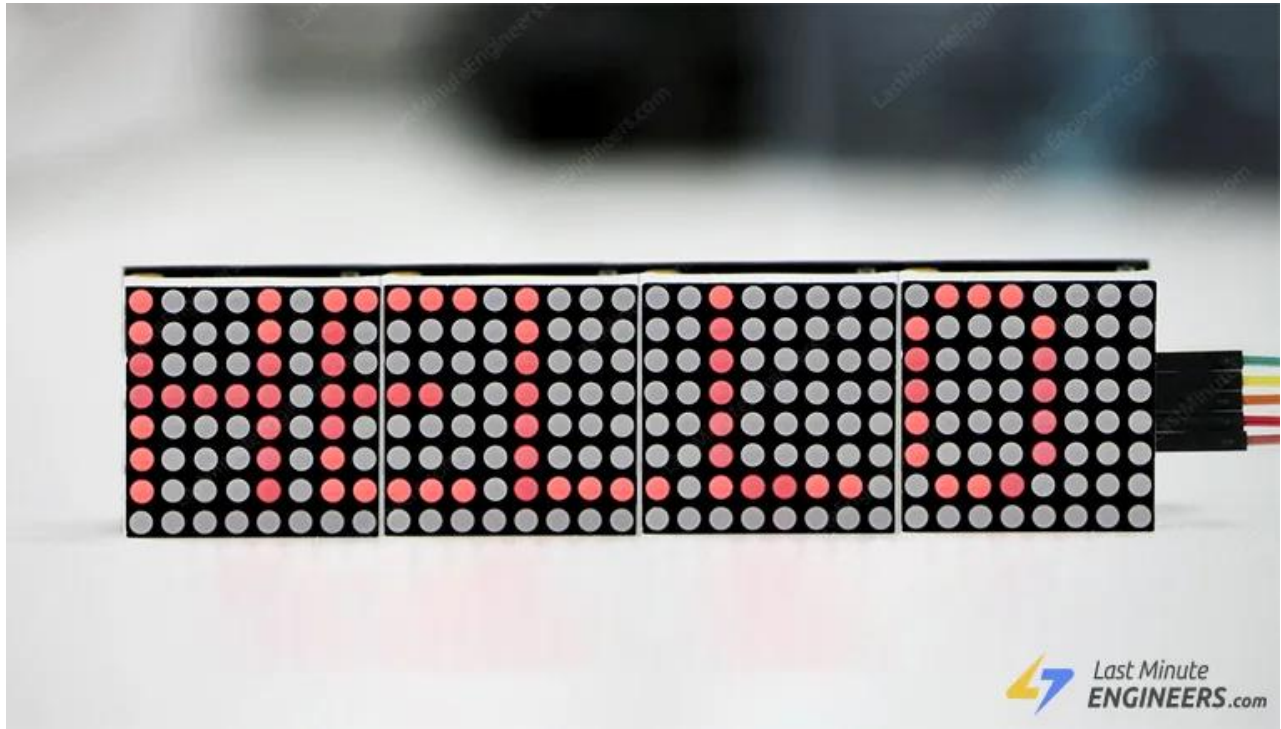


MATRIX DISPLAY



Dot matrix displays are something that all Arduino enthusiasts come across at some point. These displays are so popular that almost all modern outdoor LED displays use them to display characters, symbols, and images.

When it comes to controlling dot-matrix displays, there is hardly a better option than the MAX7219. It can easily control a single dot matrix and, for larger projects, it can be chained together to control two or more dot matrices.

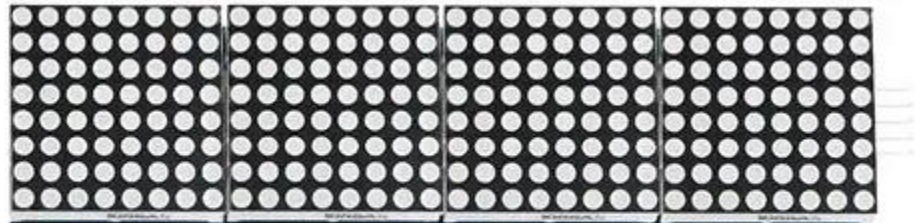
All in all, they are a lot of fun and quite useful as well, so let's get started.

MAX7219 MODULE OVERVIEW

There are several MAX7219 breakout boards available, two of which are more popular: the generic module and the FC-16 module.



