RGB-Matrix-P3-64x32

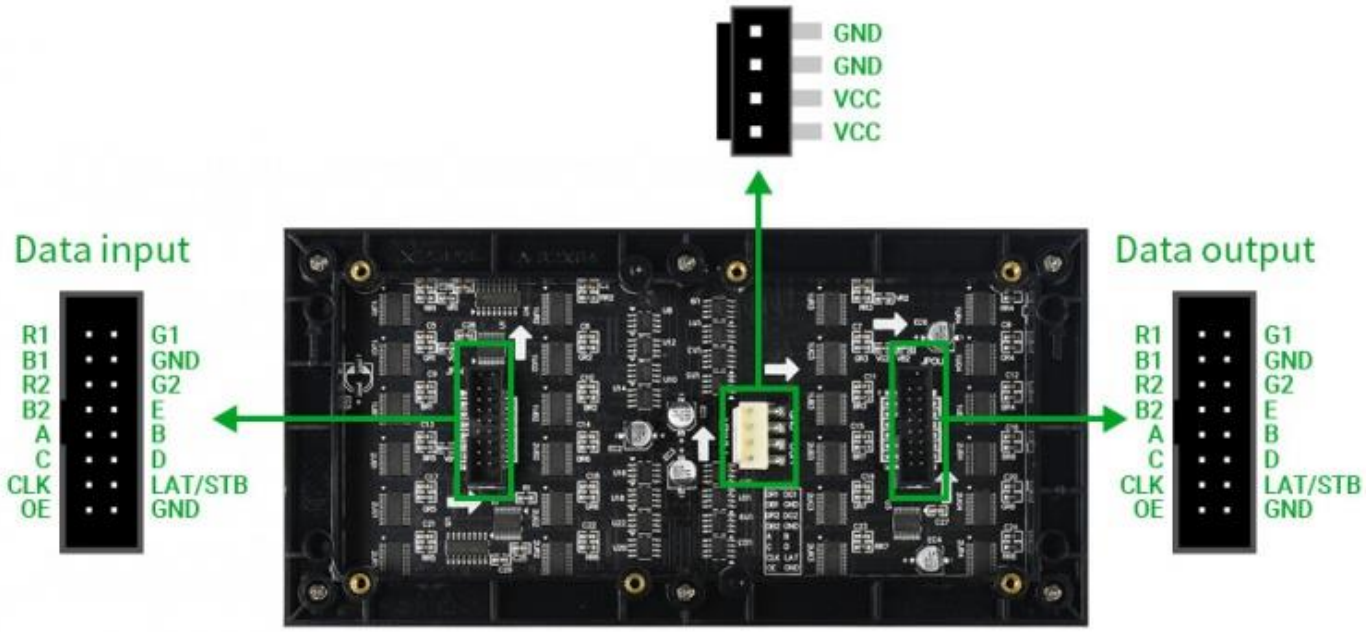
Maybe be on the wall of tall buildings, maybe on corners of unknown streets, you can always see the dazzling neon light, everywhere in the noisy yet energetic city. Sometimes, looking at these shining screens, you may want to make a unique one, and give it to the beautiful night as a gift. Now, this 64×32 RGB LED matrix panel will be the start to make your wish come true.

- 2048 individual RGB LEDs, full-color display, adjustable brightness
- 64×32 pixels, 3mm pitch, allows displaying text, colorful image, or animation
- 192×96mm dimensions, moderate size, suitable for DIY desktop display or wall mount display
- Onboard two HUB75 header, one for controller data input, one for output, chain support
- Provides open source development resources and tutorials, for use with Raspberry Pi, Pico, ESP32, Arduino, and so on

