**Uji Coba Integrasi Output Proyek Arduino Menggunakan Blynk Code Pada Web Browser dengan Wokwi dan Arduino di Visual Studio Code**

*Muhammad Fau Zan Sabani1*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*mfauzan18@student.ub.ac.id*](mailto:mfauzan18@student.ub.ac.id)

# Abstrak:

# Pada Abstrak ini membahas uji coba integrasi sistem Arduino dengan platform IoT Blynk untuk pemantauan dan pengendalian perangkat melalui web browser. Tujuan utama praktikum ini adalah mengimplementasikan sistem yang mampu mengirim data sensor (suhu dan kelembapan) dari simulasi Wokwi ke dasbor Blynk, sekaligus menerima perintah untuk mengontrol saklar lampu secara manual dari dasbor tersebut.

# Eksperimen dimulai dengan perancangan sistem di Visual Studio Code menggunakan kerangka kerja Arduino. Kode program dibuat untuk membaca data dari sensor DHT22 (suhu dan kelembapan) di dalam simulasi Wokwi. Selain mengirimkan data tersebut ke server Blynk, sistem juga dirancang untuk mendengarkan perintah dari sebuah widget saklar di dasbor Blynk yang berfungsi untuk menyalakan dan mematikan lampu secara manual.

# Hasil dari eksperimen ini menunjukkan bahwa integrasi dua arah antara simulasi Wokwi dan platform Blynk berhasil dilakukan. Data suhu dan kelembapan dapat ditampilkan secara akurat pada dasbor web, dan perintah dari saklar di dasbor Blynk berhasil mengontrol nyala lampu di simulasi Wokwi secara responsif. Uji coba ini membuktikan efektivitas Wokwi untuk memvalidasi sistem IoT yang melibatkan pemantauan sensor dan aktuator secara bersamaan sebelum implementasi fisik.

# Kata Kunci *Suhu, Kelembapan, IoT, Arduino, ESP32, DHT, LDR, Wokwi, Blynk, Laporan Praktikum*

# Abstract:

This This abstract discusses the trial integration of the Arduino system with the Blynk IoT platform for monitoring and controlling devices via a web browser. The main objective of this practicum is to implement a system capable of sending sensor data (temperature and humidity) from the Wokwi simulation to the Blynk dashboard, as well as receiving commands to manually control light switches from the dashboard.

The experiment started with designing the system in Visual Studio Code using the Arduino framework. Program code was created to read data from the DHT22 sensors (temperature and humidity) inside the Wokwi simulation. In addition to sending the data to the Blynk server, the system was also designed to listen to commands from a switch widget on the Blynk dashboard that serves to manually turn the lights on and off.

The results of this experiment show that the two-way integration between the Wokwi simulation and the Blynk platform is successful. Temperature and humidity data can be accurately displayed on the web dashboard, and commands from the switch on the Blynk dashboard successfully control the lights in the Wokwi simulation responsively. This trial proves the effectiveness of Wokwi for validating IoT systems that involve simultaneous monitoring of sensors and actuators before physical implementation.

Keywords *Temperature, Humidity, IoT, Arduino, ESP32, DHT, LDR, Wokwi, Blynk, Practical Report*

# 1. Pendahuluan

* 1. **Latar Belakang**

Pengawasan dan pengendalian perangkat di rumah atau industri secara konvensional seringkali memerlukan kehadiran fisik. Misalnya, untuk mengetahui kondisi suhu suatu ruangan atau untuk menyalakan dan mematikan lampu, seseorang harus berada di lokasi. Hal ini tidak hanya **tidak efisien dari segi waktu dan tenaga**, tetapi juga **menghilangkan kemampuan untuk merespons perubahan secara cepat**, terutama saat berada di luar lokasi.

Dengan kemajuan pesat di bidang **Internet of Things (IoT),** batasan fisik ini dapat diatasi. Teknologi IoT memungkinkan perangkat elektronik, sensor, dan aktuator untuk saling terhubung melalui internet. Hal ini membuka peluang untuk menciptakan sistem yang tidak hanya dapat **memantau kondisi lingkungan (seperti suhu dan kelembapan) dari jarak jauh,** tetapi juga **memberikan kendali penuh untuk mengoperasikan perangkat (seperti menyalakan lampu) melalui antarmuka digital** seperti dasbor web.

