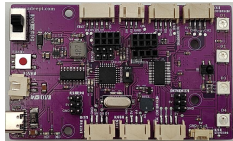




## Lesson 12 How to control the LED matrix

In this lesson, we will learn how to control the LED matrix.

### 12.1 Components used in this course

Components	Quantity	Picture
Adeept Robot Control Board	1	
Type-C USB Cable	1	
LED matrix module	1	

### 12.2 The introduction of the LED matrix

A LED matrix is a rectangular display module that consists of a uniform grid of LEDs. The following is an 8X8 monochrome LED matrix containing 64 LEDs (8 rows by 8 columns).



In order to facilitate the operation and reduce the number of ports required to drive this component, the positive poles of the LEDs in each row and negative poles of the LEDs in each column are respectively connected together inside the LED matrix module, which is called a common anode. There is another arrangement type. Negative poles of the LEDs in each row and the positive poles of the LEDs in each column are respectively connected together, which is called a common cathode. The default address of LED matrix is 0x71.

### The principle of 8\*16 LED matrix:

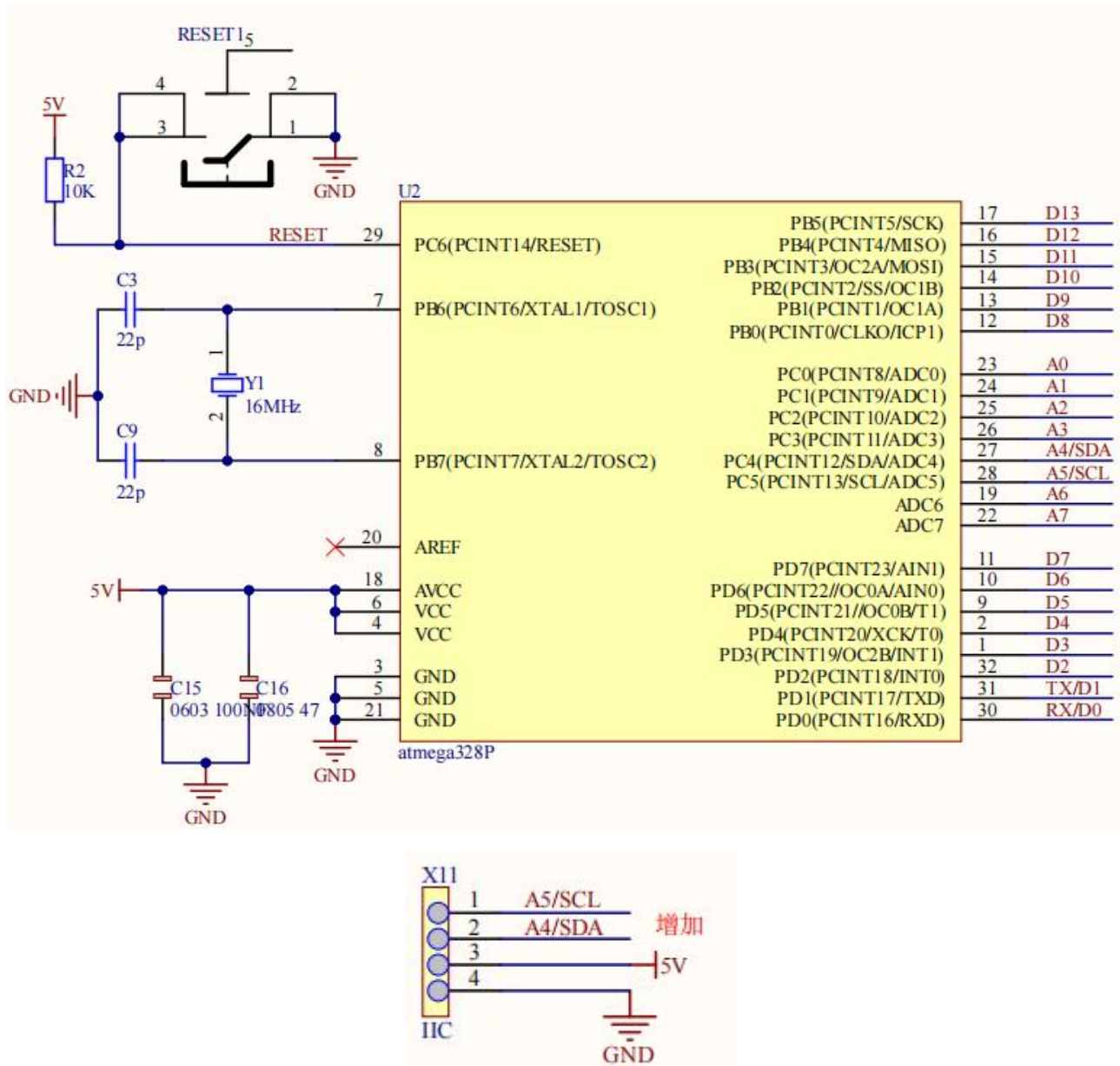
a byte has 8 bits, each bit is 0 or 1. When a bit is 0, turn off LED and when a bit is 1, turn on LED. Thereby, one byte can control the LED in a columns of dot matrix, so 16 bytes can control 16 columns of led lights, that is, 8\*16 dot matrix.

We divide the LED matrix into two sides and display “+” on the left and “o” on the right. As shown below, yellow stands for lit LED while other colors represent the OFF LED.

1 - 8								9 - 16							
			1	1							1	1			
			1	1						1			1		
	1	1	1	1	1	1			1					1	
	1	1	1	1	1	1			1					1	
			1	1						1			1		
			1	1							1	1			

