_stm32f042x6_OrbitoolO2-if00
11  restart_method: command
```

Your tool-board should now be usable with Klipper. Use the example config file to get started.
The best option is to copy the config file into the same directory as printer.cfg.
Add [include OrbitoolO2.cfg] to the beginning of your printer.cfg to include the file.
Comment out the unused I/O features of the OrbitoolO2 config section.

5 Electrical characteristics

5.1 Absolute maximum ratings

Important Note: The Orbitool O2 board can withstand these limits without electrical damage; however, long-term exposure to these limits is not recommended. The device's absolute maximum rating is not the same as the functional range.

The maximum ratings may not be exceeded under any circumstances!

Table 1. Orbitool O2 absolute maximum ratings

Nr.	Parameter	Min	Max	Unit
A1	Ambient temperature range	-20	85	°C
A2	Supply voltage	-30	30	V
A3	Extruder stepper current		1	A
A4	Hot-end heater current		6.8	A
A5	Part fan output current		1	A
A6	Hot-end Fan		1	A
A7	RGB LED 5V supply current		0.5	A
A8	RGB data line output	-1	7	V
A9	USB Data lines	-1	28*	V
A10	Hot-end temperature sensor input	-1	28*	V
A11	End stop sensor input	-1	28*	V
A12	Z sensor I/O interface	-10	+15	V
A13	Orbiter v2 sensor interface inputs	-1	3.3	V
A14	ESD-Protection level for handling (Human Body Model, HMB)	-8	8	kV

*Should not exceed the supply voltage

5.2 Full functional operational limits

Table 2. Orbitool O2 electrical characteristics

Nr.	Parameter	Symbol	Min	Typ	Max	Unit
	General					
P1	Operating temperature range	t_o	-20		60	°C
P2	Max ambient temperature with active cooling	t_{o_cooled}			80	°C
P3	Power supply voltage	v_{PWR}	22		28	V
P4	Power supply voltage measurement accuracy	a_{PWR}	-3		3	%
	Extruder stepper					
P5	Extruder stepper current (max 60°C chamber temp)	i_{MOT_60}			0.85*	A
P6	Extruder stepper current (max 80°C chamber temp)	i_{MOT_80}	0.5		0.6	A
	Hot-end heater					
P7	Hot-end heater nominal current	i_{HEATER}			5	A
P8	Hot-end heater overcurrent switch OFF threshold	i_{OV_HOT}	5.9	6.8	7.2	A
P9	Hot-end overcurrent detection time	t_{OV_HOT}			10	ms
P10	Hot-end power measurement accuracy	a_{P_HOT}	-6		6	%
	Part Fan, Tool FAN					
P11	Part fan output current	i_{PART_FAN}			1	A
P12	Part fan overcurrent switch OFF threshold	i_{OC_PART}	1.2	2	2.5	A
P13	Part fan overcurrent detection time	t_{OC_PART}			5	ms
P14	Part fan PWM frequency	f_{PART_FAN}		100		Hz
	Hot-end Cooling fan					
P15	Hot-end fan output current	i_{HOT_FAN}			0.5	A
P16	Hot-end fan current limitation	$i_{CL\_HOT\_FAN}$	2	3	3.9	A
P17	Hot-end fan PWM frequency	f_{HOT_FAN}	9	10	20	Khz
	External 5V supply					
P18	RGB LED + bed level sensor supply current	i_{5V_Prot}			0.5	A
	Internal supplies					
P19	Internal 5V supply protection current limitation	i_{SC_5V}	2	3	3.9	A
P20	Internal 3.3V supply protection current limitation	i_{SC_3V3}	220	350	550	mA

*Parameter derating with ambient temperature