8x8 Led Matrix Project Report

Team Members

1. **Mohamed Gamal**
ID: 5464
E-mail: gemy_1998@hotmail.com

2. **Verginia Ehab**
ID: 5320
E-mail:gina.lovelygi@gmail.com

3. **Omar Radwan**
ID: 6013
E-mail:omarradwanx@gmail.com

4. **Youssef Hussein**
ID:5659
E-mail:Youssef.hussein42@gmail.com

5. **Ahmed Ibrahim**
ID: 5326
E-mail:a7mad_i_elsayed10@gmail.com

Overview:

The led matrix is widely used all over the world for advertisement boards or public service announcements boards and many other applications. The main idea behind led matrices is the control of individual leds to make shapes or alphabets. This can be implemented by shift registers or a MAX7219 module.

The MAX7219 is a driver capable of alternating 64 individual LEDs using only 3 inputs from the arduino. We can also chain multiple MAX7219 modules with their 8x8 matrices using the same 3 inputs.

The HC-05 bluetooth module was used to change the text and the scrolling speed of the LEDs without having to change a single line in the code and uploading to the arduino.

The maximum number of LEDs that light up at the same time is eight. The LEDs are arranged as 8×8 set of rows and columns. So the MAX7219 activates each column for an unnoticeable period of time and at the same time it also controls each row. So by rapidly switching through the columns and rows the human eye will not notice the blinking.

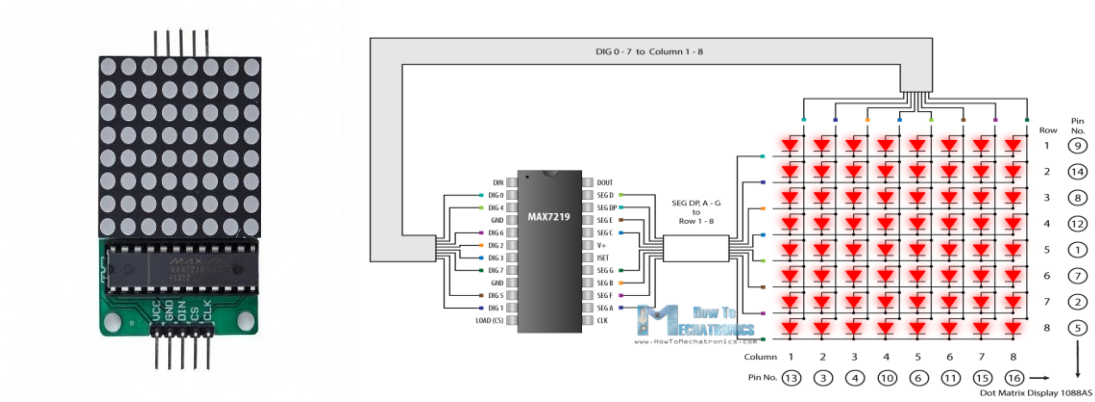
2 PCBS were used to separate the MAX7219 modules (backend) and the Matrices (frontend).

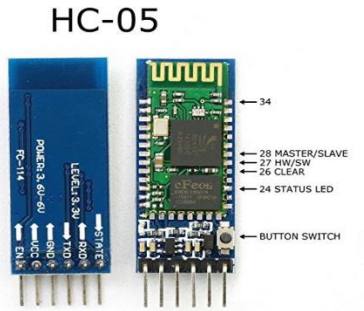
The code was hugely inspired by the MD_Parola library created by MajicDesigns.

