Automating Open Source Hardware Lightning with Home Assistant and MQTT

Leon Anavi, Senior Software Engineer, Konsulko Group

leon.anavi@konsulko.com
leon@anavi.org

Konsulko Group

- Services company specializing in Embedded Linux and Open Source Software
- Hardware/software build, design, development, and training services
- Based in San Jose, CA with an engineering presence worldwide

- http://konsulko.com/

Agenda

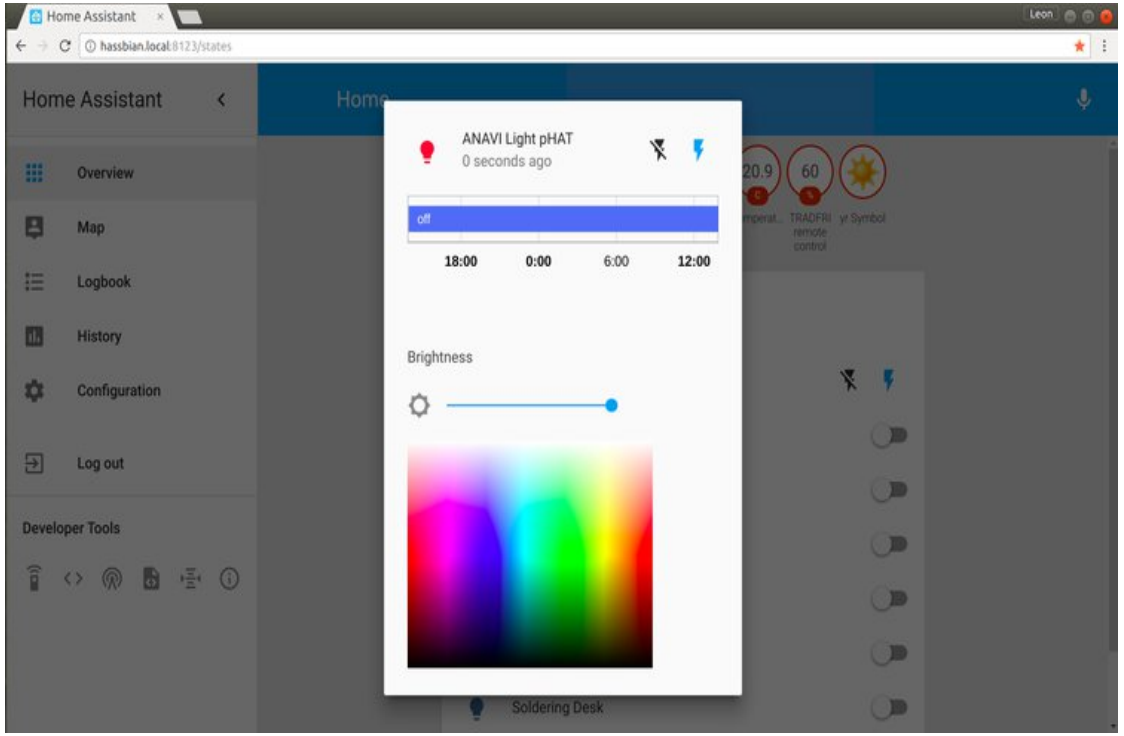
- Combining open source hardware with free and open source software for smart lightning
- Designing PCB with KiCAD
- Home Assistant
- MQTT
- Q&A

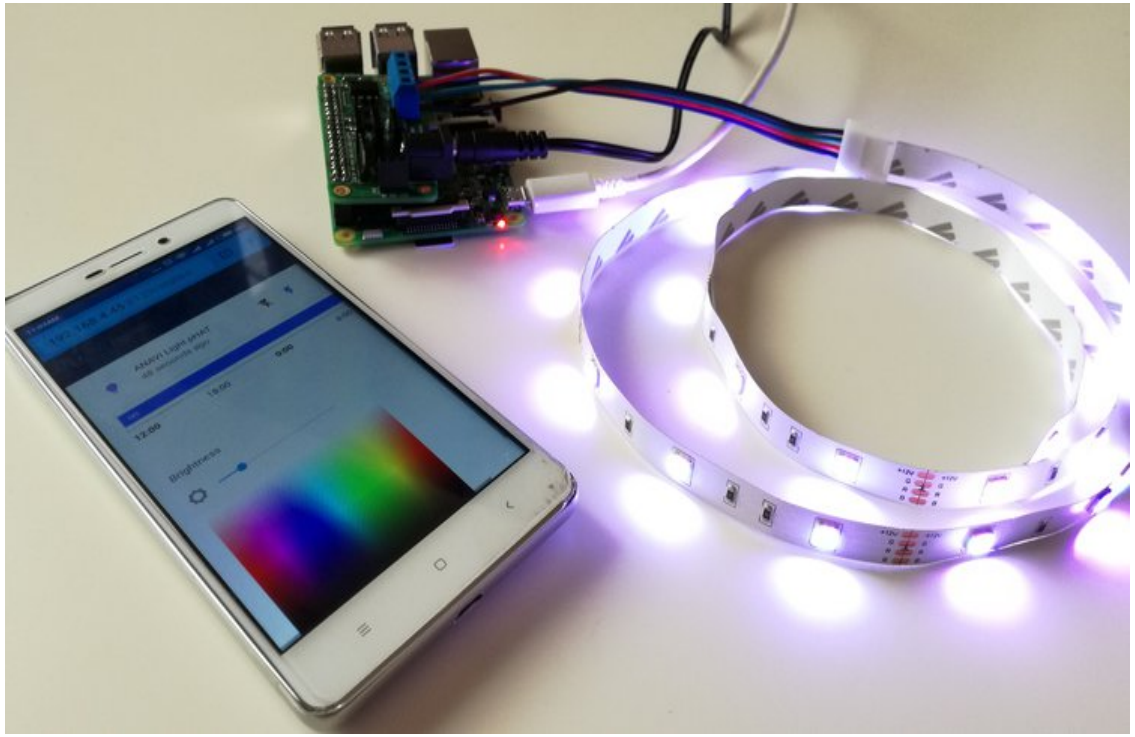
Open Source Hardware

- 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
- **Copyleft** (like Creative Commons Attribution-ShareAlike, GPL, OHL) or **permissive** licenses (like MIT)
- The Open Source Hardware Association (OSHWA) maintains the Open Source Hardware certification