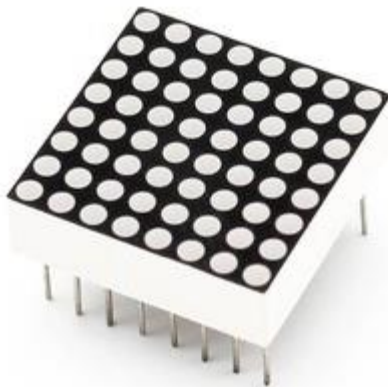
Generic Module



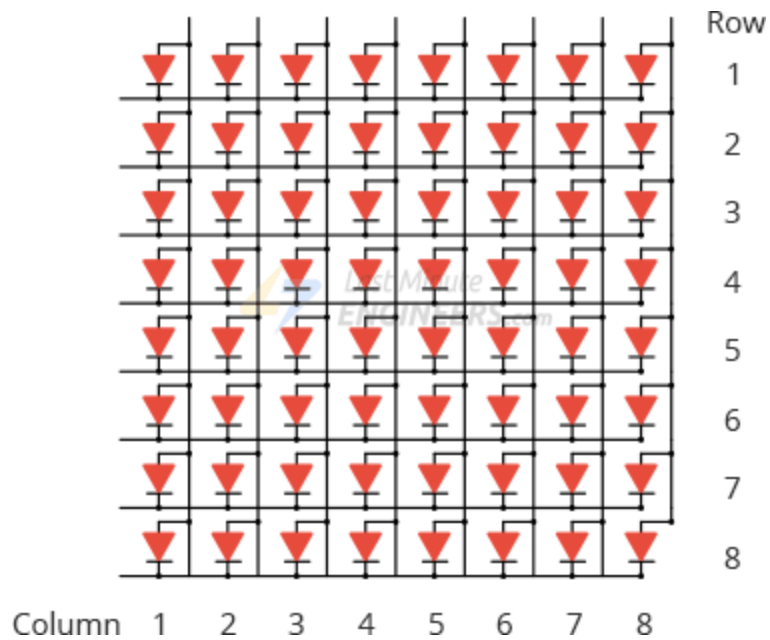
FC-16 Module

A typical MAX7219 module includes an 8×8 dot matrix display and a MAX7219 LED display driver. Let's get familiar with them.

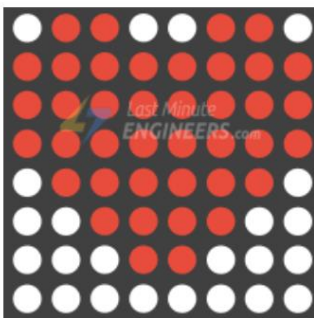
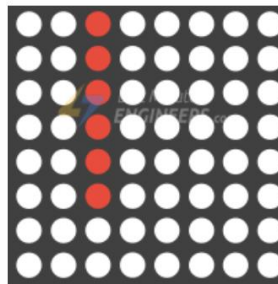
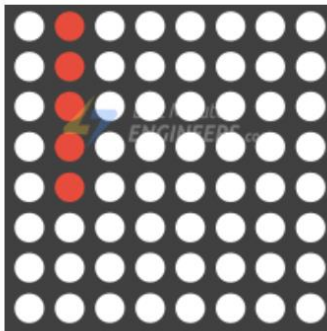
THE DOT MATRIX DISPLAY



An 8×8 dot matrix display typically has 16 pins, 8 for each row and 8 for each column. All rows and columns are wired together in order to reduce the number of pins. If this were not the case, an 8×8 dot matrix display would require 65 pins, one for each LED and one for a common anode or common cathode connector. By connecting rows and columns, only 16 pins are needed to control the entire matrix. This technique of controlling a large number of LEDs with fewer pins is referred to as Multiplexing.



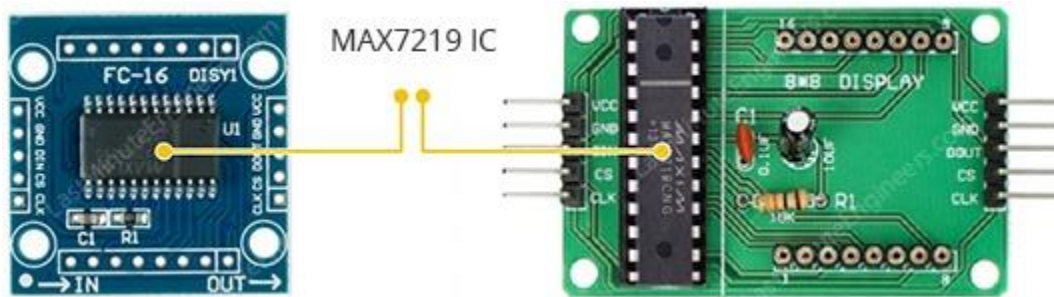
In this technique, each column is activated for a very short time, and at the same time, the LEDs on that column are lit by addressing the corresponding row. As a result, no more than eight LEDs are lit at the same time. The columns are switched so fast (hundreds or thousands of times per second) that the human eye perceives the display as fully lit.