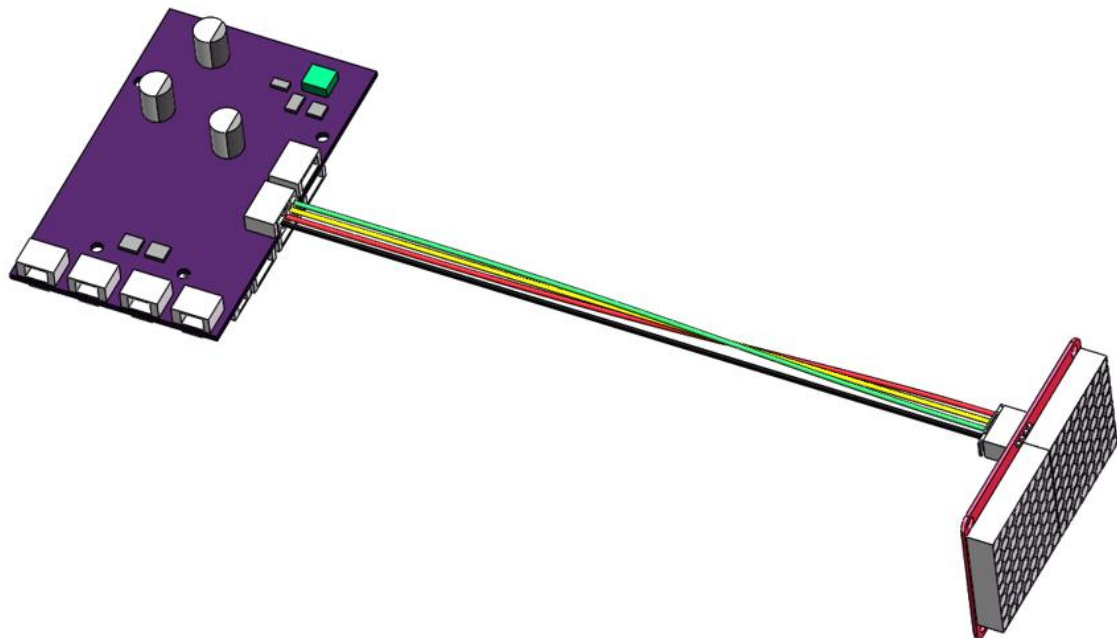
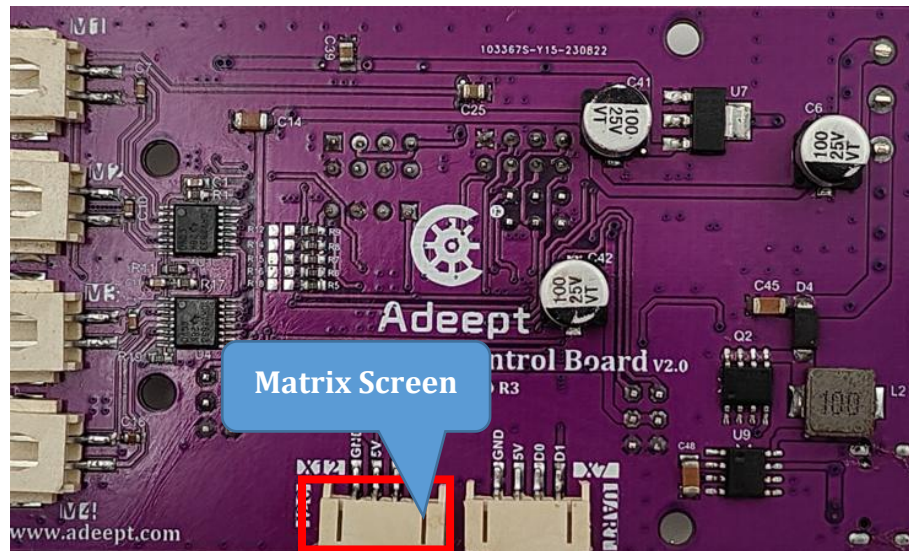
Below, the table on the left corresponds to the "+" above, and the table on the right corresponds to the "o" above.

Row	Binary	Hexadecimal	Row	Binary	Hexadecimal
1	0000 0000	0x00	9	0000 0000	0x00
2	0001 1000	0x18	10	0001 1000	0x18
3	0001 1000	0x18	11	0010 0100	0x24
4	0111 1110	0x7e	12	0100 0010	0x42
5	0111 1110	0x7e	13	0100 0010	0x42
6	0001 1000	0x18	14	0010 0100	0x24
7	0001 1000	0x18	15	0001 1000	0x18
8	0000 0000	0x00	16	0000 0000	0x00



The LED matrix is connected to the red box in the picture.

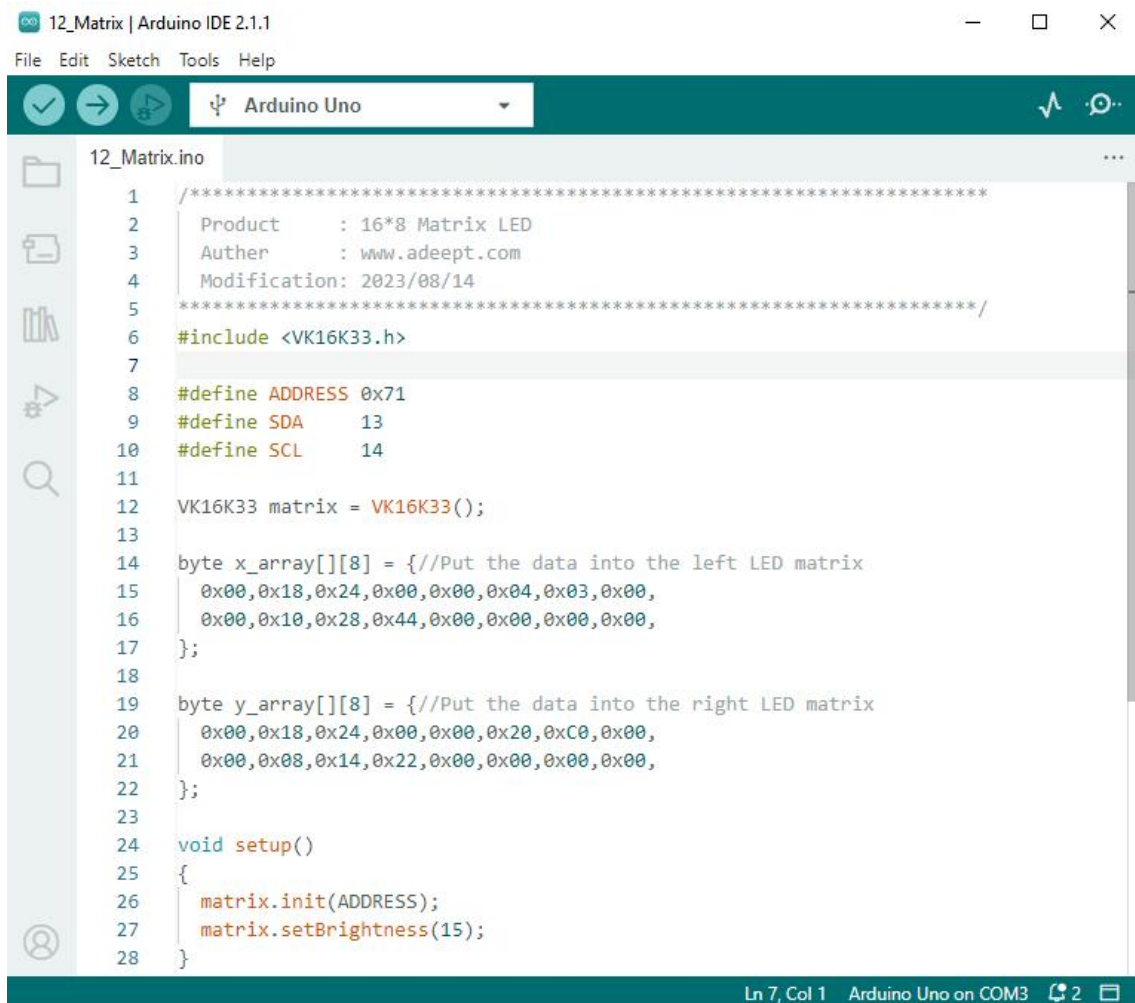
Figure as below:



The LED matrix module uses 4pin cables, the colors are as shown in the picture, and the length is 20CM.

## 12.4 How to control LED matrix

1. Connect your computer and Adeept Robot Control Board with a USB cable.
2. Open “12\_Matrix” folder in “Adeept\_UnoCar-B/Code”, double-click “12\_Matrix.ino”.



```
1  /*****
2   Product      : 16*8 Matrix LED
3   Author       : www.adeept.com
4   Modification : 2023/08/14
5   *****/
6  #include <VK16K33.h>
7
8  #define ADDRESS 0x71
9  #define SDA      13
10 #define SCL      14
11
12 VK16K33 matrix = VK16K33();
13
14 byte x_array[][8] = { //Put the data into the left LED matrix
15   0x00,0x18,0x24,0x00,0x00,0x04,0x03,0x00,
16   0x00,0x10,0x28,0x44,0x00,0x00,0x00,0x00,
17 };
18
19 byte y_array[][8] = { //Put the data into the right LED matrix
20   0x00,0x18,0x24,0x00,0x00,0x20,0xC0,0x00,
21   0x00,0x08,0x14,0x22,0x00,0x00,0x00,0x00,
22 };
23
24 void setup()
25 {
26   matrix.init(ADDRESS);
27   matrix.setBrightness(15);
28 }
```

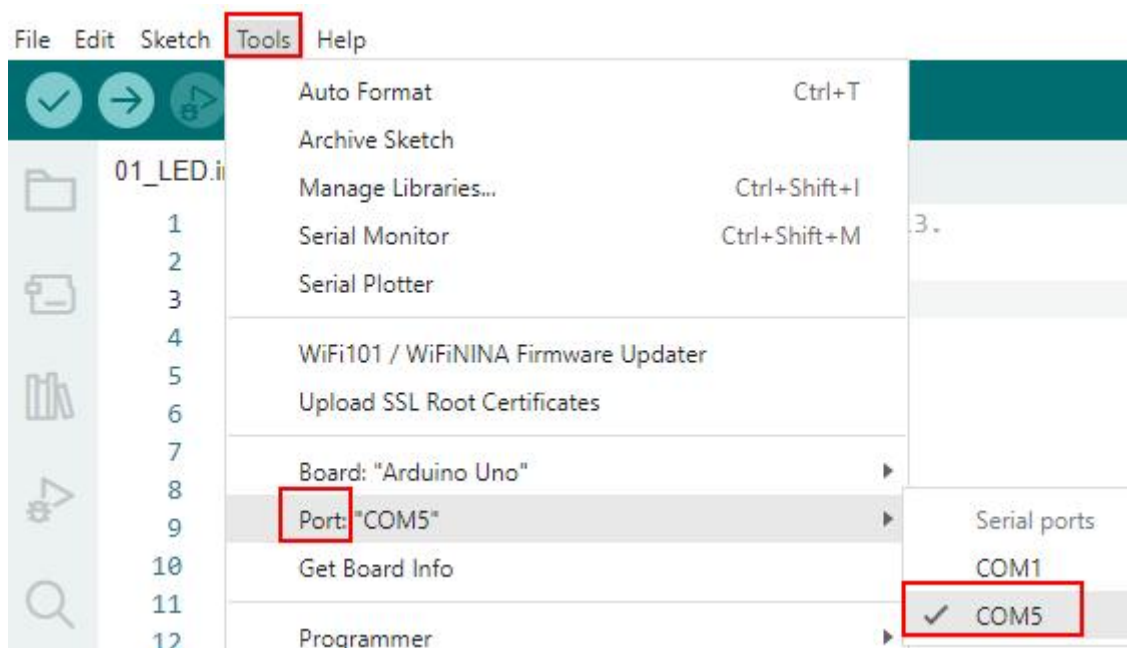
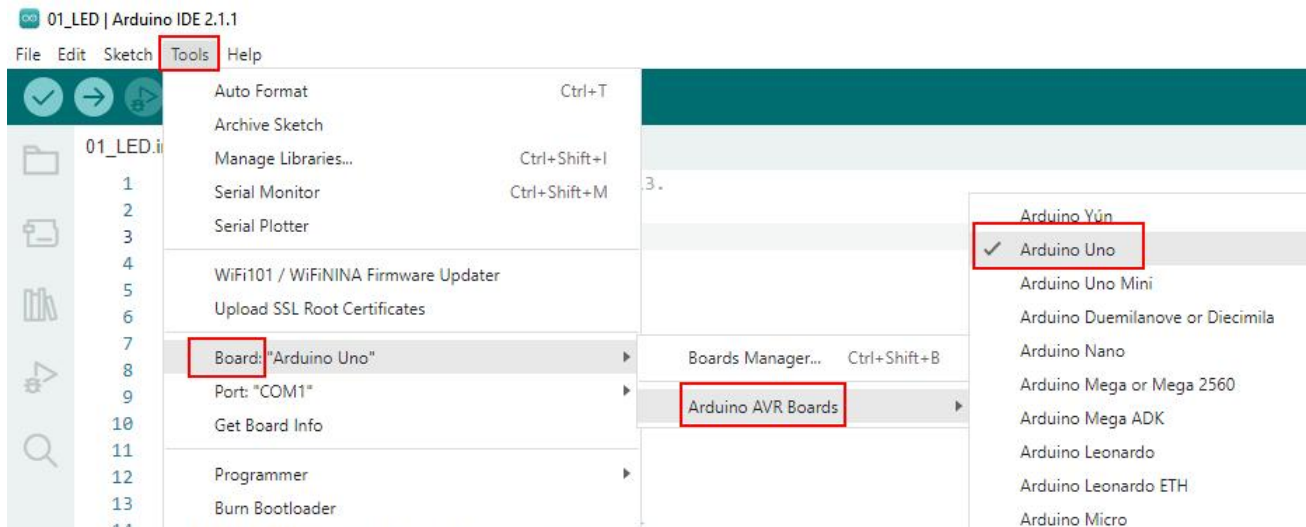
3. Select development board and serial port.


