

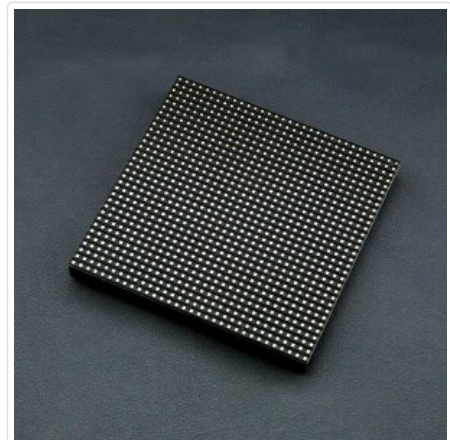


32x32 RGB LED Matrix - 4mm pitch SKU:DFR0472

 [Home](#) > [Sensors & Modules](#) > [LCDs, LEDs & Displays](#) > [LEDs](#)

Contents [\[hide\]](#)

- [1 Introduction](#)
- [2 Specification](#)
- [3 Board Overview](#)
- [4 Tutorial](#)
 - [4.1 Requirements](#)
 - [4.2 Connection Diagram](#)
 - [4.3 16P Interface Diagram](#)
 - [4.4 Sample Code 1](#)
 - [4.5 Sample Code 2](#)
 - [4.6 Expected Results](#)
- [5 FAQ](#)
- [6 More](#)



DFR0472 32x32 RGB LED



Introduction

City night is always beautiful. She is just like a charming girl, showing her beauty every time. The colorful light is jewelry, dotted on her dress.

This is a 32x32 RGB LED Matrix Panel, it has 1024 full-color RGB LEDs in all. Each LED can be independently addressed and controlled. It requires at least 13 digital GPIOs to control the LED matrix. The led matrix has 2 IDC connectors (DATA_IN, DATA_OUT) on the back, you can cascade multiple panels and make a huge screen together. BUT Arduino doesn't support this function, its speed is not enough to multiple panel.

What's more, it is a high brightness, long life, no pollution, pure color LED display module. It can be used both indoor and outdoor, safety and stability, the module can not be mutually extrusion deformation, also can be used normally in harsh environment.



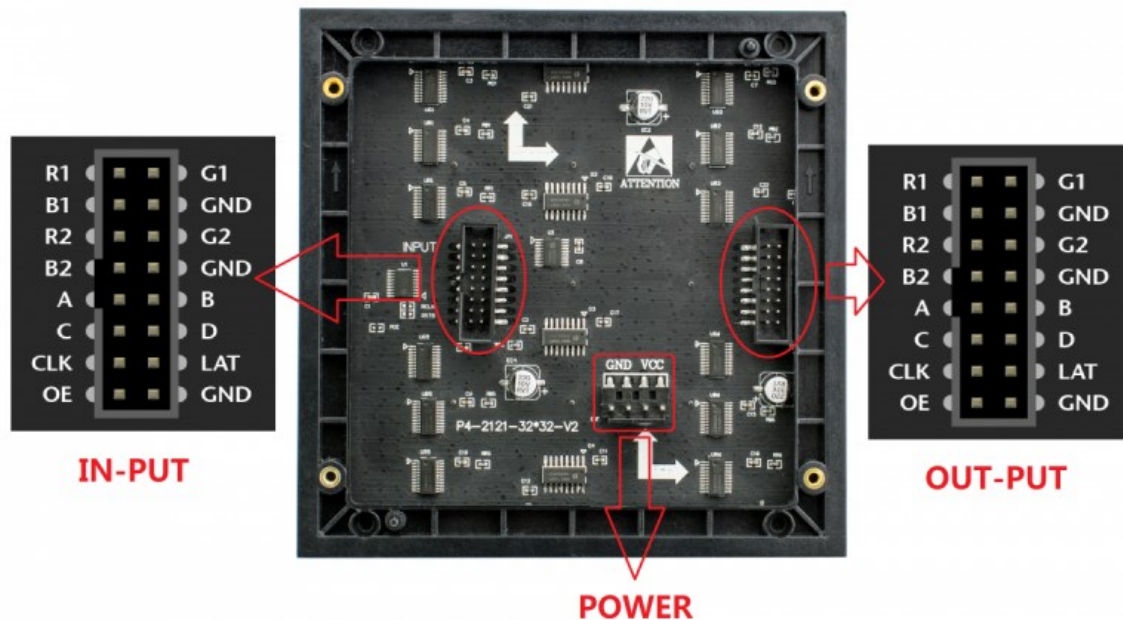
Note: This panel needs a 5V@3A power supply. And if you connect 2 or more screen, it needs a high performance controller, such as Mega 2560, Raspberry. .etc.