MAX7219 CHIP

The only problem with multiplexing is that you have to refresh the display all the time to keep the image stable.

Then there's the MAX7219 Chip, which handles all of the control and refresh work for you. All you have to do is send it serial commands through the 4-pin SPI interface, and it will take care of the rest.



It can fully control 64 individual LEDs while keeping their brightness constant. Once the microcontroller has updated the display, the MAX7219 handles the work of refreshing the display at 800 Hz. This frees up the microcontroller to do other important things.

The MAX7219 has a power saving mode in which the display can be turned off to save power. It also turns off the LEDs during startup, preventing wacky displays for the first few seconds of operation.

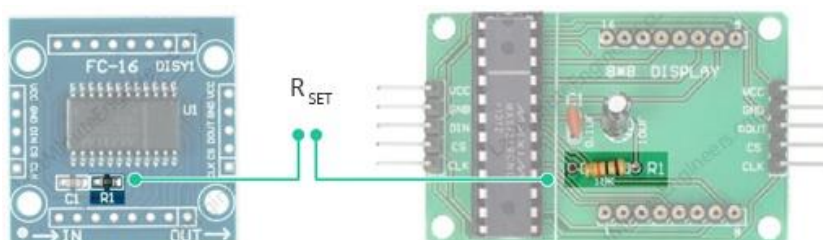
The MAX7219 communicates via the SPI interface, so it only needs 3 data pins to connect to a microcontroller. In addition, we can daisy-chain multiple modules together for a larger display using the same 3 wires.

SETTING MAXIMUM CURRENT AND BRIGHTNESS

The MAX7219 allows you to adjust the brightness of the display using either hardware or software (or both).

At the Hardware Level

The MAX7219 breakout board includes a resistor (R_{SET}) for adjusting the brightness at the hardware level.



This resistor controls the maximum current supplied to the LEDs and, hence, the overall brightness of the display.

The table below shows the resistor values you should use based on the voltage and forward current of your LED matrix. A 2V 20 mA LED, for instance, would require a 28k Ω resistor.

