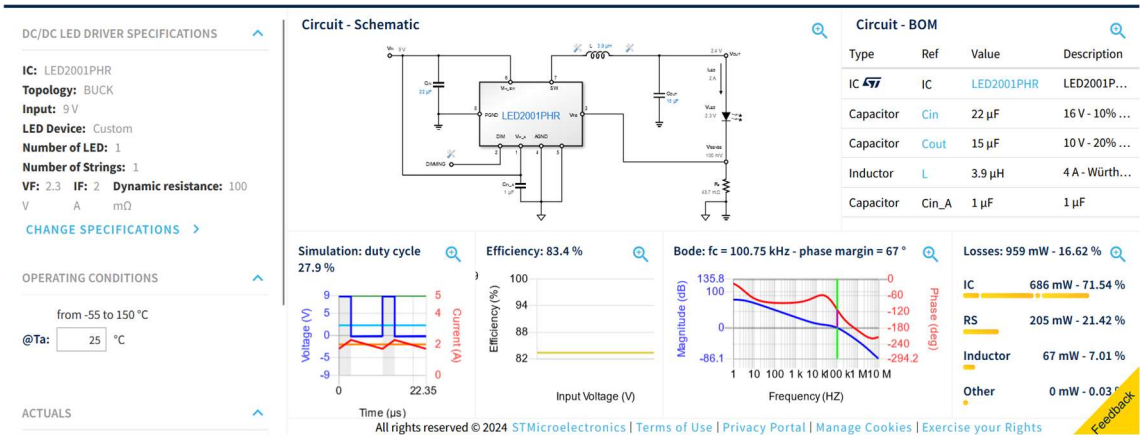
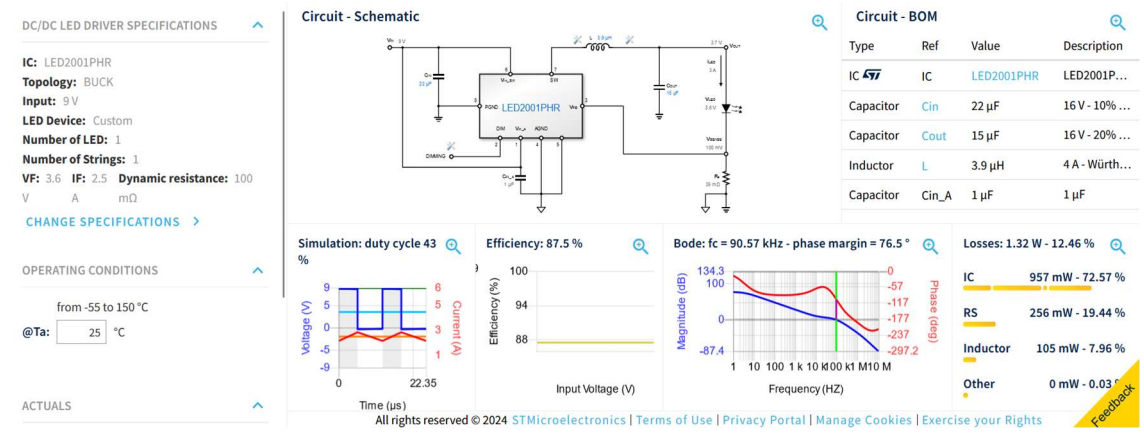
A.

