

Universidad de Costa Rica

Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE0308 – Laboratorio Eléctrico I
II ciclo 2019

Reporte del proyecto final

Vúmetro con un cubo LEDs 3x3x3 al ritmo de la música

Pablo Gómez Acuña B73201
Belinda Brown Ramírez B61254
Grupo 04, mesa 02

Profesor: Diego Dumani

Fecha de entrega
20 de noviembre de 2019

Índice

1. Resumen	1
2. Objetivos	2
2.1. Objetivos Generales	2
2.2. Objetivos específicos	2
3. Corrección del diseño	3
4. Lista de equipos y componentes	3
4.1. Equipos	3
4.2. Componentes	3
5. Resultados experimentales	4
5.1. Pruebas de LED's	5
5.2. Ruptura al ritmo de la música	6
6. Conclusiones y recomendaciones	13
6.1. Conclusiones	13
6.2. Recomendaciones	13
7. Anexos	15

Índice de figuras

1.	Conexión del jack	3
2.	Configuración en la tarjeta de arduino Uno	4
3.	Conexiones en la protoboard	5
4.	Código en Arduino, enciende LEDs del cubo de manera aleatoria	6
5.	Código en Arduino para encender los LED's al ritmo de la música, parte 1[2]	7
6.	Código en Arduino para encender los LED's al ritmo de la música, parte 2	8
7.	Código en Arduino para encender los LED's al ritmo de la música, parte 3	9
8.	Código en Arduino para encender los LED's al ritmo de la música, parte 4	10
9.	Código en Arduino para encender los LED's al ritmo de la música, parte 5	11
10.	Código en Arduino para encender los LED's al ritmo de la música, parte 6	12
11.	Estado del cubo de LED's cuando se encuentra sin música	15
12.	Estado del cubo de LED's cuando se encuentra con música, a cierto ritmo	16

Índice de tablas

1. Tabla con los componentes utilizados de la bodega 3

1. Resumen

Se define vúmetro como un dispositivo capaz de mostrar la intensidad de seřal de audio.[4] De manera general, se contruye un cubo de LEDs 3x3x3, es decir en total nueve LEDs por nivel.[2] Con el propóposito de cumplir con el objetivo de encender los LED's al ritmo de la música se utiliza un Arduino Uno.[3]

Palabras clave: Arduino Uno, cubo de LEDs, transistores, resistencias, IDLE de Arduino, música, vúmetro.

2. Objetivos

2.1. Objetivos Generales

- Implementar un vúmetro con un cubo de LEDs 3x3x3.

2.2. Objetivos específicos

- Construir un cubo de LEDs 3x3x3.
- Diseñar un vúmetro con el fin de encender los LEDs al ritmo de la música.
- Desarrollar la programación adecuada con el fin de tomar el voltaje de corte como parámetro.

3. Corrección del diseño

Se puede considerar como corrección al diseño, la configuración con el jack de audio hacia la tarjeta de arduino Uno debido a que no se utilizó el jack con sus tres conexiones para conectarlo a la tarjeta sino que se usó un cable de jack en sus dos extremos. Puesto que se necesitaba un jack con sus tres conexiones, se extrajo una la cual se encuentra en la unidad de CD-ROOM, por lo general de cualquier CPU, de una que se encontraba en desechos electrónicos.



Figura 1: Conexión del jack

4. Lista de equipos y componentes

4.1. Equipos

- Osciloscopio
- Multímetro
- Generador de señales
- Fuente DC

4.2. Componentes

En la siguiente tabla se muestran los componentes a utilizar[1]:

Valor	Componente	Cantidad
220	Resistencia	9
10k	Resistencia	3
Blaco de 5mm	LED	27
Placa	Arduino UNO	1
NTE196	Transistor	3

Tabla 1: Tabla con los componentes utilizados de la bodega

5. Resultados experimentales

Contemplando la conexión de la tarjeta de arduino Uno como se muestra en la figura 2 y en figura 3 se muestra la conexión en la protoboard de todos los elementos electrónicos, se realizó la lógica detrás de los distintos algoritmos mostrados a continuación.