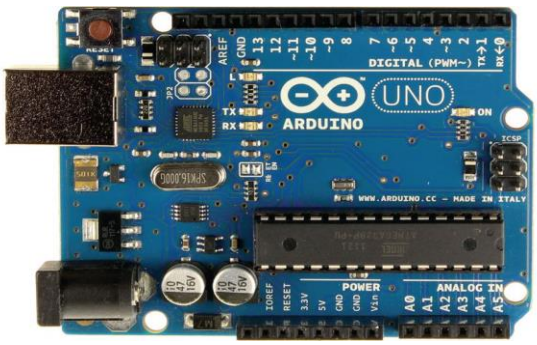
Schematics & Designs

Used Components

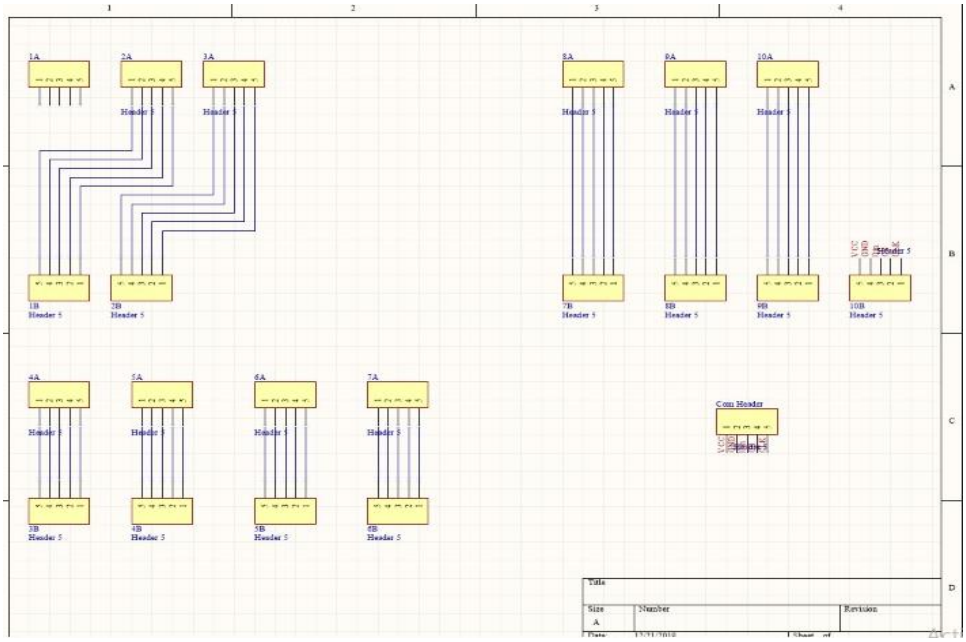
1. max7219 8x8 matrix modules
2. hc05 bluetooth module
3. Arduino board (uno)
4. 1 pcb 10x10 board, 2 pcb 20x20 boards