Specifications

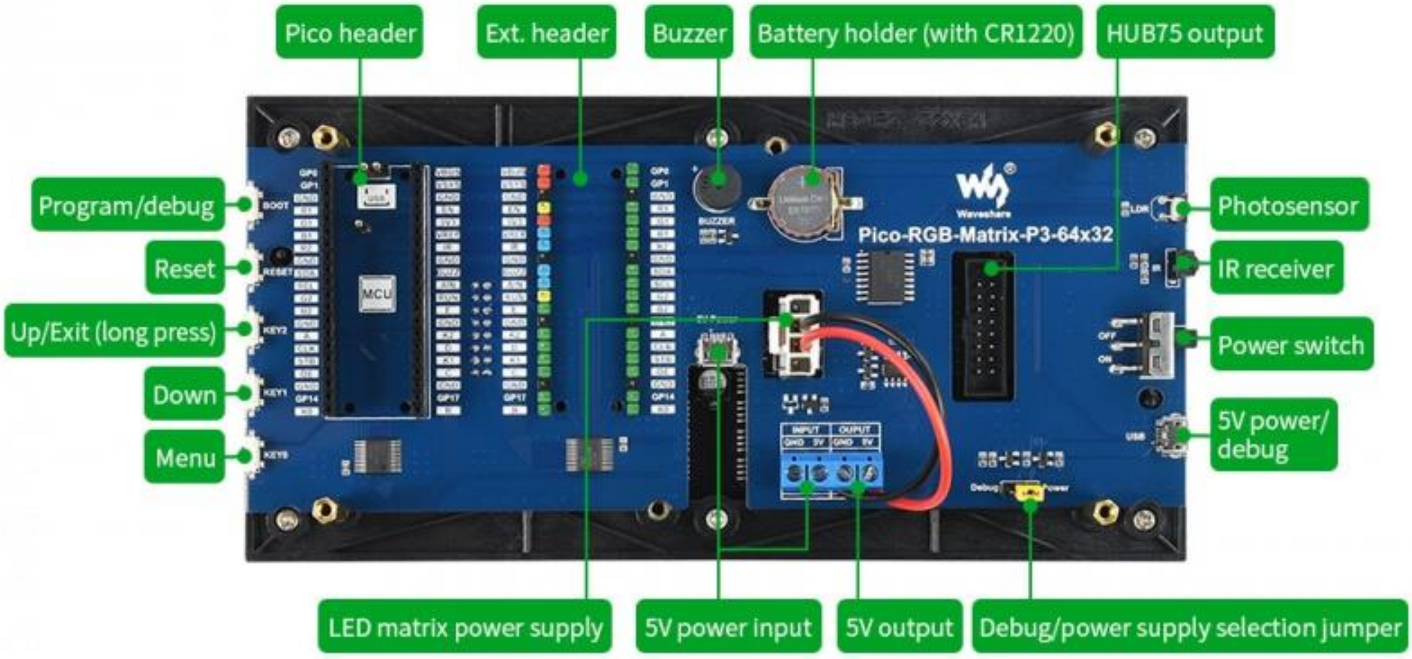
DIMENSIONS	192mm × 96mm
PIXELS	64×32=2048 DOTS
PITCH	3mm
PIXEL FORM	1R1G1B
VIEWING ANGLE	≥160°
CONTROL TYPE	synchronization
DRIVING	1/16 scan
HEADER	HUB75
POWER SUPPLY	5V / 2.5A（VH4 header input）
POWER	≤12W

