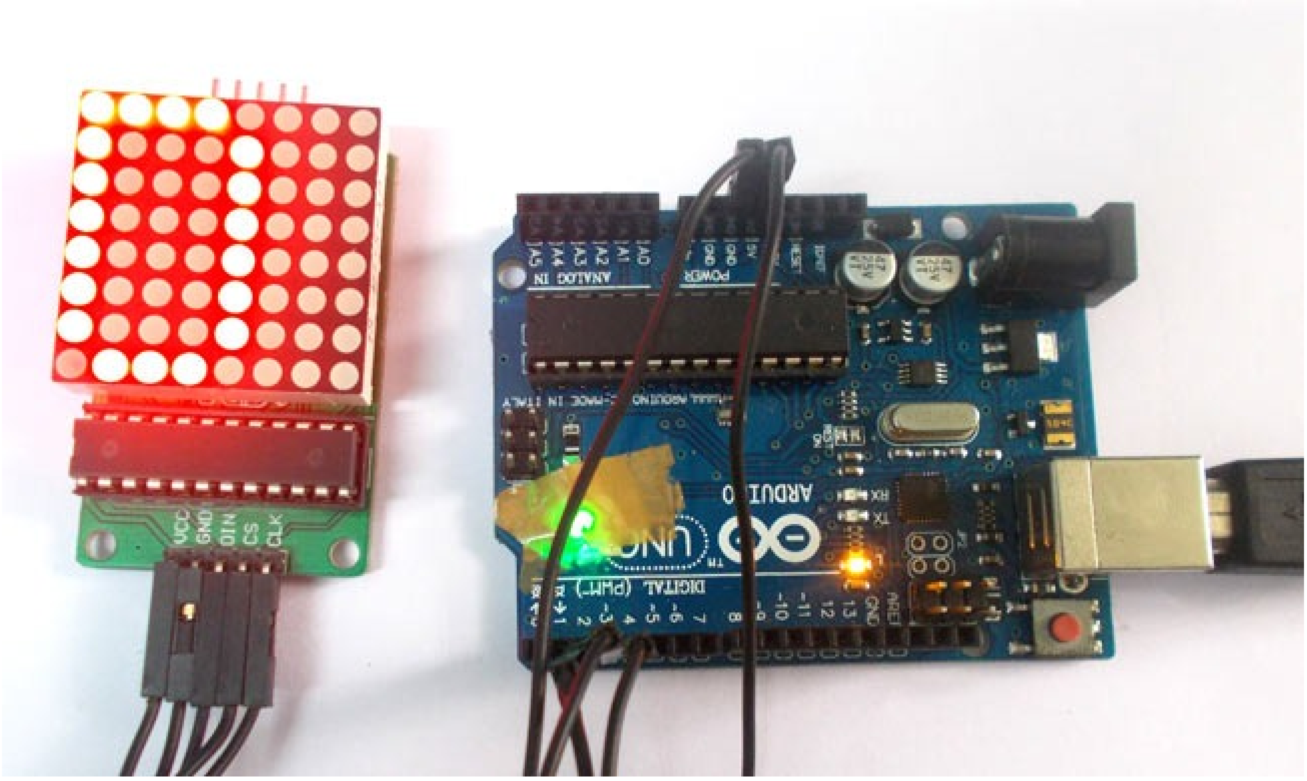
# 8x8 LED Matrix using Arduino

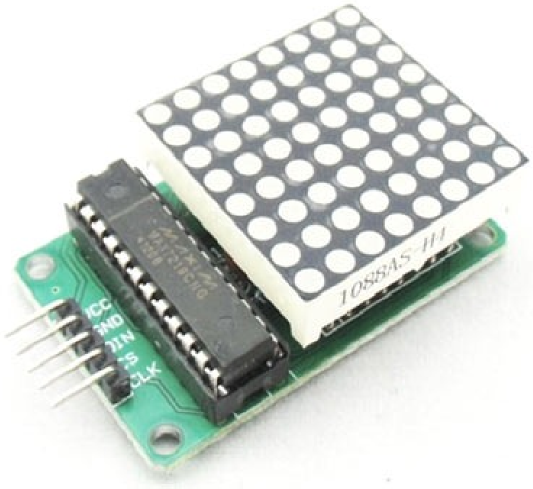


Arduino 8x8 LED Marix

In this project we are going to design an **8x8 LED matrix display**, for that we are going to interface an 8x8 LED matrix module with Arduino Uno. An 8x8 LED matrix contains 64 LEDs (Light Emitting Diodes) which are arranged in the form of a matrix, hence the name LED matrix.