# SIMULACIONES EN ST

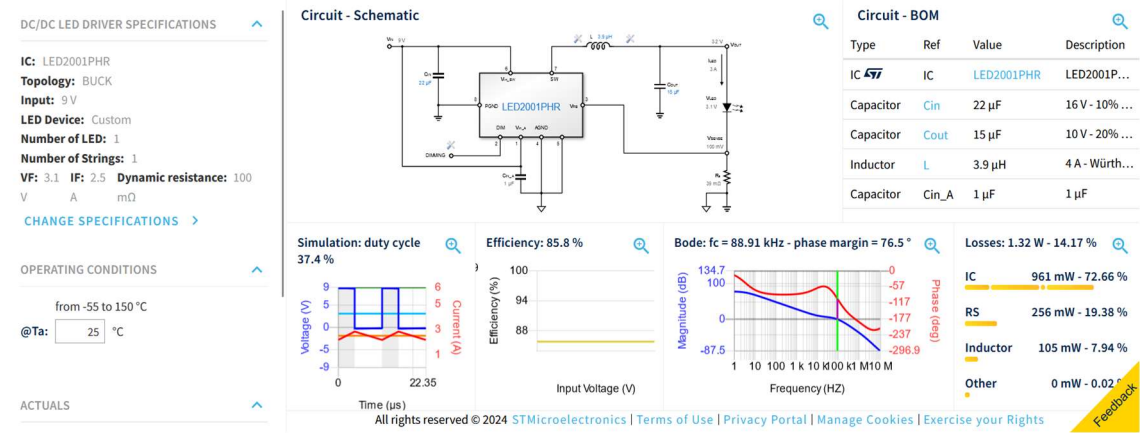
1. ROJO



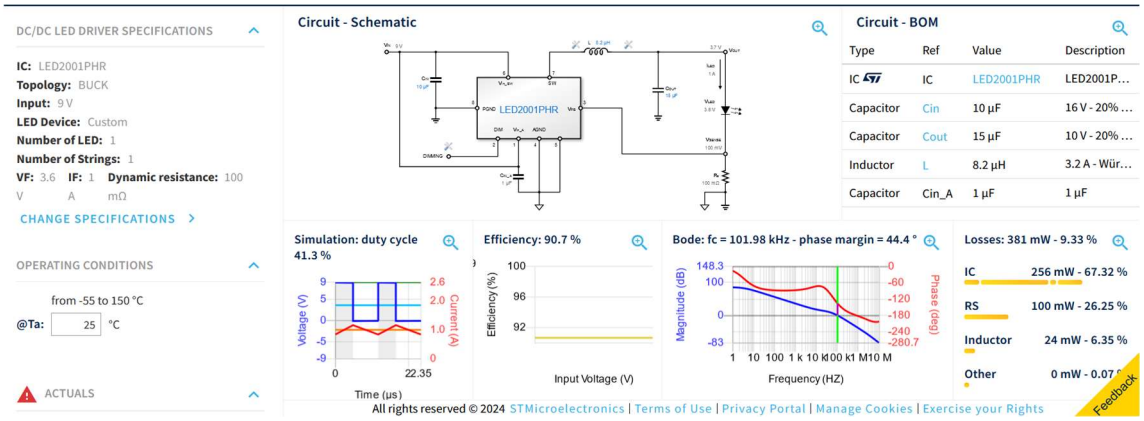

2. VERDE



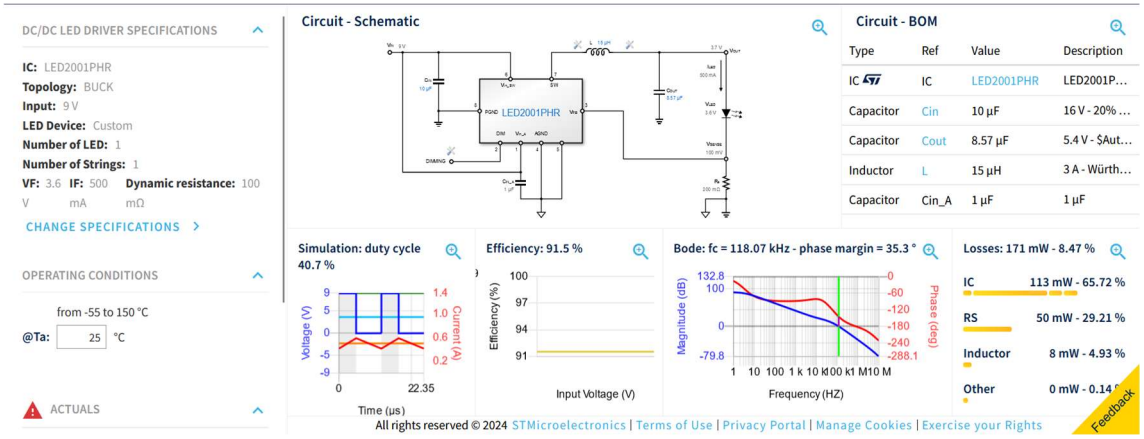

3. AZUL



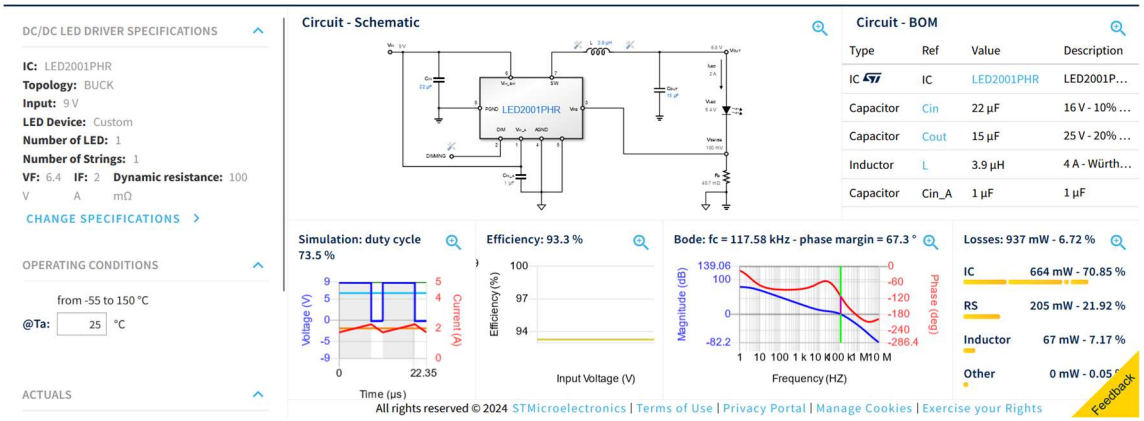
4. PC ÁMBAR



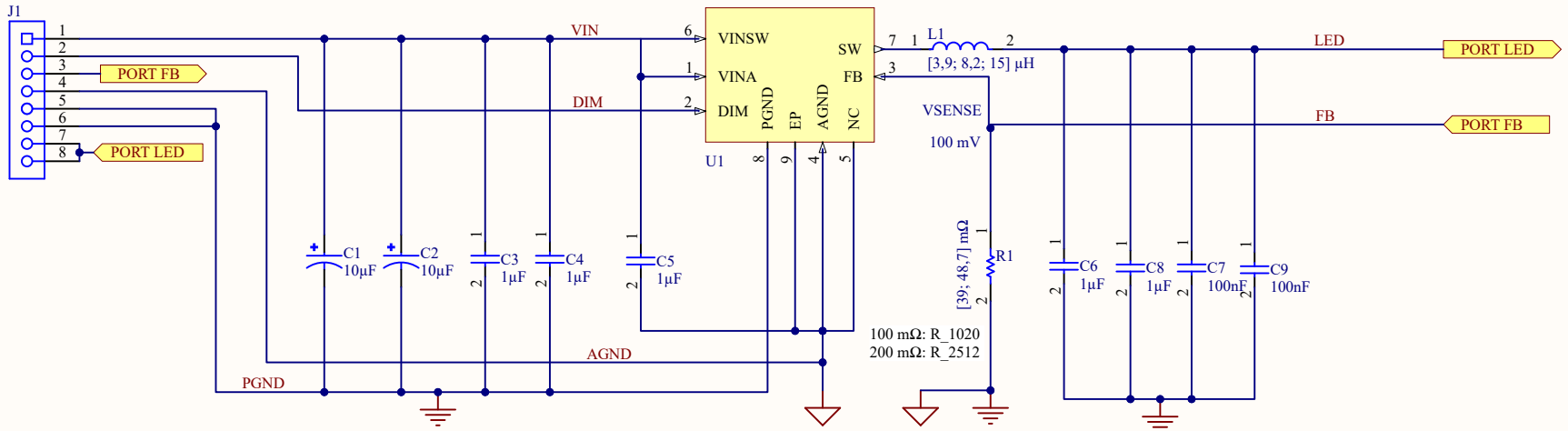

5. CIAN




6. PC LIMA



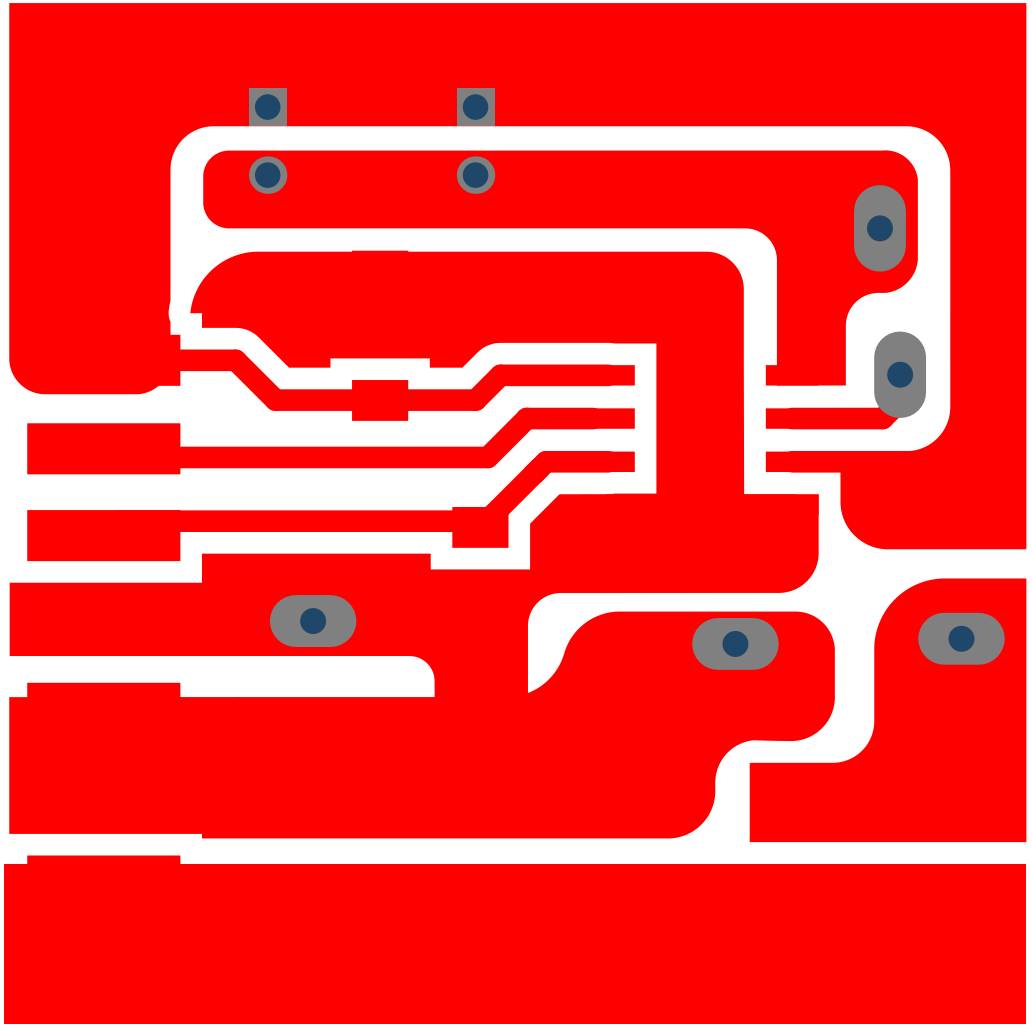
# B.1. Schematic driver

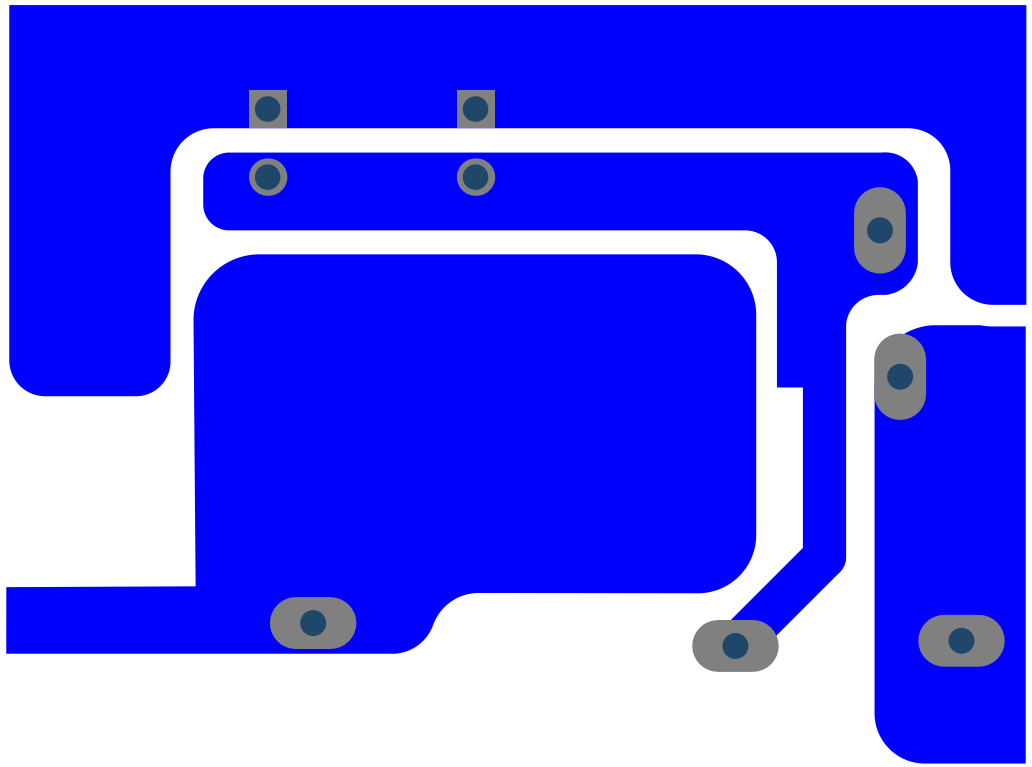



| | | |
|---|---|---|
| Title | | |
| Size A4 | Number | Revision |
| Date: 7/23/2024 | | Sheet of |
| File: C:\Users\..\ROJO SCH.SchDoc | | Drawn By: |