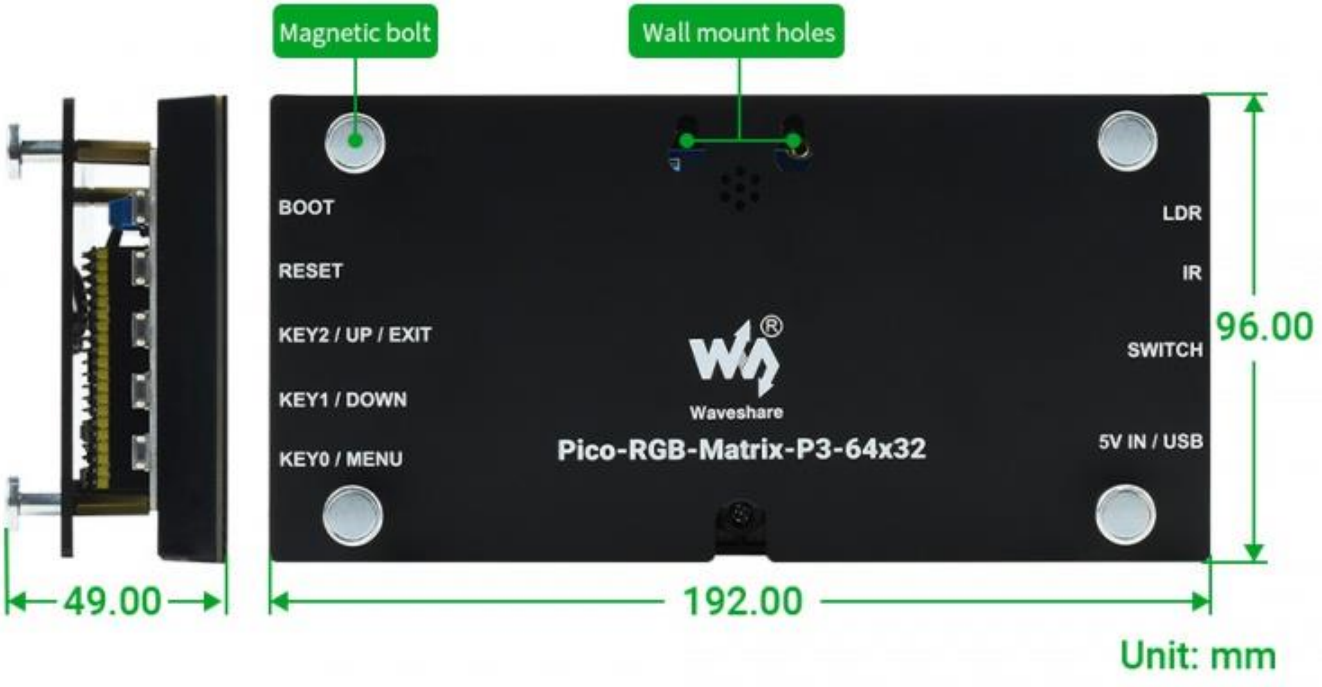
Header Definition



PIN	DESCRIPTION		PIN	DESCRIPTION
VCC	5V power input		**GND**	Ground
R1	R higher bit data		**R2**	R lower bit data
G1	G higher bit data		**G2**	G lower bit data
B1	B higher bit data		**B2**	B lower bit data
A	A line selection		**B**	B line selection
C	C line selection		**D**	D line selection
E	E line selection		**CLK**	clock input
LAT/STB	latch pin		**OE**	output enable