How difficult is it to create smart lightning solution that combines **open source hardware** with **free and open source software**?

DEMO

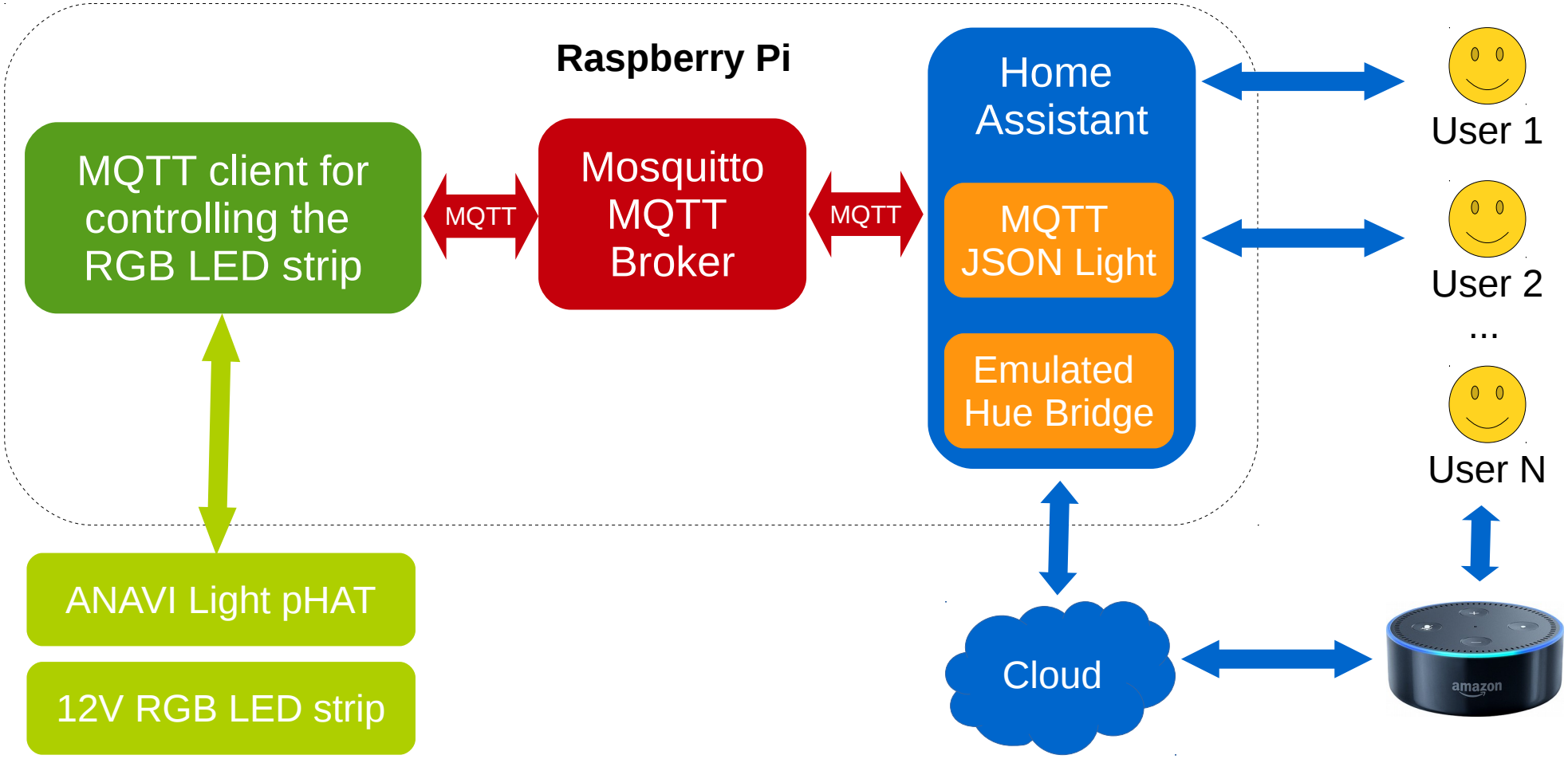

Home Assistant
hassbian.local:8123/states
Home Assistant
Overview
Map
Logbook
History
Configuration
Log out
Developer Tools
ANAVI Light pHAT
0 seconds ago
off
18:00
0:00
6:00
12:00
Brightness
Soldering Desk



Required Hardware

- Raspberry Pi 3 (or any other model and version with 40 pin header)
- ANAVI Light pHAT
- 12V 5050 RGB LED strip
- 12V power supply with 5.5*2.1mm DC jack
- 5V microUSB charger
- Amazon Echo Dot 2nd generation

How Does it Work?

