Create a Raspberry Pi HAT with KiCad 5

Making open source hardware with free and open source software

Leon Anavi
@leonanavi
Pi Wars Mini Conference 2019

Agenda

- KiCad
- Raspberry Pi HAT/uHAT
- Device tree
- Prototyping and low-volume manufacturing
- Open source hardware certification

Software vs Hardware

- Hardware is expensive
- Often you can’t built hardware at home
- Making hardware prototypes takes more time
- Soldering requires skills and practice
- Debugging hardware requires specific physical tools
- Fixing bugs requires a new version of the hardware
- Testing hardware can be dangerous

Philosophical questions

- Is it worth designing open source hardware with expensive proprietary software tools?

- Can you build a sustainable community if your contributors have to pay gazillions for software licenses to modify and contribute back to your open source hardware project?

Free and Open Source Design Software

Popular Electronic Design Automation (EDA):

- Fritzing
- gEDA
- KiCAD

Popular Computer-aided design (CAD):

- OpenSCAD
- QCAD
- FreeCAD
- Blender

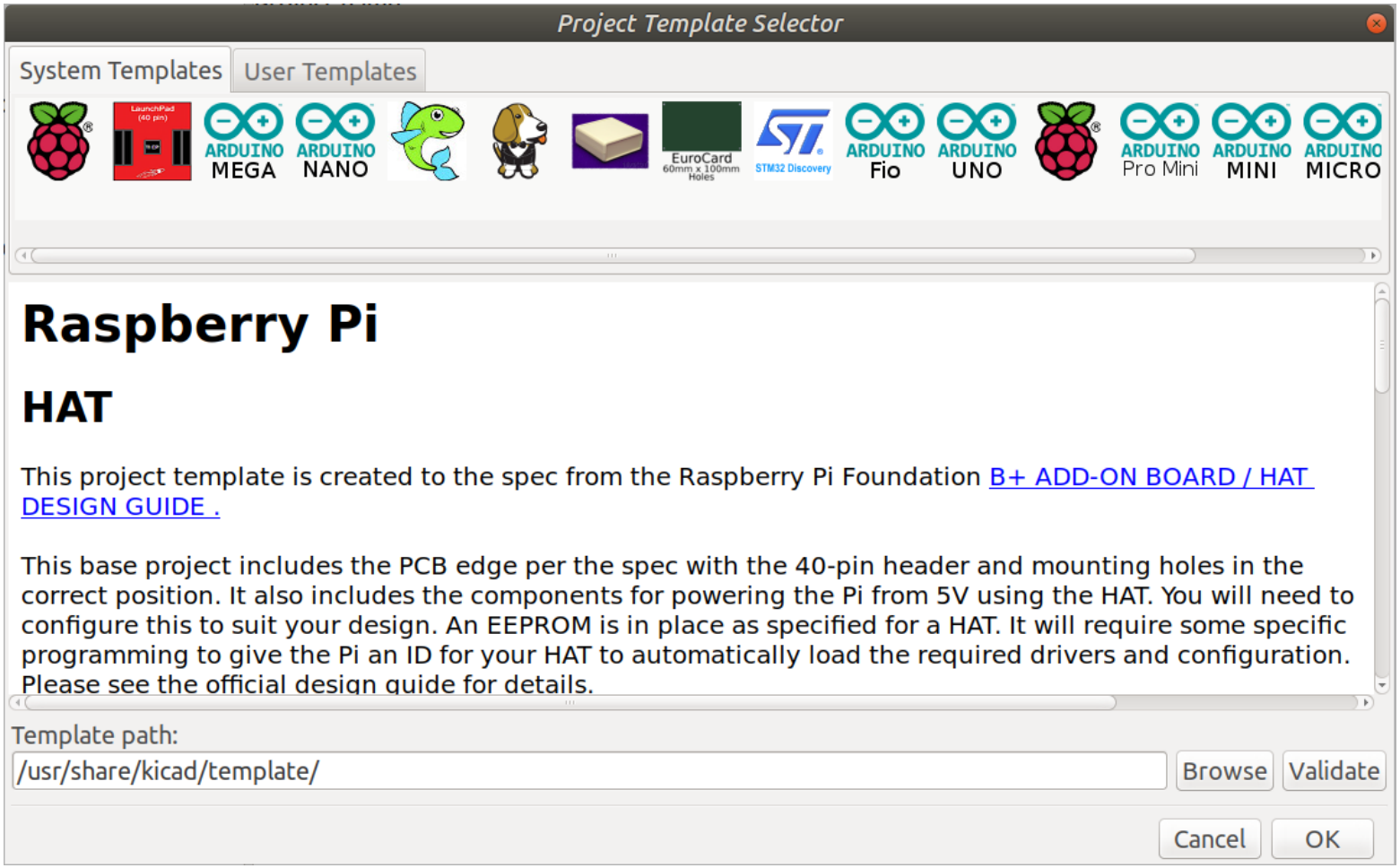
KiCad

- Free & open source EDA software (GPLv3+)
- Cross platform (works on GNU/Linux distributions, MS Windows and Mac OS X)
- Integrated 3D viewer
- Up to 32 copper layers + 14 fixed-purpose technical layers
- Contributions from CERN developers
- Already adopted by the industry
- http://kicad-pcb.org/

KiCad 5 New Features

- Released in July 2018
- New libraries for symbols, footprints and 3D models
- New 3D viewer and new 3D model plugin architecture
- More templates for common development platforms such as Arduino and Raspberry Pi
- Support direct import of Eagle projects
- http://kicad-pcb.org/blog/2018/07/KiCad-5-a-new-generation/

KiCad 5 New Project from Template



HAT/uHAT Requirements

- **H**ardware **A**ttached on **T**op
- Form factor and dimensions according to the mechanical specifications
- 40 pin header compatible with Raspberry Pi B+ and all newer models
- EEPROM with device tree fragment
- Details:

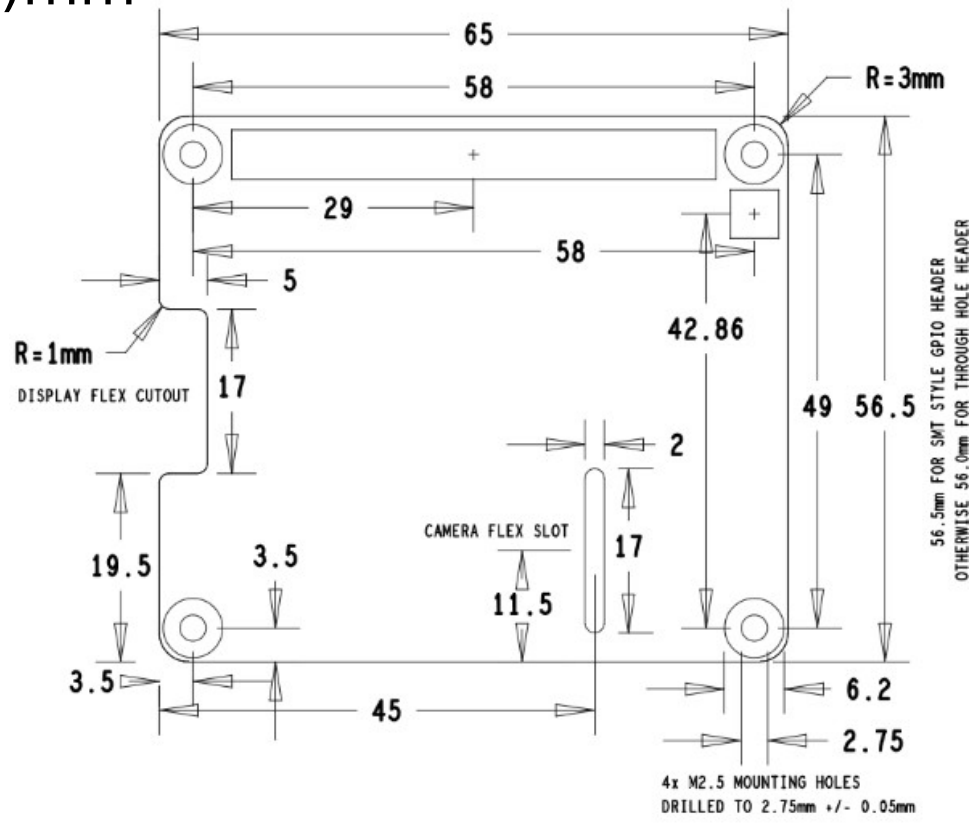
  https://www.raspberrypi.org/blog/introducing-raspberry-pi-hats/

  https://github.com/raspberrypi/hats

  https://www.raspberrypi.org/blog/raspberry-pi-tv-hat/

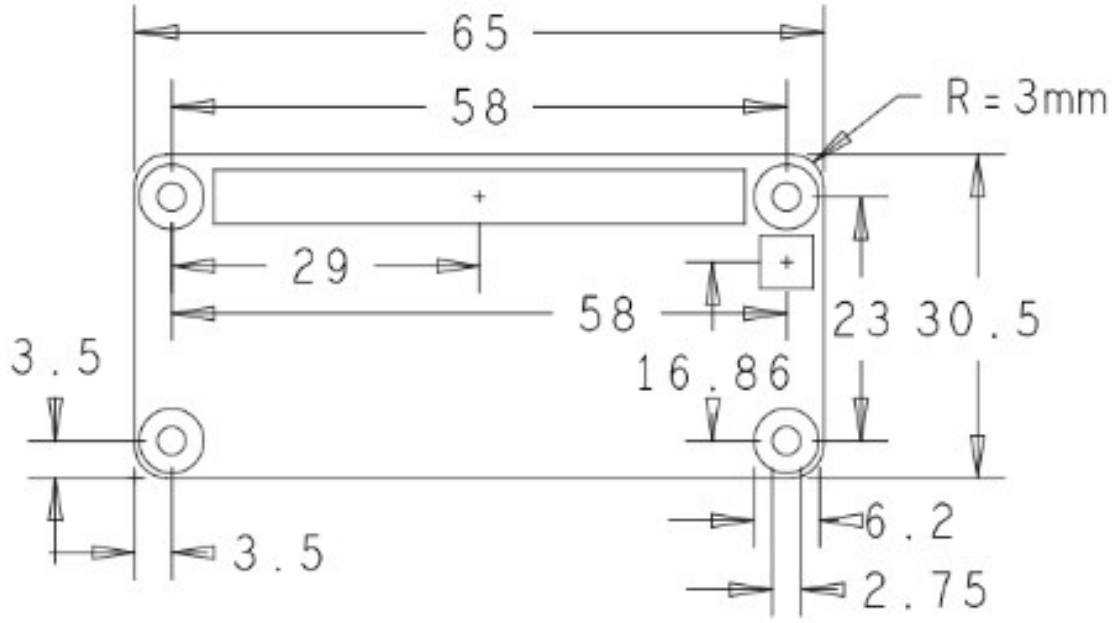
Raspberry Pi HAT Mechanical Specification

- Dimensions: 65 x 56(or 56.5)mm
- 3mm radius corners
- 40-pin GPIO connector
- 4 mounting holes