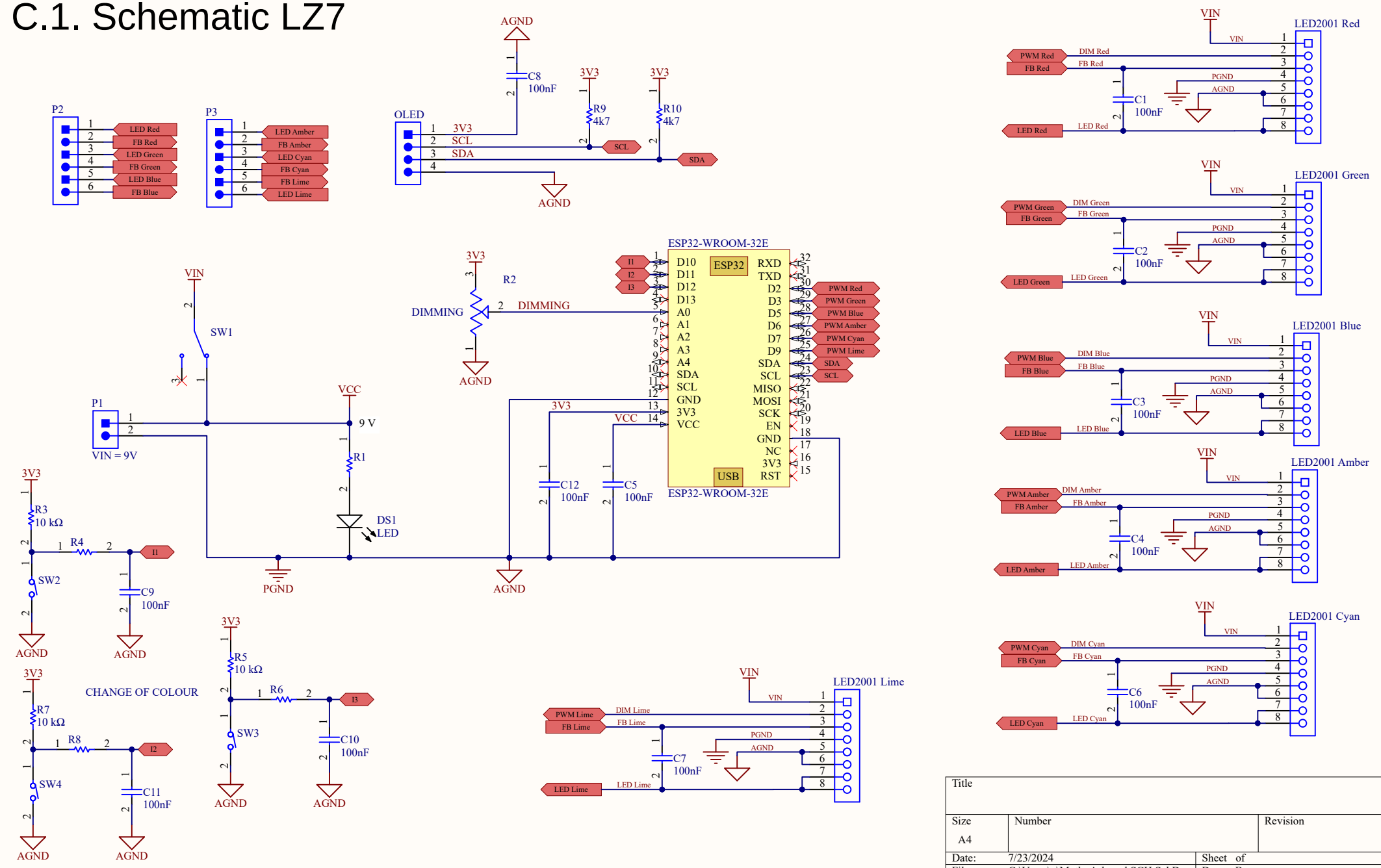
# B.2. PCB driver

Superficie aproximada de 3 mm x 3 mm



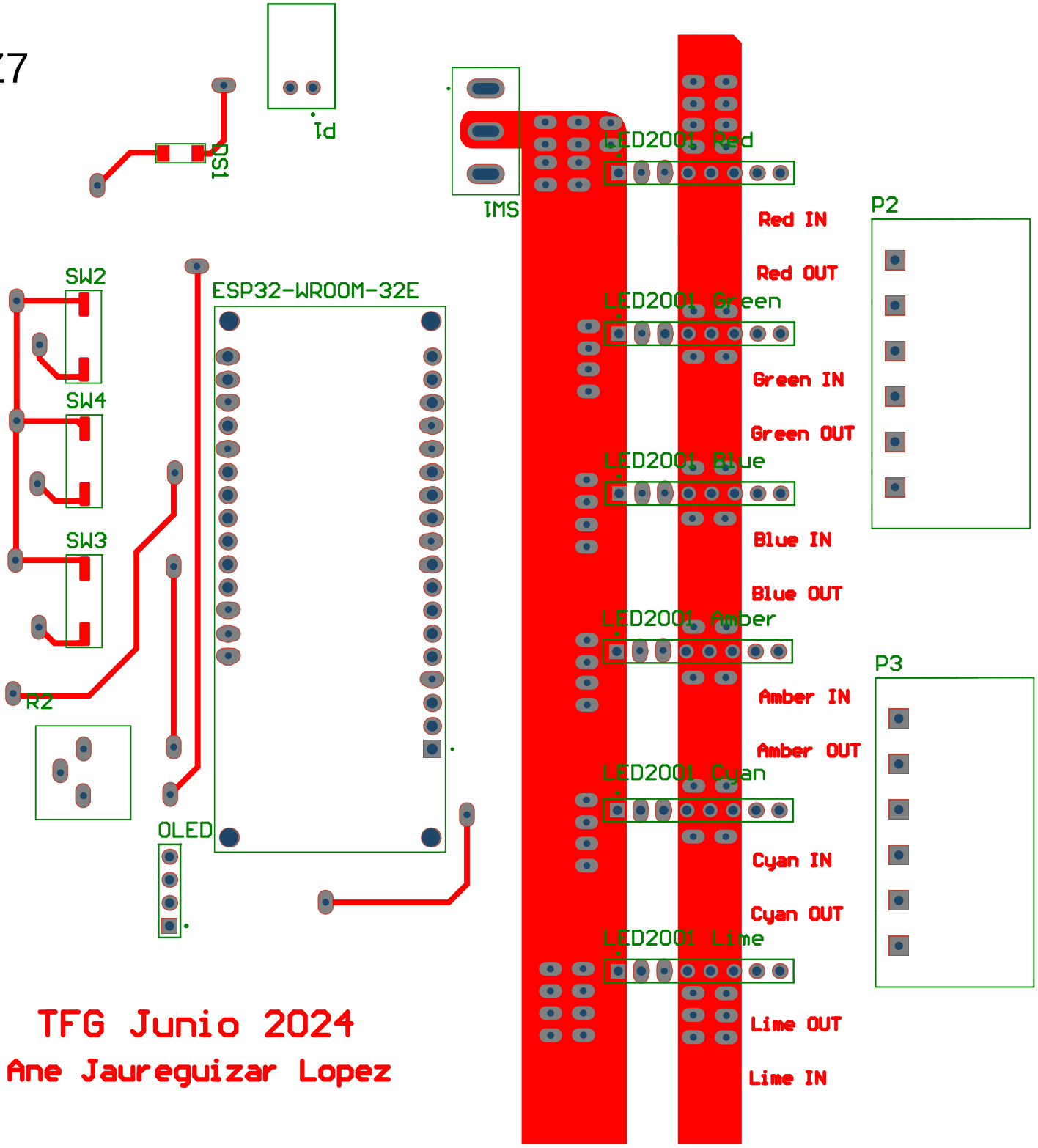


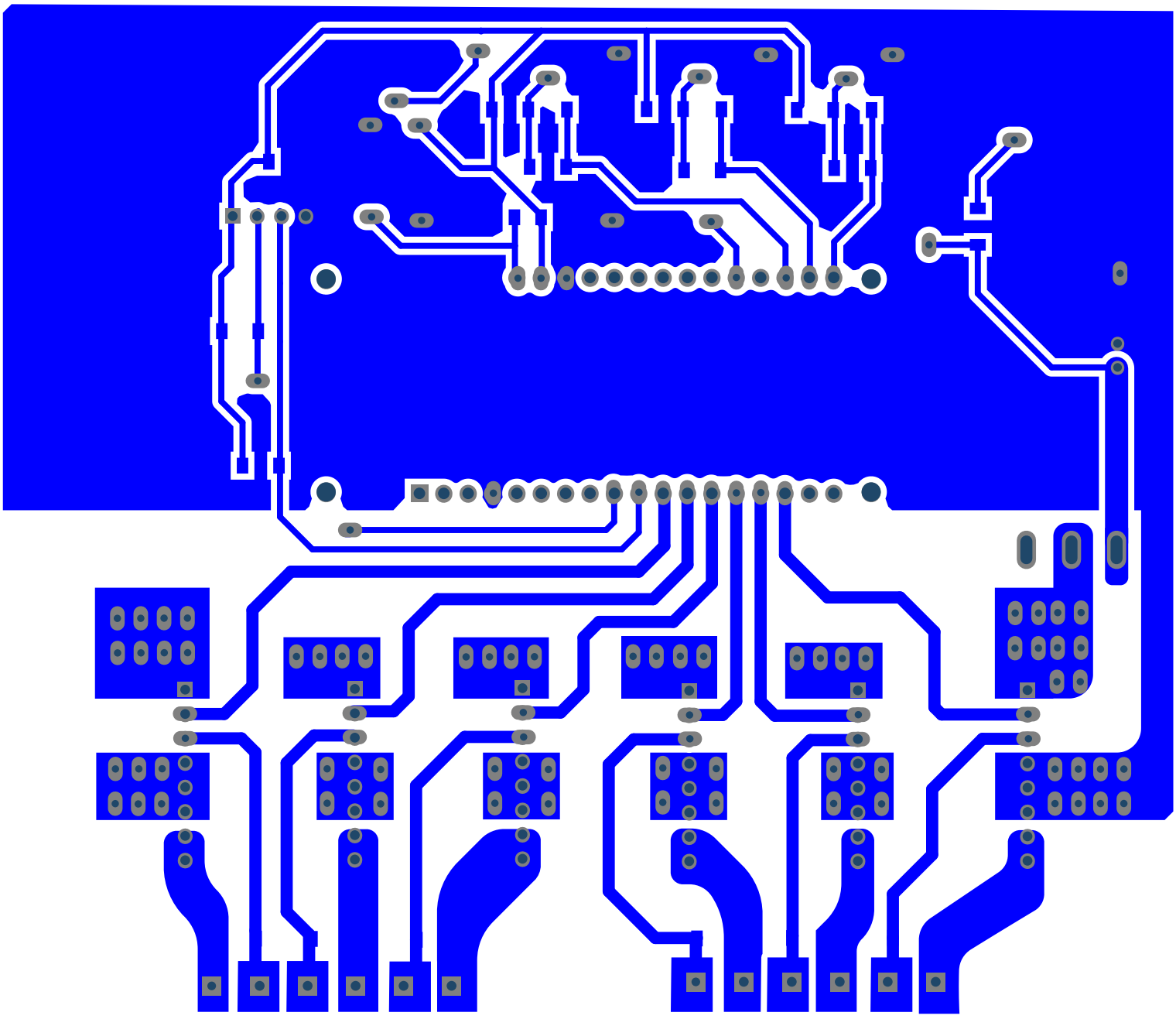
# C.1. Schematic LZ7




| | | |
|---|---|---|
| Title | | |
| Size A4 | Number | Revision |
| Date: 7/23/2024 | | Sheet of |
| File: C:\Users\..\Mother's board SCH.SchDoc | | Drawn By: |

# C.2. PCB LZ7


Id
ISI
IMS
SW2
SW4
SW3
R2
ESP32-WROOM-32E
OLED
LED2001 Red
Red IN
Red OUT
LED2001 Green
Green IN
Green OUT
LED2001 Blue
Blue IN
Blue OUT
LED2001 Amber
Amber IN
Amber OUT
LED2001 Cyan
Cyan IN
Cyan OUT
LED2001 Lime
Lime OUT
Lime IN
P2
P3
TFG Junio 2024
Ane Jaureguizar Lopez



# D. Software

```
// Librerías
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <time.h>
#include <Wire.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// OLED caracterización
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaración de la pantalla OLED
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// Constantes para la configuración MQTT
const char* wifi_ssid = "LUZ_NIVEL_2";
const char* wifi_password = "manuel_rico";
const char* mqtt_server_IP = "156.35.154.170";
const int mqtt_port = 8883;
const char* mqtt_user = "alumno";
const char* mqtt_password = "alumno";
const char* mqtt_clientID = "ESP32AJL";

// Topics del MQTT
const char* topic_suscribe1 = "tfg/ajl/manual";
```

```
const char* topic_suscribe2 = "tfg/ajl/manual/red";
const char* topic_suscribe3 = "tfg/ajl/manual/green";
const char* topic_suscribe4 = "tfg/ajl/manual/blue";
const char* topic_suscribe5 = "tfg/ajl/manual/amber";
const char* topic_suscribe6 = "tfg/ajl/manual/cyan";
const char* topic_suscribe7 = "tfg/ajl/manual/lime";
const char* topic_suscribe8 = "tfg/ajl/message";

// Constantes para la configuración del dimming PWM
const int freq = 5000;
const int resolution = 10;

// Pins de los LEDs
const int RED = 25;
const int GREEN = 26;
const int BLUE = 0;
const int AMBER = 14;
const int CYAN = 13;
const int LIME = 2;

// Constantes de envío de mensajes al MQTT
bool sending = true;
bool sending_btn = false;
bool iniciation = true;
String message = "";
String date = "";
String s_intensity = "";
```

```
// Constantes de llamada (call-back) al IoT MQTT
bool modomanualon = false;
int REDdim = 0;
int GREENdim = 0;
int BLUEdim = 0;
int AMBERdim = 0;
int CYANdim = 0;
int LIMEdim = 0;
bool REDon = false;
bool GREENon = false;
bool BLUEon = false;
bool AMBERon = false;
bool CYANon = false;
bool LIMEon = false;