These matrixes can be made by circuiting 64 LEDs, however that process is time consuming. Now a day they are available in compact forms as shown in below image. These compact modules are available in different sizes and many colors. The cost of module is same as cost of 64 LEDs, so for hobbyists this is easiest to work on.

The bare LED matrix has 16 pin outs with 8 common positive and another 8 common negative. For connecting this matrix directly to a UNO, we need to spare 16 pins on the UNO. With the output pins low on UNO, we cannot spare 16 PINS. So we need to connect this matrix to a driver chip. This driver chip along with matrix comes as a set which is shown in below figure.



This module will be interfaced with Arduino for displaying alphabets, hence the matrix display. First of all for interfacing LED matrix with Arduino, we need to download a library specifically designed for LED MATRIX. This library will be available at:
https://github.com/shaai/Arduino_LED_matrix_sketch/archive/master.zip

After downloading the Zip file, extract the contents to ARDUINO folder. (Go to local disk where ARDUINO NIGHTLY software is installed, open the folder, search for folder named "library", extract the contents of zip file in that folder, restart the program you will now be able to use features for matrix interface)

## Components Required

**Hardware:** Arduino Uno, Power supply (5v), 100 uF capacitor (connected across power supply)

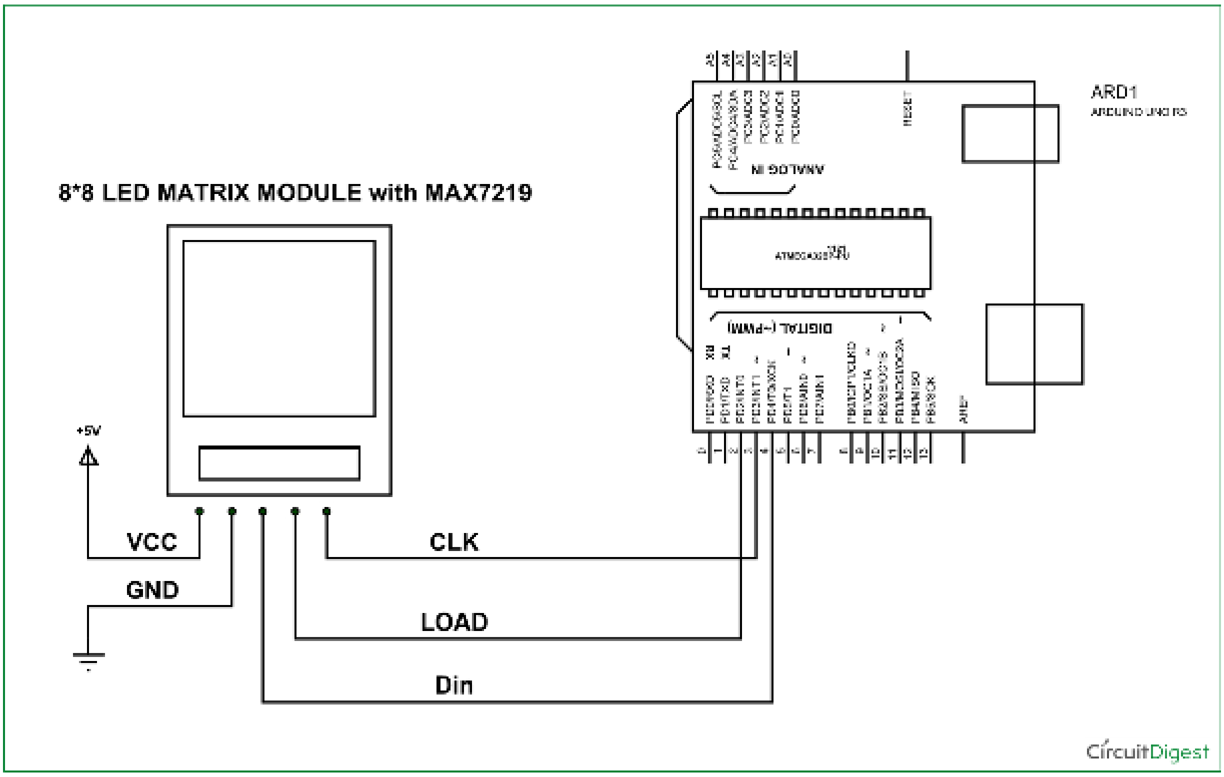
**Software:** Arduino Nightly

## Circuit Diagram and Explanation

The connections which are done between **Arduino Uno and LED matrix module** is shown in below table.