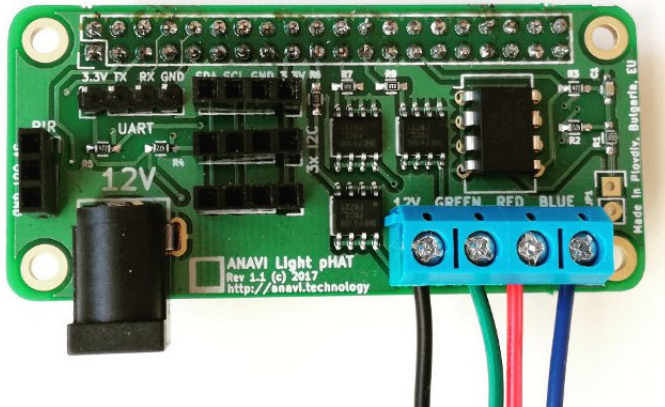


Linux
MQTT
Paho
piGPIO
Eclipse
Mosquitto
JSON
Git
wiringPi
Home Assistant
debuild
Python
Hass.io
Hassbian
Raspbian
Makefile
Docker
ResinOS
OpenSCAD
systemd
C
GCC
HTML5
Ubuntu
JavaScript
KiCAD
C++
Etcher
CSS
YAML

ANAVI Light pHAT

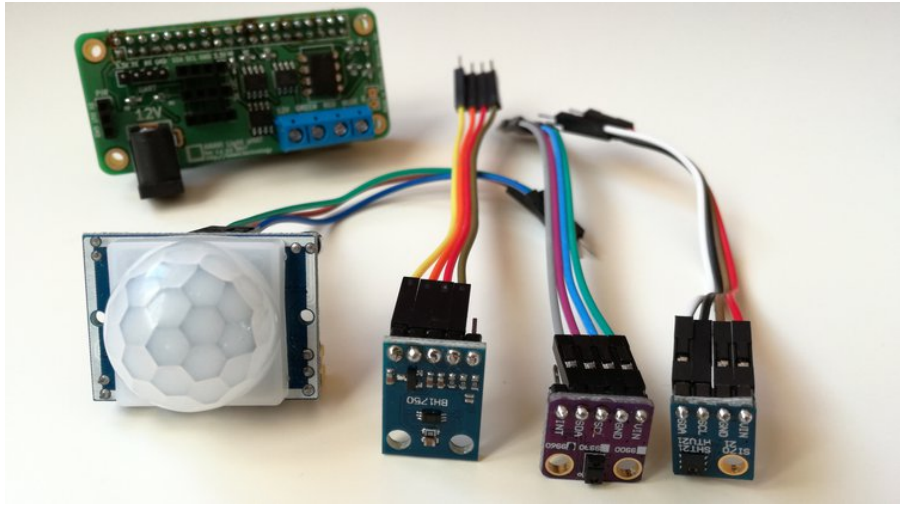
- Raspberry Pi add-on board for controlling 12V RGB LED strip
- Compatible with any model and version of Raspberry Pi with 40 pin header
- Supports PIR motion sensor and up to 3 I2C sensor modules
- Open source hardware (CC BY-SA 4.0 license) designed with KiCAD EDA

- Schematics: https://github.com/AnaviTechnology/anavi-light
- Crowdfunding: https://www.crowdsupply.com/anavi-technology/light-phat



Supported Sensors

- BH1750 I2C sensor module for ambient light intensity (Lux)
- HTU21D I2C sensor module for temperature and humidity
- APDS-9960 I2C sensor module for RGB color and gesture detection
- BMP180 I2C sensor module for temperature and barometric pressure
- PIR motion sensor



- Examples:

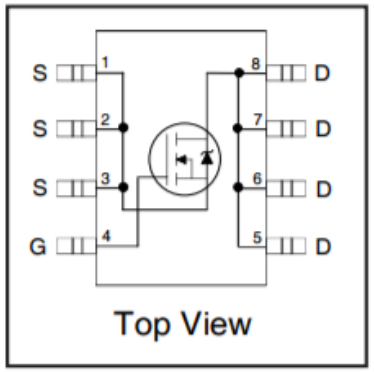
  https://github.com/AnaviTechnology/anavi-examples/tree/master/sensors

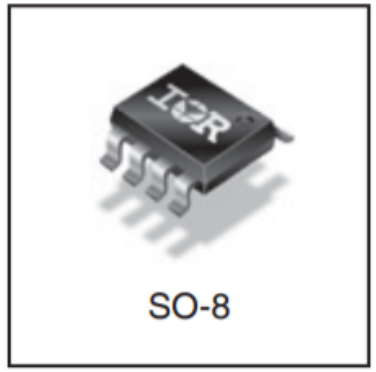
Designing PCB with KiCAD EDA

- Free & open source software (GNU GPLv3)
- Cross platform (works on GNU/Linux distributions, MS Windows and Mac OS X)
- Supports PCB with multiple layers and integrated 3D viewer
- Contributions from CERN developers
- Already well adopted by the industry
- Written in C++, source in Git repositories