Specification

- Operating voltage: DC 5V
- Average power consumption: <500W/m²
- Maxim Power Consumption: <1000w/m²
- Pixel: 32x32=1024
- Level of viewing Angle: ≥ 160°
- Control mode: Synchronous control
- Drive mode: 1/16 scan rate
- Repetition frequency: ≥ 60Hz
- White Balance Brightness: ≥ 1200cd/m²
- Refresh frequency : ≥ 300Hz
- MTTF: ≥ 5000 hours
- Service Life: 75000~100000 hours
- Pixel pitch: 6mm
- Dimension: 128mm*128mm/5.04*5.04 inches
- Thickness: 11mm

Board Overview

Note: The pin order of DATA-IN and ADTA-OUT is same, POWER SUPPLY 5V.



DATA-IN and DATA-OUT

Label	Name	Function
1	DR1	High R data
2	DG1	High G data
3	DB1	High B data
4	GND	GND
5	DR2	Low R data
6	DG2	Low G data
7	DB2	Low B data
8	GND	GND
9	A	A line selection
10	B	B line selection
11	C	C line selection
12	D	D line selection
13	CLK	CLOCK
14	LAT	LATCH
15	OE	Output Enable
16	GND	GND

POWER

Label	Name	Function
1	VCC	5V
2	VCC	5V
3	GND	GND
4	GND	GND

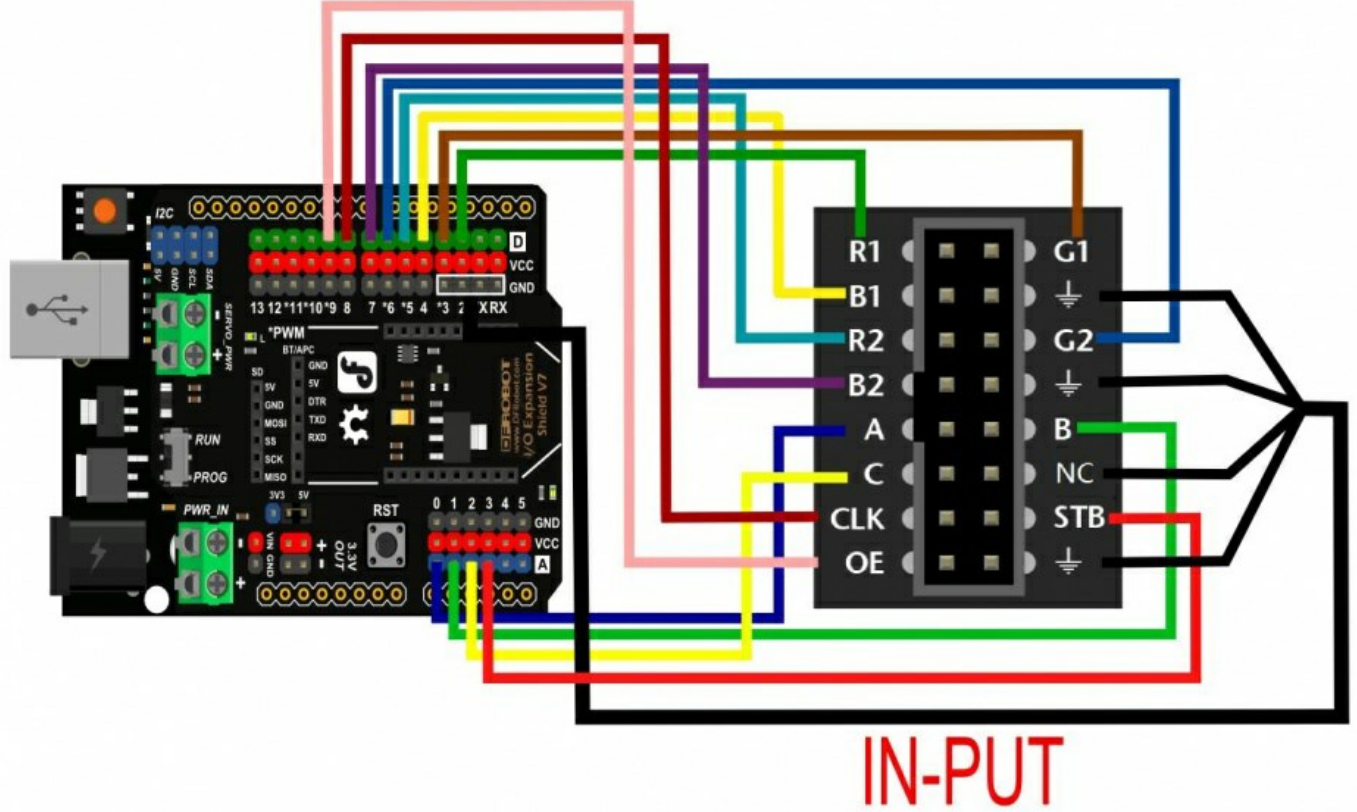
Tutorial

According to the pinout to connect, then upload the code to UNO, you will be able to see a beautiful display effect.