PIN2 ------------------LOAD or CHIPSELECT of LED module
PIN3------------------CLOCK of LED module
PIN4 ------------------DATAIN of LED module
+5V ------------------VCC of LED module
GND ------------------GND of LED module

The **circuit diagram of 8*8 LED matrix display** is shown in below figure.



Now for using the special futures called by installing new library, we need to establish few commands in program and are stated below.

```
#include "LedControlMS.h"
#define NBR_MTX 1
LedControl lc=LedControl(4,3,2, NBR_MTX);
lc.writeString(0,"CIRCUITDIGEST");
lc.clearAll();
```

First we need to call the header file for interfacing a LED matrix to Arduino Uno. That is" #include "LedControlMS.h"", this header file call the library special functions.

We have a feature with these modules we can connect many number of modules in series and program them together as a single display. This feature comes in handy when we need a display unit which could display multiple characters at a time. So we need to tell the controller how many displays we are connecting.

In this module there are mainly three pins; data flow from UNO to module takes places with these three pins. The pins are DATAIN (data receiving pin), CLOCK (clock pin), and CHIPSELECT (command receiving pin).

Now we need to tell the UNO where we are connecting these pins. This is done by command "LedControl lc=LedControl(4,3,2, NBR_MTX); ". "lc.writeString(0,"CIRCUITDIGEST");", this command is used for telling UNO which characters are to be displayed on the LED matrix. With the above the display shows" CIRCUITDIGEST", with each character once.

We need to clear the display chip memory before sending any other data, this is done by command " lc.clearAll();".

By this way we can easily interface a 8x8 LED matrix to Arduino Uno.

Code:

```
#include "LedControlMS.h"
 //pin 4 is connected to the DataIn
// pin 3 is connected to the CLK
 //pin 2 is connected to LOAD
 #define NBR_MTX 1 //number of matrices attached is one
LedControl lc=LedControl(4,3,2, NBR_MTX);//

void setup()
{
  for (int i=0; i< NBR_MTX; i++)
  {
    lc.shutdown(i,false);
  /* Set the brightness to a medium values */
    lc.setIntensity(i,8);
  /* and clear the display */
    lc.clearDisplay(i);
                delay(100);
  }
}

void loop()
{
                  lc.writeString(0,"CIRCUITDIGEST");//sending characters to display
                  lc.clearAll();//clearing the display
                  delay(1000);
}
```

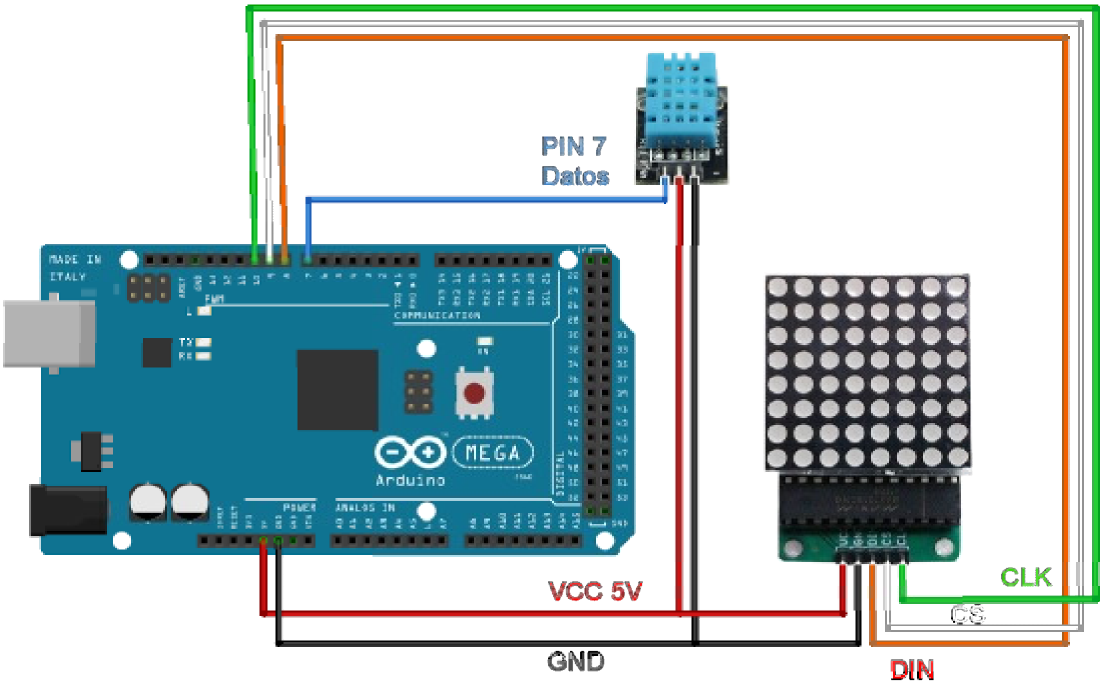
# Módulo de Matriz de LED y Sensor DHT11