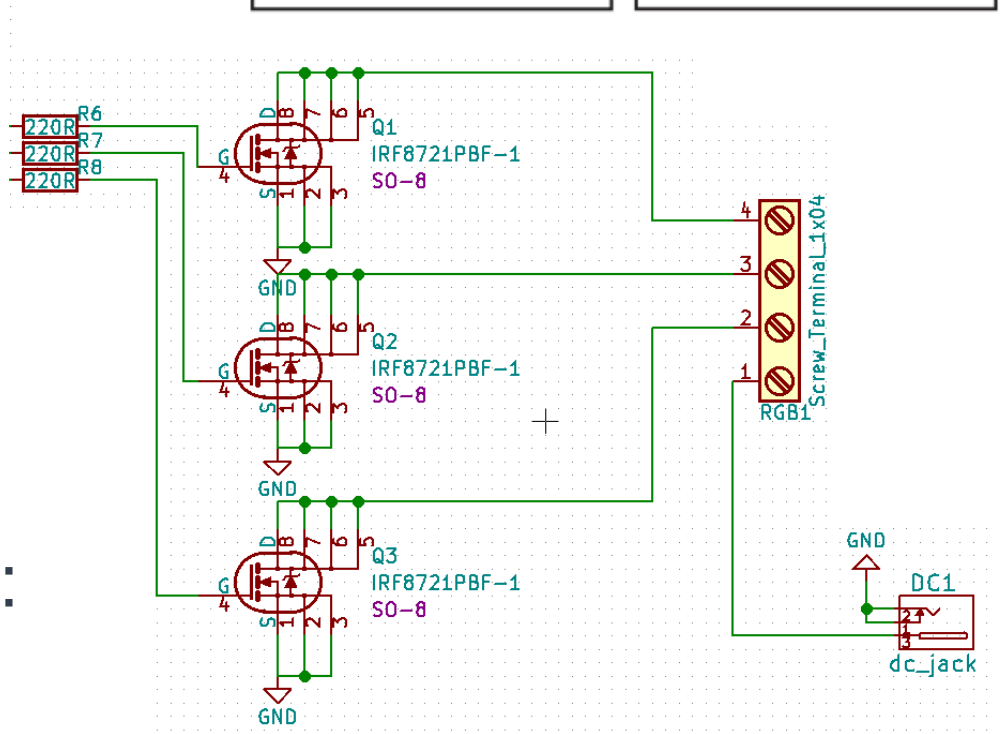
- http://kicad-pcb.org/

Wiring 12V RGB 5050 LED Strip

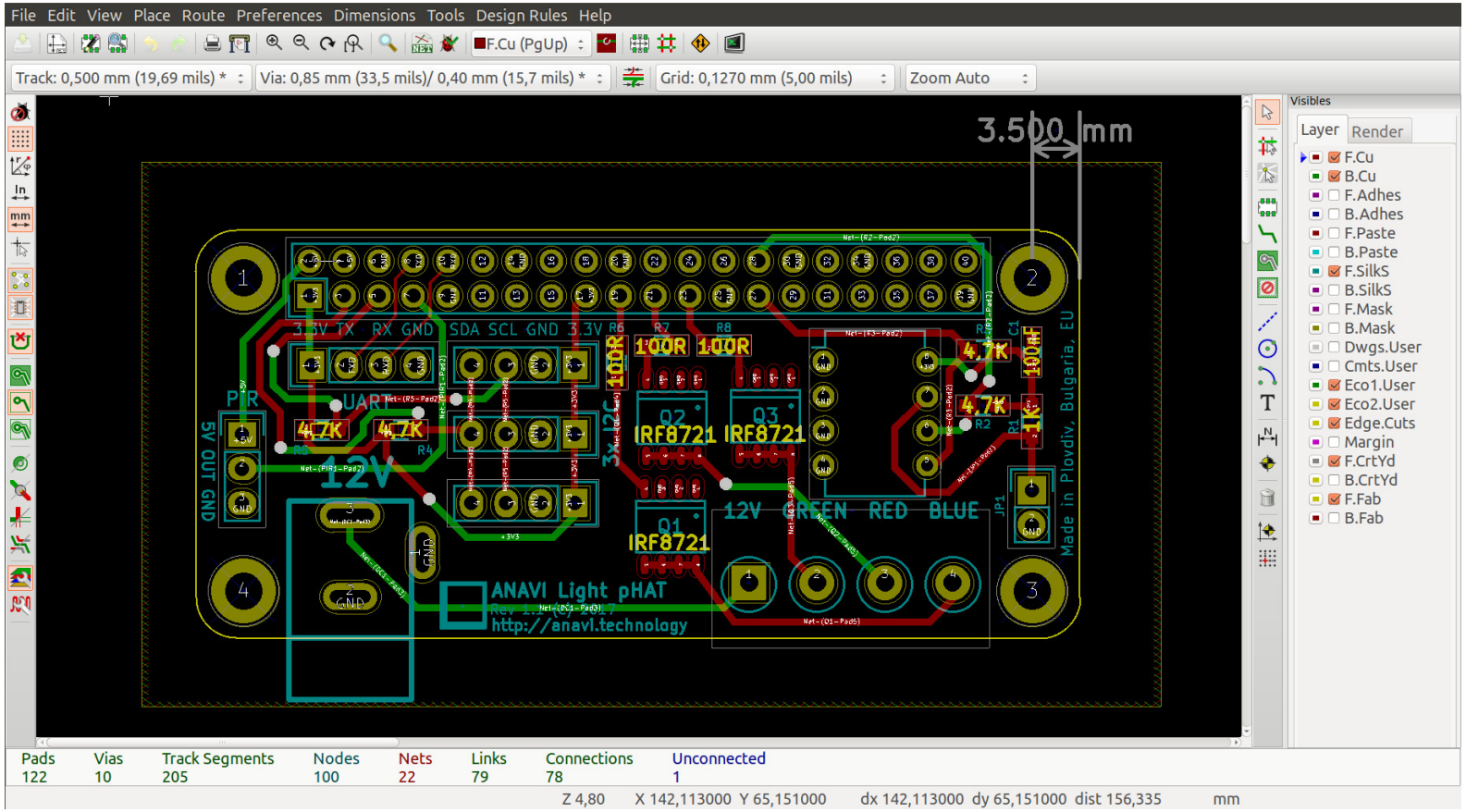




- 30V Single N-Channel HEXFET Power MOSFET IRF8721PbF per each color
- Packed in SO-8 for SMT
- 12V Power supply for the LED strip
- Pulse-width modulation (PWM) to set a color from 16 777 216 (2563) possible combinations through the 3 main colors: red, green and blue