// Constantes para el funcionamiento por Botones
int category = 0;
int menu = 0;
int submenu0 = 0;
int submenu1 = 0;
int subsubmenu0 = 0;
const int btn1 = 17; // UP
const int btn2 = 16; // DOWN
const int btn3 = 4; // ACCEPT
const int DIM = 36;
float intensity;
int percentage;
bool messageprinted = false;
```

```
bool switch_off = false;
bool dimming_available = false;

// Cronómetro
unsigned long previousMillis = 0;// Almacena el último tiempo en milisegundos
unsigned long currentMillis;
const long interval = 500;       // Intervalo de tiempo para el contador (500ms)
int counter = 0;

// Certificado de la Autoridad Certificadora (CA_Cert)
//------------------------------------------------------------------------------
static const char CA_cert[] PROGMEM = R"EOF(
-----BEGIN CERTIFICATE-----
MIIEETCCAvmgAwIBAgIUebjUqWHarDoBKLzH78XkunwNlrEwDQYJKoZIhvcNAQEL
BQAwgZcxCzAJBgNVBAYTAkVTMREwDwYDVQQIDAhBc3R1cmlhczEOMAwGA1UE
BwwF
R2lqb24xDzANBgNVBAoMBlVOSU9WSTEOMAwGA1UECwwFQ0UzSTIxIjAgBgNVB
AMM
GWVsZWN0cm9uaWNvLmVkdi51bmlvdmkuZXMxIDAeBgkqhkiG9w0BCQEWEWVs
b3Bl
emNAdW5pb3ZpLmVzMB4XDTIxMDYyMjIxNTgwOFoXDTMxMDYyMDIxNTgwOFowg
Zcx
CzAJBgNVBAYTAkVTMREwDwYDVQQIDAhBc3R1cmlhczEOMAwGA1UEBwwFR2lq
b24x
DzANBgNVBAoMBlVOSU9WSTEOMAwGA1UECwwFQ0UzSTIxIjAgBgNVBAMMGWVs
ZWN0
cm9uaWNvLmVkdi51bmlvdmkuZXMxIDAeBgkqhkiG9w0BCQEWEWVsb3BlemNAd
W5p
b3ZpLmVzMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA0VFz0NxAxerK
aSA50KRFLaGP/tMLBhyJdfpgtmFy61Yi+q5ppqmQBh9+VMxVtfKZ8RZlcteYMSRy
```

KHWHUyy/t0aMvaB4d/ftwX4X8xzAt6DayTmhp7cuqJU/Tdfje0iWI4J1yN7vDnKt

//jzBqloLnnuqljKBHc/lSJm+kAAYRPDoOdsSYtiHubgMIIEWp7p7OXRPZurHoF3

GnBwqTOE3+a9/RxtUTck1g+wv5b8CK7RWtmMtMXlp6eUdgk48rDAPv8r+zpxPQdc

xFYK6CnGa1Sv6hMERQ26GuopfbiDfEmSoV8b9qonIcjlQW5UvJGuLJsYJo+befM5

YfPF5VIB+QIDAQABo1MwUTAdBgNVHQ4EFgQULPRJpQp3iLG0y3WwzJhXz3g2WaIw

HwYDVR0jBBgwFoAULPRJpQp3iLG0y3WwzJhXz3g2WaIwDwYDVR0TAQH/BAUwAwEB

/zANBgkqhkiG9w0BAQsFAAOCAQEASPAq1oNcMyK/6DvGHCKlujrctBLOTx9kkdpE

YcLV7s0ySyIvaMadraK4tAJr/dFqFCIJb23ie229SJN3x9e4H52a7twHFlK/wZOA

B1Q/sZq2uTKS1Pw99gzQKYiycIqnRUnD/TACaGTr+6TEzOnm4Q3VuR75WWPE8bcl

APLFyPF5wviOy4E9TPRxYqAdfMSUFUm7oh1Fv92wFUgWC02ycfKtkMaa/oZoNchF

lyk81V8EnfhmZseDVRyRvGxlNYG9312uLQLzqhf8C/Czsco0pI0K7W6Sd1XQ1MkF

eEATGimuO2alKYek4upq3jZkSe3dJiPeu6J7fyhow6bI1WGcxQ==

-----END CERTIFICATE-----

)EOF";

WiFiClientSecure mqtt_cliente;

PubSubClient PubSubClient_cliente(mqtt_cliente);

// Configuración del WiFi

void setup_wifi() {

  delay(1000);

  Serial.println();

  Serial.print("Conectando a ");

  Serial.println(wifi_ssid);