Oleh karena itu, praktikum yang mengintegrasikan simulasi perangkat dengan platform IoT menjadi langkah penting untuk memahami konsep ini secara praktis. Dalam proyek ini, **Blynk** digunakan sebagai platform cloud yang menyediakan antarmuka dasbor web interaktif. **Wokwi** berperan sebagai lingkungan simulasi di mana mikrokontroler membaca data sensor suhu dan kelembapan, sekaligus menerima perintah dari dasbor Blynk untuk mengendalikan saklar lampu. Kombinasi ini memungkinkan pengujian sistem kontrol dan monitoring secara menyeluruh tanpa memerlukan perangkat keras fisik.

* 1. **Tujuan Eksperimen**

# Tujuan dari eksperimen integrasi sistem Arduino dengan platform Blynk ini adalah sebagai berikut:

# Merancang dan mengimplementasikan sistem IoT dua arah yang mampu mengirimkan data sensor (suhu dan kelembapan) ke dasbor Blynk dan menerima perintah untuk mengendalikan saklar lampu secara real-time.

# Menguji integrasi antara simulasi Wokwi dan platform cloud Blynk untuk memvalidasi fungsionalitas pengiriman data (uplink) dan penerimaan perintah (downlink) tanpa memerlukan perangkat keras fisik.

# Memahami arsitektur sistem kontrol IoT secara menyeluruh, mulai dari pembacaan data sensor di perangkat, komunikasi dengan layanan cloud, hingga interaksi pengguna melalui antarmuka dasbor web.

# Mengembangkan kemampuan teknis dalam pemrograman mikrokontroler untuk berinteraksi dengan layanan IoT serta mengonfigurasi dasbor kontrol yang fungsional dan interaktif.

# Menyediakan dasar untuk pengembangan aplikasi Smart Home atau otomasi sederhana, yang membuktikan kelayakan konsep sebelum diimplementasikan dengan biaya dan perangkat keras yang sebenarnya.

# 2. Metodologi

* 1. **Alat dan Bahan**

Dalam perancangannya, untuk membuat simulasi menampilkan indikator suhu, kelembapan dan mengatur saklar pada IoT menggunakan dashboard Blynk dan Arduino dibutuhkan beberapa alat dan bahan, diantaranya:

* 1 buah mikrokontroller ESP32,
* 1 buah sensor DHT (untuk mendeteksi/mengukur suhu dan kelembapan),
* 1 buah LED,
* Platform Wokwi, yang digunakan untuk simulasi rangkaian dan pemrograman secara virtual.
* Software Arduino IDE/Extension PlatformIo (jika menggunakan Visual Studio Code).
  1. **Langkah Implementasi**

Berikut langkah-langkah implementasi simulasi lampu lalu lintas dengan menggunakan Mikrokontroler ESP32 di Wokwi/Arduino:

* Install Arduino IDE atau jika menggunakan VsCode install Extension PlatfromIO di bagian ekstensinya. Buka juga website wokwi untuk tampilan simulasi nya secara virtual (jangan lupa untuk install ekstensi Wokwi juga di VsCode).



* Buat proyek baru di Wokwi dengan memilih ESP32 sebagai mikrokontroler. Tambahkan juga komponen yang dibutuhkan seperti sensor DHT, dan juga layar LED. Smbungkan tiap pin sensor maupun LED ke tiap pin yang sesuai di mikrokontroller ESP32 menggunakan kabel jumper virtual. Seperti dibawah:

A computer screen shot of a circuit board

Description automatically generated

* Untuk pemogramannya menggunakan bahasa Arduino (C/C++) di Arduino IDE/PlatformIO (bisa juga langsung dimasukkan ke file ‘main.c’ di website Wokwi). Kodenya kita atur agar output dari pemograman ini nantinya dapat menampilkan hasil indikator suhu, tingkat kelembapan dan intensitas cahaya pada OLED Display. Seperti berikut kode pemogramannya:

#define BLYNK\_PRINT Serial

#define BLYNK\_TEMPLATE\_ID "TMPL6dkzfxdTx"

#define BLYNK\_TEMPLATE\_NAME "CobaImplement"

