Documentación
PRÁCTICA 5

Estructura de computadores

UNIVERSIDAD
DE GRANADA

Fernández Mertanen, Jonathan

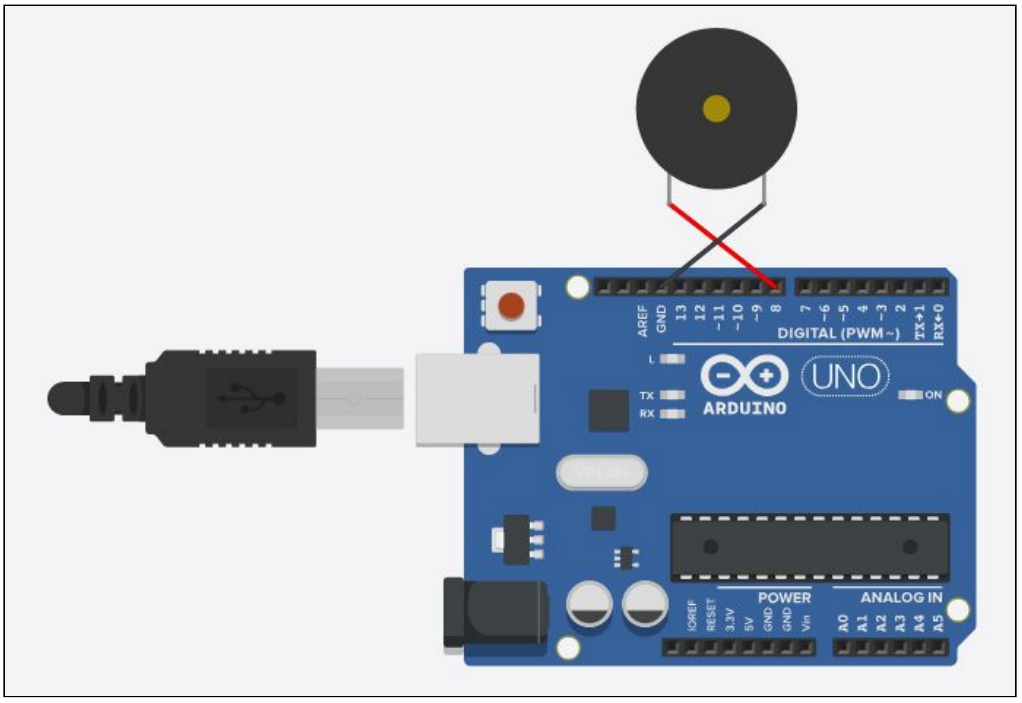
Índice

Primer circuito: Passive Buzzer

La intención de este circuito es emitir ocho sonidos a diferentes frecuencias, cada uno durante 0,5 segundos.
Componentes requeridos:

- (1) x Placa Elegoo Mega 2560 R3
- (1) x Passive buzzer
- (2) x H-M cables

Diagrama



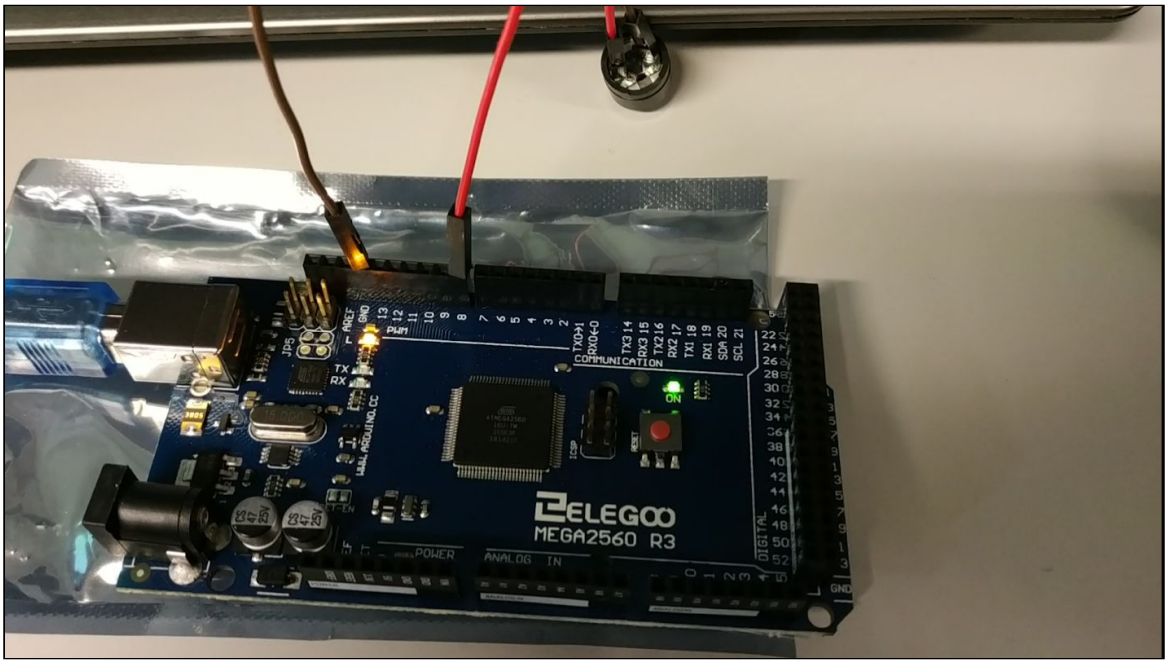
Código

Proporcionado en las sesiones de el guión de Elegoo.

```
#include <pitches.h>

// notes in the melody:
int melody[] = {
  NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5,
NOTE_C6};
int duration = 500;  // 500 miliseconds

void setup() {
```

```
}

void loop() {
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    // pin8 output the voice, every scale is 0.5 sencond
    tone(8, melody[thisNote], duration);

    // Output the voice after several minutes
    delay(500);
  }

  // restart after two seconds
  delay(1000);
}
```

Montaje

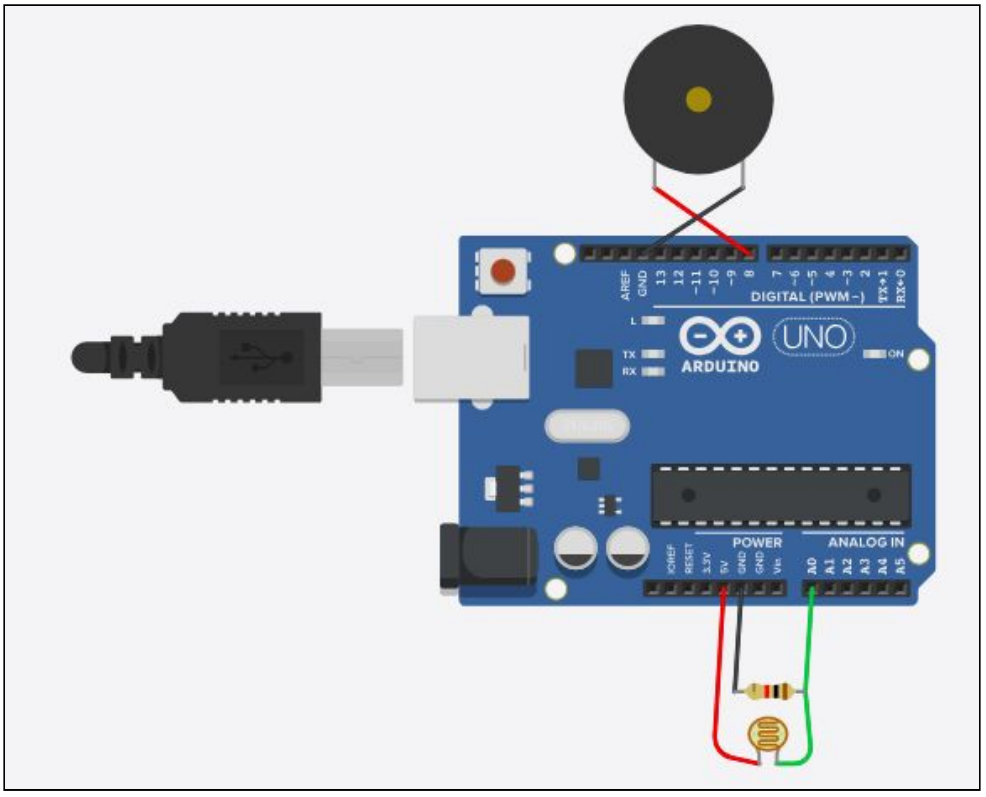


Segundo circuito: Light Theremin

La intención de este circuito es implementar un theremin para que funcione con una célula fotovoltaica como sensor de luz para controlar la frecuencia.
Componentes requeridos:

- (1) x Placa Elegoo Mega 2560 R3
- (1) x Passive buzzer
- (1) x Célula fotovoltaica
- (1) x Resistencia 1kΩ
- (X) x H-M cables

Diagrama



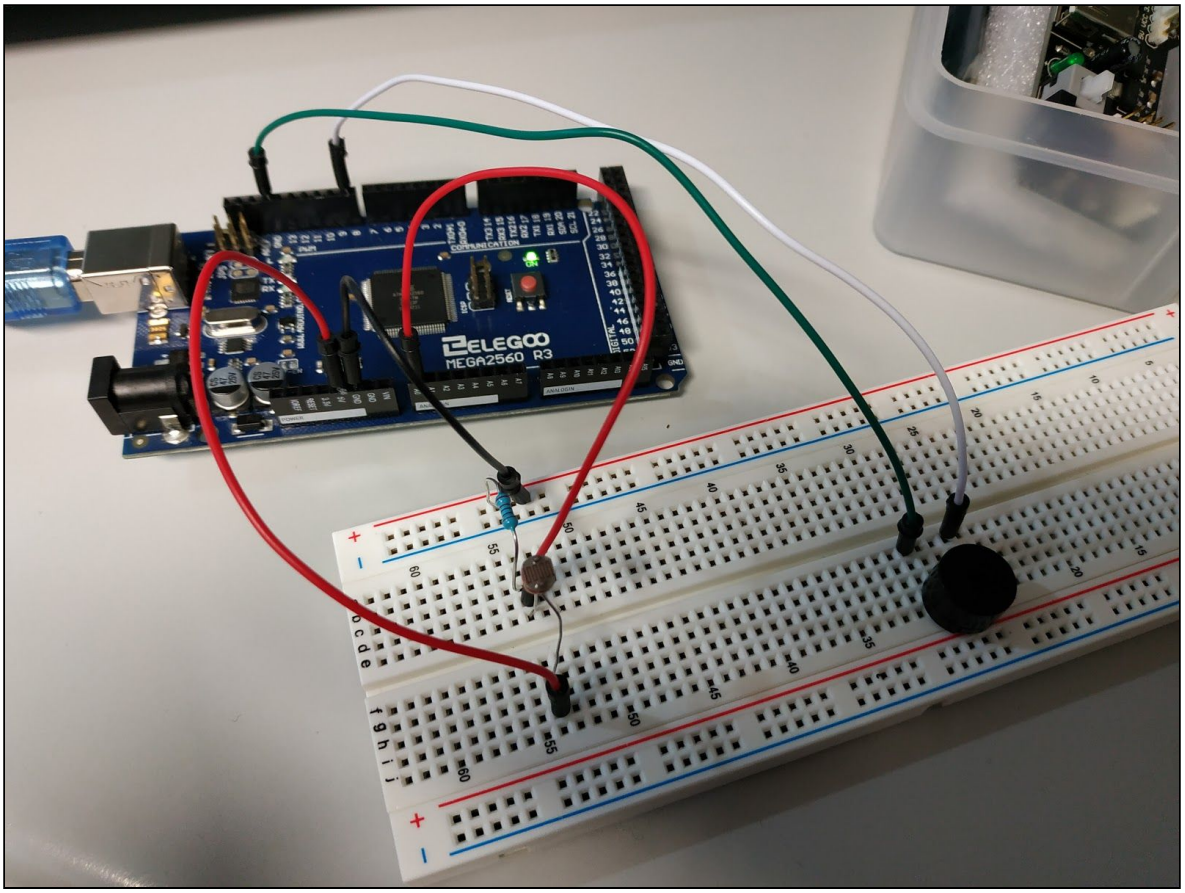
Código

Proporcionado por arduino como ejemplo.

```
// variable to hold sensor value
int sensorValue;
// variable to calibrate low value
int sensorLow = 1023;
// variable to calibrate high value
int sensorHigh = 0;
// LED pin
const int ledPin = 13;

void setup() {
  // Make the LED pin an output and turn it on
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);

  // calibrate for the first five seconds after program runs
  while (millis() < 5000) {
    // record the maximum sensor value
```

```
    sensorValue = analogRead(A0);
    if (sensorValue > sensorHigh) {
      sensorHigh = sensorValue;
    }
    // record the minimum sensor value
    if (sensorValue < sensorLow) {
      sensorLow = sensorValue;
    }
  }
  // turn the LED off, signaling the end of the calibration period
  digitalWrite(ledPin, LOW);
}

void loop() {
  //read the input from A0 and store it in a variable
  sensorValue = analogRead(A0);

  // map the sensor values to a wide range of pitches
  int pitch = map(sensorValue, sensorLow, sensorHigh, 50, 4000);

  // play the tone for 20 ms on pin 8
  tone(8, pitch, 20);

  // wait for a moment
  delay(10);
}
```