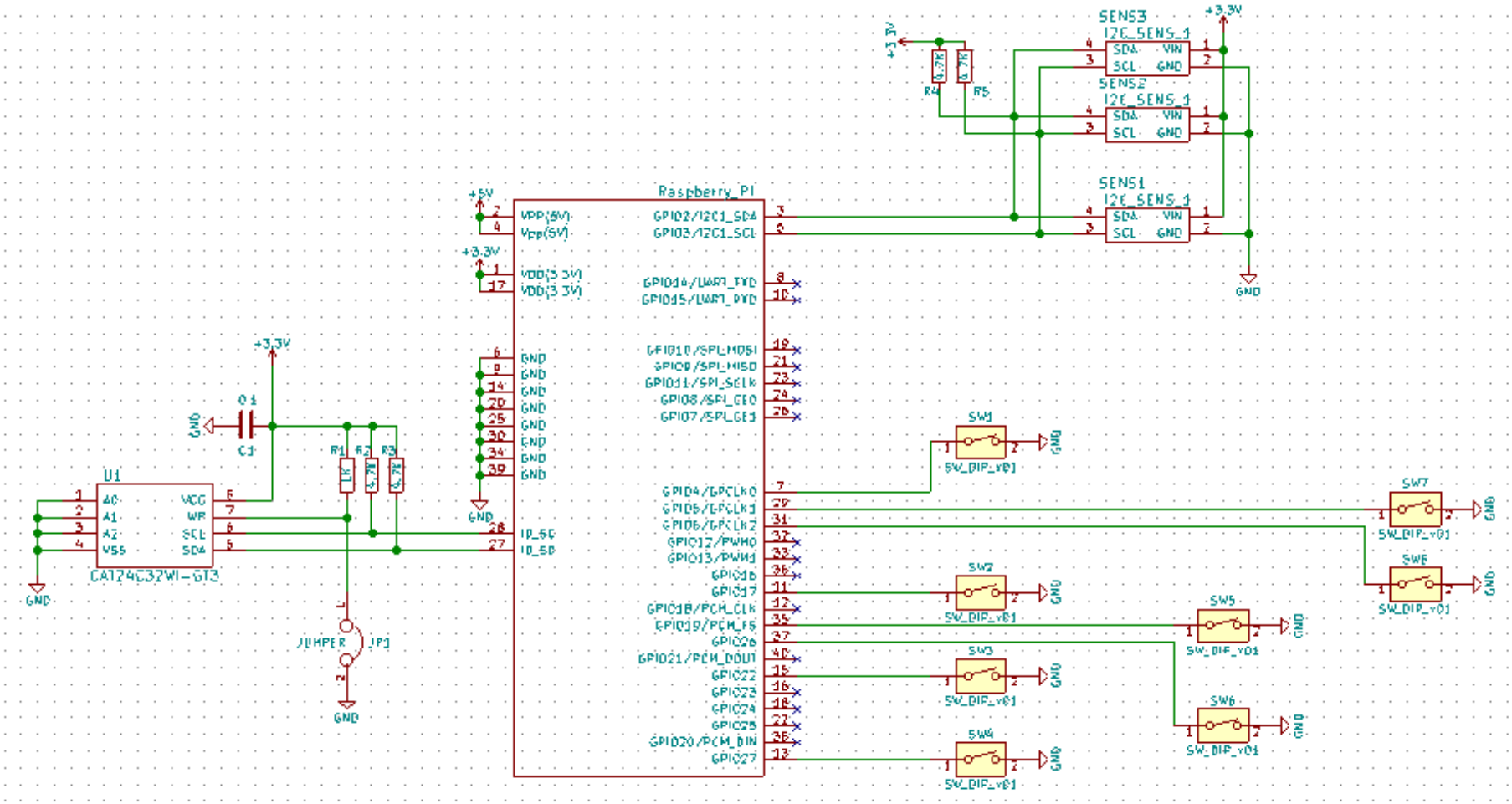


Raspberry Pi uHAT Mechanical Specification

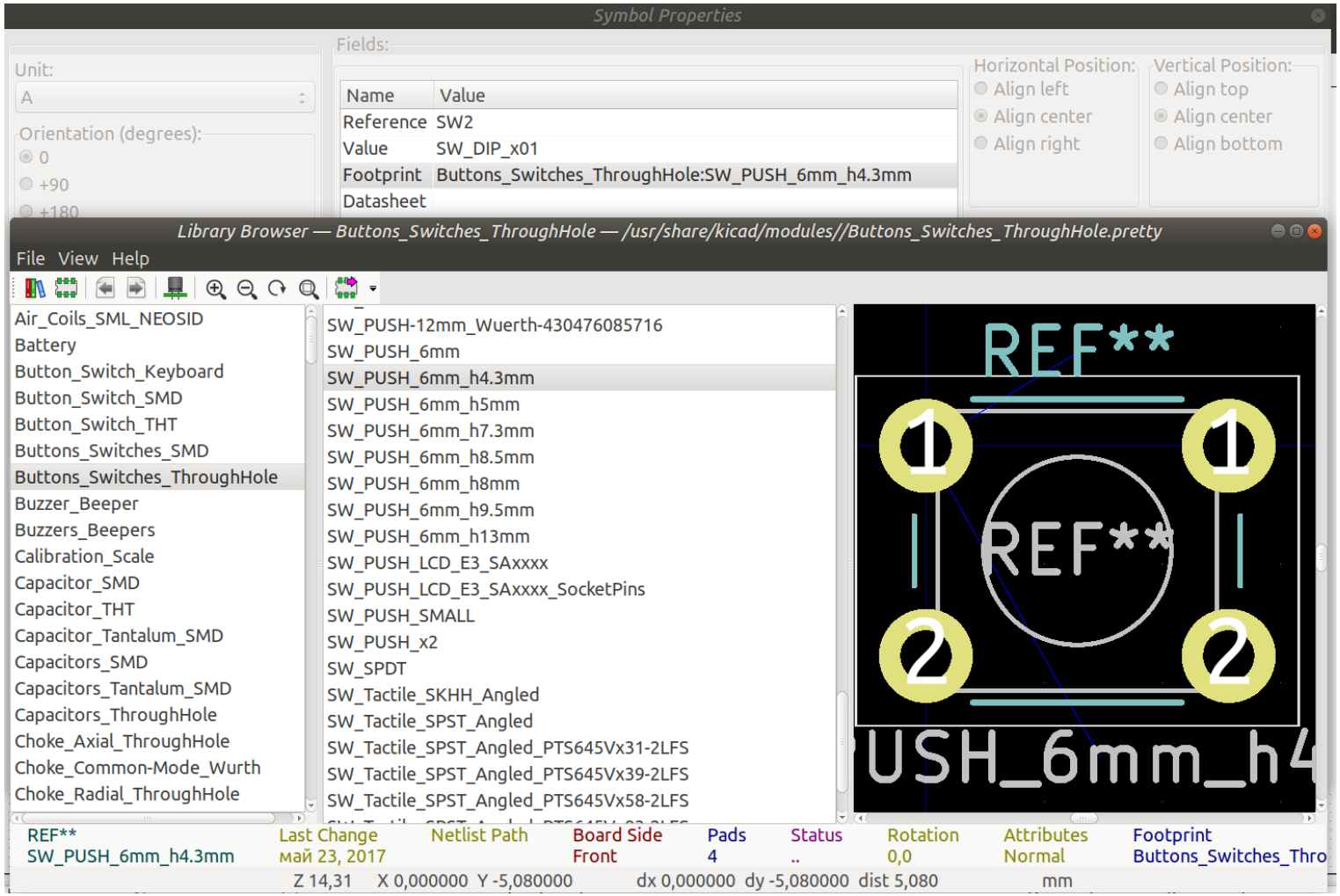
- Dimensions: 65 x 30(or 30.5) mm
- 3mm radius corners
- 40-pin GPIO connector
- At least 3 mounting hol