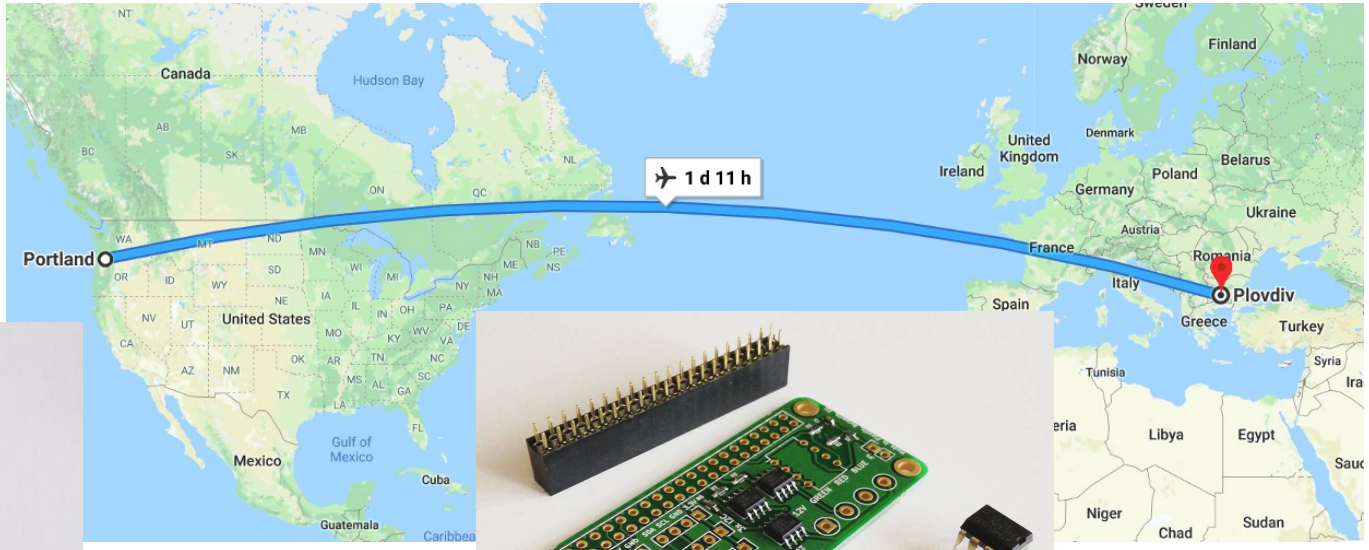


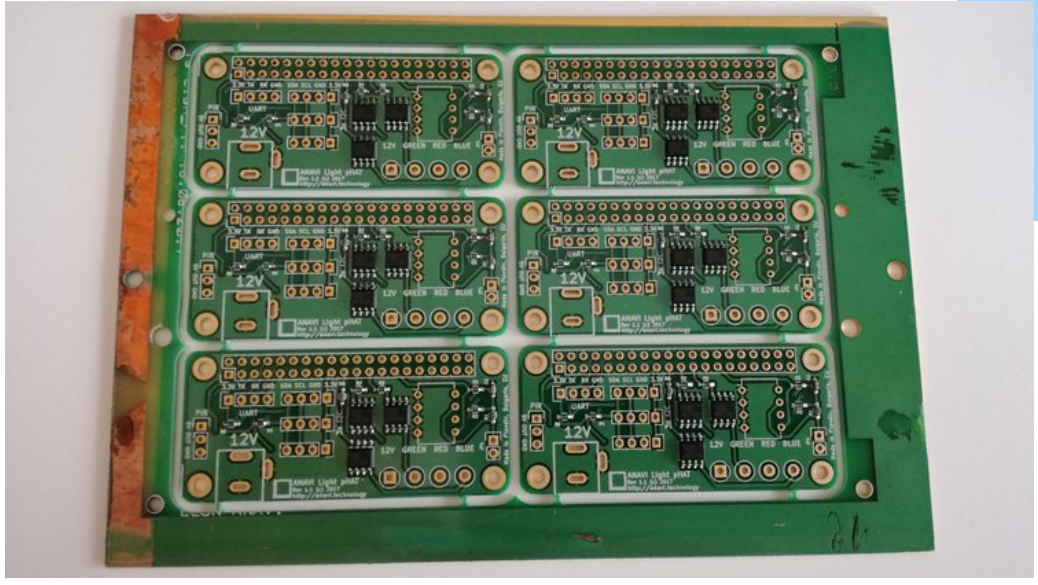
The PCB for ANAVI Light pHAT in KiCAD

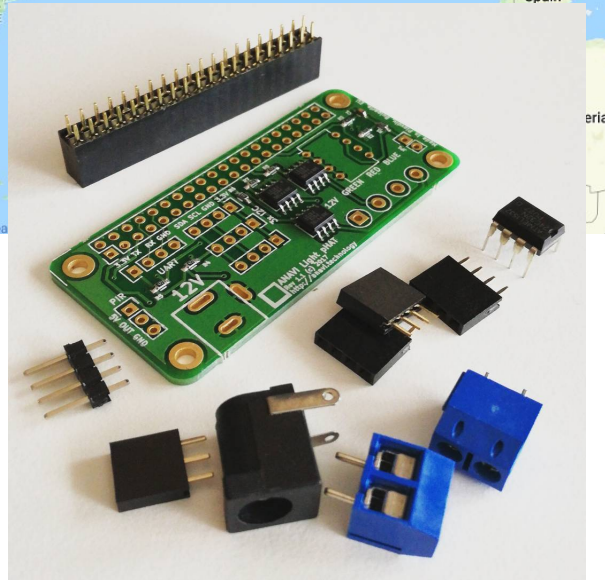


Low Volume Hardware Manufacturing

- Panel with printed circuit boards (PCB)
- SMT and THT assembly
- Made in Plovdiv, Bulgaria