  WiFi.begin(wifi_ssid, wifi_password);

  int attempt = 0;

```
  while (WiFi.status() != WL_CONNECTED && attempt < 20) {
    delay(500);
    Serial.print(".");

    attempt++;
  }


  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("");
    Serial.println("WiFi conectado");
    Serial.print("Dirección IP: ");
    Serial.println(WiFi.localIP());

  } else {
    Serial.println("");
    Serial.println("Fallo al conectar WiFi");
  }
}



// Función para ajustar la hora usando NTP
void ajusta_hora() {
  Serial.println("");
  Serial.println("");
  Serial.println("Esperando sincronización de hora mediante NTP: ");
```

```
  configTime(0, 0, "pool.ntp.org", "time.nist.gov"); // Configura servidores NTP

  time_t now = time(nullptr);
  while (now < 2 * 3600 * 2) {
    delay(500);
    Serial.print(".");

    now = time(nullptr);
  }

  struct tm timeinfo;
  gmtime_r(&now, &timeinfo);
  Serial.println("");
  Serial.print("La hora actual es: ");
  Serial.println(asctime(&timeinfo));
  date = asctime(&timeinfo);
}

// Configuración del IoT MQTT
void setup_mqtt() {
  // Configurar certificado CA para conexión segura SSL/TLS
  mqtt_cliente.setCACert(CA_cert);

  // Configurar servidor y puerto MQTT
  PubSubClient_cliente.setServer(mqtt_server_IP, mqtt_port);
}
```

```
// Función para la reconexión del MQTT
void reconnect() {
  while (!PubSubClient_cliente.connected()) {
    Serial.print("Intentando conexión MQTT...");

    if (PubSubClient_cliente.connect(mqtt_clientID, mqtt_user, mqtt_password)) {
      Serial.println("conectado");
      PubSubClient_cliente.subscribe(topic_suscribe1);
      PubSubClient_cliente.subscribe(topic_suscribe2);
      PubSubClient_cliente.subscribe(topic_suscribe3);
      PubSubClient_cliente.subscribe(topic_suscribe4);
      PubSubClient_cliente.subscribe(topic_suscribe5);
      PubSubClient_cliente.subscribe(topic_suscribe6);
      PubSubClient_cliente.subscribe(topic_suscribe7);
      PubSubClient_cliente.subscribe(topic_suscribe8);

    } else {
      Serial.print("falló, rc=");
      Serial.print(PubSubClient_cliente.state());
      Serial.println(" intentando de nuevo en 5 segundos");
      delay(5000);
    }
  }
}

// Función para la llamada (call-back) al MQTT
void callback_MQTT(const char* topic_suscribe, byte* payload, unsigned int
length) {
```

```
 char payloadString[length + 1];
 memcpy(payloadString, payload, length);
 payloadString[length] = '\0'; // Añadir el terminador nulo al final de la cadena

 //Serial.print("Mensaje recibido: ");
 //Serial.println(payloadString);


if (strstr(topic_suscribe,topic_suscribe1)) {

   if (strcmp(payloadString, "manualoff") == 0) {
    if (iniciation == true) {
     message = "LZ7 conectado a IoTMQTTPanel";
    } else {
     message = "Modo manual desactivado y LZ7 apagado";
     sending = true;
     dimming_available = false;
     Serial.println("Modo manual desactivado");
     ledcWrite(RED, 0);
     ledcWrite(GREEN, 0);
     ledcWrite(BLUE, 0);
     ledcWrite(AMBER, 0);
     ledcWrite(CYAN, 0);
     ledcWrite(LIME, 0);
     Serial.println("LZ7 apagado");
     modomanualon = false;
     }
```

```
} else if (strcmp(payloadString, "manualon") == 0) {
  iniciation = false;
  message = "Modo manual activado";
  sending = true;
  Serial.println("Modo manual activado");
  modomanualon = true;
  if (REDon == true) {
  ledcWrite(RED, map(REDdim, 0, 100, 0, 1023));
  Serial.print("Brillo ajustado a: ");
  Serial.print(REDdim);
  Serial.println("%");
  } if (GREENon == true) {
  ledcWrite(GREEN, map(GREENdim, 0, 100, 0, 1023));
  Serial.print("Brillo ajustado a: ");
  Serial.print(GREENdim);
  Serial.println("%");
  } if (BLUEon == true) {
  ledcWrite(BLUE, map(BLUEdim, 0, 100, 0, 1023));
  Serial.print("Brillo ajustado a: ");
  Serial.print(BLUEdim);
  Serial.println("%");
  } if (AMBERon == true) {
  ledcWrite(AMBER, map(AMBERdim, 0, 100, 0, 1023));
  Serial.print("Brillo ajustado a: ");
  Serial.print(AMBERdim);
  Serial.println("%");
  } if (CYANon == true) {
  ledcWrite(CYAN, map(CYANdim, 0, 100, 0, 1023));
```

```
      Serial.print("Brillo ajustado a: ");
      Serial.print(CYANdim);
      Serial.println("%");
      } if (LIMEon == true) {
      ledcWrite(LIME, map(LIMEdim, 0, 100, 0, 1023));
      Serial.print("Brillo ajustado a: ");
      Serial.print(LIMEdim);
      Serial.println("%");
      }
   }
}

if (strstr(topic_suscribe,topic_suscribe2)) {
  REDdim = atoi(payloadString);
  if (modomanualon == true) {
    if (dimming_available == true) {
      message = "Brillo ajustado en ROJO a: " + String(percentage) + "%";
      sending = true;
    } else {
      ledcWrite(RED, map(REDdim, 0, 100, 0, 1023));
      message = "Brillo ajustado en ROJO a: " + String(REDdim) + "%";
      sending = true;
      Serial.print("Brillo ajustado a: ");
      Serial.print(REDdim);
      Serial.println("%");
      REDon = true;
    }
  } else if (modomanualon == false) {
```

```
        if (iniciation == true) {
          message = "LZ7 conectado a IoTMQTTPanel";
        } else {
          Serial.println("Modo manual desactivado, si desea puede activarlo con el interruptor");
          message = "¡Error! Modo manual desactivado";
          sending = true;
          dimming_available = false;
          REDon = true;
        }
      }
    }

    if (strstr(topic_suscribe,topic_suscribe3)) {
      GREENdim = atoi(payloadString);
      if (modomanualon == true) {
        ledcWrite(GREEN, map(GREENdim, 0, 100, 0, 1023));
        message = "Brillo ajustado en VERDE a: " + String(GREENdim) + " %";
        sending = true;
        Serial.print("Brillo ajustado a: ");
        Serial.print(GREENdim);
        Serial.println("%");
        GREENon = true;
      } else if (modomanualon == false) {
        Serial.println("Modo manual desactivado, si desea puede activarlo con el interruptor");
        message = "¡Error! Modo manual desactivado";
        sending = true;
        GREENon = true;
```

```
    }
  }

  if (strstr(topic_suscribe,topic_suscribe4)) {
   BLUEdim = atoi(payloadString);
   if (modomanualon == true) {
      ledcWrite(BLUE, map(BLUEdim, 0, 100, 0, 1023));
      message = "Brillo ajustado en AZUL a: " + String(BLUEdim) + " %";
      sending = true;
      Serial.print("Brillo ajustado a: ");
      Serial.print(BLUEdim);
      Serial.println("%");
      BLUEon = true;
    } else if (modomanualon == false) {
      Serial.println("Modo manual desactivado, si desea puede activarlo con el
interruptor");
      message = "¡Error! Modo manual desactivado";
      sending = true;
      BLUEon = true;
    }
  }

  if (strstr(topic_suscribe,topic_suscribe5)) {
   AMBERdim = atoi(payloadString);
   if (modomanualon == true) {
      ledcWrite(AMBER, map(AMBERdim, 0, 100, 0, 1023));
      message = "Brillo ajustado en ÁMBAR a: " + String(AMBERdim) + " %";
      sending = true;
```

```
      Serial.print("Brillo ajustado a: ");
      Serial.print(AMBERdim);
      Serial.println("%");
      AMBERon = true;
    } else if (modomanualon == false) {
      Serial.println("Modo manual desactivado, si desea puede activarlo con el interruptor");
      message = "¡Error! Modo manual desactivado";
      sending = true;
      AMBERon = true;
    }
  }

  if (strstr(topic_suscribe,topic_suscribe6)) {
   CYANdim = atoi(payloadString);
   if (modomanualon == true) {
      ledcWrite(CYAN, map(CYANdim, 0, 100, 0, 1023));
      message = "Brillo ajustado en CIAN a: " + String(CYANdim) + " %";
      sending = true;
      Serial.print("Brillo ajustado a: ");
      Serial.print(CYANdim);
      Serial.println("%");
      CYANon = true;
    } else if (modomanualon == false) {
      Serial.println("Modo manual desactivado, si desea puede activarlo con el interruptor");
      message = "¡Error! Modo manual desactivado";
      sending = true;
      CYANon = true;
```

```
    }
  }

  if (strstr(topic_suscribe,topic_suscribe7)) {
   LIMEdim = atoi(payloadString);
    if (modomanualon == true) {
       ledcWrite(LIME, map(LIMEdim, 0, 100, 0, 1023));
       message = "Brillo ajustado en LIMA a: " + String(LIMEdim) + " %";
       sending = true;
       Serial.print("Brillo ajustado a: ");
       Serial.print(LIMEdim);
       Serial.println("%");
       LIMEon = true;
     } else if (modomanualon == false) {
       Serial.println("Modo manual desactivado, si desea puede activarlo con el
interruptor");
       message = "¡Error! Modo manual desactivado";
       sending = true;
       LIMEon = true;
     }
  }
}

// Función para enviar mensajes al MQTT
void enviar_MQTT() {
  if (iniciation == true) {
    message = "LZ7 conectado a IoTMQTTPanel";
    percentage = 0;
```

```
    s_intensity = String(percentage);
    String init = "manualoff";
    PubSubClient_cliente.publish(topic_suscribe1, init.c_str());
    PubSubClient_cliente.publish(topic_suscribe2, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe3, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe4, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe5, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe6, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe7, s_intensity.c_str());
  } else if (sending_btn == true) {
    PubSubClient_cliente.publish(topic_suscribe2, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe3, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe4, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe5, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe6, s_intensity.c_str());