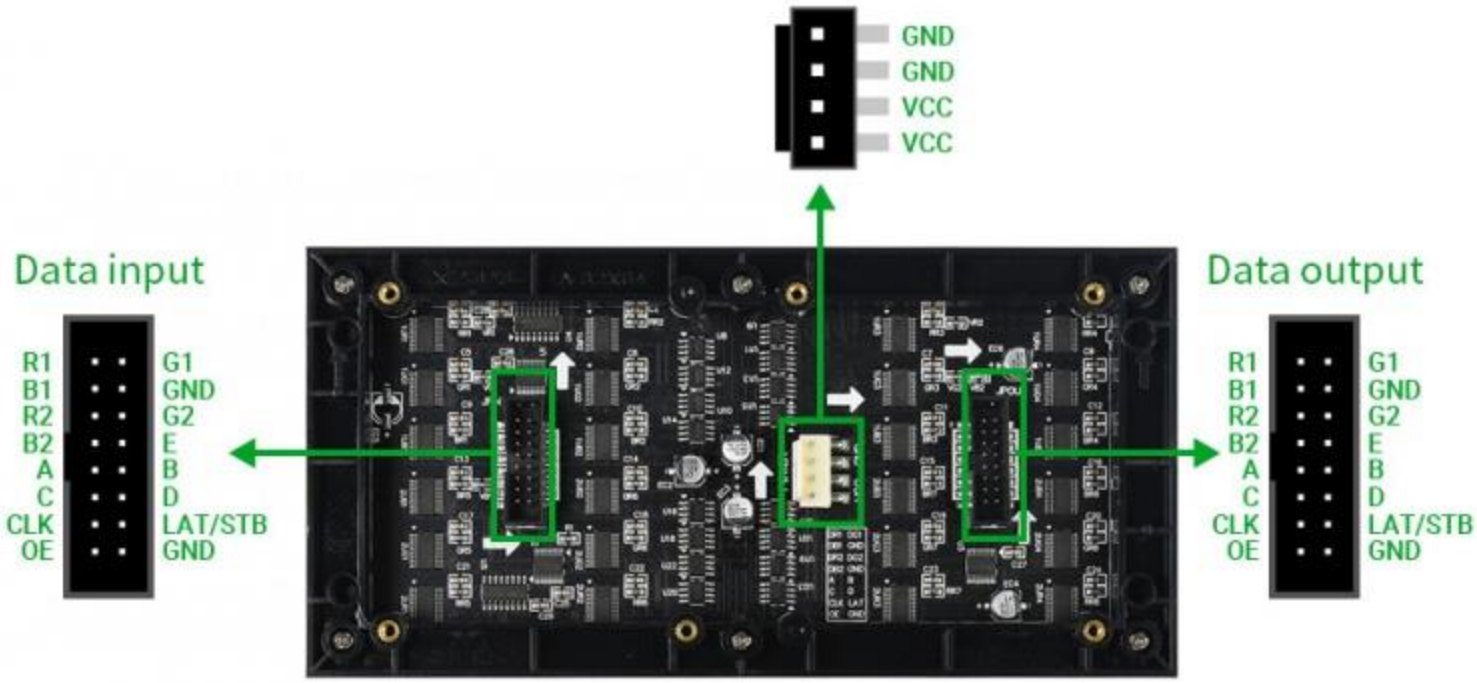
Header Definition





Pico connection pins

Board	Pico	Pin description
Pins used by RGB LED Matrix (HUB75 interface)		
R1	GP02	R higher bit data
G1	GP03	G higher bit data
B1	GP04	B higher bit data
R2	GP05	R lower bit data
G2	GP08	G lower bit data
B2	GP09	B lower bit data
A	GP10	A line selection
B	GP16	B line selection
C	GP18	C line selection
D	GP20	D line selection
E	GP22	E line selection
CLK	GP11	clock input
STB/LAT	GP12	latch pin
OE	GP13	output enable



Board	Pico	Pin description
Pins used by other resources of the board		
K0	GP15	KEY0 button, the MENU menu of the digital clock, can also be customized
K1	GP19	KEY1 button, + / Down button of digital clock, can also be customized
K2	GP21	KEY2 button,-/ UP button of digital clock, can also be customized
RUN	RUN	RESET button, can be used for Pico reset
BOOTSET	BOOTSET	BOOT button, can be used for Pico burning program (long press BOOT, then press RESET to enter the firmware download mode)
SDA	GP06	I2C data pin, used to control DS3231 RTC clock chip
SCL	GP07	I2C clock pin, used to control DS3231 RTC clock chip
BUZZ	GP27	Buzzer control pin
AIN	GP26	Photoresistor control pin
IRM	GP28	Infrared receiving control pin

See detailed hardware design of the circuit diagram.

Setup environment

You can refer to Raspberry Pi's guide: https://www.raspberrypi.org/documentation/pico/getting-started
Or check the guide#Set up C/C++ Environment

Demo example

Hardware connection

Materials needed

- Pico-RGB-Matrix-P3-64x32 (this product)
- Raspberry Pi Pico (must be purchased separately, if not, it is recommended to buy a version with soldered headers, which is convenient for direct insertion and use)
- Micro USB cable (must be purchased separately)

Hardware connection steps

1. Align the pin header which is marked in red and then connect the RGB LED Matrix panel to the driver board.
2. Cut the adapter cable (about 10cm) by plier
3. Connect the cable which is cut in the last step to the RGB LED Matrix and the driver board

Connect your Pico board to the Matrix device, please take care of the direction.