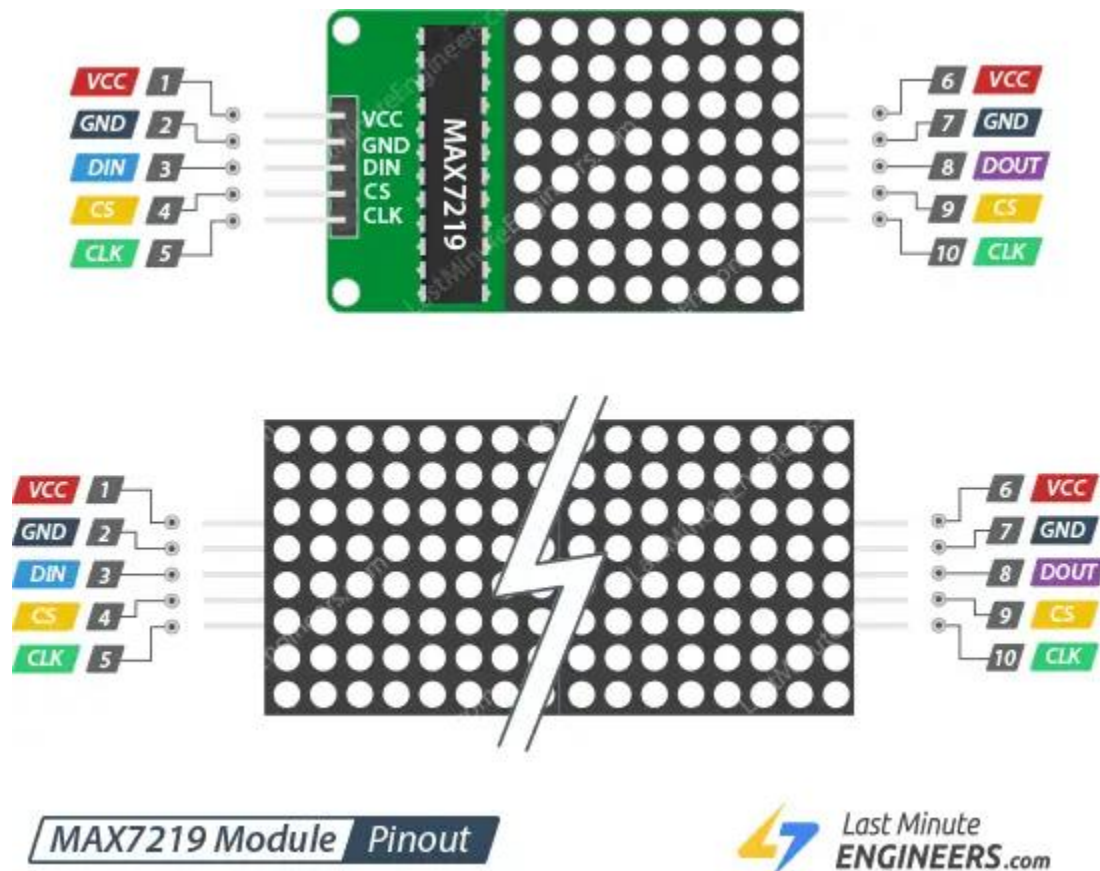
I_{SEG} (mA)	V_{LED} (V)				
	1.5	2.0	2.5	3.0	3.5
40	12.2	11.8	11.0	10.6	9.69
30	17.8	17.1	15.8	15.0	14.0
20	29.8	28.0	25.9	24.5	22.6
10	66.7	63.7	59.3	55.4	51.2

At the Software Level

We'll discuss adjusting brightness using software later in this tutorial.

MAX7219 MODULE PINOUT

Regardless of which variant you select, the module will have two connectors.



INPUT CONNECTOR

The breakout pins on one end of the module are used to communicate with the microcontroller.

VCC is connected to 5V. Because the display draws a lot of current (up to 1A at maximum brightness), it's best to use an external power supply instead of the Arduino's 5V supply. If you want to use the Arduino's 5V supply, keep the brightness below 25% to avoid overheating the voltage regulator.

GND is the common ground pin.

DIN is the Data pin. Connect it to any digital pin of the microcontroller.

CS/LOAD is Chip Select (sometimes labeled as LOAD). Connect it to any digital pin of the microcontroller.

CLK stands for Clock pin. Connect it to any digital pin of the microcontroller.

OUTPUT CONNECTOR

The breakout pins on the other end of the module are used to daisy-chain displays.

VCC connects to 5V on the next module.

GND connects to GND on the next module.

DOUT is Data Out and connects to the DIN pin of the next module.

CS/LOAD connects to CS / LOAD on the next module.

CLK connects to CLK on the next module.