    PubSubClient_cliente.publish(topic_suscribe7, s_intensity.c_str());
    sending_btn = false;
  }
   PubSubClient_cliente.publish(topic_suscribe8, message.c_str());
   sending = false;
}

void setup() {

  // LZ7 apagada al principio del programa
  ledcWrite(RED, 0);
  ledcWrite(GREEN, 0);
  ledcWrite(BLUE, 0);
```

```
ledcWrite(AMBER, 0);
ledcWrite(CYAN, 0);
ledcWrite(LIME, 0);

Serial.begin(115200);
setup_wifi();
setup_mqtt();
ajusta_hora();

if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D for 128x64
Serial.println(F("SSD1306 allocation failed"));
for(;;);
}
delay(2000);

display.clearDisplay();
display.setTextSize(2);

PubSubClient_cliente.setCallback(callback_MQTT);

pinMode(btn1, INPUT);
pinMode(btn2, INPUT);
pinMode(btn3, INPUT);
pinMode(RED, OUTPUT);
pinMode(GREEN, OUTPUT);
pinMode(BLUE, OUTPUT);
pinMode(AMBER, OUTPUT);
pinMode(CYAN, OUTPUT);
```

```
pinMode(LIME, OUTPUT);
analogReadResolution(10);
ledcAttach(RED,freq,resolution);
ledcAttach(GREEN,freq,resolution);
ledcAttach(BLUE,freq,resolution);
ledcAttach(AMBER,freq,resolution);
ledcAttach(CYAN,freq,resolution);
ledcAttach(LIME,freq,resolution);
//Serial.begin(9600); // ¿Necesario?
delay (500);

ledcWrite(RED, 1023);
delay(5);
ledcWrite(RED, 0);

ledcWrite(GREEN, 1023);
delay(5);
ledcWrite(GREEN, 0);

ledcWrite(BLUE, 1023);
delay(5);
ledcWrite(BLUE, 0);

ledcWrite(AMBER, 1023);
delay(5);
ledcWrite(AMBER, 0);

ledcWrite(CYAN, 1023);
```

```
  delay(5);
  ledcWrite(CYAN, 0);

  ledcWrite(LIME, 1023);
  delay(5);
  ledcWrite(LIME, 0);
}

void loop() {
  if (!WiFi.isConnected()) {
    Serial.println("WiFi desconectado. Reintentando conexión...");
    setup_wifi(); // Intenta reconectar WiFi
  }
  if (!PubSubClient_cliente.connected()) {
    reconnect(); // Intentar reconectar MQTT si no está conectado
  }
  PubSubClient_cliente.loop();

  if (sending==true) {
   enviar_MQTT();
   //PubSubClient_cliente.loop();
  }
  menudisplay();
  modeselector ();
  if (switch_off == true) {
   switching_off();
  }
  if (dimming_available == true) {
```

```
    dimming ();
  }
}

void menudisplay () {
 display.fillRect(0, 0, SCREEN_WIDTH, SCREEN_HEIGHT, SSD1306_BLACK);
 display.setTextColor(WHITE);
 if(category == 0 && messageprinted == false) {
  switch(menu) {
   case 0: display.setCursor(0, 24); display.println("MANUAL"); display.display();
messageprinted = true; break;
   case 1: display.setCursor(0, 24); display.println("AUTOMATIC");
display.display(); messageprinted = true; break;
  }
  delay(50);
 }
 if(category == 1) {
  if (menu == 0 && messageprinted == false) {
   switch(submenu0) {
      case 0: display.setCursor(0, 24); display.println("RED"); display.display();
messageprinted = true; break;
      case 1: display.setCursor(0, 24); display.println("GREEN"); display.display();
messageprinted = true; break;
      case 2: display.setCursor(0, 24); display.println("BLUE"); display.display();
messageprinted = true; break;
      case 3: display.setCursor(0, 24); display.println("AMBER"); display.display();
messageprinted = true; break;
      case 4: display.setCursor(0, 24); display.println("CYAN"); display.display();
messageprinted = true; break;
      case 5: display.setCursor(0, 24); display.println("LIME"); display.display();
messageprinted = true; break;
```

```
      case 6: display.setCursor(0, 24); display.println("Swith all off?");
display.display(); messageprinted = true; break;

      case 7: display.setCursor(0, 24); display.println("Back"); display.display();
messageprinted = true; break;

    }

  }

  if(menu == 1 && messageprinted == false) {

    switch(submenu1) {

      case 0: display.setCursor(0, 24); display.println("ON"); display.display();
messageprinted = true; break;

      case 1: display.setCursor(0, 24); display.println("Back"); display.display();
messageprinted = true; break;

    }

  }

  delay(50);

 }

 if (category == 2) {

  if(menu == 0 && messageprinted == false) {

    switch(subsubmenu0) {

     case 0: display.setCursor(0, 12); display.println("DIMMING");
display.setCursor(0, 36); display.print("Available"); display.display();
dimming_available = true; currentMillis = millis(); messageprinted = true; break;

     case 1: display.setCursor(0, 24); display.println("Back"); display.display();
sending = true; sending_btn = true; message = "Brillo ajustado en ROJO a: " +
String(percentage) + "%"; previousMillis = 0; counter = 0; messageprinted = true;
break;

    }

  }

  delay(50);

 }

}
```

```
void modeselector () {
  int lectura1 = digitalRead(btn1);
  int lectura2 = digitalRead(btn2);
  int lectura3 = digitalRead(btn3);

  if(lectura1 == LOW) { // Asumiendo que el botón está presionado cuando la lectura es baja
    delay(50);
    display.clearDisplay();
    messageprinted = false;
    if(category == 0) {
      if(menu == 0) {
        menu = 1;
      } else {
        menu--;
      }
    }
    if(category == 1) {
      if(menu == 0) {
        if (submenu0 == 0) {
          submenu0 = 7;
        } else {
          submenu0--;
        }
      }
      if(menu == 1) {
        if (submenu1 == 0) {
```

```
      submenu1 = 1;
     } else {
      submenu1--;
     }
    }
   }
   if (category == 2) {
    if (menu == 0) {
     if(subsubmenu0 == 0) {
      subsubmenu0 = 1;
     } else {
      subsubmenu0--;
     }
    }
   }
   // Esperar a que el botón se suelte para evitar rebotes
   while(digitalRead(btn1) == LOW);
   delay(50); // Pequeña demora para debouncing
  }
  if(lectura2 == LOW) { // Asumiendo que el botón está presionado cuando la
lectura es baja
   delay(50);
   display.clearDisplay();
   messageprinted = false;
   if(category == 0) {
    if(menu == 1) {
     menu = 0;
    } else {
```

```
    menu++;
  }
 }
 delay(10);
 if(category == 1) {
   if(menu == 0) {
     if (submenu0 == 7) {
       submenu0 = 0;
     } else {
       submenu0++;
     }
   }
   if(menu == 1) {
     if (submenu1 == 1) {
       submenu1 = 0;
     } else {
       submenu1++;
     }
   }
 }
 delay(10);
 if (category == 2) {
   if (menu == 0) {
     if(subsubmenu0 == 1) {
       subsubmenu0 = 0;
     } else {
       subsubmenu0++;
     }
```

```
    }
   }
   delay(10);
   // Esperar a que el botón se suelte para evitar rebotes
   while(digitalRead(btn2) == LOW);
   delay(50); // Pequeña demora para debouncing
  }
  if(lectura3 == LOW) { // Asumiendo que el botón está presionado cuando la lectura es baja
   delay(50);
   //display.clearDisplay();
   display.fillRect(0, 0, SCREEN_WIDTH, SCREEN_HEIGHT, SSD1306_WHITE); // Fondo blanco
   display.setTextColor(SSD1306_BLACK);
   //delay(1000);
   if (category == 1 && submenu0 == 7) {
    submenu0 = 0;
    category--;
    messageprinted = false;
   }
   else if (category == 1 && submenu0 == 6) {
    switch_off = true;
    messageprinted = true;
   }
   else if (category == 1 && submenu1 == 1) {
    submenu1 = 0;
    category--;
    messageprinted = false;
   }
```

```
else if (category == 2 && subsubmenu0 == 0) {
  category = 2;
  subsubmenu0 = 0;
  messageprinted = true;
}
else if (category == 2 && subsubmenu0 == 1) {
  subsubmenu0 = 0;
  dimming_available = false;
  category--;
  messageprinted = false;
}
else if (category == 1 && menu == 0) {
  category++;
  messageprinted = false;
}
else if (category == 1 && submenu1 == 0) {
  category = 1;
  messageprinted = true;
}
else if (category == 0 && menu == 1) {
  switch_off = true;
  category++;
  messageprinted = false;
}
else if (category == 0 && menu == 0) {
  category++;
  messageprinted = false;
}
```

```
    // Esperar a que el botón se suelte para evitar rebotes
    while(digitalRead(btn3) == LOW);
    delay(50); // Pequeña demora para debouncing
  }
}

void switching_off() {
  switch_off = false;
  sending = true;
  sending_btn = true;
  percentage = 0;
  s_intensity = String(percentage);
  message = "Brillo ajustado en LZ7 a: " + String(percentage) + "%";
  ledcWrite(RED, 0);
  ledcWrite(GREEN, 0);
  ledcWrite(BLUE, 0);
  ledcWrite(AMBER, 0);
  ledcWrite(CYAN, 0);
  ledcWrite(LIME, 0);
  delay(20);
}

void dimming() {
  float DIMMING = analogRead(DIM);
  intensity = DIMMING*1024/1024;
  percentage =  round(map(intensity, 0, 1023, 0, 100));
  s_intensity = String(percentage);
  if (submenu0 == 0) {
```

```
  ledcWrite(RED, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en ROJO a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 } else if (submenu0 == 1) {
  ledcWrite(GREEN, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en VERDE a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 } else if (submenu0 == 2) {
```

```
  ledcWrite(BLUE, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en AZUL a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 } else if (submenu0 == 3) {
  ledcWrite(AMBER, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en ÁMBAR a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 } else if (submenu0 == 4) {
```

```
  ledcWrite(CYAN, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en CIAN a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 } else if (submenu0 == 5) {
  ledcWrite(LIME, intensity);
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;  // Actualiza el último tiempo
    // Incrementa el contador
    counter++;
  } else {
    previousMillis = 0;
    counter = 0;
    sending = true;
    sending_btn = true;
    message = "Brillo ajustado en LIMA a: " + String(percentage) + "%";
    currentMillis = millis();
  }
 }
```

```
  delay (50);
}
```

