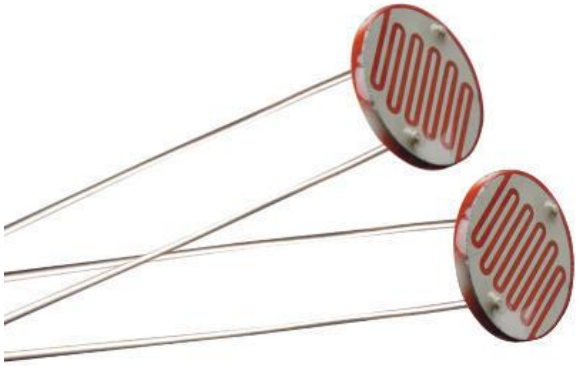
Instrumento Musical

Grupo 9

Leonardo Moniz Sodré Lopes Teixeira
Vinícius dos Santos Tavares

Materiais Utilizados



Primeiros Testes

```
int L1 = 10;
int L2 = 8;
//Sensor de luminosidade LDR
int lr1 = A2;
int lr2 = A1;
//potenciometro
int pot = A4;
//armazenamento de notas
int a[100][4];
int aux = 0;
//botão
int but = 5;
//Som
int caixa1 = 6;
int caixa2 = 9;
void setup() {
  pinMode(L1, OUTPUT);
  pinMode(L2, OUTPUT);
  pinMode(lr1, INPUT);
  pinMode(lr2, INPUT);
  pinMode(pot, INPUT);
  pinMode(caixa1, OUTPUT);
  pinMode(caixa2, OUTPUT);
  Serial.begin(9600);
}
```

```
// variaveis auxiliares
int luz1, luz2, p, estado = 0;
unsigned int ton;
int estadoluz1 = 0, estadoluz2 = 0;
unsigned int tNota1, tNota2;
unsigned int tPausa =0;
int estadoP = 0;
void loop() {
  p = analogRead(pot);
  p = map(p, 0, 1010, 0, 8);
  //  Serial.print("pot = ");
  //  Serial.println(p);
  luz1 = analogRead(lr1);
  //    Serial.print("SEM ATT luz 1
  //    Serial.println(luz1);
  if (luz1 > 960) {
    ton = 44;
    //    digitalWrite(L1, HIGH);
    tone(caixa1, ton * pow(2, p));
    estadoluz1 = 1;
  if (butL()) {
    tNota1 = millis();
    if(estadoP==0){
      estadoP=1;
      tPausa = millis() - tPausa;
      }
  }
  //      Serial.println(ton*pow(2,p));
} else if (estadoluz1 == 1) {
  //    digitalWrite(L1, LOW);
  noTone(caixa1);
  estadoluz1 = 0;
  if (butL()) {
    tNota1 = millis() - tNota1;
    gravar(caixa1,44* pow(2, p),tNota1,tPausa);
    if(estadoP ==1){
      estadoP=0;
      tPausa = millis();
      }
  }
}


luz2 = analogRead(lr2);
//    Serial.print("SEM ATT luz 2 =");
//    Serial.println(luz2);
if (luz2 > 900) {
  ton = 62;
  //    digitalWrite(L2, HIGH);
  tone(caixa2, ton * pow(2, p));
  //      Serial.print("2=");
  //      Serial.println(ton*pow(2,p));
  estadoluz2 = 1;
  if (butL()) {
    tNota2 = millis();
    if(estadoP==0){
      estadoP=1;
      tPausa = millis() - tPausa;
      }
  }
} else if (estadoluz2 == 1) {
  //    digitalWrite(L2, LOW);
  noTone(caixa2);
  estadoluz2 = 0;
  if (butL()) {
    tNota2 = millis() - tNota1;
    gravar(caixa2,62* pow(2, p),tNota2,tPausa);
    if(estadoP ==1){
      estadoP=0;
      tPausa = millis();
      }
  }
 }
}
// maquina de estados do butão
bool butL() {
  if (estado == 0 && !digitalRead(but)) {
    delay(200);
    estado = 1;
  }
  if (estado == 1 && digitalRead(but)) {
    estado = 2;
    return true;
  }


  if (estado == 2 && !digitalRead(but)) {
    delay(200);
    estado = 3;
  }
  if (estado == 3 && digitalRead(but)) {
    estado = 0;
    return false;
  }
}

// tentativa de implemenação do gravar
void gravar(int porta, unsigned int fre, unsigned int timerNota, unsigned int timerP ) {
  a[aux][0] = porta;
  a[aux][1] = fre;
  a[aux][2] = timerNota;
  a[aux][3] = timerP;
  aux++;
}
```

