Laumio Documentation

Release 1.0.0

HAUM

February 26, 2017

Contents

Laumio is a hacked Ikea lamp from the HAUM with 13 RGB LED included. It can be programmed over Wifi directly using `socat`, the Python wrapper or by... looking at the API.

All the code is available on the dedicated Github repo. Feel free to use it, clone it hack it, release it.

A french version of this documentation is also available here.

CHAPTER 1

Designing the Laumio

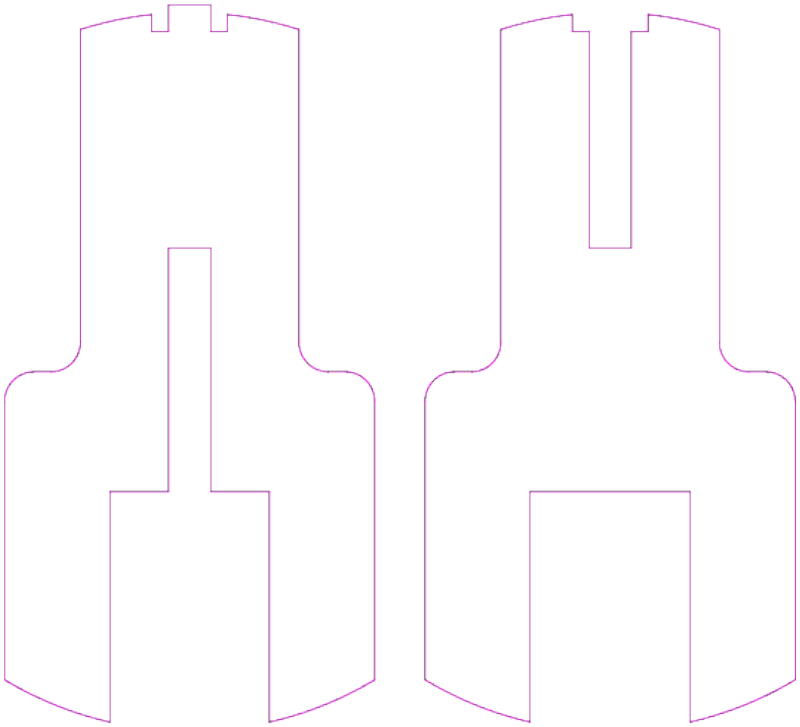
Laumio is based on a Ikea FADO lamp and a homemade system bearing 13 RGB LED.

The physical system itself is composed of the mechanical part supporting the lamps and the electronic board and system.

Mechanical Part

Tip: All the design files can be found on the Git repository.

The mechanical part is composed of a central structure designed to stand on the lightbulb socket. It's is machined using a in-house CNC mill and dry-assembled.



Two different propositions were made to hold the LED themselves.

The first one is based on foldable arms with LED glued at the ends. The main issues are both the fragility of the design and the effort needed to fit it in the glass.

The second one, still under study, is based on flexible plastic branches cruve from the to to the bottom of the structure.

Electronic system

The electronic system consist of a ESP Wemos d1 Mini and on 13 LED WS2812. To be able to reuse the Wemos in other projects, it was decided to create a shield to dock it on. The KiCAD design files for the shield itself can be downloaded here.

The code running on the ESP is given on the repo too, please follow the README to be able to compile and flash it.

Once flashed and started, the Laumio will look for the specified Wifi network (see `wifi-config.h`) and wait for UDP or HTTP instructions.

Booting

At startup, the Laumio blinks in purple and then starts a red animation : it's looking for the specified Wifi network and trying to connect.

If the connection is successful, the Laumio will show a rainbow animation, if not, it will light up in orange, indicating the *Access Point* mode was started. This mode will allow one to specify the correct information to connect to a Wifi network.

CHAPTER 2

Laumio's anatomy and HEX commands

The Laumio can be controlled using the bare-UDP API using simple HEX commands sent on the **port 6969**. It can alternatively receive commands from a REST API. Four different features can be set : the pixel, ring, column and the whole Laumio at once. Additionnaly, some animations can be triggered.

The leds are disposed in 4 branches of three LED plus one at the top. They are numbered starting from the bottom of one branch to its top, skipping the led at the center, going down on the other side and then from the bottom of one of the other two branches to the bottom of the other (with the top middle LED included).