Anteriormente hemos hablado del módulo de sensor DHT11 (Puedes verlo en este enlace) y hoy vamos a usar este sensor para visualizar los datos que nos proporciona en un módulo de matriz de LED de 8x8. Para ello vamos a usar el Módulo de Matriz MAX7219

## Elementos Necesarios

***1x Placa Arduino (En este caso usaremos un Arduino Mega 2560)***
***1x Módulo de Sensor de Temperatura DHT11***
***1x Módulo de Matriz de LED MAX7219***
***8x Conectores Jumper Macho-Hembra***

## Esquema Eléctrico



**NOTA: Los pines del módulo de sensor de temperatura pueden variar según el distribuidor del módulo. En mi caso S es el pin de datos - es el pin de GND y el restante es VCC. ¡Siempre leer las letras escritas en la placa antes de conectar!**

## Códigos y Librerías

**Descargar la librería DHT.h: Enlace a MEGA**
**Descargar la librería MaxMatrix.h: Enlace de DESCARGA**

**Código Modificado para usar:**
*el Pin 7 para el sensor DHT11*
*el Pin 8 para DIN en la matriz*
*el Pin 9 para CS en la matriz*
*el Pin 10 para CLK en la matriz*

```
//ARDUINOESP.BLOGSPOT.COM
//Basado en el proyecto de ARDUINOARTS.com y modificado en Español y para mostrar la
Humedad -

#include <MaxMatrix.h>
#include <avr/pgmspace.h>
#include <stdlib.h>
#include "DHT.h"
#define DHTPIN 7     // Pin al que esta conectado el DHT11
#define DHTTYPE DHT11   // Tipo de sensor en este caso DHT11
DHT dht(DHTPIN, DHTTYPE);

PROGMEM unsigned char const CH[] = {
3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space
1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !
3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "
5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #
4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $
5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %
5, 8, B00110110, B01001001, B01010110, B00100000, B01010000, // &
1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '
3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (
3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )
5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *
5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +
2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,
4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -
2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .
4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0
3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1
4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2
4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3
4, 8, B00011000, B00010100, B00010010, B01111111, B00000000, // 4
4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
2, 8, B01010000, B00000000, B00000000, B00000000, B00000000, // :
2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;
3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <
3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =
3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >
4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?
5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @
4, 8, B01111110, B00010001, B00010001, B01111110, B00000000, // A
4, 8, B01111111, B01001001, B01001001, B00110110, B00000000, // B
4, 8, B00111110, B01000001, B01000001, B00100010, B00000000, // C
4, 8, B01111111, B01000001, B01000001, B00111110, B00000000, // D
4, 8, B01111111, B01001001, B01001001, B01000001, B00000000, // E
4, 8, B01111111, B00001001, B00001001, B00000001, B00000000, // F
4, 8, B00111110, B01000001, B01001001, B01111010, B00000000, // G
```

```
4, 8, B01111111, B00001000, B00001000, B01111111, B00000000, // H
3, 8, B01000001, B01111111, B01000001, B00000000, B00000000, // I
4, 8, B00110000, B01000000, B01000001, B00111111, B00000000, // J
4, 8, B01111111, B00001000, B00010100, B01100011, B00000000, // K
4, 8, B01111111, B01000000, B01000000, B01000000, B00000000, // L
5, 8, B01111111, B00000010, B00001100, B00000010, B01111111, // M
5, 8, B01111111, B00000100, B00001000, B00010000, B01111111, // N
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // O
4, 8, B01111111, B00001001, B00001001, B00000110, B00000000, // P
4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q
4, 8, B01111111, B00001001, B00001001, B01110110, B00000000, // R
4, 8, B01000110, B01001001, B01001001, B00110010, B00000000, // S
5, 8, B00000001, B00000001, B01111111, B00000001, B00000001, // T