Home Assistant

- Open-source home automation platform running on Python 3 started in 2013 by Paulus Schoutsen
- Perfect to run on a Raspberry Pi
- More than 950 components for integration with popular Internet of Things such as IKEA Trådfri, Philips Hue, Google Assistant, Alexa / Amazon Echo, Nest, KODI, etc.
- Huge community, more than 830 contributors
- Source code available at GitHub under Apache 2.0 license
- https://home-assistant.io/

Home Assistant on Raspberry Pi

- **Hass.io**

  An operating system based on ResinOS and Docker for running Home Assistant. Started by Pascal Vizeli in 2017. Compatible with Raspberry Pi, Intel NUC or generic Linux servers.

- **Hassbian (used for the demo)**

  GNU/Linux distribution for Raspberry Pi with Home Assistant based on Raspbian that uses the same repositories.

MQTT

- Lightweight publish/subscribe machine-to-machine protocol on top of TCP/IP
- Near real-time communication
- Message broker
- Small source code footprint for embedded devices
- Open source MQTT brokers: Mosquitto, HiveMQ, Mosca, emqttd, etc.

- http://mqtt.org/

Mosquitto

- Open source MQTT broker implemented in the C programming language
- Supports MQTT protocol version 3.1 and 3.1.1
- Supports web sockets
- Available for all popular GNU/Linux distributions, Windows, FreeBSD and Mac
- Project of iot.eclipse.com

- https://mosquitto.org/

Installing Mosquitto on Hassbian

- Script hassbian-config simplifies the installation of the latest version of the Mosquitto package from the official repository:

```
sudo hassbian-config install mosquitto
```

- Configure client authentication in Mosquitto using a username and password (optional but highly recommended)
- Set the MQTT broker in configuration.yaml for Home Assistant

```
mqtt:
  discovery: true
  broker: hassbian.local
  port: 1883
```

anavid

- Open source Linux daemon application written C for managing ANAVI Light pHAT on Raspberry Pi through MQTT
- Supports the format of MQTT JSON Light component of Home Assistant
- Uses Paho MQTT C library for implementation of MQTT client
- Uses PiGPIO library for PWM control of the RGB LED strip
- Uses WiringPi library for retrieving data from the supported I2C sensor modules
- Available at GitHub under GNU General Public License v3.0:
- https://github.com/AnaviTechnology/anavid

Installing anavid

- Building from source

```
git clone https://github.com/AnaviTechnology/anavid.git
cd anavid
make
sudo make install
```

- Using deb package (created with debuild)

```
sudo dpkg -i anavi_0.0.1_armhf.deb
```

- Configurations stored at /etc/anavilight.ini

Eclipse Paho Project

- Provides open-source client implementations of MQTT and MQTT-SN messaging protocols
- Supports various programming language: C, C++, Java, JavaScript, Python, Go, Rust, C#
- Project of the Eclipse Foundation
- anavid uses Paho MQTT C client for Posix and Windows:

```
MQTTClient_create(&client, config.address, config.clientId,
        MQTTCLIENT_PERSISTENCE_NONE, NULL);
```

```
MQTTClient_setCallbacks(client, NULL, connlost, msgarrvd, delivered);
```

```
MQTTClient_publishMessage(client, mqttTopic, &pubmsg, &token);
```

The piGPIO Library

- Written in the C programming language
- The pigpio daemon offers a socket and pipe interface to the underlying C library
- Source code at GitHub: https://github.com/joan2937/pigpio
- Used by anavid for controlling the color of the RGB LED strip through software PWM:

```
cmdCmd_t cmd;
cmd.cmd = PI_CMD_PWM;
cmd.p1 = pin;
cmd.p2 = value;
cmd.p3 = 0;
```

```
send(sock, &cmd, sizeof(cmdCmd_t), 0)

…

recv(sock, &cmd, sizeof(cmdCmd_t), MSG_WAITALL)
```

Systemd

- Init system in Linux distributions
- Runs as PID 1 and starts the rest of the system
- anavi systemd service:

```
[Unit]
Description=ANAVI Daemon Service
After=pigpio.service
Requires=pigpio.service

[Service]
Type=simple
ExecStart=/usr/local/bin/anavid
Restart=on-failure
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Managing anavi.service

- Enable systemd service at launch of Raspberry Pi:

```
sudo systemctl enable anavi
```

- Start the systemd service:

```
sudo systemctl start anavi
```

- Stop the systemd service:

```
sudo systemctl stop anavi
```

- Check the status of the service:

```
sudo systemctl status -l anavi
```

- See all log messages:

```
sudo journalctl -u anavi
```

Machine ID

- Unique machine ID of the local system that is initialized by systemd-machine-id-setup during installation or boot
- Stored at at single newline-terminated, hexadecimal, 32-character, lowercase text at /etc/machine-id
- Used by anavid as MD5 hash in MQTT topics to identify different devices

- For details visit: https://www.freedesktop.org/software/systemd/man/machine-id.html

MQTT JSON Light Component

- Home Assistant component for controlling a MQTT-enabled light that can receive JSON messages
- https://home-assistant.io/components/light.mqtt_json/
- Example configuration for ANAVI Light pHAT, where YOURMACHINEID should match /etc/machine-id:

```
light:
  - platform: mqtt_json
    name: "ANAVI Light pHAT"
    command_topic: "YOURMACHINEID/action/rgbled"
    brightness: true
    rgb: true
```

MQTT JSON Light Messages

- MQTT topic:

```
<MD5 OF YOUR MACHINE ID>/action/rgbled
```

- Setting brightness in range from 0 to 255:

```
{
	"brightness": 255
}
```

- Setting specific color in RGB color model in range from 0 to 255 for each LED, for example pink:

```
{
	"red": 255,
	"green": 20,
	"blue": 147
}
```

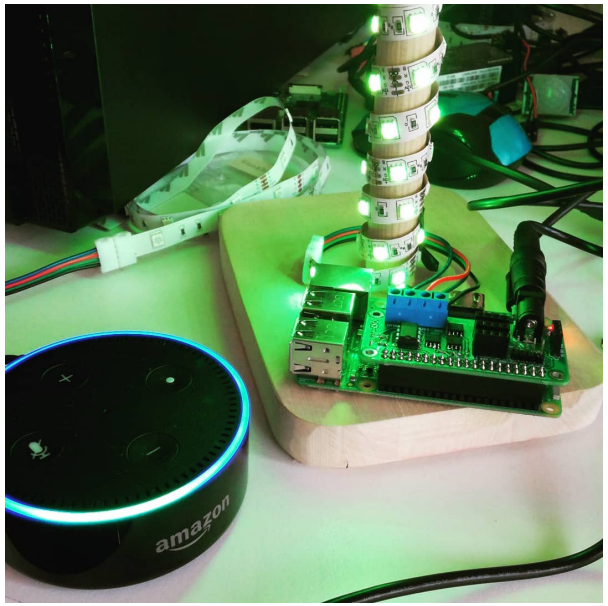
Integration with Alexa and Amazon Echo

- Home assistant is compatible with Alexa and Amazon Echo
- Basic integration using the Emulated Hue Bridge component of Home Assistant
- Emulated Hue Bridge allows non-Philips Hue devices to be controlled though with voice the built-in support of Amazon Echo