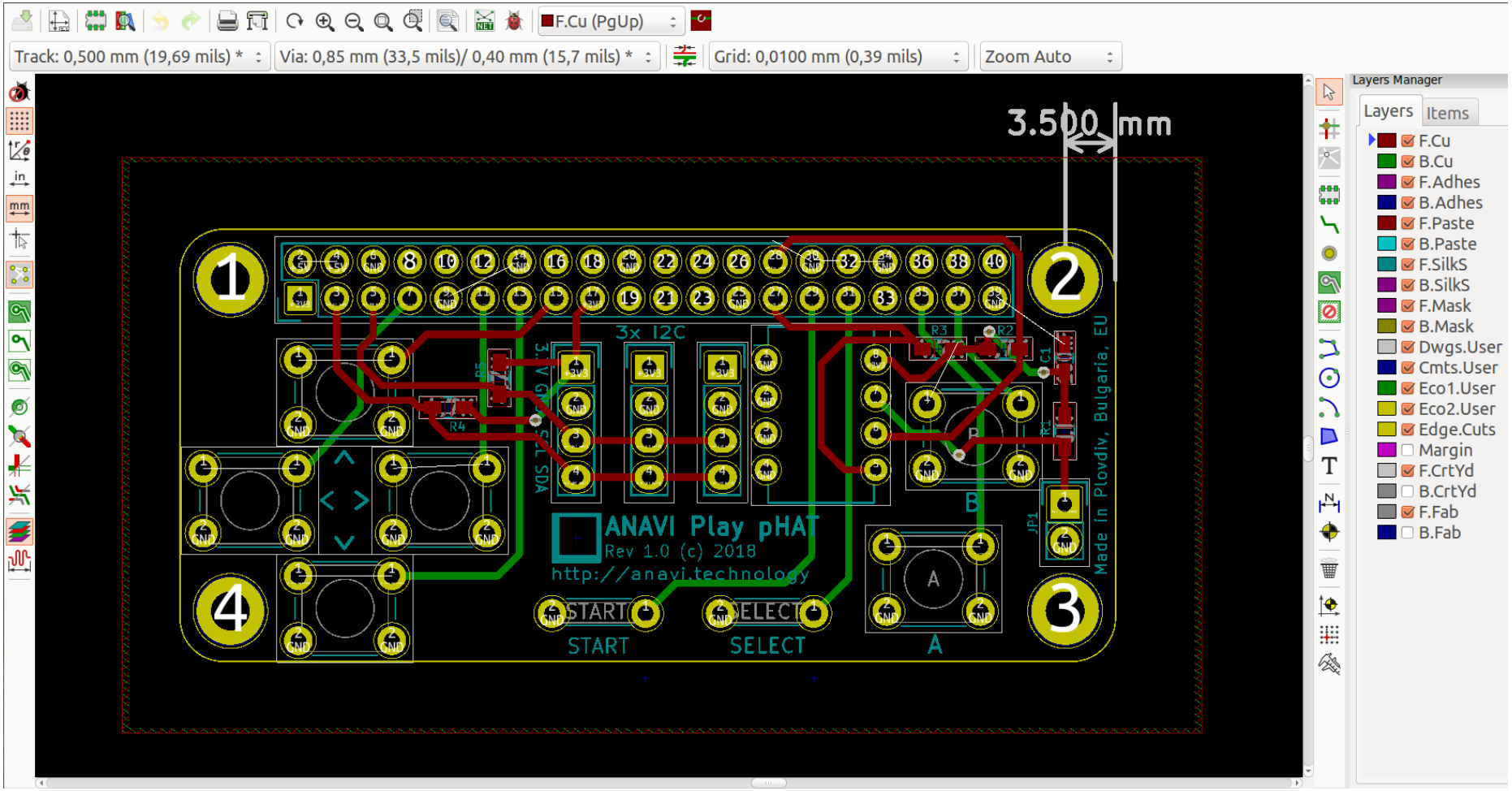
Step 1: Schematics



Step 2: Footprints



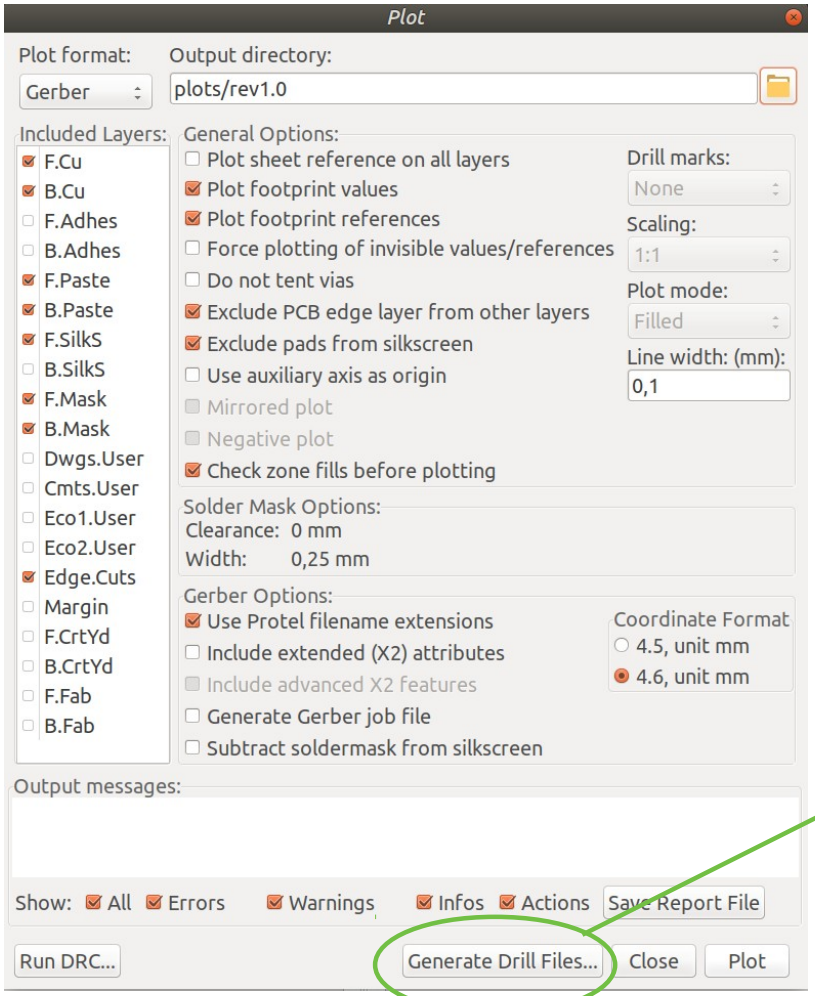
Step 3: PCB Layout

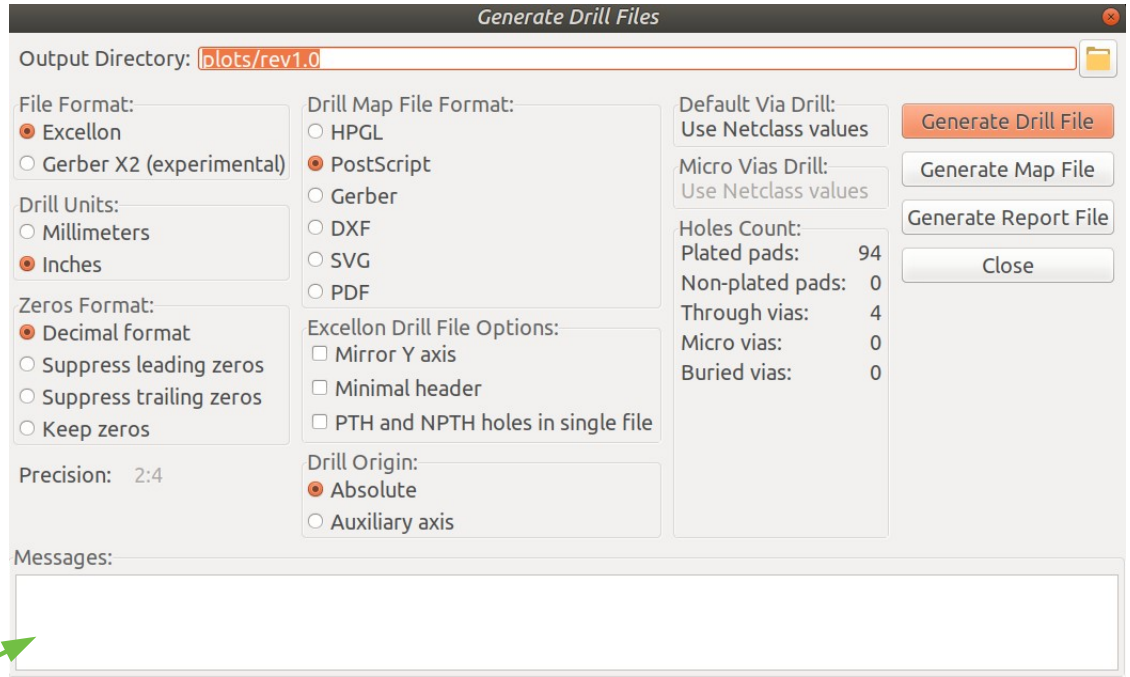


Recommendations

- Comply with the minimum requirements of the PCB manufacturer for trace spaces, drills and angular rings
- Read the datasheets of all components carefully
- Keep in mind the complexity of the assembly process while designing the PCB