Requirements

- **Hardware**
 - [DFRduino UNO R3](#) x1
 - DFR0471 x1
 - DuPont cables
- **Software**
 - Arduino IDE [Click to Download Arduino IDE from Arduino®](#)

Connection Diagram

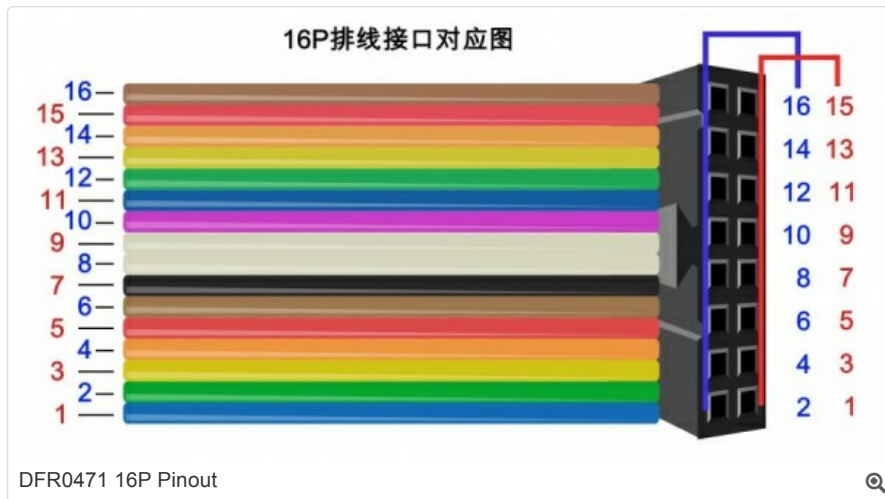


DFR0471 Connection Diagram



Note: It needs an external power supply, the USB is only 5V@500mA, not enough power.

16P Interface Diagram



Sample Code 1

Click to download the library [Adafruit-GFX-Library](#). [RGB-matrix-Panel](#). [How to install the library?](#)

```

/*****
 *
 * For 32x32 RGB LED matrix.
 *
 * @author lg.gang
 * @version V1.0
 * @date 2016-10-28
 *
 * GNU Lesser General Public License.
 * See
 *
 * for details.
 * All above must be included in any redistribution
 * *****/

```

```

#include

// Core graphics library

#include

// Hardware-specific library

// If your 32x32 matrix has the SINGLE HEADER input,
// use this pinout:
#define CLK 8 // MUST be on PORTB! (Use pin 11 on Mega)
#define OE 9
#define LAT 10
#define A A0
#define B A1
#define C A2
#define D A3
// If your matrix has the DOUBLE HEADER input, use:
// #define CLK 8 // MUST be on PORTB! (Use pin 11 on Mega)
// #define LAT 9
// #define OE 10
// #define A A3
// #define B A2
// #define C A1
// #define D A0
RGBmatrixPanel matrix(A, B, C, D, CLK, LAT, OE, false);

void setup() {
    int x, y, hue;
    float dx, dy, d;
    uint8_t sat, val;
    uint16_t c;

    matrix.begin();

    for(y=0; y < matrix.width(); y++) {
        dy = 15.5 - (float)y;
        for(x=0; x < matrix.height(); x++) {
            dx = 15.5 - (float)x;
            d = dx * dx + dy * dy;
            if(d <= (16.5 * 16.5)) { // Inside the circle(ish)?
                hue = (int)((atan2(-dy, dx) + PI) * 1536.0 / (PI * 2.0));
                d = sqrt(d);
                if(d > 15.5) {
                    // Do a little pseudo anti-aliasing along perimeter
                    sat = 255;
                    val = (int)((1.0 - (d - 15.5)) * 255.0 + 0.5);
                } else {
                    // White at center
                    sat = (int)(d / 15.5 * 255.0 + 0.5);
                    val = 255;
                }
                c = matrix.ColorHSV(hue, sat, val, true);
            } else {
                c = 0;
            }
            matrix.drawPixel(x, y, c);
        }
    }
}

void loop() {
    // do nothing
}

```

Smample Code 2

```

/*****
 *
 * For 32x32 RGB LED matrix.
 *
 * @author lg.gang
 * @version V1.0
 * @date 2016-10-28
 *
 * GNU Lesser General Public License.
 * See
 *
 * for details.