Figura 2: Configuración en la tarjeta de arduino Uno

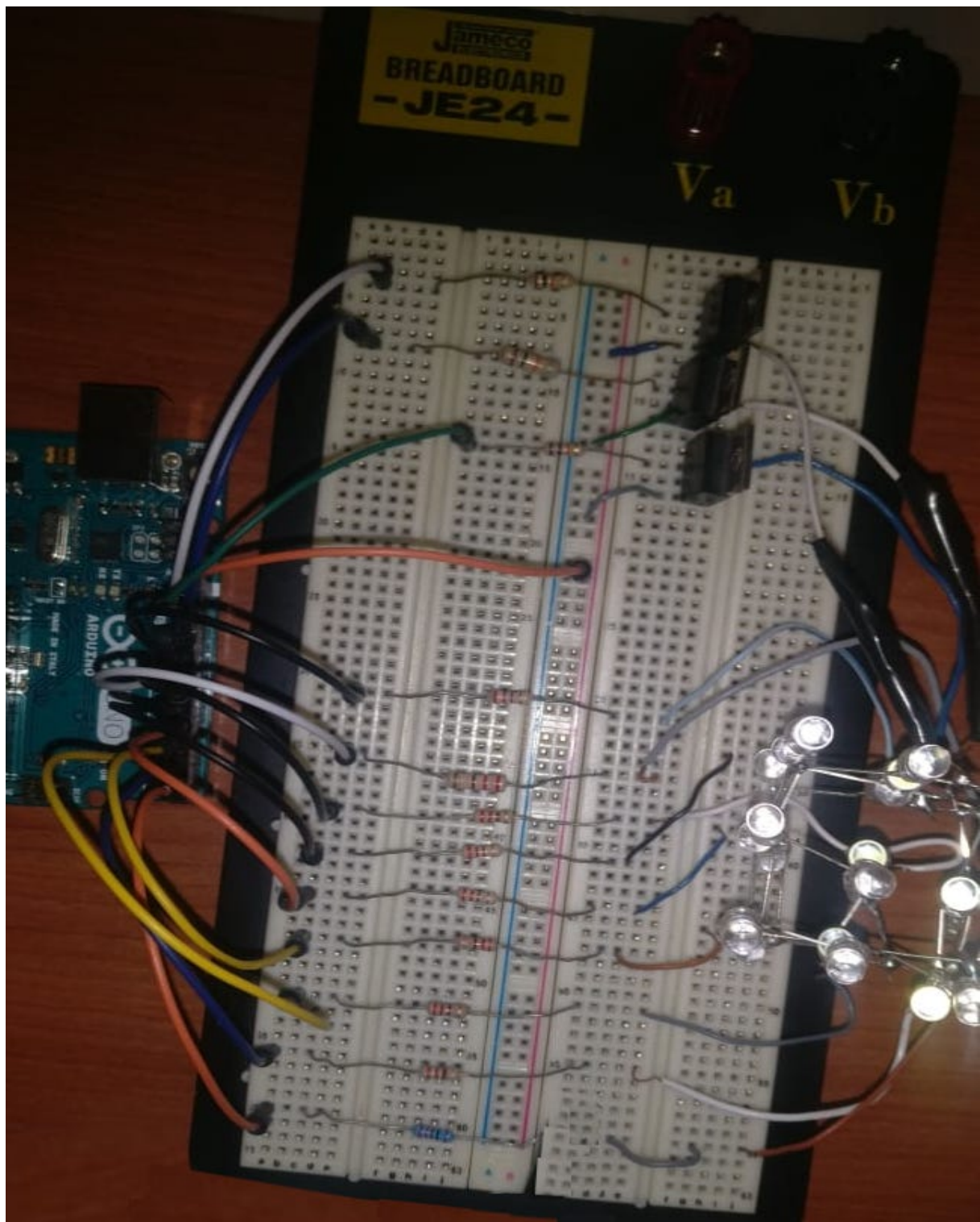


Figura 3: Conexiones en la protoboard

5.1. Pruebas de LED's

Con lo que respecta al funcionamiento de los LEDs se requirió realizar un código en arduino para cual se encienden de manera aleatoria. Esto como primera prueba donde corresponde el siguiente código:

```
vumetro3x3x3.ino | aleatorio.ino
1  int Columnas[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
2
3  int Filas[] = {12, 11, 10};
4  int RandFila;      //Variable para fila aleatoria
5  int RandColumna;  //Variable para columna aleatoria
6
7  void setup()
8  {
9      int contador;
10
11     for (int contador = 1; contador < 10; contador++){
12         pinMode(Columnas[contador], OUTPUT); }
13
14     for (int contador = 1; contador < 4; contador++){
15         pinMode(Filas[contador], OUTPUT); }
16 }
17 void loop()
18 {
19     RandLed();
20 }
21 void RandLed()
22 {
23     RandFila = random(0,3);
24     RandColumna = random(0,9);
25
26     digitalWrite(Filas[RandFila], HIGH);
27     digitalWrite(Columnas[RandColumna], HIGH);
28     delay(75);
29     digitalWrite(Filas[RandFila], LOW);
30     digitalWrite(Columnas[RandColumna], LOW);
31     delay(50);
32 }
33
```