#define BLYNK\_AUTH\_TOKEN "fqIH2gCLk4\_9-z7LF7lt2jlyWcCVP5tU"

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <DHTesp.h> // Library untuk DHT

char auth[] = BLYNK\_AUTH\_TOKEN; // Auth Token

char ssid[] = "Wokwi-GUEST"; // Nama hotspot yang digunakan

char pass[] = ""; // Password hotspot yang digunakan

const int DHT\_PIN = 15;

int value0, value1, value2, value3, value6;

byte LED\_R = 26;

byte LED\_Y = 27;

byte LED\_G = 14;

byte LED\_B = 12;

DHTesp dht;

BlynkTimer timer;

// Fungsi untuk mengirimkan data sensor ke Blynk

void sendSensor() {

TempAndHumidity data = dht.getTempAndHumidity();

Serial.print("% Temperature: ");

Serial.print(data.temperature);

Serial.println("C ");

Serial.print("% Kelembaban: ");

Serial.print(data.humidity);

Serial.println("% ");

Blynk.virtualWrite(V0, data.temperature); // Kirim suhu ke Blynk

Blynk.virtualWrite(V1, data.humidity); // Kirim kelembaban ke Blynk

}

// Fungsi untuk menyalakan LED berdasarkan perintah dari Blynk

BLYNK\_WRITE(V2) {

int nilaiBacaIO = param.asInt();

digitalWrite(LED\_R, nilaiBacaIO);

Blynk.virtualWrite(V3, nilaiBacaIO);

}

void setup() {

Serial.begin(115200);

dht.setup(DHT\_PIN, DHTesp::DHT22);

pinMode(LED\_R, OUTPUT);

pinMode(LED\_Y, OUTPUT);

pinMode(LED\_G, OUTPUT);

pinMode(LED\_B, OUTPUT);

Blynk.begin(auth, ssid, pass); // Mulai Blynk

// Jika menggunakan Wokwi, gunakan mode simulasi WiFi

#ifdef WOKWI

Blynk.connectWiFi(ssid, pass);

Blynk.config(auth);

#endif

timer.setInterval(1000, sendSensor); // Kirim data setiap 1000ms

}

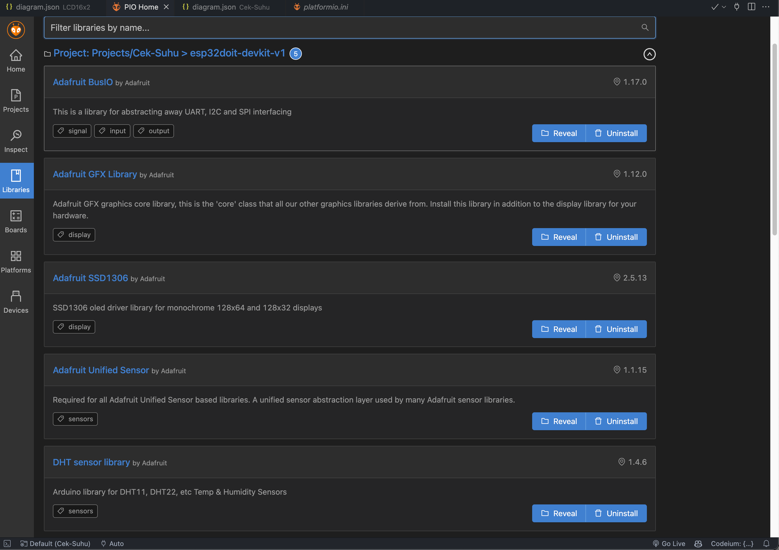
void loop() {

Blynk.run();

timer.run();