```

```

* *****/

#include

    // Core graphics library

#include

    // Hardware-specific library

// If your 32x32 matrix has the SINGLE HEADER input,
// use this pinout:
#define CLK 8 // MUST be on PORTB! (Use pin 11 on Mega)
#define OE 9
#define LAT 10
#define A A0
#define B A1
#define C A2
#define D A3
// If your matrix has the DOUBLE HEADER input, use:
// #define CLK 8 // MUST be on PORTB! (Use pin 11 on Mega)
// #define LAT 9
// #define OE 10
// #define A A3
// #define B A2
// #define C A1
// #define D A0
RGBmatrixPanel matrix(A, B, C, D, CLK, LAT, OE, false);

void setup() {

    matrix.begin();

    // whew!
}

void loop() {
    // fix the screen with red
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(7, 0, 0));
    delay(500);
    // fix the screen with green
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(0, 7, 0));
    delay(500);
    // fix the screen with blue
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(0, 0, 7));
    delay(500);
    // fix the screen with yellow
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(7, 7, 0));
    delay(500);
    // fix the screen with purple
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(7, 0, 7));
    delay(500);
    // fix the screen with cyan
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(0, 7, 7));
    delay(500);
    // fix the screen with white
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(3, 3, 3));
    delay(500);

    // fill the screen with 'black'
    matrix.fillRect(matrix.Color333(0, 0, 0));
    // draw a pixel in solid white
    matrix.drawPixel(0, 0, matrix.Color333(7, 7, 7));
    delay(500);

    // fix the screen with green
    matrix.fillRect(0, 0, 32, 32, matrix.Color333(0, 7, 0));
    delay(500);

    // draw a box in yellow
    matrix.drawRect(0, 0, 32, 32, matrix.Color333(7, 7, 0));
    delay(500);

    // draw an 'X' in red
    matrix.drawLine(0, 0, 31, 31, matrix.Color333(7, 0, 0));
    matrix.drawLine(31, 0, 0, 31, matrix.Color333(7, 0, 0));
    delay(500);

    // draw a blue circle
    matrix.drawCircle(10, 10, 10, matrix.Color333(0, 0, 7));
    delay(500);
}

```

```

matrix.fillCircle(21,21, 10, matrix.Color333(7, 0, 7));
delay(500);

// fill the screen with 'black'
matrix.fillRect(matrix.Color333(0, 0, 0));

// draw some text!
matrix.setCursor(1, 0);    // start at top left, with one pixel of spacing
matrix.setTextSize(1);    // size 1 == 8 pixels high
matrix.setTextWrap(false); // Don't wrap at end of line - will do ourselves

matrix.setTextColor(matrix.Color333(0,7,0));
matrix.println("green");
matrix.setTextColor(matrix.Color333(0,7,7));
matrix.println("robot");

// print each letter with a rainbow color
matrix.setTextColor(matrix.Color333(7,0,0));
matrix.print('3');
matrix.setTextColor(matrix.Color333(7,4,0));
matrix.print('2');
matrix.setTextColor(matrix.Color333(7,7,0));
matrix.print('x');
matrix.setTextColor(matrix.Color333(4,7,0));
matrix.print('3');
matrix.setTextColor(matrix.Color333(0,7,0));
matrix.println('2');

matrix.setTextColor(matrix.Color333(0,7,7));
matrix.print('*');
matrix.setTextColor(matrix.Color333(0,4,7));
matrix.print('R');
matrix.setTextColor(matrix.Color333(0,0,7));
matrix.print('G');
matrix.setTextColor(matrix.Color333(4,0,7));
matrix.print('B');
matrix.setTextColor(matrix.Color333(7,0,4));
matrix.print('*');
delay(5000);
}

```

Expected Results

- Sample Code 1: A color filled circle
- Sample Code 2: The LED module will take turns display: a white point, full screen green, full screen yellow, a yellow rectangle, a yellow rectangle, a red X and a blue circle,

FAQ

For any questions, advice or cool ideas to share, please visit the [DFRobot Forum](#).

More

 Shopping from [32x32 RGB LED Matrix Panel \(4mm pitch\)](#) or [DFRobot Distributor](#).

[What links here](#)
[Related changes](#)
[Special pages](#)
[Printable version](#)
[Permanent link](#)
[Page information](#)

This page was last modified on 11 October 2017, at 10:01.

Content is available under [GNU Free Documentation License 1.3 or later](#) unless otherwise noted.

[Privacy policy](#)
[About DFRobot Electronic Product Wiki and Tutorial: Arduino and Robot Wiki-DFRobot.com](#)
[Disclaimers](#)