- Consider the design of the case simultaneously with the design of the PCB
- Submit often to version control system to get early feedback from the community

Step 4: Export Gerber and Drill Files

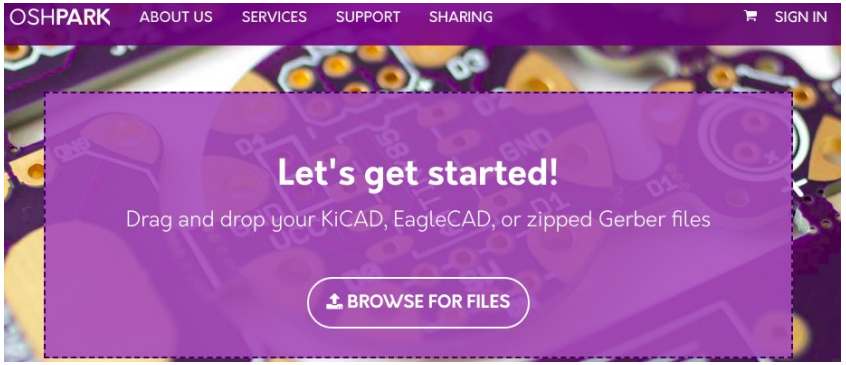


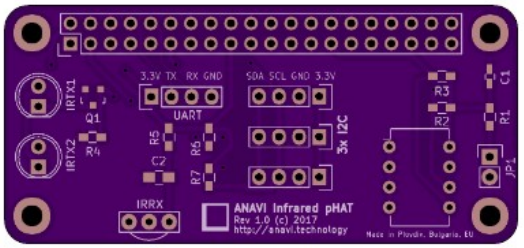


PCB Prototyping

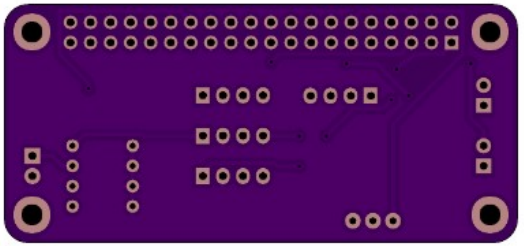
PCB printing services from:

- OSHPark (USA)
- Aisler (Germany)
- PCB train (UK)
- Other local European factories
- JLCPCB (China)
- PCBWAY (China)
- Seeed Studio (China)
- ALLPCB (China)
- Other Chinese factories



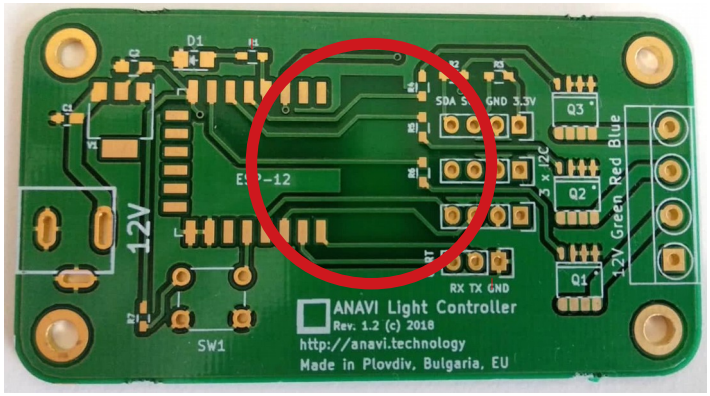


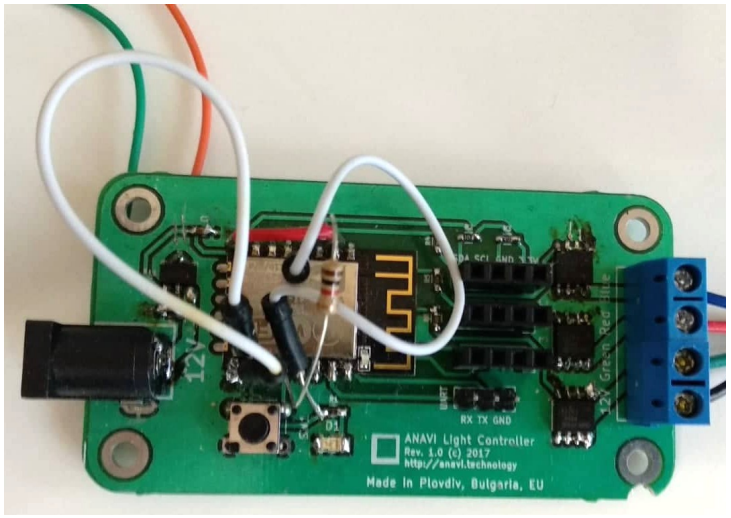
Top

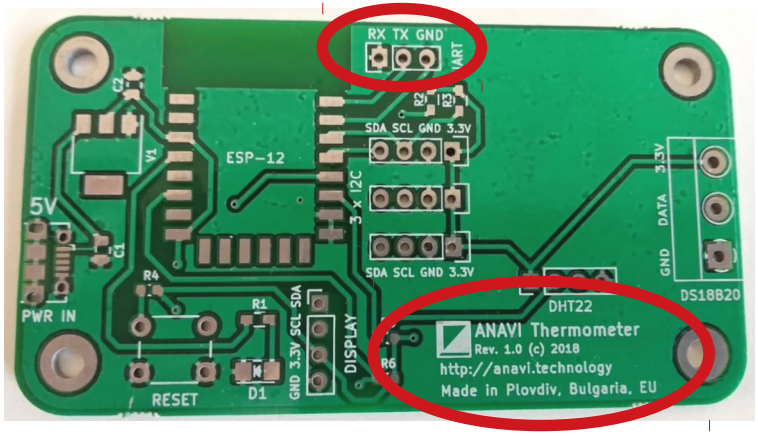


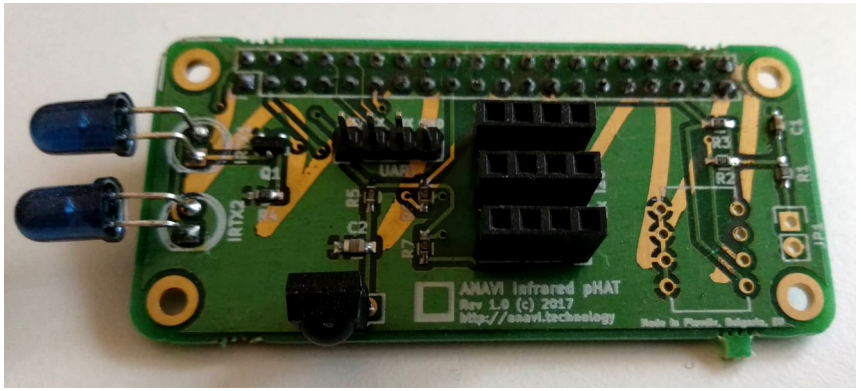
Bottom

Everyone Makes Mistakes



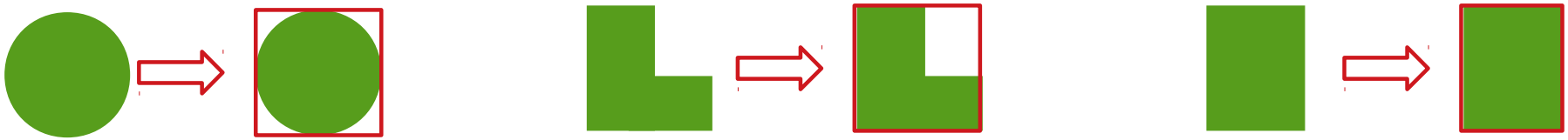




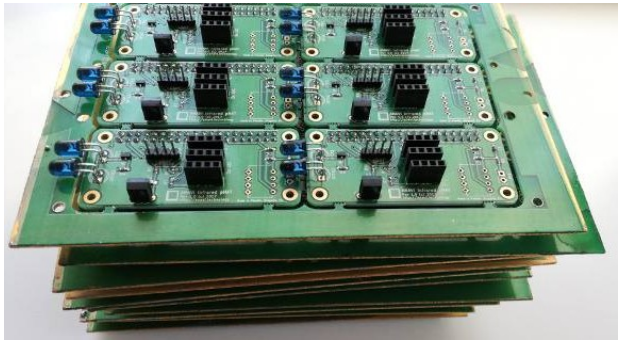


Low Volume Manufacturing (1/3)