4. Assemble the Acrylic backplane and fix it with magnetic screws

If you need to program Pico and debug, you can skip this step first, and operate it when it is ready to use

5. Optional: If you feel that the RGB LED Matrix is too bright, you can stick the black Acrylic font panel on the Matrix.

- 

1. Assemble Matrix panel

- 

2. Cut the cable

- 

3. Connect the cable

- 

4. Assemble Acrylic backplane

- 

5（optional） Assemble Acrylic font panel

Example display

Multi-Features Digital Clock

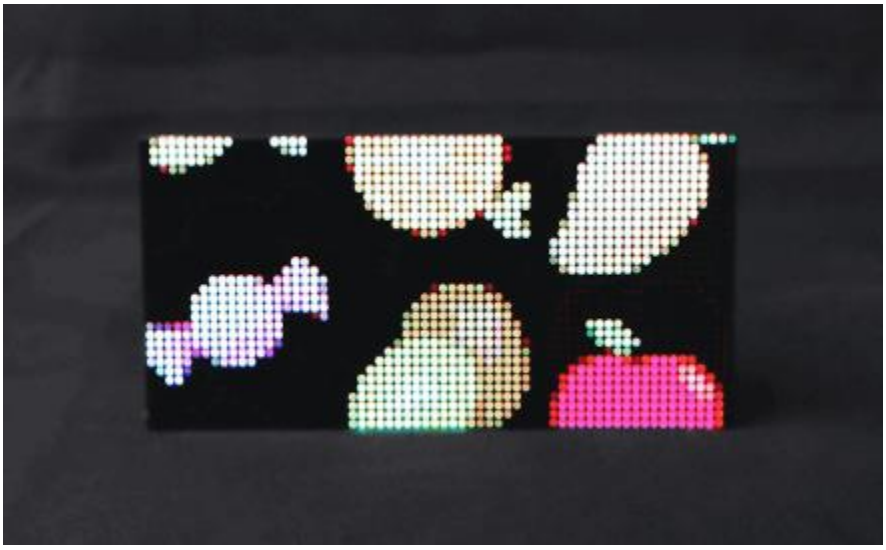
- This example is developed based on the C++ SDK. In order to quickly demonstrate the effects and functions of the example.
- Download the demo codes from the Resources and unzip to get the uf2 file.
    - You can refer to the #Set up C/C++ Environment to build and program the C codes.
- Press and hold the BOTSEL button of your Pico, and connect it to the host PC. A portable disk RPI-RP2 will be recognized
- Copy the uf2 file to the Pico (RPI-RP2) and it will reconnect automatically.
- After the burning is completed, the running effect of the example is shown in the figure below:

【Function Description】

- Time display screen:
    - Display date, day of the week, hour, minute, lunar calendar and temperature
- Function setting menu
    - Date setting
    - time setting
    - BEEP setting (buzzer setting)
    - Auto brightness
    - Language setting (under development)

Fruit machine

This example is developed based on CircuitPython[1] , and the program is downloaded[2] . The effects and functions of the example are as follows:



【Function Description】

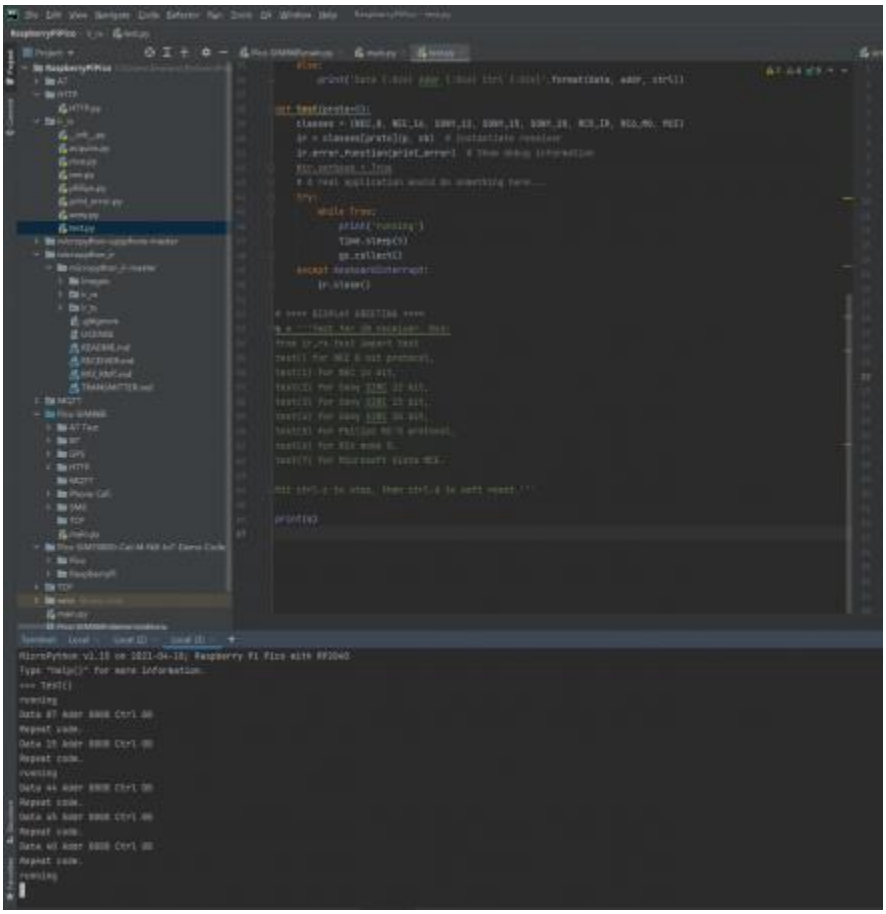
- The display can display a variety of fruits or other small BMP icons

- Automatically scroll icons at regular intervals, and randomly display the results of the scrolling

1. Jump up↑ CircuitPython is a fork of MicroPython, For specific usage, please refer to the RGB-Matrix related CircuitPython tutorial
2. Jump up↑ Pico must first install CircuitPython, and then copy the corresponding CircuitPythond code to the recognized U disk to complete the download.

Infrared transceiver test

This example is developed based on MicroPython, the program is downloaded , the example effects and functions are as follows: :



【Function Description】

- The infrared serial code generated by the corresponding button of the infrared remote control can be recognized

Set up C/C++ Environment

You can program the Pico with Raspberry Pi or Windows PC.
You can directly refer to the Offical manual of Raspberry Pi.

Raspberry Pi

- Please make sure that you use the Newest Raspberry Pi OS and the Pi 4 is recommended.

Pico SDK

- Open the terminal of the Raspberry Pi
- Run the following commands to get the SDK

```
cd ~/
mkdir pico
cd pico
git clone -b master https://github.com/raspberrypi/pico-sdk.git
cd pico-sdk
git submodule update --init
```

- Install the ToolChain

```
sudo apt update
sudo apt install cmake gcc-arm-none-eabi libnewlib-arm-none-eabi build-essential
```

- **[Optional]** If you want to update the SDK, you can run the following commands:

```
cd ~/pico/pico-sdk
git pull
git submodule update
```

Pico Examples

Here we use the pico-examples of Raspberry Pi for examples

- Download the examples

```
cd ~/pico
git clone -b master https://github.com/raspberrypi/pico-examples.git
```

- Create build directory for Blink example

```
cd ~/pico/pico-examples
mkdir build
cd build
```

- Set the PICO_SDK_PATH
    - Note that the path of the SDK maybe different according to the directory you saved
    - If you follow this guide step by step, the relative path of the SDK should be ../../pico-sdk and the absolute path should be ~/pico/pico-sdk

- If you change the path of the sdk, please use the correct path

```
export PICO_SDK_PATH=../../pico-sdk
```

- Build the example

```
cmake ..
make -j4
```

- After building, a .uf2 file and a .elf file are generated
- Press and hold the BOOTSEL button of the pico and connect it to Raspberry Pi by USB cable
- A portable disk RPI-RP2 is recognized, copy the .uf2 file to the RPi-RP2
- Pico will reboot and run the codes.