Montaje

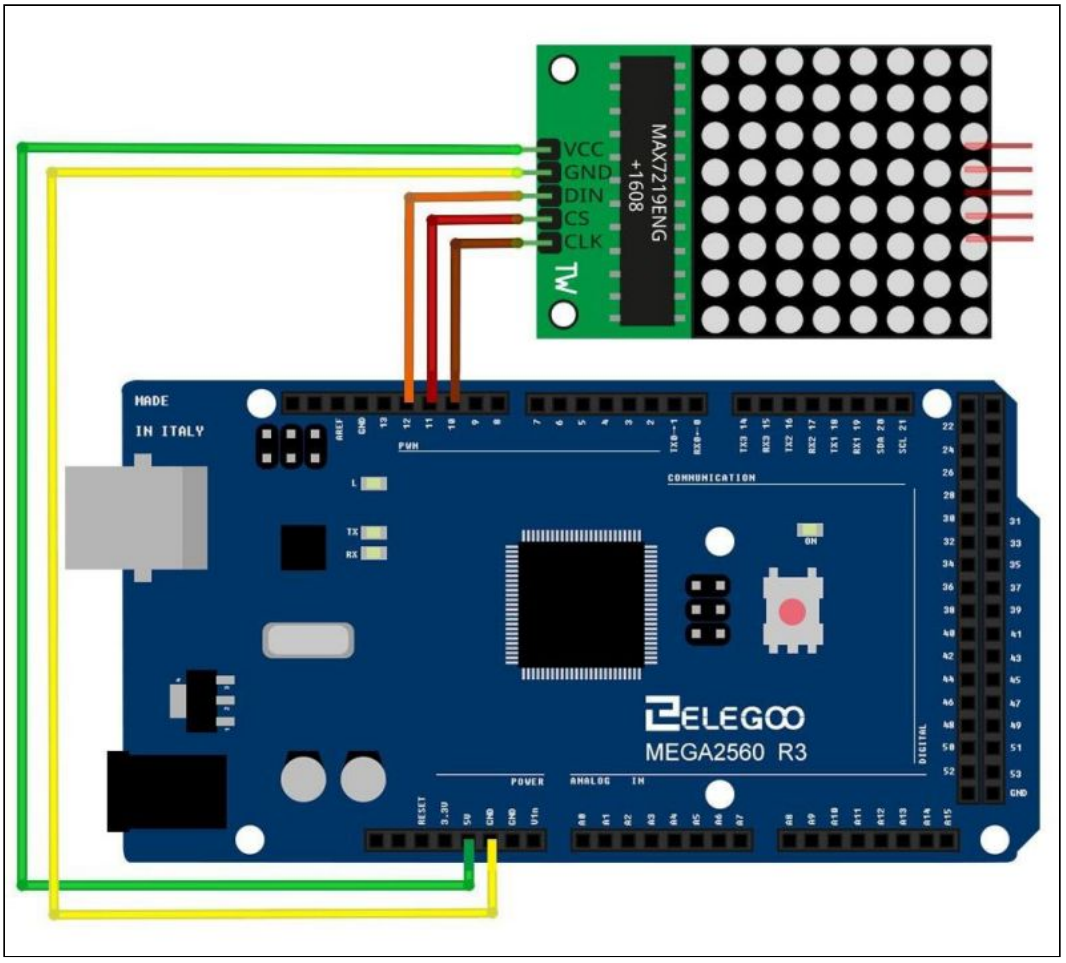


Tercer circuito: Led Matrix 8x8 + controller

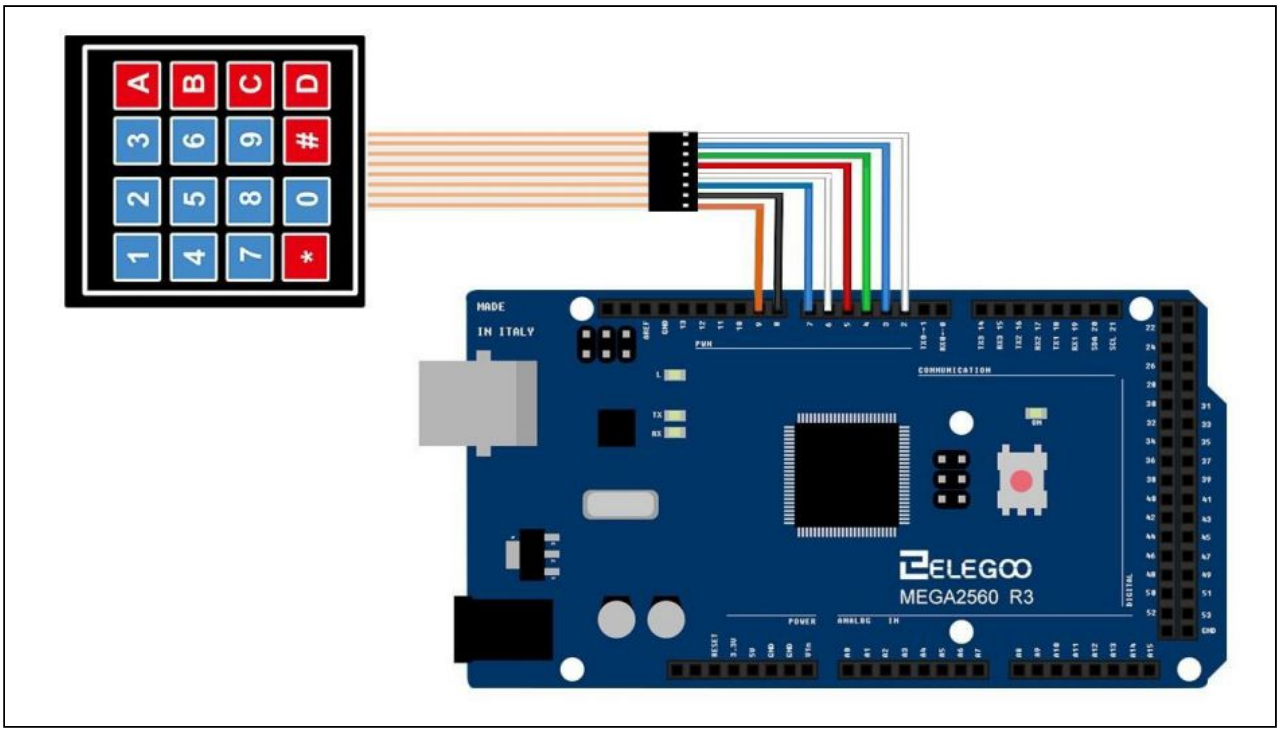
Implementación de un circuito que permite controlar una matriz 8x8 de luces led con un controlador de membrana.
Componentes requeridos:

- (1) x Placa Elegoo Mega 2560 R3
- (1) x Led Matrix 8x8
- (1) x Controlador de membrana
- (X) x H-M cables