}

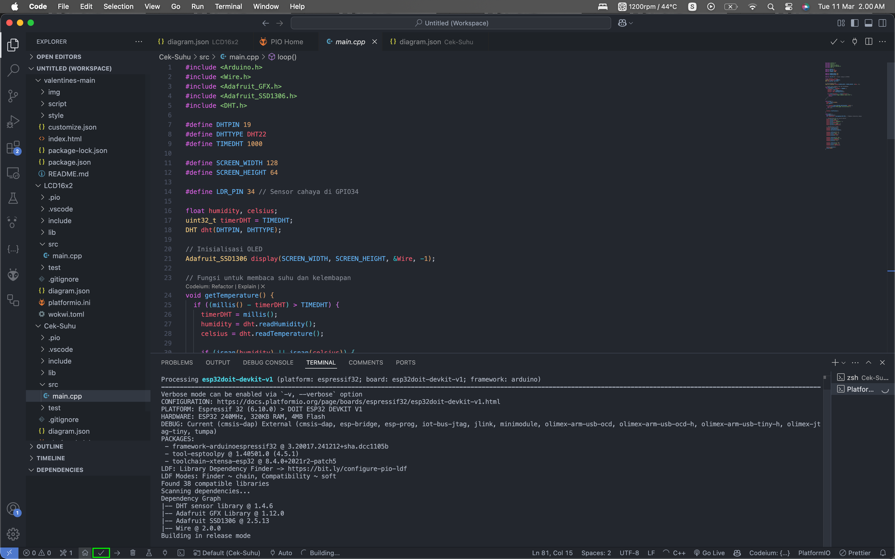
* Karena di dalam eksperimen kali ini menggunakan beberapa part yang memerlukan library untuk menjalankannya seperti library untuk sensor DHT maka dari itu perlu penginstalannya dari library yang sudah disediakan di Library Manager. Berikut library yang dibutukan pada eksperimen kali ini:



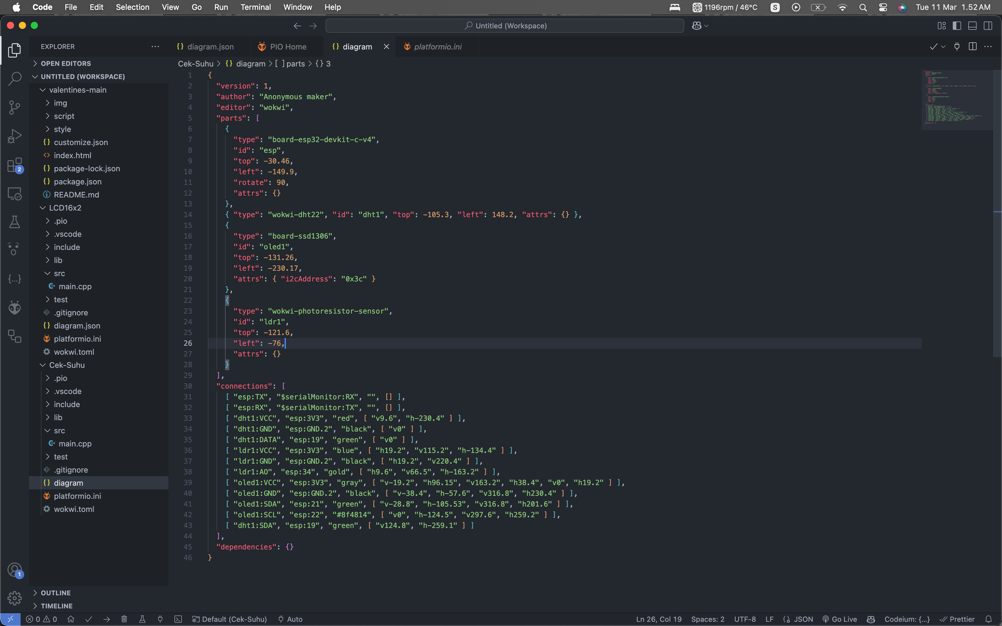
A screenshot of a computer

Description automatically generated

* Langkah selanjutnya jalankan semua kodenya agar bisa menampilkan outputnya.
* Jika di platform Wokwi file ‘main.c’ tidak bisa ter-compile/dijalankan maka kita bisa menguji apakah sukses atau tidak program kode tersebut di PlatformIO dengan menggunakan tombol centang dibawah.