WIRING MAX7219 MODULE WITH ARDUINO UNO

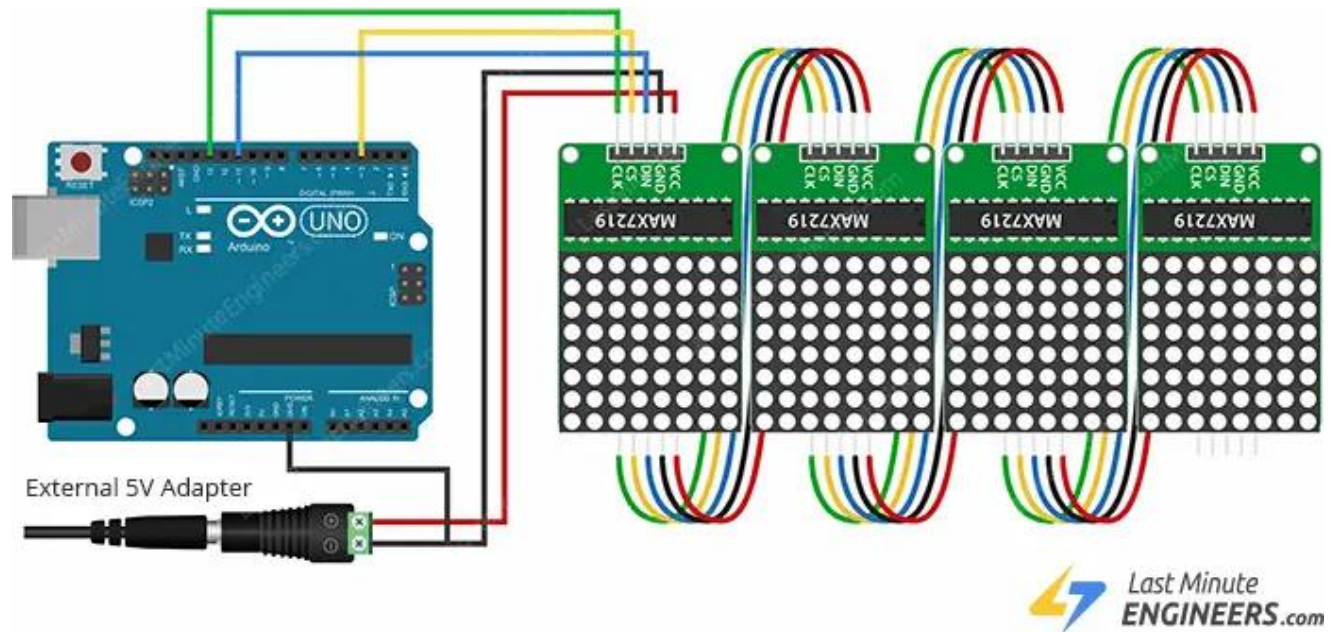
Now that we know everything about the module, we can start hooking it up to our Arduino!

Let's start with the module's power supply connections. Because the display consumes a lot of current, we'll use an external power supply instead of the Arduino board's 5V supply. If you are only using a single MAX7219 module, you can power it directly from the Arduino, but you should avoid doing so if possible.

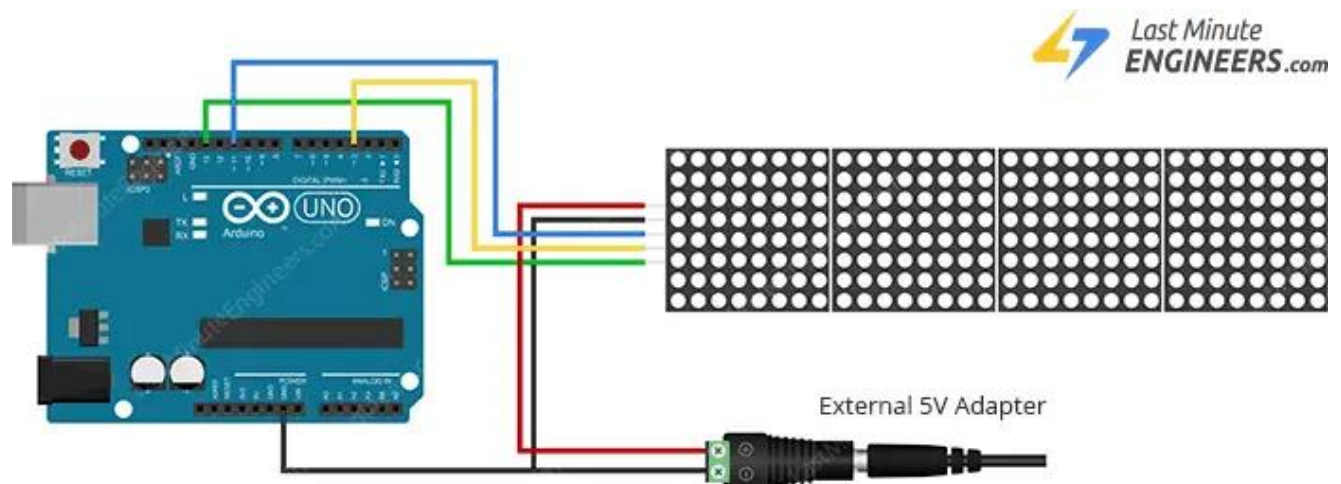
Let's wire up the SPI pins. Note that each Arduino board has a unique set of SPI pins that must be connected accordingly. For Arduino boards such as the UNO/Nano V3.0, these pins are digital 13 (SCK), 12 (MISO), 11 (MOSI), and 10 (SS).

If you're using a different Arduino board, check the official documentation for SPI pin locations before proceeding.