Figura 4: Código en Arduino, enciende LEDs del cubo de manera aleatoria

5.2. Ruptura al ritmo de la música

Visto que el objetivo se enfoca en encender los LEDs siguiendo el ritmo de la música, se plantea el siguiente código donde funciona mediante condicionales amplios con el fin de encender los LEDs según la señal que pasa por el jack. Cabe recalcar que el jack utilizado fue obtenido de la unidad de disco de una CPU con defecto en tarjeta madre, para comprobar su funcionamiento se hicieron pruebas de continuidad entre la entrada auxiliar libre y la que contiene el jack conectado, se observan como resultado estados de continuidad para las tres partes.

```
vumetro3x3x3.ino
1 //Se declaran los Leds
2
3 int LED1 = 3;
4
5 int LED2 = 4;
6
7 int LED3 = 5;
8
9 int LED4 = 6;
10
11 int LED5 = 7;
12
13 int LED6 = 8;
14
15 int LED7 = 9;
16
17 int LED8 = 10;
18
19 int LED9 = 11;
20
21
22
23 //Se declaran 5 variables para manipular los tiempos de silencio
24
25 int Valor;
26
27 int Valor1;
28
29 int Valor2;
30
```

vumetro3x3x3.ino* 9:14

Figura 5: Código en Arduino para encender los LED's al ritmo de la música, parte 1[2]

```

vumetro3x3x3.ino
21
22
23 //Se declaran 5 variables para manipular los tiempos de silencio
24
25 int Valor;
26
27 int Valor1;
28
29 int Valor2;
30
31 int Valor3;
32
33 int Valor4;
34
35
36
37 //Configuración de los pines y el serial para poder visualizar
38 //la entrada analógica, en nuestro caso ver los valores que entran
39 //para programar los valores de ruptura
40
41 void setup () {
42
43   Serial.begin(9600);
44
45   pinMode(LED1, OUTPUT);
46
47   pinMode(LED2, OUTPUT);
48
49   pinMode(LED3, OUTPUT);
50
51   pinMode(LED4, OUTPUT);
52
53   pinMode(LED5, OUTPUT);
54
55   pinMode(LED6, OUTPUT);
56
57   pinMode(LED7, OUTPUT);
58
59   pinMode(LED8, OUTPUT);
60
61   pinMode(LED9, OUTPUT);
62
63
64 }
65

```

Figura 6: Código en Arduino para encender los LED's al ritmo de la música, parte 2

```

vumetro3x3x3.ino
67
68 void loop (){
69     //Para leer el valor
70
71     Valor = analogRead(A0);
72
73     //La idea es transferir los valores
74     //para saber cual era el estado anterior
75
76     Valor4 = Valor3;
77
78     Valor3 = Valor2;
79
80     Valor2 = Valor1;
81
82     Valor1 = Valor;
83
84
85
86
87     //Para visualizar los
88     //valores en Serial Monitor
89
90     Serial.print("Valor: ");
91
92     Serial.print(Valor);
93
94     Serial.print("\t Valor1: ");
95
96     Serial.print(Valor1);
97
98     Serial.print("\t Valor2: ");
99
100    Serial.print(Valor2);
101
102    Serial.print("\t Valor3: ");
103
104    Serial.print(Valor3);
105
106    Serial.print("\t Valor4: ");
107
108    Serial.println(Valor4);
109
110