# E. Planificación del proyecto

## Proyecto: Luminaria LZ7-04M2PD

Área de Tecnología Electrónica
Ane Jaureguizar López

Inicio del proyecto: 20/4/2024

Semana para mostrar: 1

| TAREA | ASIGNADO A | PROGRESO | INICIO | FIN |
|---|---|---|---|---|
| **FASE 1: Estado del arte y recopilación de información** | | | | |
| Investigación inicial para el estado del arte | Ane Jaureguizar López | 100% | 20-4-24 | 22-4-24 |
| Redacción del estado del arte | Ane Jaureguizar López | 80% | 22-4-24 | 29-4-24 |
| Tutorial para el manejo de Altium Designer | Ane Jaureguizar López | 100% | 29-4-24 | 14-5-24 |
| Recopilación de información sobre luminaria LZ7-04M2PD y sobre driver LED2001 | Ane Jaureguizar López | 100% | 6-5-24 | 11-5-24 |
| Cálculo teórico de componentes a través de la plataforma de eDesign | Ane Jaureguizar López | 100% | 12-5-24 | 14-5-24 |
| **FASE 2: *Hardware*** | | | | |
| Diseño del esquemático y PCB driver LED2001 | Ane Jaureguizar López | 100% | 15-5-24 | 26-5-24 |
| Acopio de componentes PCB driver LED2001 | Ane Jaureguizar López | 100% | 27-5-24 | 28-5-24 |
| Fabricación de PCBs driver LED2001 | Ane Jaureguizar López | 100% | 26-5-24 | 7-6-24 |
| Pruebas de funcionamiento PCBs driver LED2001 y corrección de errores | Ane Jaureguizar López | 100% | 7-6-24 | 10-6-24 |
| Diseño del esquemático y PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 13-6-24 | 17-6-24 |
| Acopio de componentes PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 21-6-24 |
| Fabricación de PCBs general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 27-6-24 |
| Pruebas de funcionamiento PCB general de la luminaria LZ7-04M2PD y corrección de errores | Ane Jaureguizar López | 100% | 27-6-24 | 28-6-24 |
| **FASE 3: *Software*** | | | | |
| Tutorial para el manejo de entorno Arduino IDE | Ane Jaureguizar López | 100% | 30-6-24 | 2-7-24 |
| Programación de menú pantalla OLED y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 3-7-24 | 5-7-24 |
| Tutorial para el manejo de entorno IoT MQTT | Ane Jaureguizar López | 100% | 6-7-24 | 8-7-24 |
| Programación de menú en entorno IoT MQTT y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 9-7-24 | 11-7-24 |
| Programación de menú que habilite tanto el control a través de pantalla OLED como mediante IoT MQTT | Ane Jaureguizar López | 100% | 12-7-24 | 15-7-24 |
| **FASE 4: Memoria y anexos** | | | | |
| Preinscripción | Ane Jaureguizar López | 70% | 15-7-24 | 15-7-24 |
| Redacción de la memoria | Ane Jaureguizar López | 100% | 13-7-24 | 22-7-24 |
| Desarrollo de presupuesto | Ane Jaureguizar López | 10% | 20-7-24 | 21-7-24 |
| Pago de defensa TFG | Ane Jaureguizar López | 100% | 17-7-24 | 17-7-24 |

# Proyecto: Luminaria LZ7-04M2PD

Área de Tecnología Electrónica
Ane Jaureguizar López

Inicio del proyecto: 20/4/2024

Semana para mostrar: 6


| TAREA | ASIGNADO A | PROGRESO | INICIO | FIN |
|---|---|---|---|---|
| **FASE 1: Estado del arte y recopilación de información** | | | | |
| Investigación inicial para el estado del arte | Ane Jaureguizar López | 100% | 20-4-24 | 22-4-24 |
| Redacción del estado del arte | Ane Jaureguizar López | 80% | 22-4-24 | 29-4-24 |
| Tutorial para el manejo de Altium Designer | Ane Jaureguizar López | 100% | 29-4-24 | 14-5-24 |
| Recopilación de información sobre luminaria LZ7-04M2PD y sobre driver LED2001 | Ane Jaureguizar López | 100% | 6-5-24 | 11-5-24 |
| Cálculo teórico de componentes a través de la plataforma de eDesign | Ane Jaureguizar López | 100% | 12-5-24 | 14-5-24 |
| **FASE 2: *Hardware*** | | | | |
| Diseño del esquemático y PCB driver LED2001 | Ane Jaureguizar López | 100% | 15-5-24 | 26-5-24 |
| Acopio de componentes PCB driver LED2001 | Ane Jaureguizar López | 100% | 27-5-24 | 28-5-24 |
| Fabricación de PCBs driver LED2001 | Ane Jaureguizar López | 100% | 26-5-24 | 7-6-24 |
| Pruebas de funcionamiento PCBs driver LED2001 y corrección de errores | Ane Jaureguizar López | 100% | 7-6-24 | 10-6-24 |
| Diseño del esquemático y PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 13-6-24 | 17-6-24 |
| Acopio de componentes PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 21-6-24 |
| Fabricación de PCBs general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 27-6-24 |
| Pruebas de funcionamiento PCB general de la luminaria LZ7-04M2PD y corrección de errores | Ane Jaureguizar López | 100% | 27-6-24 | 28-6-24 |
| **FASE 3: *Software*** | | | | |
| Tutorial para el manejo de entorno Arduino IDE | Ane Jaureguizar López | 100% | 30-6-24 | 2-7-24 |
| Programación de menú pantalla OLED y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 3-7-24 | 5-7-24 |
| Tutorial para el manejo de entorno IoT MQTT | Ane Jaureguizar López | 100% | 6-7-24 | 8-7-24 |
| Programación de menú en entorno IoT MQTT y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 9-7-24 | 11-7-24 |
| Programación de menú que habilite tanto el control a través de pantalla OLED como mediante IoT MQTT | Ane Jaureguizar López | 100% | 12-7-24 | 15-7-24 |
| **FASE 4: Memoria y anexos** | | | | |
| Preinscripción | Ane Jaureguizar López | 70% | 15-7-24 | 15-7-24 |
| Redacción de la memoria | Ane Jaureguizar López | 100% | 13-7-24 | 22-7-24 |
| Desarrollo de presupuesto | Ane Jaureguizar López | 10% | 20-7-24 | 21-7-24 |
| Pago de defensa TFG | Ane Jaureguizar López | 100% | 17-7-24 | 17-7-24 |