4, 8, B00111111, B01000000, B01000000, B00111111, B00000000, // U
5, 8, B00001111, B00110000, B01000000, B00110000, B00001111, // V
5, 8, B00111111, B01000000, B00111000, B01000000, B00111111, // W
5, 8, B01100011, B00010100, B00001000, B00010100, B01100011, // X
5, 8, B00000111, B00001000, B01110000, B00001000, B00000111, // Y
4, 8, B01100001, B01010001, B01001001, B01000111, B00000000, // Z
2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [
4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backslash
2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, // ]
3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat
4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _
2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `
4, 8, B00100000, B01010100, B01010100, B01111000, B00000000, // a
4, 8, B01111111, B01000100, B01000100, B00111000, B00000000, // b
4, 8, B00111000, B01000100, B01000100, B00101000, B00000000, // c
4, 8, B00111000, B01000100, B01000100, B01111111, B00000000, // d
4, 8, B00111000, B01010100, B01010100, B00011000, B00000000, // e
3, 8, B00000100, B01111110, B00000101, B00000000, B00000000, // f
4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g
4, 8, B01111111, B00000100, B00000100, B01111000, B00000000, // h
3, 8, B01000100, B01111101, B01000000, B00000000, B00000000, // i
4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j
4, 8, B01111111, B00010000, B00101000, B01000100, B00000000, // k
3, 8, B01000001, B01111111, B01000000, B00000000, B00000000, // l
5, 8, B01111100, B00000100, B01111100, B00000100, B01111000, // m
4, 8, B01111100, B00000100, B00000100, B01111000, B00000000, // n
4, 8, B00111000, B01000100, B01000100, B00111000, B00000000, // o
4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p
4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q
4, 8, B01111100, B00001000, B00000100, B00000100, B00000000, // r
4, 8, B01001000, B01010100, B01010100, B00100100, B00000000, // s
3, 8, B00000100, B00111111, B01000100, B00000000, B00000000, // t
4, 8, B00111100, B01000000, B01000000, B01111100, B00000000, // u
5, 8, B00011100, B00100000, B01000000, B00100000, B00011100, // v
5, 8, B00111100, B01000000, B00111100, B01000000, B00111100, // w
5, 8, B01000100, B00101000, B00010000, B00101000, B01000100, // x
4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y
3, 8, B01100100, B01010100, B01001100, B00000000, B00000000, // z
3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {
1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |
3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};

int data = 8;    // DIN pin del módulo MAX7219
int load = 9;    // CS pin del módulo MAX7219
int clock = 10;  // CLK pin del módulo MAX7219
int maxInUse = 1;    // Cuantos módulos MAX7219 se estan usando
MaxMatrix m(data, load, clock, maxInUse);
byte buffer[10];

void setup(){
  pinMode(2,INPUT); // Entrada de Botón
  m.init();
  m.setIntensity(15); // Intensidad luminosa de la matriz
  Serial.begin(3600);
  Serial.println("Iniciando sensor");
  dht.begin();
}

void loop(){

  printStringWithShift(" ArduinoESP.blogspot.com ", 75);
  int t = dht.readTemperature();
  char temp[4];
  itoa(t,temp,10);
  Serial.println(temp);
  printStringWithShift("Temperatura: ", 100);
  printStringWithShift(temp, 100);
  printStringWithShift("C ", 100);
  delay(100);
  m.shiftLeft(false, true);
  int h = dht.readHumidity();
  char hum[4];
  itoa(h,hum,10);
  Serial.println(hum);
  printStringWithShift(" Humedad: ", 100);
  printStringWithShift(hum, 100);
  printStringWithShift("% ", 100);
  delay(100);
  m.shiftLeft(false, true);
  }


void printCharWithShift(char c, int shift_speed){
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(32, 0, buffer);
  m.setColumn(32 + buffer[0], 0);

  for (int i=0; i<buffer[0]+1; i++)
  {
    delay(shift_speed);
    m.shiftLeft(false, false);
  }
}

void printStringWithShift(char* s, int shift_speed){
  while (*s != 0){
    printCharWithShift(*s, shift_speed);
    s++;
  }
}

void printString(char* s)
{
  int col = 0;
  while (*s != 0)
  {
    if (*s < 32) continue;
    char c = *s - 32;
    memcpy_P(buffer, CH + 7*c, 7);
    m.writeSprite(col, 0, buffer);
    m.setColumn(col + buffer[0], 0);
    col += buffer[0] + 1;
    s++;
  }
}
```

## Resultado Final

### CIRCUITO MONTADO