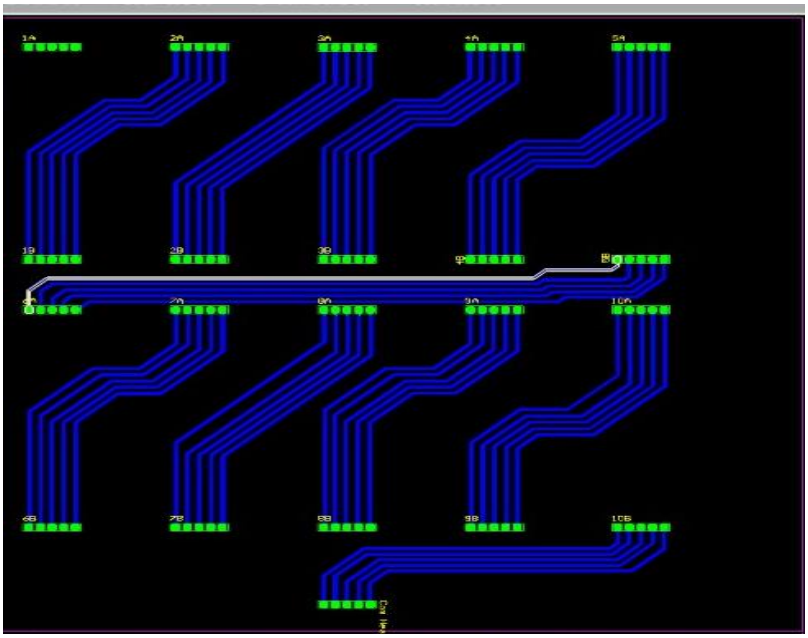




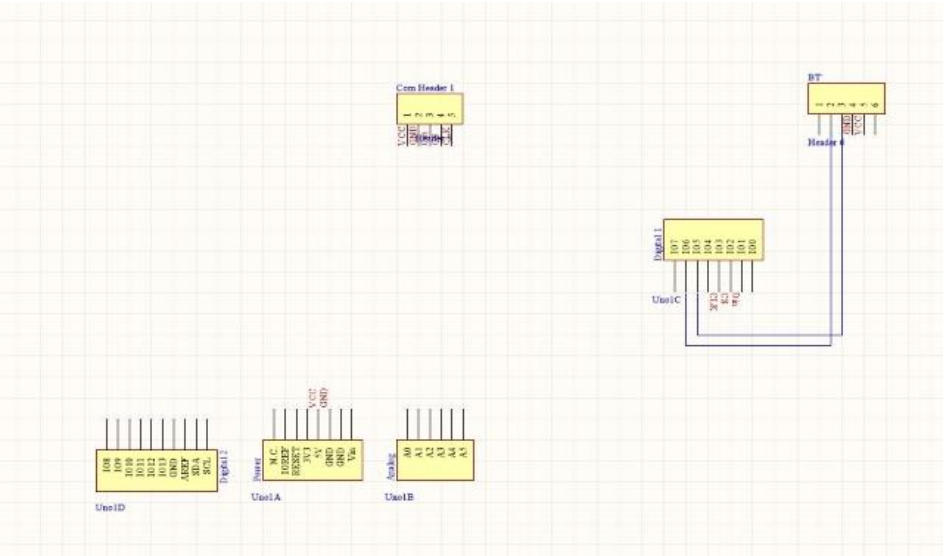
Modules Schematic



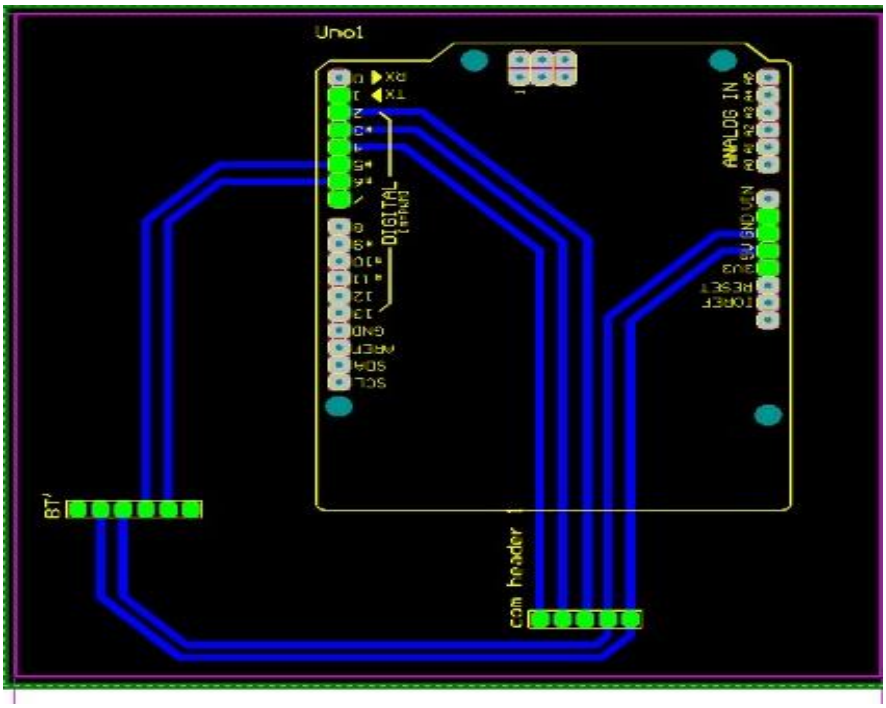
Modules PCB Design



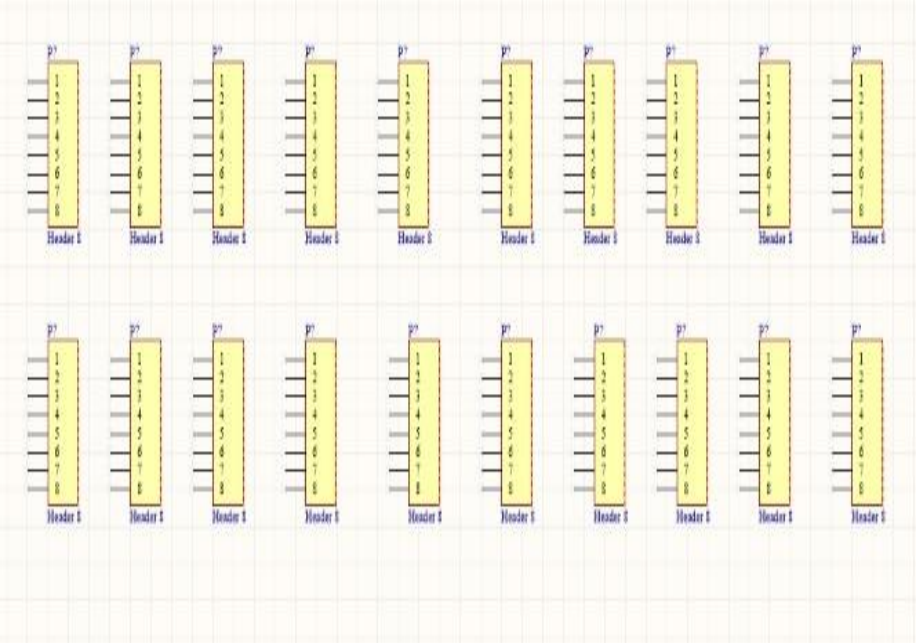
Arduino Board Schematic



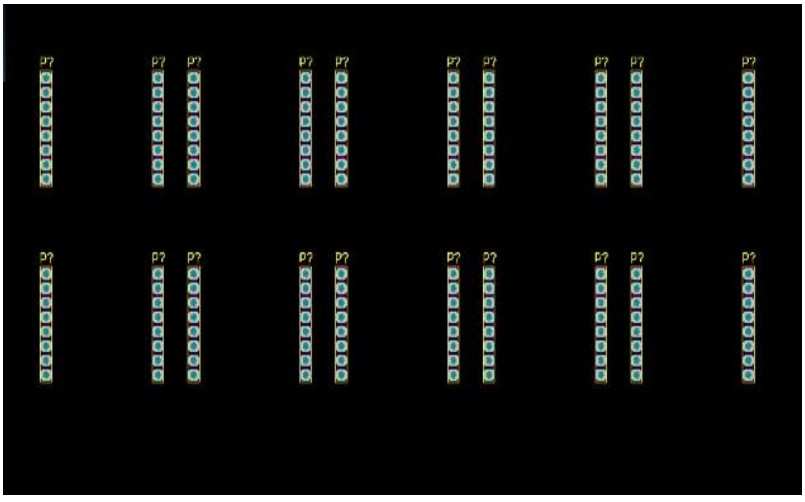
Arduino Board PCB Design



Matrices Schematic



Matrices PCB Design

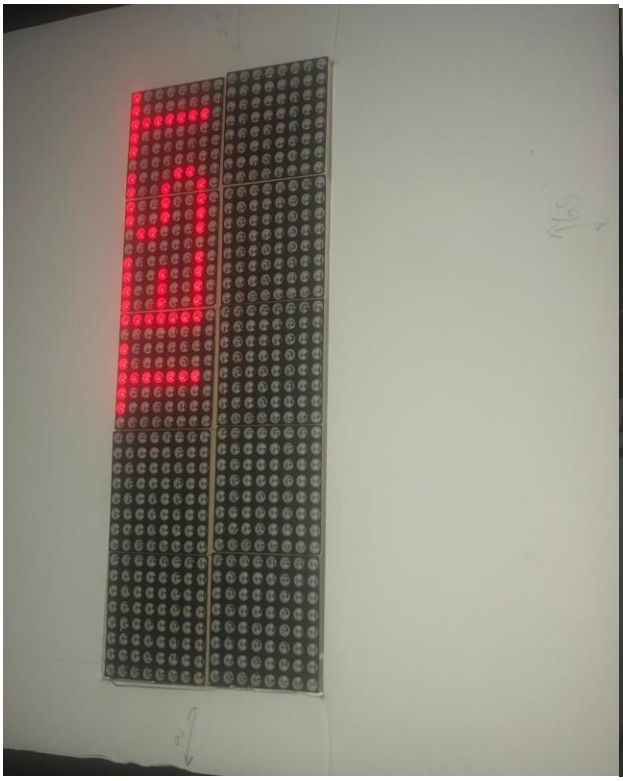


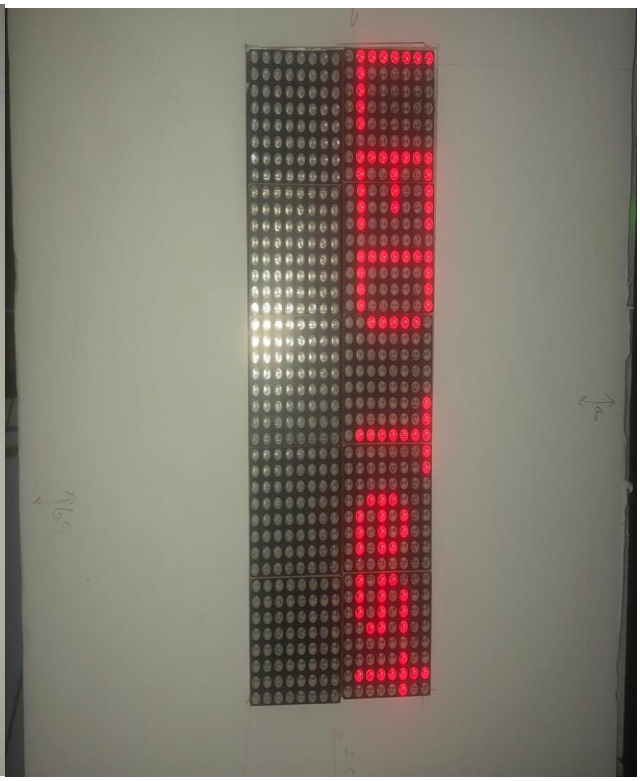
These designs and schematic were created by Altium Pcb design software

Pins used in Arduino

1. V_{cc} and Ground pins
2. Pin 2: $Module_0$ D_{in}
3. Pin 3: $Module_0$ cs
4. Pin 4: $Module_0$ clk
5. Pin 5: Tx of the Bluetooth module
6. Pin 6: Rx of the Bluetooth module

Pictures of the project





Commands:

1. /mode_append: default writing mode, when new message is written it's appended to the already written message on the screen.
2. /mode_clear: when a new message it replaces the message which is already written on the screen.
3. /clear: removes the message written on the screen.
4. /invert: switches between active low and active high mode.
5. /scroll: starts the scrolling mode (if the message is static).
6. /hold: stops the scrolling mode (if the message is scrolling).
7. /left: makes scrolling direction to the left.
8. /right: makes scrolling direction to the right.
9. /si: increases the scrolling speed (makes scrolling faster).
10. /sd: decreases the scrolling speed (makes scrolling slower).
11. Typing text without "/" in the beginning won't be considered as command and message will be shown on the screen

Android App Used

Name: Android Bluetooth control

Link: https://play.google.com/store/apps/details?id=com.broxcode.arduinobluetoothfree&hl=en

you can use any other Arduino Bluetooth control app as long as it contains terminal

Youtube Link:

https://www.youtube.com/watch?v=2X2ARv1Ss_g&feature=youtu.be

References:

https://github.com/MajicDesigns/MD_Parola
https://majicdesigns.github.io/MD_Parola/class_m_d___parola.html