- Price depends on PCB size



- Several boards are grouped in a panel



Low Volume Manufacturing (2/3)

- Assembly could be more expensive than the PCB
- Handsoldering SMD (Surface Mount Devices) might be OK for a prototype but is extremely time consuming and not a reasonable option for low volume manufacturing
- Stencil, solder paste and a reflow oven is required for SMT (Surface Mount Technology)
- Pick and place machine might be too expensive for low volume manufacturing of PCB with just a few SMD

Low Volume Manufacturing (3/3)

- Planning low volume manufacturing from local suppliers avoids delays caused by import procedures and holidays in the country of origin of the components
- Local manufacturing allows better QA during the process
- Local manufacturing may be more expensive but cuts the costs for shipping
- Bonus: it is awesome to see your hometown on a PCB

Device Tree

- Specification for a software data structure that describes hardware components
- Allows compiled Linux kernel to support different hardware configurations within a wider architecture family
- The device tree is compiled as an external file from the Linux kernel binary files called **DTB** (Device Tree Blob) from source code in **DTS** (Device Tree Source)
- Device Tree Compiler (DTC) compiles the source into a binary
- For more details:

  https://www.devicetree.org/

  https://elinux.org/images/f/f9/Petazzoni-device-tree-dummies_0.pdf

Device Tree Overlay (DTO)

- DTO enables a central **DTB** (device tree blob) to be overlaid on the device tree
- DTO includes a number of fragments
- With the default bootloader on Raspberry Pi, DTO can be set at **config.txt** on the FAT partition with **device_tree=**
- The EEPROM on a Raspberry Pi add-on board, connected to the secondary I2C bus on pins 27 and 28, should include the overlay required to automatically enable the board
- On Raspbian and other GNU/Linux distributions for Raspberry Pi the information in the EEPROM can be seen from user-space at **/proc/device-tree/hat/**

Example: Mapping Keys in DTS

```
fragment@0 {
   target-path = "/";
   __overlay__ {
      keypad: anavi-play-phat {
         compatible = "gpio-keys";
         #address-cells = <1>;
         #size-cells = <0>;
         autorepeat;
         button@17 {
            label = "right";
            linux,code = <106>;
            gpios = <&gpio 17 1>;
         };
```

- Linux key codes as defined in /usr/include/linux/input-event-codes.h
- Compile **DTBO**:
  dtc -I dts -O dtb -o anavi-play-phat.dtbo anavi-play-phat.dts
- Create settings.txt file for the EEPROM and specify the DTO
- Create binary **.eep** file from the .txt file using **eepmake**
  ./eepmake settings.txt settings.eep anavi-play-phat.dtbo
- DTS for the DTO of Play pHAT:
  https://github.com/AnaviTechnology/hats/blob/anavi/eepromutils/anavi-play-phat.dts

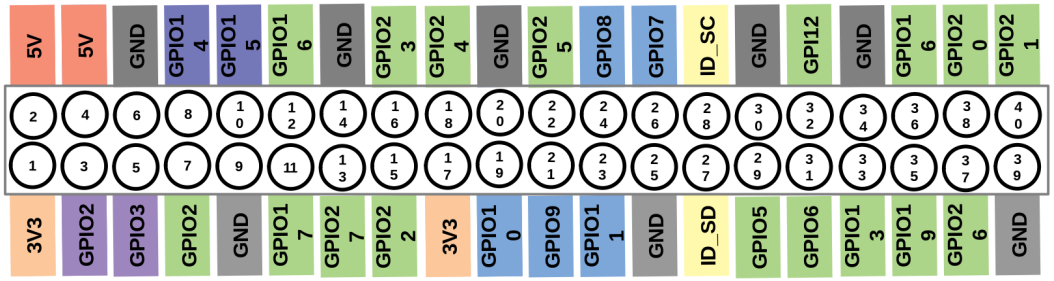
Flashing an EEPROM for a HAT

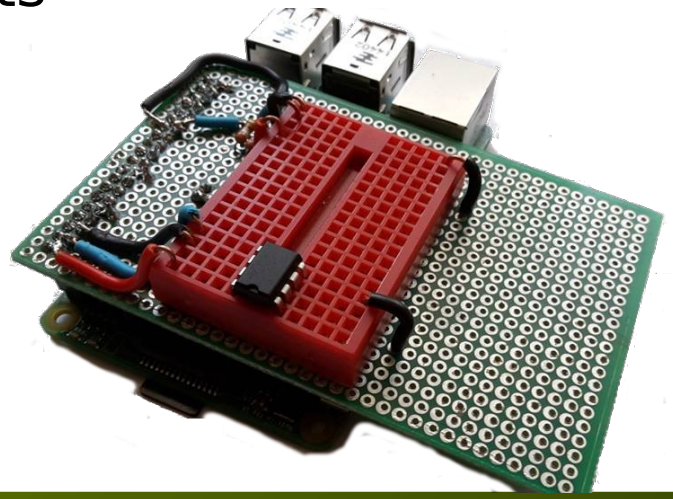
- Writing to the EEPROM:
  pin 3 and 5 (1st I2C)

  Reading from the EEPROM:
  pin 27 and 28 (2nd I2C)