Board: Tools--->Board--->Arduino AVR Boards--->Arduino Uno

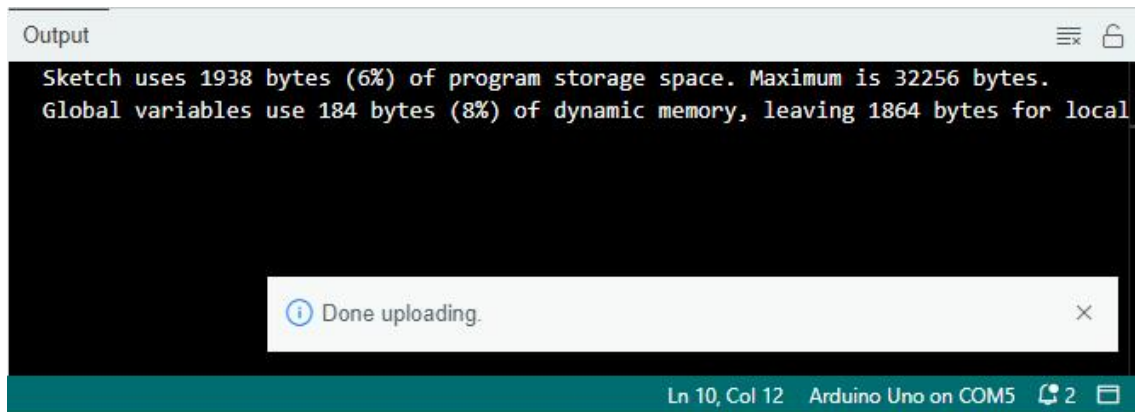
Port: Tools --->Port--->COMx

Note: The port number will be different in different computers.





4. After opening, click  to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.



5. After successfully running the program, You will be able to see the LED matrix module display a smiley face.

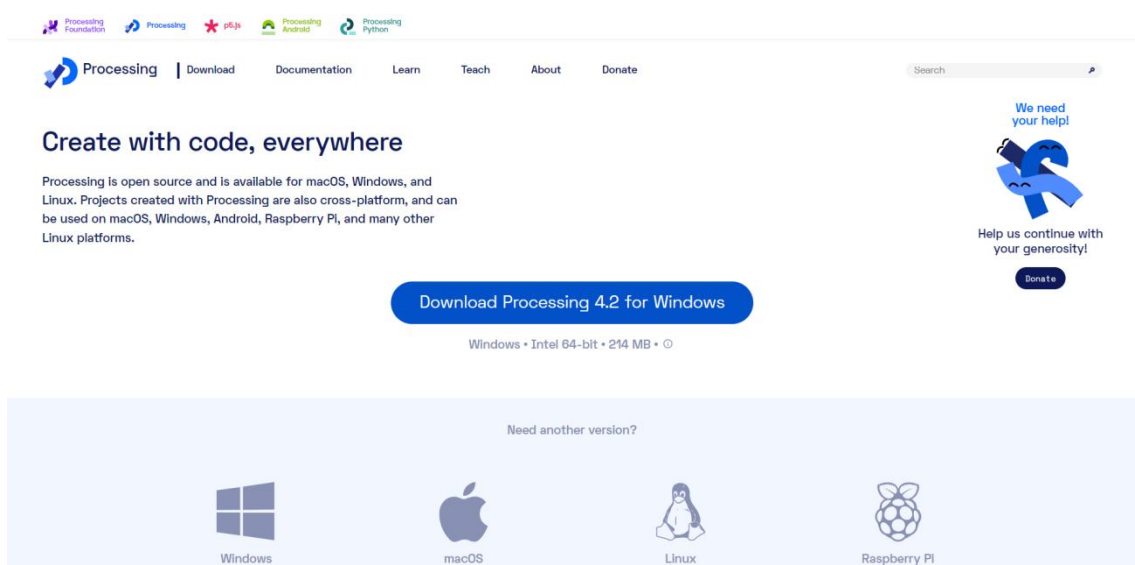
## 12.5 Dot Matrix Tool

### Install Processing

use Processing to build a simple Led Matrix platform.

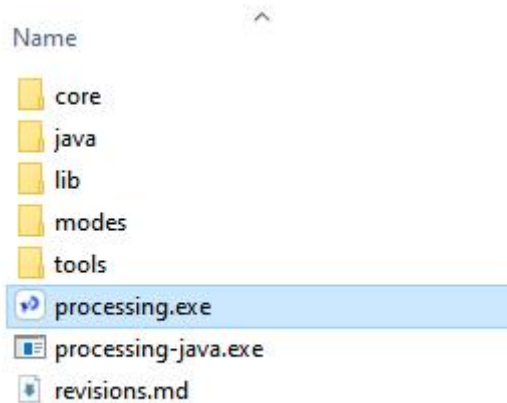
You can download Processing via the link: <https://processing.org/download/>

You can choose an appropriate version to download according to your PC system.