The Code

```
#include <SoftwareSerial.h>
#include <MD_Parola.h>
#include <MD_MAX72xx.h>
#include <SPI.h>

SoftwareSerial mySerial(5, 6); //(RXArduino=TxBluetooth), (TXArduino=RXBluetooth)

#define HARDWARE_TYPE MD_MAX72XX::GENERIC_HW
#define MAX_DEVICES 10

#define CLK_PIN   4
#define DATA_PIN  2
#define CS_PIN    3

// code variables

boolean staticWord = true;
char  message [100] = "TEXT!!";
int messageEnd =0;
String data = "";
int mode = 0 ; // type 0 append, type 1 clear and write new
boolean invertState = false;

uint8_t scrollSpeed = 40;    // default frame delay value
textEffect_t scrollEffect = PA_SCROLL_LEFT;
textPosition_t scrollAlign = PA_RIGHT;
uint16_t scrollPause = 10; // in milliseconds

MD_Parola P = MD_Parola(HARDWARE_TYPE, DATA_PIN, CLK_PIN, CS_PIN, MAX_DEVICES);


void setup(void)
{

 P.begin();
 P.setIntensity(0);
 mySerial.begin(9600);
 Serial.begin(9600);
   P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);
   while (message[messageEnd]!='\0'){
     messageEnd++;
   }
 Serial.print(messageEnd);
}


void loop(void)
{
 if (!staticWord){
      if (P.displayAnimate())
 {
   P.displayReset();
 }
 }
 else {
   P.print(message);
 }
      bluetoothRead();
}
```

```
void bluetoothRead() {
 char x ;
 if (mySerial.available()) {
  x = mySerial.read();
  if (x == '/') {
   parseCommand();
  }
  else {
   parseString(x);
  }
 }


}

void parseCommand() {
 Serial.print("command start"); // debug

 String command = "" ;

 while (mySerial.available()) {
  char x = mySerial.read();
  command += x;
 }

 if (command.equals("clear")) {
   message[0] = '\0';
   messageEnd = 0 ;

  P.displayClear();
  P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);

  Serial.print("clear command"); // debug
 }

 else if (command.equals("mode_append")) {
  mode = 0;
 }

 else if (command.equals("mode_clear")) {
  mode = 1 ;
 }

 else if (command.equals("invert")) {
  invertState = !invertState;
  P.setInvert(invertState);
 }

 else if (command.equals("scroll")) {
  staticWord = false;
  scrollAlign = PA_LEFT;

   P.displayClear();
  P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);

 }
 else if (command.equals("hold")){
      staticWord = true;
    scrollAlign = PA_RIGHT;
    P.displayClear();
    P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);
 }

 else if (command.equals("si")){
   if (scrollSpeed<=21){
```

```
    scrollSpeed = 1;
   }
   else {
       scrollSpeed-=20;
   }

   P.setSpeed(scrollSpeed);
   Serial.print(scrollSpeed);
 }
  else if (command.equals("sd")){
   if (scrollSpeed>210){
    scrollSpeed = 210;
   }
   scrollSpeed+=20;
   P.setSpeed(scrollSpeed);
 }

 else if (command.equals("left")){
    scrollEffect = PA_SCROLL_LEFT;
       P.displayClear();
    P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);
 }

  else if (command.equals("right")){
     scrollEffect = PA_SCROLL_RIGHT;
       P.displayClear();
    P.displayText(message, scrollAlign, scrollSpeed, scrollPause, scrollEffect, scrollEffect);
 }

}

void parseString(char firstChar) {
 String tmp = "";
 tmp += firstChar;

 while (mySerial.available() > 0) {
  char x = mySerial.read();
  tmp += x;
 }

 if (mode == 0) {
  int i = 0;
  while (messageEnd<99&&i<tmp.length()){
       message[messageEnd++] = tmp.charAt(i++);
  }
  message[messageEnd] = '\0';
 }

 else if (mode == 1) {
  int i  = messageEnd = 0;

  while (messageEnd<99&&i<tmp.length()){
       message[messageEnd++] = tmp.charAt(i++);
  }
  message[messageEnd] = '\0';
 }
}
```