Código Publisher

```
#include <WiFi.h>
#include <PubSubClient.h>
//pinos sensores
int lr1 = 34;
int lr2 = 32;
int pot = 33;
int a[100][4];
int aux = 0;

//WiFi
const char* SSID = "Leonardo";// SSID / nome da rede WiFi que deseja se conectar
const char* PASSWORD = "12345678";   // Senha da rede WiFi que deseja se conectar
WiFiClient wifiClient;

//MQTT Server
const char* BROKER_MQTT = "test.mosquitto.org"; //URL do broker MQTT que se deseja utilizar
int BROKER_PORT = 1883;                      // Porta do Broker MQTT

#define ID_MQTT  "ppsadiohnk"//Informe um ID unico e seu. Caso sejam usados IDs repetidos a ultima conexão irá sobrepor a anterior.
#define TOPIC_PUBLISH "pp"    //Informe um Tópico único. Caso sejam usados tópicos em duplicidade, o último irá eliminar o anterior.
PubSubClient MQTT(wifiClient);// Instancia o Cliente MQTT passando o objeto espClient

//Declaração das Funções
void mantemConexoes();  //Garante que as conexoes com WiFi e MQTT Broker se mantenham ativas
void conectaWiFi();     //Faz conexão com WiFi
void conectaMQTT();     //Faz conexão com Broker MQTT
void enviaPacote();     //
```

```
 void setup() {
   pinMode(lr1, INPUT);
   pinMode(lr2, INPUT);
   pinMode(pot, INPUT);
   conectaWiFi();
   MQTT.setServer(BROKER_MQTT, BROKER_PORT);
   Serial.begin(9600);
 }
 int luz1, luz2, p;

 void loop() {
 mantemConexoes();
 enviaValores();
  MQTT.loop();
 }
 void mantemConexoes() {
   if (!MQTT.connected()) {
     conectaMQTT();
   }
   conectaWiFi(); //se não há conexão com o WiFI, a conexão é refeita
 }
void conectaWiFi() {
  if (WiFi.status() == WL_CONNECTED) {
    return;
  }
  Serial.print("Conectando-se na rede: ");
  Serial.print(SSID);
  Serial.println("  Aguarde!");

  WiFi.begin(SSID, PASSWORD); // Conecta na rede WI-FI
  //WiFi.begin(SSID); // Conecta na rede WI-FI

  while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    Serial.print(".");
  }

  Serial.println();
  Serial.print("Conectado com sucesso, na rede: ");
  Serial.print(SSID);
  Serial.print("  IP obtido: ");
  Serial.println(WiFi.localIP());
}
```

```
void conectaMQTT() {
  while (!MQTT.connected()) {
    Serial.print("Conectando ao Broker MQTT: ");
    Serial.println(BROKER_MQTT);
    if (MQTT.connect(ID_MQTT)) {
      Serial.println("Conectado ao Broker com sucesso!");
    }
    else {
      Serial.println("Noo foi possivel se conectar ao broker.");
      Serial.println("Nova tentatica de conexao em 10s");
      delay(10000);
    }
  }
}
String conversor() {
  String  v1, v2, v3;
  p = analogRead(pot);
  p = map(p, 100, 4070, 0, 8);
  luz1 = analogRead(lr1);
  luz2 = analogRead(lr2);
  Serial.print(luz1);
  Serial.print(" ");
  Serial.print(luz2);
  Serial.print(" ");
  if (luz1 > 200)
  {
    v1 = "on";
    v2 = "off";
  }
  if (luz2 > 200) {
    v1 = "off";
    v2 = "on";
  }

  if (luz2 > 200 && luz1 > 200) {
    v1 = "on";
    v2 = "on";
  }
  if (luz2 < 200 && luz1 < 200) {
    v1 = "off";
    v2 = "off";
  }

  v3 = String(p);
  String json[] = {"ldr1:", v1, ",ldr2:", v2, ",f:", v3, "."};

  String msg;
  for (int i = 0; i < 7; i++) {
    msg += json[i];
  }
  return msg;
}
```

```
  void enviaValores() {
  String msg=conversor();
    const char* dados = msg.c_str();
    Serial.println(msg);


    delay(300);
  MQTT.publish(TOPIC_PUBLISH, dados);
  delay(300);

  }
```