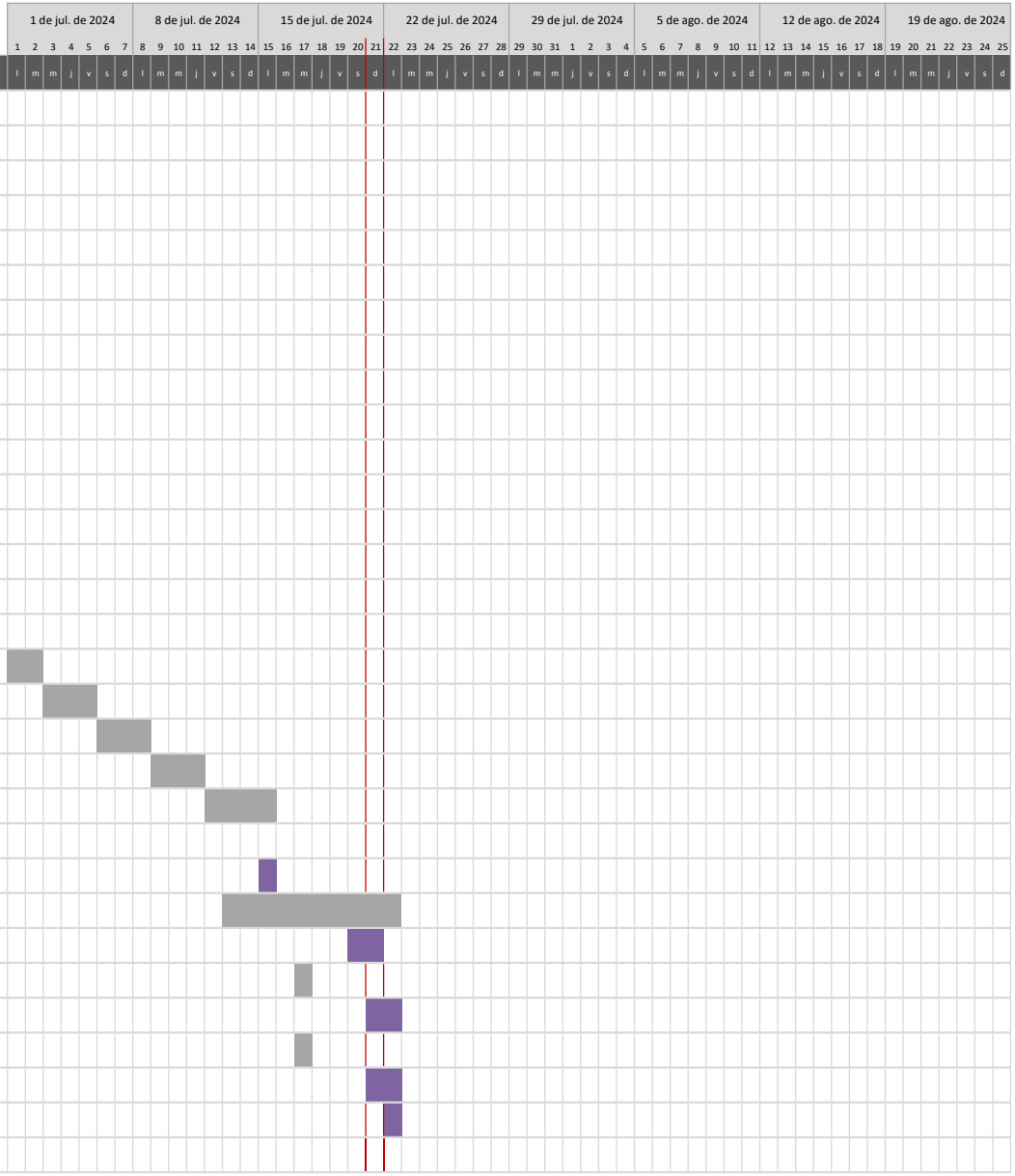
# Proyecto: Luminaria LZ7-04M2PD

Área de Tecnología Electrónica
Ane Jaureguizar López

Inicio del proyecto: 20/4/2024

Semana para mostrar: 12

| TAREA | ASIGNADO A | PROGRESO | INICIO | FIN |
|---|---|---|---|---|
| **FASE 1: Estado del arte y recopilación de información** | | | | |
| Investigación inicial para el estado del arte | Ane Jaureguizar López | 100% | 20-4-24 | 22-4-24 |
| Redacción del estado del arte | Ane Jaureguizar López | 80% | 22-4-24 | 29-4-24 |
| Tutorial para el manejo de Altium Designer | Ane Jaureguizar López | 100% | 29-4-24 | 14-5-24 |
| Recopilación de información sobre luminaria LZ7-04M2PD y sobre driver LED2001 | Ane Jaureguizar López | 100% | 6-5-24 | 11-5-24 |
| Cálculo teórico de componentes a través de la plataforma de eDesign | Ane Jaureguizar López | 100% | 12-5-24 | 14-5-24 |
| **FASE 2: *Hardware*** | | | | |
| Diseño del esquemático y PCB driver LED2001 | Ane Jaureguizar López | 100% | 15-5-24 | 26-5-24 |
| Acopio de componentes PCB driver LED2001 | Ane Jaureguizar López | 100% | 27-5-24 | 28-5-24 |
| Fabricación de PCBs driver LED2001 | Ane Jaureguizar López | 100% | 26-5-24 | 7-6-24 |
| Pruebas de funcionamiento PCBs driver LED2001 y corrección de errores | Ane Jaureguizar López | 100% | 7-6-24 | 10-6-24 |
| Diseño del esquemático y PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 13-6-24 | 17-6-24 |
| Acopio de componentes PCB general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 21-6-24 |
| Fabricación de PCBs general de la luminaria LZ7-04M2PD | Ane Jaureguizar López | 100% | 20-6-24 | 27-6-24 |
| Pruebas de funcionamiento PCB general de la luminaria LZ7-04M2PD y corrección de errores | Ane Jaureguizar López | 100% | 27-6-24 | 28-6-24 |
| **FASE 3: *Software*** | | | | |
| Tutorial para el manejo de entorno Arduino IDE | Ane Jaureguizar López | 100% | 30-6-24 | 2-7-24 |
| Programación de menú pantalla OLED y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 3-7-24 | 5-7-24 |
| Tutorial para el manejo de entorno IoT MQTT | Ane Jaureguizar López | 100% | 6-7-24 | 8-7-24 |
| Programación de menú en entorno IoT MQTT y sus correspondientes pruebas | Ane Jaureguizar López | 100% | 9-7-24 | 11-7-24 |
| Programación de menú que habilite tanto el control a través de pantalla OLED como mediante IoT MQTT | Ane Jaureguizar López | 100% | 12-7-24 | 15-7-24 |
| **FASE 4: Memoria y anexos** | | | | |
| Preinscripción | Ane Jaureguizar López | 70% | 15-7-24 | 15-7-24 |
| Redacción de la memoria | Ane Jaureguizar López | 100% | 13-7-24 | 22-7-24 |
| Desarrollo de presupuesto | Ane Jaureguizar López | 10% | 20-7-24 | 21-7-24 |
| Pago de defensa TFG | Ane Jaureguizar López | 100% | 17-7-24 | 17-7-24 |
| Compilación de anexos | Ane Jaureguizar López | 10% | 21-7-24 | 22-7-24 |
| Entrega del borrador | Ane Jaureguizar López | 100% | 17-7-24 | 17-7-24 |
| Corrección de errores | Ane Jaureguizar López | 0% | 21-7-24 | 22-7-24 |
| Entrega del final de la memoria, anexos y carta de pago | Ane Jaureguizar López | 0% | 22-7-24 | 22-7-24 |



# F. Presupuesto

**Ane Jaureguizar López**
71731841Q
Escuela Politécnica de Ingeniería de Gijón
33203 Oviedo, Asturias, España
Telf. 625374665

## CLIENTE

**Universidad de Oviedo**
Plaza de Riego (Palacio Quieros), 4 - 2º Planta
33003 Oviedo, Asturias, España

## FACTURA

**Nº de factura** **F20241**
Fecha factura 28/05/2024

| Conceptos | Cant. | Precio uni. | Imp. | Total |
|---|---|---|---|---|
| Luminaria LZ7-04M2PD<br>OSRAM ref. LZ7-04M2PD | 1,00 | 20,39 € | 21% | 24,67 € |
| Resistencia 50mΩ<br>Ref. 6836215 | 1,00 | 3,85 € | 21% | 4,66 € |
| Resistencias 39mΩ<br>Ref. 2172669 | 1,00 | 3,25 € | 21% | 3,93 € |
| Resistencias 100mΩ<br>Ref. 572631 - ERJB1CFR10U | 1,00 | 0,66 € | 21% | 0,80 € |
| Resistencias 200mΩ<br>Ref. 7216211AEC-Q200 | 1,00 | 0,77 € | 21% | 0,93 € |
| Bobinas 3,9µH<br>Ref. SDE0604A-3R9M | 10,00 | 0,382 € | 21% | 4,62 € |
| Bobinas 8,2µH<br>Ref. SDE0604A-8R2M | 10,00 | 0,221 € | 21% | 2,67 € |
| Bobinas 15µH<br>Ref. SDE0604A-150M | 1,00 | 0,382 € | 21% | 0,46 € |
| Condensadores electrolíticos 10µF<br>Ref. ECA1HM100 | 20,00 | 0,168 € | 21% | 4,07 € |
| Condensadores 1µF<br>Ref. GRM319R71C105KAA3D | 50,00 | 0,14 € | 21% | 8,47 € |
| Condensadores 100nF<br>Ref. VJ1206Y104KXAMT | 50,00 | 0,14 € | 21% | 8,47 € |
| Resistencias ref. 100Ω<br>Ref. RC1206FR-07100RL | 5,00 | 0,09 € | 21% | 0,54 € |
| Resistencias 10kΩ<br>Ref. RCC120610K0FKEA | 5,00 | 0,27 € | 21% | 1,63 € |
| Resistencias 4k7Ω<br>Ref. BSMD4725025 | 5,00 | 0,83 € | 21% | 5,02 € |

| Conceptos | Cant. | Precio uni. | Imp. | Total |
|---|---|---|---|---|
| Resistencias 820Ω<br>Ref. WAL WR12X8200FTL | 5,00 | 0,03 € | 21% | 0,18 € |
| Pantalla OLED<br>AZDelivery 1,3 Pulgadas OLED Display I2C SSH1106 Chip | 1,00 | 9,99 € | 21% | 12,09 € |
| Pulsadores<br>Ref. 179-TS026643BK100LCR | 5,00 | 0,09 € | 21% | 0,54 € |
| Potenciómetro<br>Ref. 3386F-1-502TLF | 10,00 | 1,82 € | 21% | 22,02 € |
| Microprocesador ESP32-WROOM-32E<br>FireBeetle 2 ESP32-E (N16R2) IoT Microcontroller (16M Flash, 2M PSRAM, Supports Wi-Fi & Bluetooth) | 1,00 | 11,84 € | 21% | 14,33 € |
| Fuente de alimentación<br>XP Power 9V CC, 6.7A, 60W | 1,00 | 57,06 € | 0% | 57,06 € |

| | |
|---|---|
| Base Imponible | 156,32 € |
| IVA 0% | 0,00 € |
| IVA 21% | 20,84 € |
| **Total** | **177,16 €** |