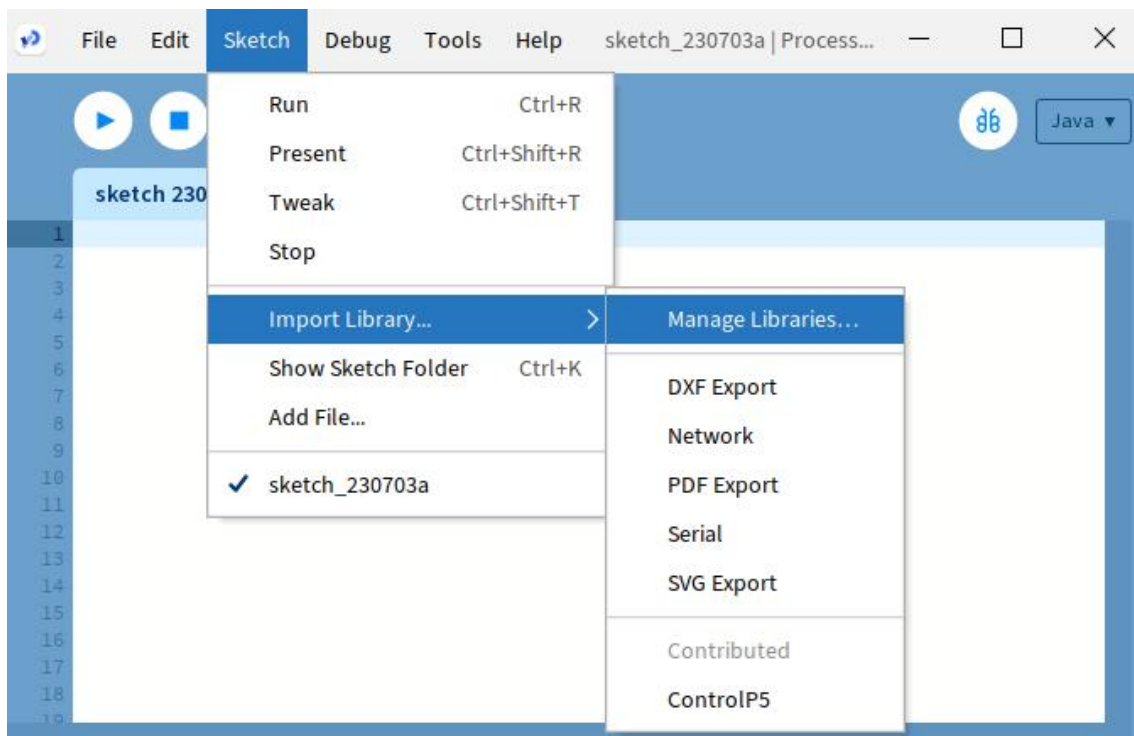




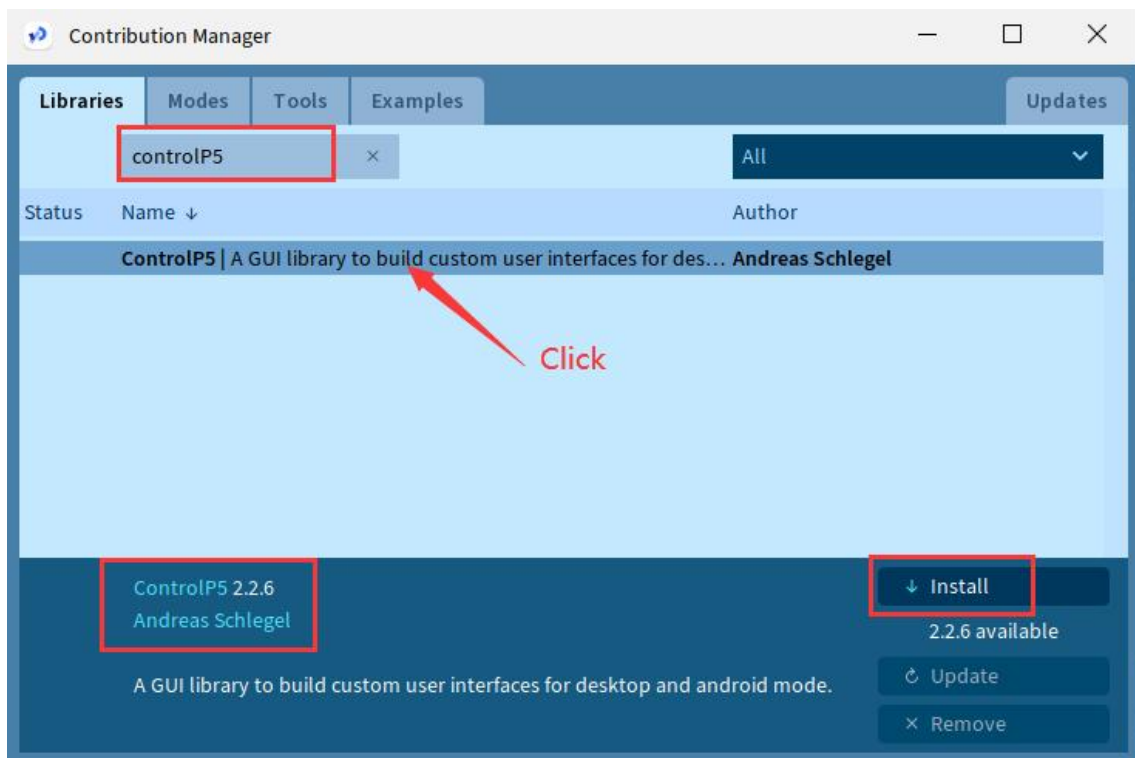
Unzip the downloaded file to your computer. Click "[processing.exe](#)" as the figure below to run this software.



In the interface of Processing, click [Sketch](#) on Menu bar, select "[Import Library...](#)" and then click "[Manage Libraries...](#)"