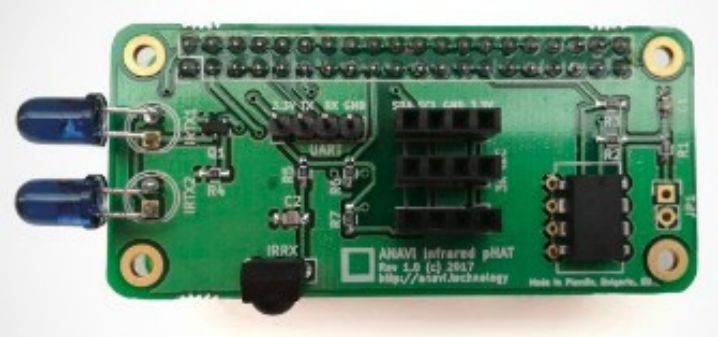
```
pi@raspberrypi: ~
pi@raspberrypi:~ $ ls /proc/device-tree/hat/
name  product  product_id  product_ver  uuid  vendor
pi@raspberrypi:~ $ cat /proc/device-tree/hat/product
ANAVI Infrared pHATpi@raspberrypi:~ $
pi@raspberrypi:~ $ cat /proc/device-tree/hat/vendor
ANAVIpi@raspberrypi:~ $
pi@raspberrypi:~ $
```

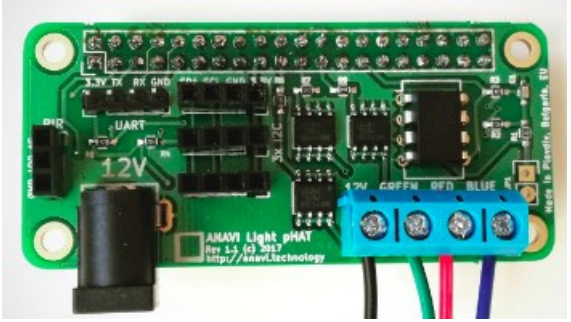
- Flash the **.eep** file with **eepflash**:
  https://github.com/raspberrypi/hats

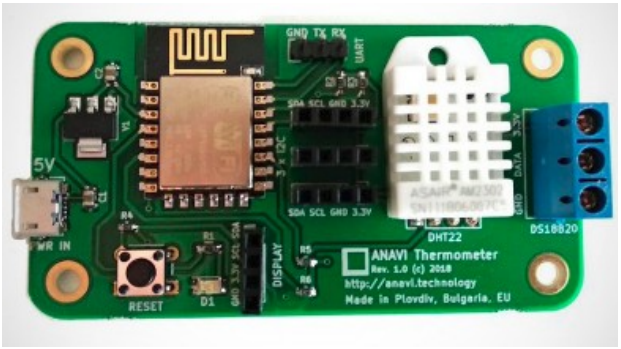


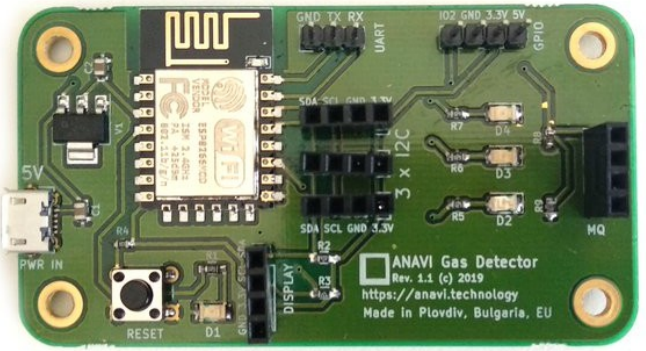


Final Results









One more thing... Open Source Hardware

- Design of physical objects that is publicly available so that anyone can study, modify, distribute, make, and sell the design or hardware based on that design
- To be really open source hardware the project needs 4 main elements: hardware, software, documentation, branding
- A lot of companies and indivuals have been already involved in the open source hardware movement: Arduino, Olimex, SparkFun, Adafruit, Intel, Google, IBM, Prusa...

Open Source Hardware Benefits

- Giving confidence that the design will be available if the original manufacturer stops production
- Keeping the prices low
- Enabling customizations and integration in 3rd party products
- Sharing knowledge, educating students and getting feedback and contributions from the community

Open Source Hardware Licenses

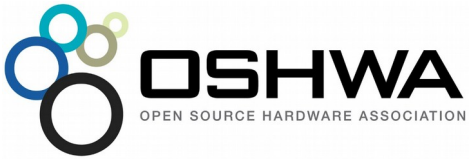
Popular viral licenses:

- GNU General Public License (GPL)
- Creative Commons Attribution-ShareAlike
- CERN Open Hardware License (OHL)
- TAPR Open Hardware License (OHL)

Popular permissive licenses:

- FreeBSD
- MIT
- Solderpad Hardware Licence

Open Source Hardware Certification



Open Source Hardware Association(OSHWA):

- Maintains Certified Projects Directory
- Ensures that the definition of “open source hardware” used by a specific project matches the community definition of open source hardware
- Provides an unique ID for certified hardware based on the country code and a number, for example: BG000007
- Provides an unique logo for the certified ID
- https://certification.oshwa.org/

Conclusions

- KiCad is a high quality free and open source software tool for designing hardware
- Not all Raspberry Pi add-on boards are HATs
- Building and testing a hardware prototype is often more expensive and time consuming (compared to software)
- Open source hardware is a viable business model already used by a lot of well-known companies in the industry
- Open source hardware certification by OSHWA is free and guarantees that the products are really open source

Thank you!

Useful links

- https://www.oshwa.org/
- https://certification.oshwa.org/process.html
- http://kicad-pcb.org/
- https://oshpark.com/
- https://docs.oshpark.com/design-tools/kicad/generating-kicad-gerbers/
- https://github.com/AnaviTechnology/
- https://www.raspberrypi.org/blog/introducing-raspberry-pi-hats/
- https://www.raspberrypi.org/magpi/make-your-own-hat/
- http://pinout.xyz/
- https://github.com/raspberrypi/hats