Código Subscriber

```
void setup() {
  Serial.begin(9600);
  pinMode(s1,OUTPUT);
  pinMode(s2,OUTPUT);

  conectaWiFi();
  MQTT.setServer(BROKER_MQTT, BROKER_PORT);
  MQTT.setCallback(callback);
  mantemConexoes();
  mantemConexoes();
  MQTT.subscribe("pp");
  json[0]="cas";
}

void loop() {
  MQTT.loop();
}

void mantemConexoes() {
  if (!MQTT.connected()) {
    conectaMQTT();
  }

  conectaWiFi(); //se não há conexão com o WiFI, a conexão é refeita
}
```

```
void conectaWiFi() {

  if (WiFi.status() == WL_CONNECTED) {
    return;
  }

  Serial.print("Conectando-se na rede: ");
  Serial.print(SSID);
  Serial.println("  Aguarde!");

  WiFi.begin(SSID, PASSWORD); // Conecta na rede WI-FI
  //WiFi.begin(SSID); // Conecta na rede WI-FI

  while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    Serial.print(".");
  }

  Serial.println();
  Serial.print("Conectado com sucesso, na rede: ");
  Serial.print(SSID);
  Serial.print("  IP obtido: ");
  Serial.println(WiFi.localIP());
}
```

```
void conectaMQTT() {
  while (!MQTT.connected()) {
    Serial.print("Conectando ao Broker MQTT: ");
    Serial.println(BROKER_MQTT);
    if (MQTT.connect(ID_MQTT)) {
      Serial.println("Conectado ao Broker com sucesso!");

    }
    else {
      Serial.println("Não foi possivel se conectar ao broker.");
      Serial.println("Nova tentatica de conexao em 10s");
      delay(10000);
    }
  }
}
```

```
void callback(char* topic, byte*message, unsigned int length) {

  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();
  Serial.println(messageTemp);
  Serial.println();
  String a = messageTemp.substring(messageTemp.indexOf(":")+1);
  String ldr1 = a.substring(0,a.indexOf(","));
  Serial.println(a);
  Serial.print("LDR1: ");
  Serial.println(ldr1);
  a = a.substring(a.indexOf(":")+1);
  String ldr2 = a.substring(0,a.indexOf(","));
  Serial.print("LDR2: ");
  Serial.println(ldr2);
  a = a.substring(a.indexOf(":")+1);
  String mult = a.substring(0,a.indexOf("."));
  int f1 = 200, f2=500;
  f1 = f1*mult.toInt();
  f2 = f2*mult.toInt();
  exc(ldr1,ldr2,f1,f2);
}
String makestr(String t){
  String ret;
  ret = "LDR1:"+ t[0];
  ret+= ",LDR2:"+t[1];
  Serial.println(ret);
  return ret;
}
void exc(String l1,String l2,int f1,int f2){
    if(l1=="on"){
      Serial.println("on1");
      tone(s1,f1,1000);
      }
     if(l2=="on"){
      Serial.println("on2");
      tone(s2,f2,1000);
      }

}
```