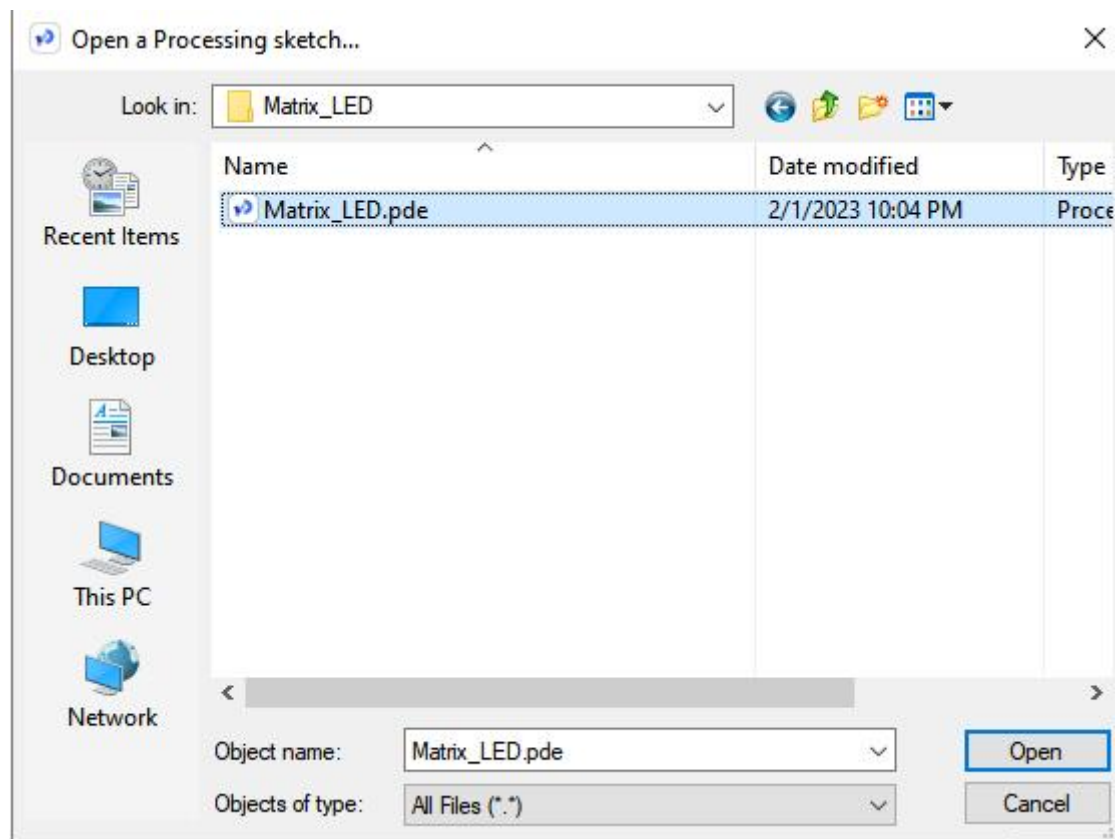


Enter "[ControlP5](#)" in the input field of the pop-up window. Click the searching result and then click "[install](#)"

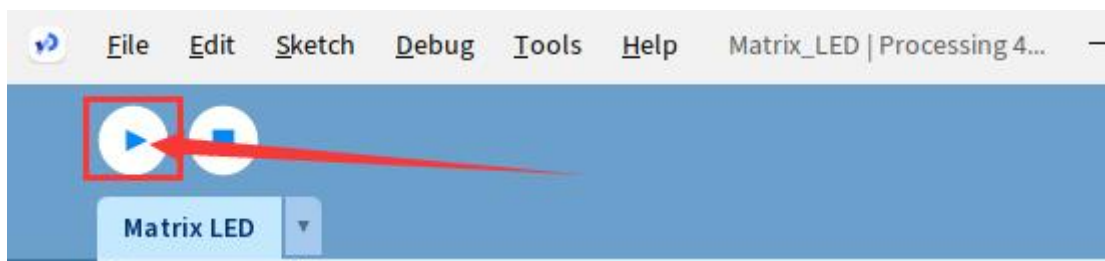


Open the folder Matrix\_LED in 01.5\_Matrix of the “[Adeept\\_UnoCar-B/Code/12\\_Matrix](#)”.

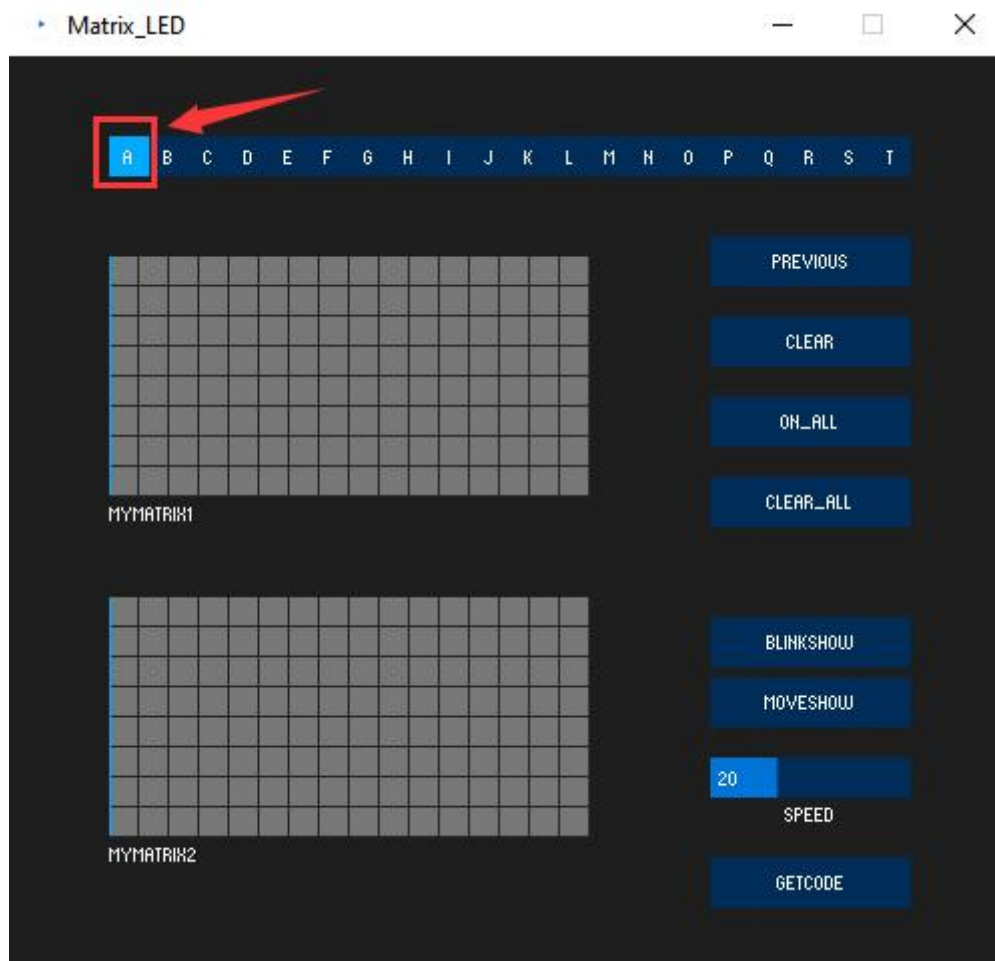
Here we take Windows as an example. Click to open **Matrix\_LED.pde**.