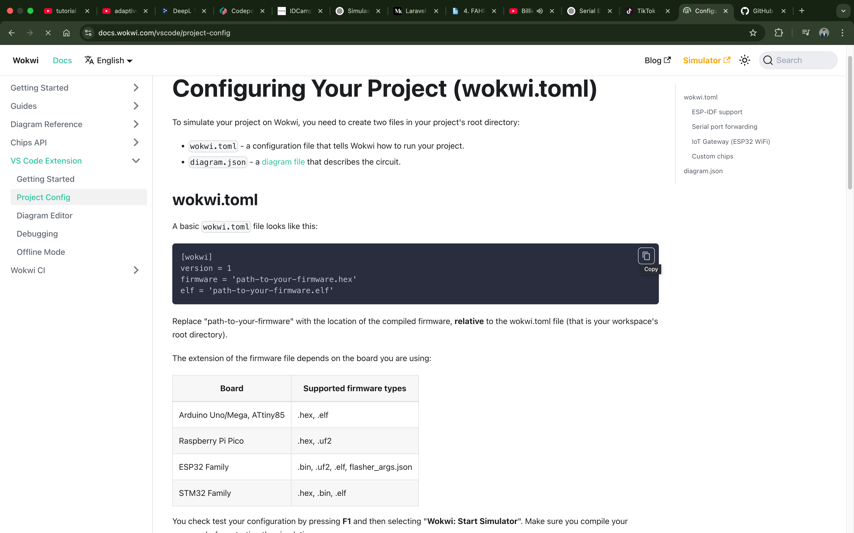
* Jika kode di file ‘main.cpp’ sukses di *Build* maka langkah selanjutnya untuk dapat melakukan virtualisasi tampilan sensor-sensor dan LED nya dengan membuat file bernama ‘diagram’ yang nanti diisi oleh kode dari file ‘diagram.json’ di projek Wokwi sebelumnya. Lalu kalau sudah, ubah nama file ‘diagram’ sbelumnya menjadi ‘diagram.json’ seperti di platform Wokwi sebelumnya.