```

Figura 7: Código en Arduino para encender los LED's al ritmo de la música, parte 3

```

vumetro3x3x3.ino
---
111
112 //Condicionamos que si el programa lee 4 veces seguidas 0 que
113 //se apaguen los leds,
114 // si no ponemos esto los leds van a parpadear mucho
115
116 if (Valor1+Valor2+Valor3+Valor4==0){
117
118     digitalWrite(LED1,LOW);
119
120     digitalWrite(LED2,LOW);
121
122     digitalWrite(LED3,LOW);
123
124     digitalWrite(LED4,LOW);
125
126     digitalWrite(LED5,LOW);
127
128     digitalWrite(LED6,LOW);
129
130     digitalWrite(LED7,LOW);
131
132     digitalWrite(LED8,LOW);
133
134     digitalWrite(LED9,LOW);
135
136
137 }
138
139 //Encendemos los leds segun el valor de entrada.
140 // En este caso la salida de mi pc como mucho me llega a 400 asi
141 //que lo dividimos entre ocho leds.
142
143 else{
144
145     if ((Valor < 10) || (Valor > 10) || (Valor == 0)){
146         digitalWrite(LED1, HIGH);
147     }
148
149     else{
150         digitalWrite(LED1, LOW);
151     }

```

Figura 8: Código en Arduino para encender los LED's al ritmo de la música, parte 4

```

vumetro3x3x3.ino
152
153   if ((Valor < 30) || (Valor > 30)){
154       digitalWrite(LED2, HIGH);
155   }
156
157   else{
158       digitalWrite(LED2, LOW);
159   }
160
161   if ((Valor < 40) || (Valor > 40)){
162       digitalWrite(LED3, HIGH);
163   }
164
165   else{
166       digitalWrite(LED3, LOW);
167   }
168
169   if ((Valor < 50) || (Valor > 50)){
170       digitalWrite(LED4, HIGH);
171   }
172
173   else{
174       digitalWrite(LED4, LOW);
175   }
176
177   if ((Valor < 55) || (Valor > 55)){
178       digitalWrite(LED5, HIGH);
179   }
180   else{
181       digitalWrite(LED5, LOW);
182   }
183
184   if ((Valor < 60) || (Valor > 60)){
185       digitalWrite(LED6, HIGH);
186   }
187
188   else{
189       digitalWrite(LED6, LOW);
190   }
191   }
192

```

Figura 9: Código en Arduino para encender los LED's al ritmo de la música, parte 5

```

192
193   if ((Valor < 70) || (Valor > 70)){
194       digitalWrite(LED7, HIGH);
195   }
196
197   else{
198       digitalWrite(LED7, LOW);
199   }
200
201   if ((Valor < 75) || (Valor > 75)){
202       digitalWrite(LED8, HIGH);
203   }
204
205   else{
206       digitalWrite(LED8, LOW);
207   }
208
209   if ((Valor < 350) || (Valor > 350)){
210       digitalWrite(LED9, HIGH);
211   }
212
213   else{
214       digitalWrite(LED9, LOW);
215   }
216   //fin del else del loop
217   }
218   // Fin del loop
219   }
220
vumetro3x3x3.ino*   168:1

```

Figura 10: Código en Arduino para encender los LED's al ritmo de la música, parte 6

Es importante recalcar que se realizaron pruebas sobre como actúan los LEDs con respecto a varias canciones. Las imágenes adjuntas son las pruebas realizadas con la canción "Bee Gees - Stayin' Alive (1977)". La figura 11 muestra el estado de apagado del cubo debido a la ausencia de música, en cambio, una vez con la música se empiezan a encender los LED's de acuerdo al valor de ruptura de entrada como se observa en la figura 12.

6. Conclusiones y recomendaciones

6.1. Conclusiones

1. Se comprobó el correcto funcionamiento de las conexiones eléctricas mediante el código de reproducción aleatoria.
2. Con lo que respecta a la ruptura al ritmo de la música se presentaron dificultades a la hora de encender tres LED's se considera que es debido a la programación ya que los rangos de encendido de los LED's varían respecto a 9 diferentes condiciones y su negación.
3. Se concluye que el cambio en los valores de ruptura si afecta como se espera al comportamiento de los LED's.

6.2. Recomendaciones

1. Es importante tener sumo cuidado cuando se sueldan los ánodos y cátodos del arduino ya que si se come el error de conectarse al revés este no va a funcionar.
2. Se recomienda instalar el IDLE de Arduino sobre cualquier computadora que posea Windows como sistema operativo debido a que sobre Linux puede generar algunas complicaciones a la hora de cargar el código en el arduino.