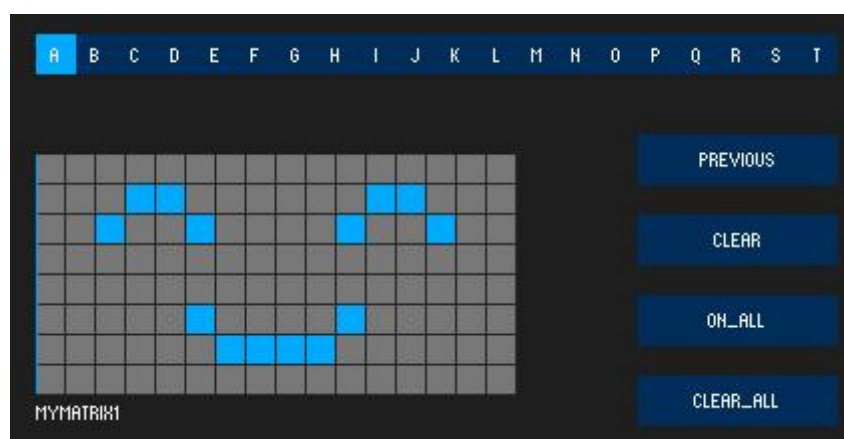
Click **Run**.



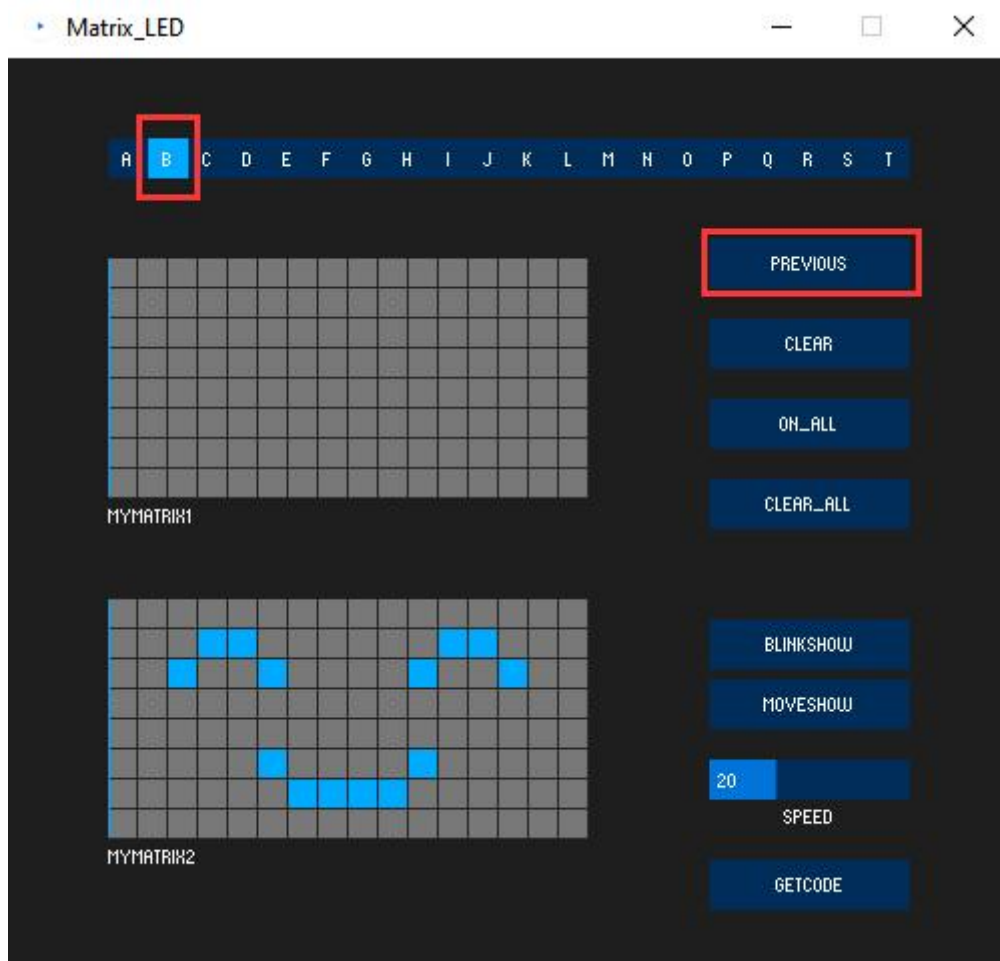
There are 20 pages from A to T. Select Page A.