Here is the wiring for the Generic MAX7219 Module:



Here is the wiring for the FC-16 MAX7219 Module



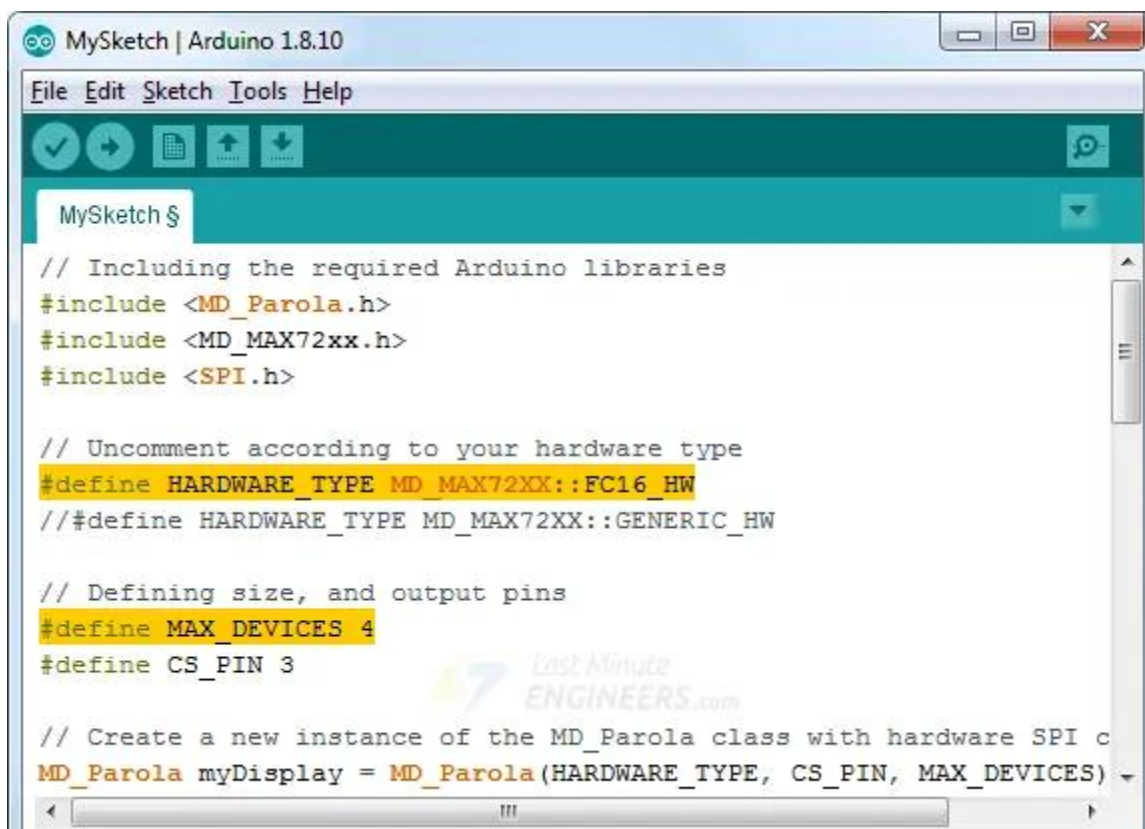
If you want to daisy-chain multiple displays to make a larger display, connect the DOUT of the first display to the DIN of the next display. VCC, GND, CLK, and CS will all be shared between displays.

Once your module is connected to the Arduino, it's time to write some code!

Code

Our first experiment involves displaying a simple text without any animation.

But, before you upload the sketch, you must modify the following two variables.

A screenshot of the Arduino IDE window titled "MySketch | Arduino 1.8.10". The window shows a sketch with the following code:

```
// Including the required Arduino libraries
#include <MD_Parola.h>
#include <MD_MAX72xx.h>
#include <SPI.h>

// Uncomment according to your hardware type
#define HARDWARE_TYPE MD_MAX72XX::FC16_HW
// #define HARDWARE_TYPE MD_MAX72XX::GENERIC_HW

// Defining size, and output pins
#define MAX_DEVICES 4
#define CS_PIN 3

// Create a new instance of the MD_Parola class with hardware SPI c
MD_Parola myDisplay = MD_Parola(HARDWARE_TYPE, CS_PIN, MAX_DEVICES)
```

The code is displayed in a text editor with a light blue background. The IDE window has a menu bar with "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checking, running, uploading, and downloading. The sketch is named "MySketch" and is shown in a tab. A watermark for "Last Minute ENGINEERS.com" is visible in the background of the code editor.

REFERENCE:

<https://lastminuteengineers.com/max7219-dot-matrix-arduino-tutorial/>