Referencias

- [1] Montero. A. *Electrónica*. Editorial Editex. Descargado de https://books.google.co.cr/books?id=_g0xi3Til6AC, 2009.
- [2] Svoboda. J. Dorf. R. *Gs eléctricos*. México D.F. Alfaomega., 2011.
- [3] Harper. G. *El abc del control electrónico de las máquinas eléctricas*. Limusa. Descargado de <https://books.google.co.cr/books?id=2Jp7EpxiMbwC>, 2003.
- [4] Espinoza. M. *OrCAD 9.1. Análisis y diseño de circuitos analógicos asistido por computadora*. Obtenido el 19 de noviembre de 2019 desde la fuente <http://eie.ucr.ac.cr/uploads/file/software/GuiaOrCAD%209,1.pdf>, 2008.

7. Anexos

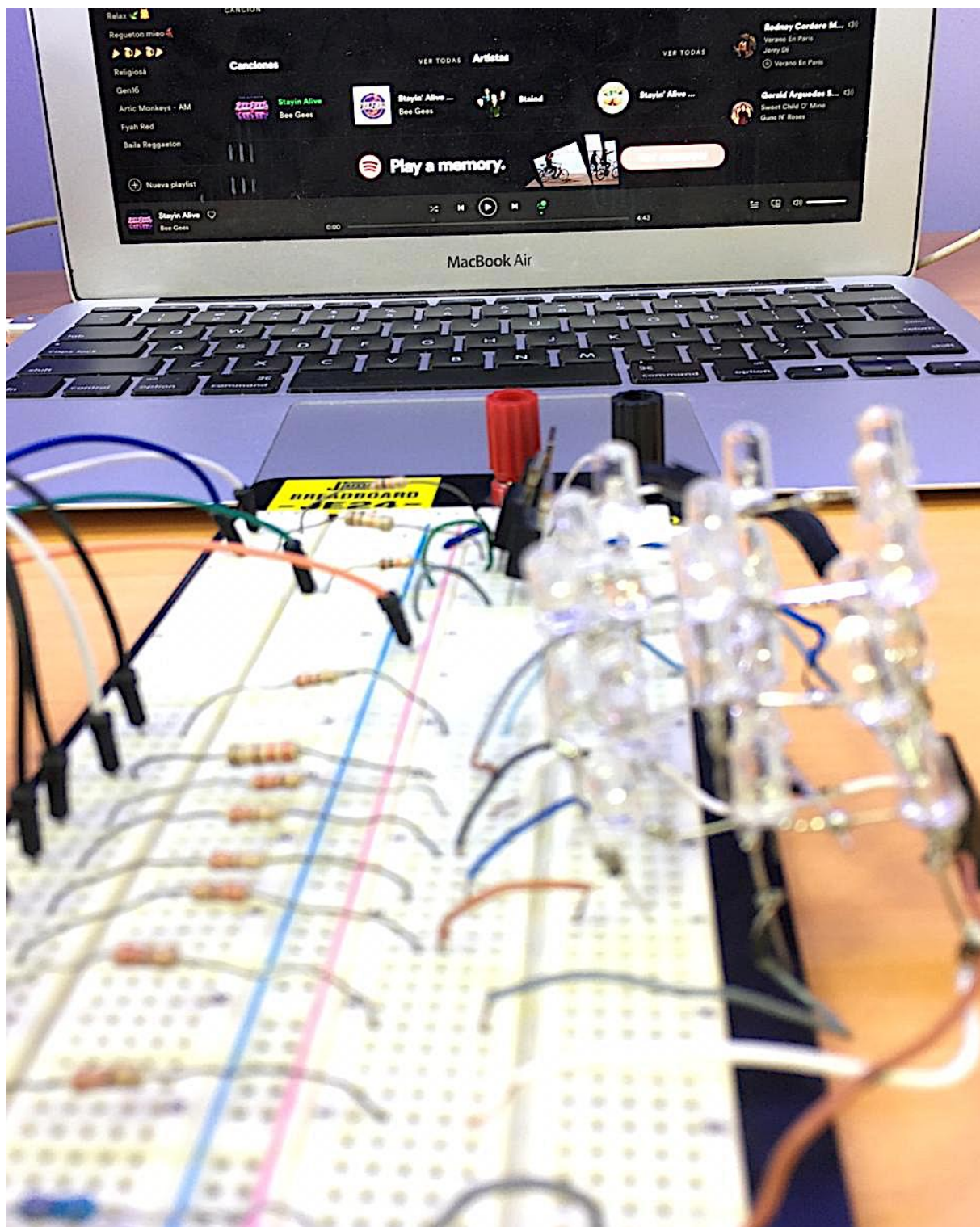


Figura 11: Estado del cubo de LED's cuando se encuentra sin música

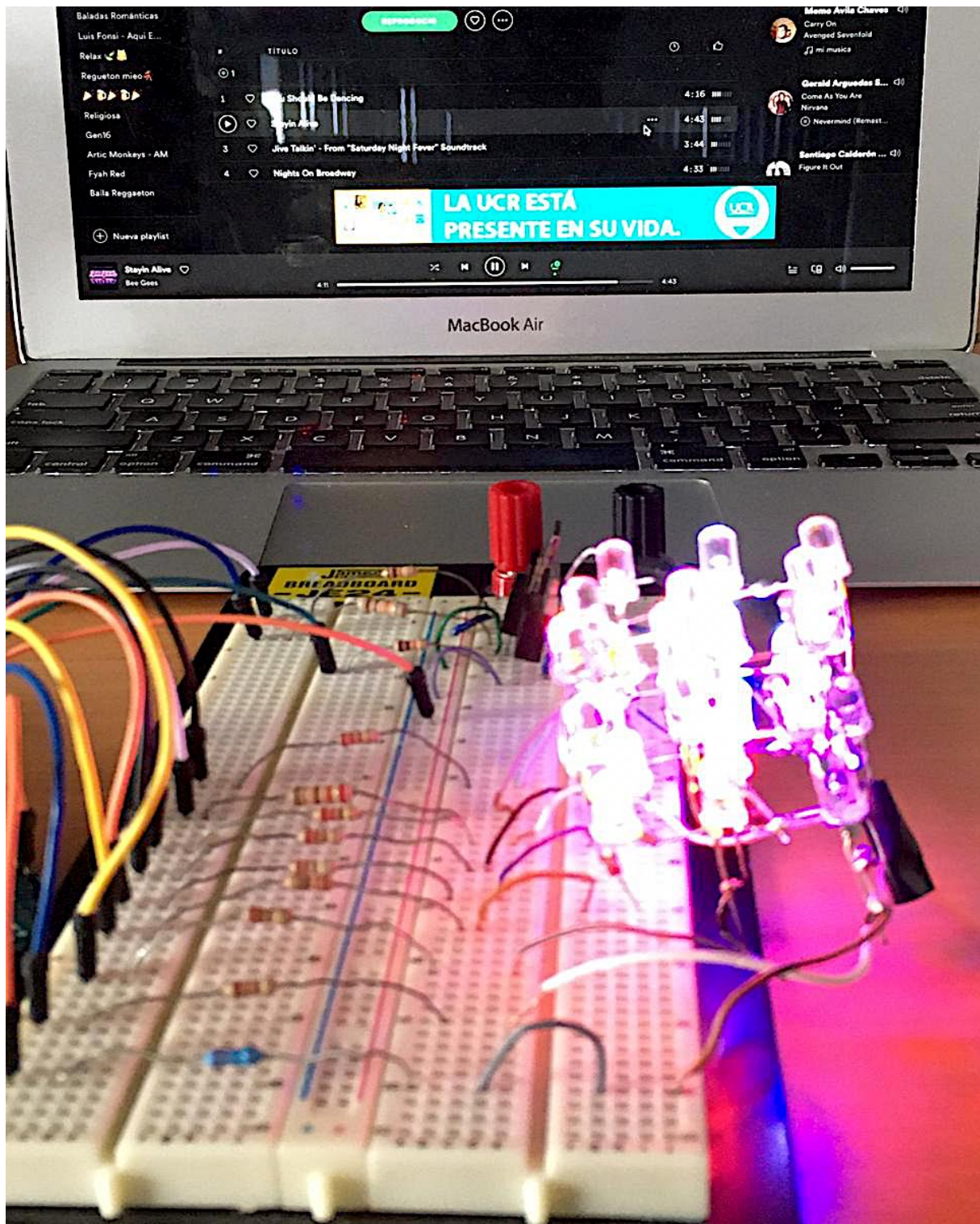


Figura 12: Estado del cubo de LED's cuando se encuentra con música, a cierto ritmo