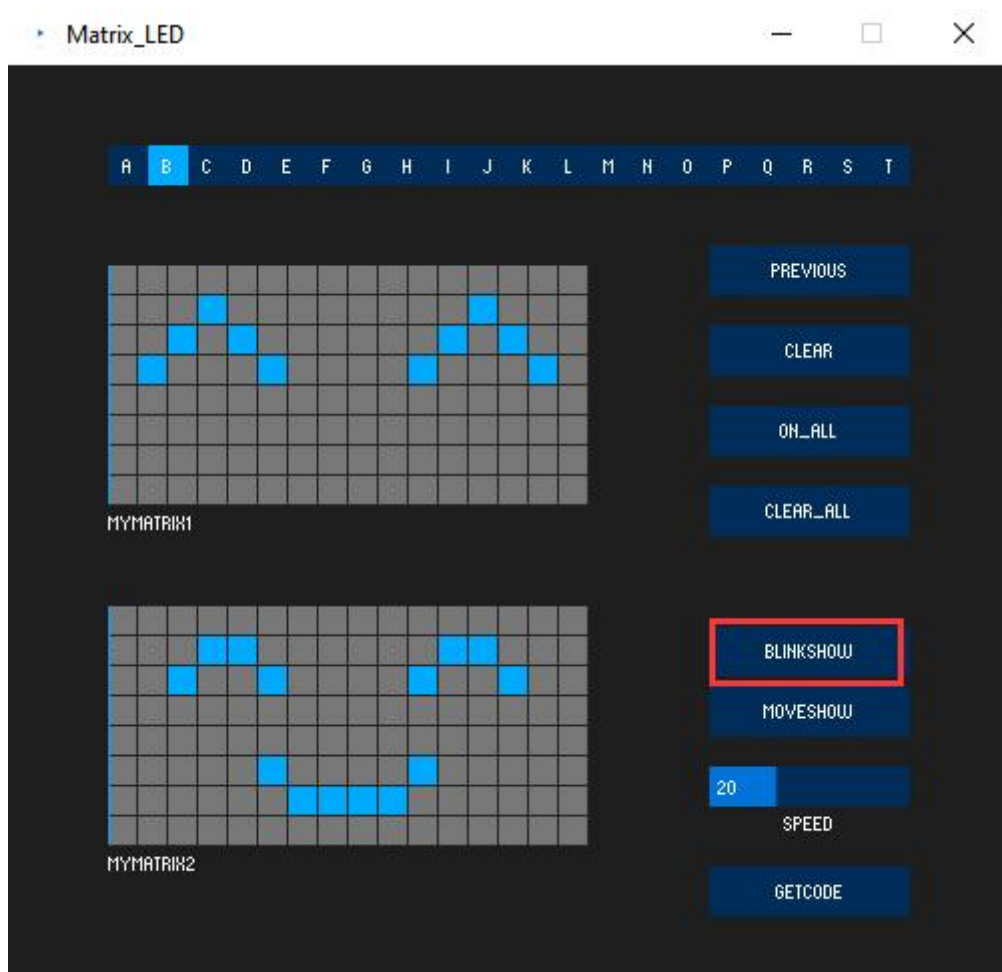


You can design your own pattern by clicking on the squares. Left-click to select a point.



Next select Page B and click **PREVIOUS**. PREVIOUS will copy the previous pattern to B.

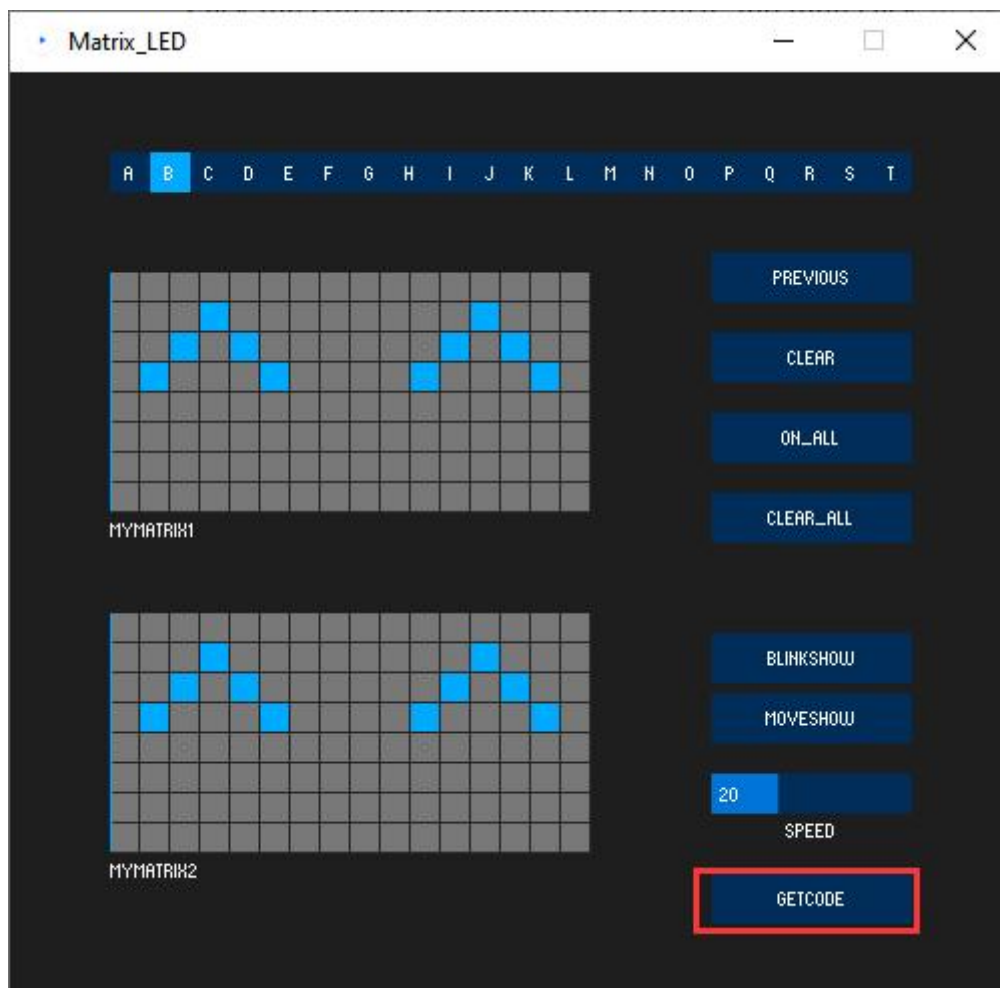




Click the squares to modify the pattern, and then click [BLINKSHOW](#), you can browse the overlay effect of different pages.

Click [GETCODE](#) to generate array.





```

-----y
-----get HEX-----
0x00,0x18,0x24,0x00,0x00,0x04,0x03,0x00,0x00,0x10,0x28,0x44,0x00,0x00,0x00,0x00,-----x
0x00,0x18,0x24,0x00,0x00,0x20,0xC0,0x00,0x00,0x08,0x14,0x22,0x00,0x00,0x00,0x00,-----y

```

The data on the left of the LED matrix are stored together and end with "----x", and the data on the right are stored together and end with "----y". Copy these two sets of dot matrix data and replace the array content in "12\_Matrix.ino".

In the array, every 8 values is a pattern, usually a row with 8 values in the program.

## 12.6 Code

```
1. #include <VK16K33.h>
2.
3. #define ADDRESS 0x71
4.
5. VK16K33 matrix = VK16K33();
6.
7. byte x_array[][8] = { //Put the data into the left LED matrix
8.     0x00,0x18,0x24,0x00,0x00,0x04,0x03,0x00,
9.     0x00,0x10,0x28,0x44,0x00,0x00,0x00,0x00,
10. };
11.
12. byte y_array[][8] = { //Put the data into the right LED matrix
13.     0x00,0x18,0x24,0x00,0x00,0x20,0xC0,0x00,
14.     0x00,0x08,0x14,0x22,0x00,0x00,0x00,0x00,
15. };
16.
17. void setup()
18. {
19.     matrix.init(ADDRESS);
20.     matrix.setBlink(VK16K33_BLINK_OFF);
21. }
22.
23. void loop()
24. {
25.     int count = sizeof(x_array) / sizeof(x_array[0]);
26.     for (int i = 0; i < count; i++) {
27.         matrix.showStaticArray(x_array[i], y_array[i]);
28.         delay(500);
29.     }
30. }
```