```
emulated_hue:
  type: alexa
  expose_by_default: true
```

Simple Voice Control with Alexa

- Example voice commands for MQTT JSON Light component configured with name "*ANAVI Light pHAT*" in *configuration.yaml*:

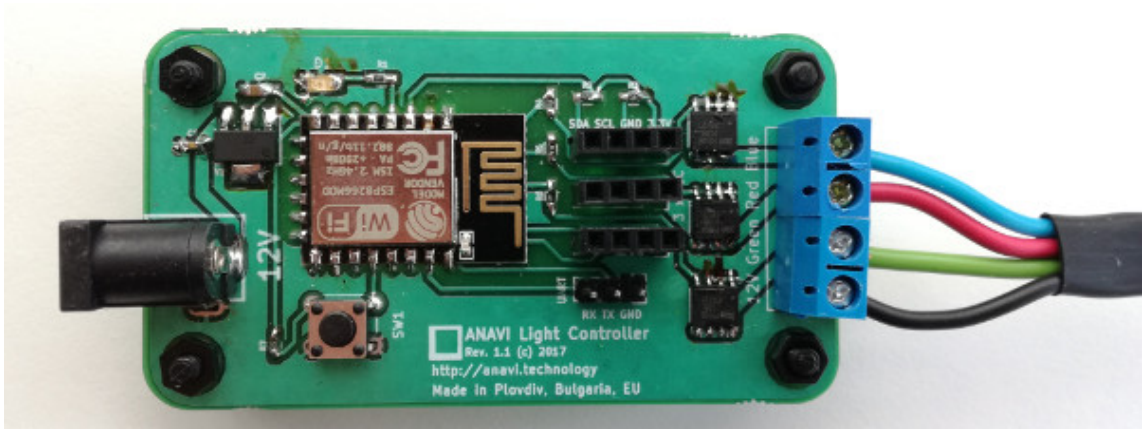


Alexa, turn **ON** ANAVI Light pHAT

Alexa, turn **OFF** ANAVI Light pHAT

ANAVI Light Controller

- Low cost alternative as a standalone embedded device for controlling 12V RGB LED strip with ESP8266 WiFi microcontroller (work in progress)
- Open source hardware designed in KiCAD EDA and software powered by ESP8266 core for Arduino



- https://github.com/AnaviTechnology/anavi-light-controller

Software for ANAVI Light Controller

- Flashing through UART to USB debug cable and by holding the button connected to GPIO0 at boot time
- Relies on **ESP8266 core for Arduino** and depends on the libraries WiFiManager, PubSubClient and ArduinoJson
- Gets chip ID as a 32-bit integer from the ESP-specific API ESP.getChipId()
- Software defined PWN using analogWrite() with default frequency of 1kHz

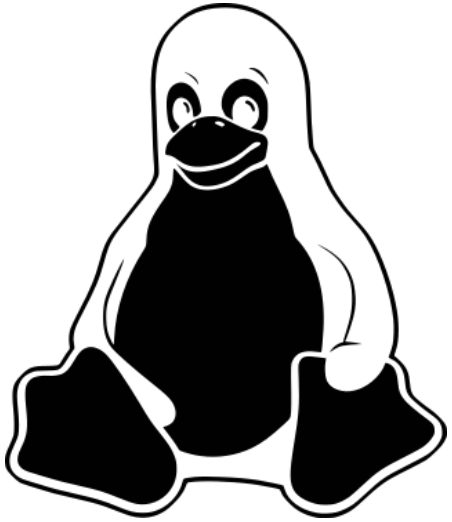
Comparison

	ANAVI Light pHAT	ANAVI Light Controller
12V RGB LED	**Yes**	**Yes**
WiFi	**Yes** (through Raspberry Pi)	**Yes**
Open Source Hardware	**Yes**	**Yes**
Slots for I2C sensors	**3**	**3**
PIR motion sensor	**Yes**	**No**
Stand-alone	**No** (requires Raspberry Pi with 40 pin header)	**Yes** (with built-in ESP8266 module)
Power Supply	**12V 5.5*2.1 DC jack + 5V microUSB** for Raspberry Pi	**12V 5.5*2.1 DC jack**

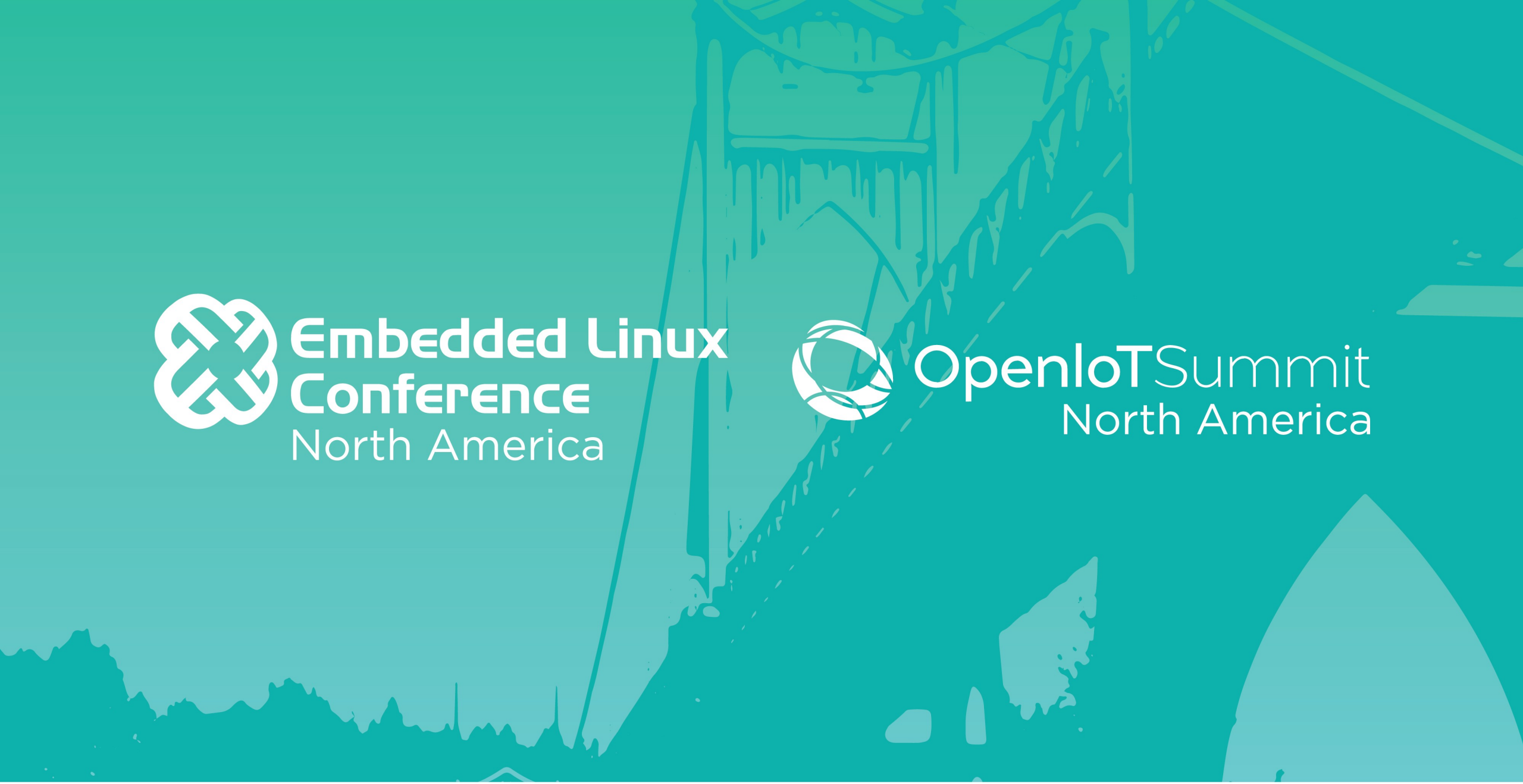
Conclusions

- Combining open source hardware with free and open source software brings value to the community as it allows anyone to study, modify, make and improve the design
- The integration of new embedded devices through the MQTT JSON Light component in Home Assistant is straight-forward
- Further efforts are needed to make the proposed open source solution for smart lightning more secure, user-friendly and easier to setup for non-technical users
- Using microcontrollers with WiFi, like ESP8266, significantly reduces the cost of the smart lightning

Useful Links



- https://home-assistant.io/
- http://mqtt.org/
- https://mosquitto.org/
- https://www.eclipse.org/paho/
- https://www.linux.com/news/home-assistant-python-approach-home-automation-video
- http://abyz.me.uk/rpi/pigpio/
- http://esp8266.github.io/Arduino/
- https://github.com/AnaviTechnology/
- https://www.crowdsupply.com/anavi-technology/light-phat


Embedded Linux
Conference
North America
OpenIoTSummit
North America