The so-called rings correspond to a layer of LED. There's three of them, numbered from bottom to top.

As for the columns, they correspond to the four branches of LEDs, of course without the top LED.

Bare UDP

Pixel-wise control

The command is `0x00` and the data to be sent is first the pixel ID and the three byte of the (RGB) color:

```
0x00 PixelID R G B
```

Ring-wise control

The command is `0x01` and one must specify first the ring (`0x00`, `0x01` or `0x02`) and then the three bytes of the color:

```
0x01 RingID R G B
```

Column-wise control

The command is `0x02` and one must specify first the column (`0x00`, `0x01`, `0x02` or `0x03`) and then the three bytes of the color:

```
0x02 ColumnID R G B
```

Full span control

The command is `0xff` and the only thing to specify is the RGB color:

```
0xff R G B
```

Animations

This is still a work in progress but 2 animations can be triggered on the Laumio.

Color Wipe

The color wipe animation changes the color on the full Laumio one LED at a time. It takes two parameters, the 3 bytes of color and the delay between two LED:

```
0x0b R G B Delay
```

Rainbow

The rainbow animation takes no parameters, it's the one used when the Laumio manage to connect to a Wifi network:

```
0x0a
```

JSON API

The JSON API is still a work in progress and every UDP-supported call has not been translated yet.

Status

The status of the Laumio can be retrieved using the `/api/` endpoint:

```
GET http://<laumio's ip>/api/
```

The response is given in the form :

```
{"name":"laumio","version":"devel"}
```

Pixel-wise control

The Laumio can be controlled through simple POST requests, specifying both the `led` and `rgb` key:

```
{
  'led': PixelID,
  'rgb': [R, G, B]
}
```

Full span control

To control the full Laumio at once, just set the `led` parameter to 255 in the previous JSON snippet:

```
{
  'led': 255,
  'rgb': [R, G, B]
}
```

CHAPTER 3

Simple Bash Interface

The Laumio can be controlled simply using socat or any other well-designed network tool.

One can use the following boilerplate :

```
#!/bin/bash

# Consts
IP=              # Laumio's IP
ANIM_TIME=0.05
PAUSE_TIME=0.3

# Utils
fill() {
  echo -en "\xff\x$1\x$2\x$3"
}

led() {
  echo -en "\x00\x$1\x$2\x$3\x$4"
}

ring() {
  echo -en "\x01\x$1\x$2\x$3\x$4"
}

column() {
  echo -en "\x02\x$1\x$2\x$3\x$4"
}

# Program
(
fill 00 00 00
sleep $ANIM_TIME

while true; do

  # write your animation here...

done
) | socat - udp-sendto:$IP:6969
```

You can also send various commands in one request as in the following example:

In this command, the Laumio is entirely red colored; then the top LED is set to white color.

CHAPTER 4

Python Library

The class `Laumio` is built as a wrapper around the primitives exposed in the API page.

class Laumio

__init__(*ip*)
Constructor to control a Laumio responding on IP address *ip*.

wipeOut()
Shuts down all the LED

Actually just an alias for :py:method:‘fillColor‘.

fillColor(*r*, *g*, *b*)
Fill the whole Laumio with a color (*r*, *g*, *b*).

fillRing(*ringid*, *r*, *g*, *b*)
Fill a ring with a color. *ringid* can be 0~3 and *r*, *g*, *b* specify the color. The top LED is not addressed.

fillColumn(*colmunid*, *r*, *g*, *b*)
Fill a column with a color. *columnid* can be 0~4 and *r*, *g*, *b* specify the color. The top LED is not addressed.

setPixelColor(*pixel*, *r*, *g*, *b*)
Set a pixel *pixel* (0~12) to a given color (*r*, *g*, *b*).

colorWipe(*r*, *g*, *b*, *delay*)
Start a color wipe animation with color (*r*, *g*, *b*) and inter-LED delay *delay*.

rainbow()
Start a rainbow animation.

status()
Return the JSON **string** for the Laumio’s status

_send(*payload*)
Conventionnaly private, this method is used to transmit a raw `bytearray` payload to a Laumio. It can be used to trigger custom animations.

Symbols

C

F

L

R

S

W