Diagrama


VCC
GND
DIN
CS
CLK
MAX7219ENG
+1608
MADE
IN ITALY
ELEGOO
MEGA2560 R3

+



Código

```
#include <LedControl.h>
#include <Keypad.h>

const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the symbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {9, 8, 7, 6}; //connect to the row pinouts of the
keypad
byte colPins[COLS] = {5, 4, 3, 2}; //connect to the column pinouts of
the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins,
ROWS, COLS);

/*
 Now we need a LedControl to work with.
 ***** These pin numbers will probably not work with your hardware *****
```

```
 pin 12 is connected to the DataIn
 pin 11 is connected to LOAD(CS)
 pin 10 is connected to the CLK
 We have only a single MAX72XX.
 */
LedControl lc=LedControl(12,10,11,1);

/* we always wait a bit between updates of the display */
unsigned long delaytime1=1000;
unsigned long delaytime2=50;

/*
 This method will display the characters that are pressed
 on the keypad on the matrix .
 */

void writeMatrix(byte C[8]){
  lc.setRow(0,0,C[0]);
  lc.setRow(0,1,C[1]);
  lc.setRow(0,2,C[2]);
  lc.setRow(0,3,C[3]);
  lc.setRow(0,4,C[4]);
  lc.setRow(0,5,C[5]);
  lc.setRow(0,6,C[6]);
  lc.setRow(0,7,C[7]);
}

void printKey(char key){
  byte
A[8]={B00011000,B00100100,B00100100,B00111100,B00111100,B00100100,B00100
100,B00100100};
  byte
B[8]={B00111000,B00100100,B00100100,B00111000,B00111000,B00100100,B00100
100,B00111000};
  byte
C[8]={B00011000,B00100100,B00100000,B00100000,B00100000,B00100000,B00100
100,B00011000};
  byte
D[8]={B00111000,B00100100,B00100100,B00100100,B00100100,B00100100,B00100
100,B00111000};
  byte
n0[8]={B00111100,B00100100,B00100100,B00100100,B00100100,B00100100,B0010
0100,B00111100};
  byte
n1[8]={B00001000,B00011000,B00101000,B00001000,B00001000,B00001000,B0000
```

```
1000,B00001000};
  byte
n2[8]={B00011000,B00100100,B00000100,B00001000,B00010000,B00100000,B0010
0000,B00111100};
  byte
n3[8]={B00011000,B00100100,B00000100,B00011000,B00011000,B00000100,B0010
0100,B00011000};
  byte
n4[8]={B00000100,B00001100,B00010100,B00100100,B00111100,B00000100,B0000
0100,B00000100};
  byte
n5[8]={B00111100,B00100000,B00100000,B00111000,B00000100,B00000100,B0000
0100,B00111000};
  byte
n6[8]={B00011000,B00100100,B00100000,B00100000,B00111000,B00100100,B0010
0100,B00011000};
  byte
n7[8]={B00111100,B00000100,B00000100,B00001000,B00001000,B00010000,B0001
0000,B00010000};
  byte
n8[8]={B00011000,B00100100,B00100100,B00011000,B00011000,B00100100,B0010
0100,B00011000};
  byte
n9[8]={B00011000,B00100100,B00100100,B00011100,B00000100,B00000100,B0010
0100,B00011000};
  byte
cross[8]={B00000000,B00010100,B01001000,B00111010,B01011100,B00010010,B0
0101000,B00000000};
  byte
hashtag[8]={B00010100,B00010100,B01111110,B00100100,B00100100,B01111110,
B00101000,B00101000};

  switch(key){
    case 'A':
      Serial.println(key);
      writeMatrix(A);
      break;
    case 'B':
      Serial.println(key);
      writeMatrix(B);
      break;
    case 'C':
      Serial.println(key);
      writeMatrix(C);
      break;
```

```
    case 'D':
      Serial.println(key);
      writeMatrix(D);
      break;
    case '0':
      Serial.println(key);
      writeMatrix(0);
      break;
    case '1':
      Serial.println(key);
      writeMatrix(n1);
      break;
    case '2':
      Serial.println(key);
      writeMatrix(n2);
      break;
    case '3':
      Serial.println(key);
      writeMatrix(n3);
      break;
    case '4':
      Serial.println(key);
      writeMatrix(n4);
      break;
    case '5':
      Serial.println(key);
      writeMatrix(n5);
      break;
    case '6':
      Serial.println(key);
      writeMatrix(n6);
      break;
    case '7':
      Serial.println(key);
      writeMatrix(n7);
      break;
    case '8':
      Serial.println(key);
      writeMatrix(n8);
      break;
    case '9':
      Serial.println(key);
      writeMatrix(n9);
      break;
    case '*':
```

```
      Serial.println(key);
      writeMatrix(cross);
      break;
    case '#':
      Serial.println(key);
      writeMatrix(hashtag);
      break;
    default:
      break;
  }
}

void setup() {
  /*
   The MAX72XX is in power-saving mode on startup,
   we have to do a wakeup call
   */
  lc.shutdown(0,false);
  /* Set the brightness to a medium values */
  lc.setIntensity(0,8);
  /* and clear the display */
  lc.clearDisplay(0);
  /* open a port to print */
  Serial.begin(9600);
}

void loop() {
  char customKey = customKeypad.getKey();
  if (customKey){
    lc.clearDisplay(0);
    printKey(customKey);
  }
}
```

Montaje