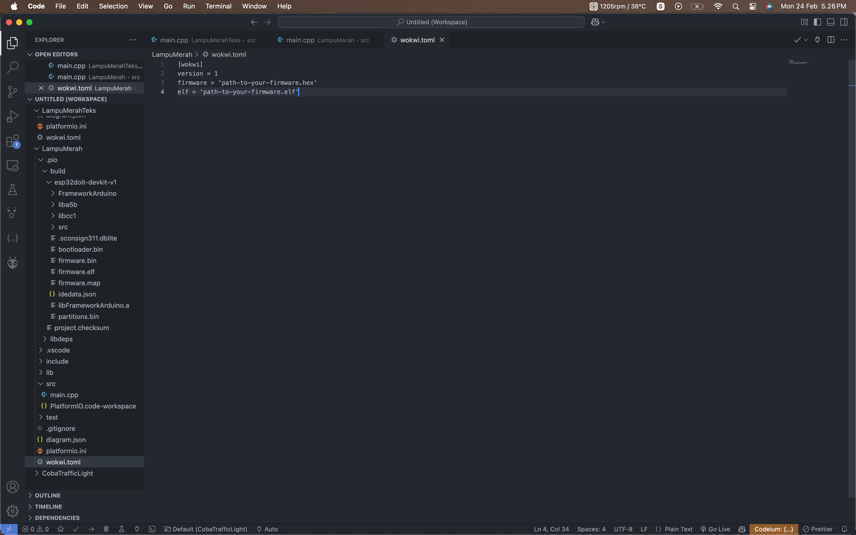


A computer screen shot of a computer

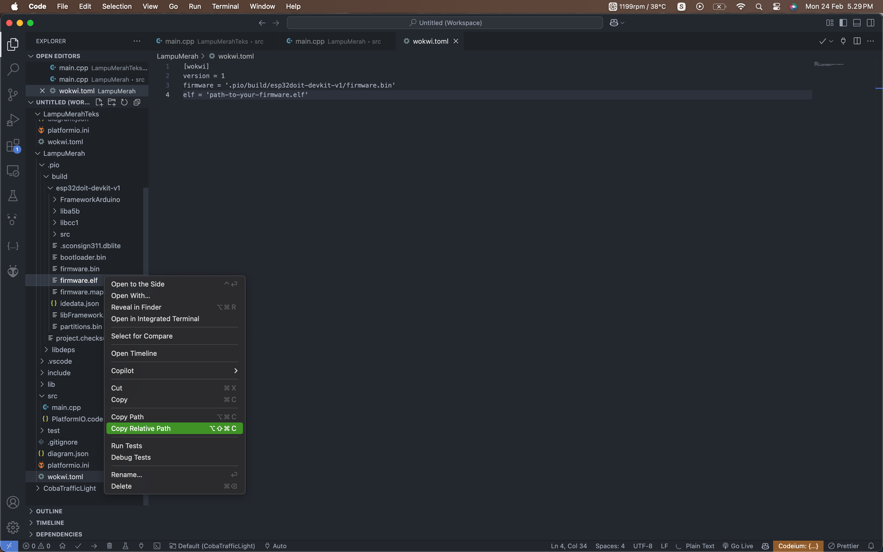
Description automatically generated

* Untuk dapat menjalankan simulasi dari file ‘diagram.json’ tersebut maka kita harus menambahkan file yang kita namakan ‘wokwi.toml’. Langkah berikutnya cari di web browser dengan keyword ‘Wokwi.toml’ dan salin kode konfigurasinya yang berasal dari halaman web dokumentasi wokwi tersebut ke file ‘wokwi.toml’ yang sudah kita buat.





* Langkah selanjutnya untuk dapat mengkonfigurasi file ‘wokwi.toml’ tersebut agar dapat membantu menjalankan file diagramnya maka kita perlu untuk mengganti path ‘firmware’ dan ‘elf’ path file yang berada di dalam rute folder ‘pio/build/esp32doit-devkit-v1’. Salin tiap file dengan format file yang bertuliskan ‘.hex’ dan ‘.elf’ dengan menggunakan klik kanan dan klik ‘Copy relative path’ lalu tempel ke path yang harus diisi di file ‘wokwi.toml’.



* Jalankan ulang file ‘diagram.json’ untuk dapat melihat hasil akhir simulasi yang sudah kita buat.
* Kemudian untuk mengintegrasikannya ke dalam platform blynk, maka kita perlu login atau mendaftarkan akun (jika belum memiliki akun) terlebih dahulu ke dalam website resmi dai platform blynk.
* Tambahkan template untuk device baru proyek kita. Kemudian tambahkan virtual pin dan sesuaikan dengan virtual pin yang sudah kita atur dalam kode di main.cpp, sesuaikan juga untuk nilai, nama, dan satuannya.

A screenshot of a computer

Description automatically generated

* Jika sudah mendapatkan token, ganti token yang ada di kode main untuk bisa mengintegrasikan hasil output dari wokwi ke dalam dashboard blynk.
* Langkah terakhir lakukan testing atau uji coba dengan menjalankan file diagram.json dan cek apakah data dari hasil output terintegrasikan ke dalam platform blynk.

A screenshot of a computer

Description automatically generated

# 3. Hasil dan Pembahasan

1. **Hasil Eksperimen**

Dari simulasi yang sudah dibuat, di dapatkan hasil yang sudah sesuai yang kita inginkan. Berikut hasil simulasinya:

* Tampilan Wokwi:

A computer screen shot of a computer

Description automatically generated

* Konfigurasi Virtual Pin:

A screenshot of a computer

Description automatically generated

* Hasil uji coba:

A screenshot of a computer

Description automatically generated

# 4. Lampiran

* Kode Program diagram.json:

{

"version": 1,

"author": "Anonymous maker",

"editor": "wokwi",

"parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -278.9, "left": 52.76, "attrs": {} },

{

"type": "wokwi-led",

"id": "led1",

"top": -306.4,

"left": -89.47,

"attrs": { "color": "red" }

},

{

"type": "wokwi-resistor",

"id": "r5",

"top": -274.74,

"left": -44.52,

"attrs": { "value": "1000" }

},

{

"type": "wokwi-dht22",

"id": "dht1",

"top": -260.42,

"left": 247.56,

"attrs": { "temperature": "58.7", "humidity": "77" }

}

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "led1:A", "r5:1", "red", [ "v0" ] ],

[ "r5:2", "esp:D26", "red", [ "v1.2", "h17.93", "v81.46" ] ],

[ "dht1:VCC", "esp:VIN", "red", [ "v87.6", "h-228.22", "v-54.65" ] ],

[ "dht1:GND", "esp:GND.1", "black", [ "v93.06", "h-109.48", "v-76.5" ] ],

[ "dht1:SDA", "esp:D15", "green", [ "v26.39", "h-81.44", "v-19.67" ] ],

[ "led1:C", "esp:GND.2", "black", [ "v0" ] ]

],

"dependencies": {}

}