StarFive
赛昉科技

使用昉·星光 2点亮LED矩阵

Python语言版本
应用说明
版本：1.1
日期：2023/06/08
Doc ID: VisionFive 2-ANCH-003

法律声明

阅读本文件前的重要法律告知。

版权注释

本文档中的说明均基于“视为正确”提供，可能包含部分错误。内容可能因产品开发而定期更新或修订。上海赛昉科技有限公司（以下简称“赛昉科技”）保留对本协议中的任何内容进行更改的权利，恕不另行通知。

赛昉科技明确否认任何形式的担保、解释和条件，无论是明示的还是默示的，包括但不限于适销性、特定用途适用性和非侵权的担保或条件。

赛昉科技无需承担因应用或使用任何产品或电路而产生的任何责任，并明确表示无需承担任何及所有连带责任，包括但不限于间接、偶然、特殊、惩戒性或由此造成的损害。

联系我们：

地址： 浦东新区盛夏路61弄张润大厦2号楼502，上海市，201203，中国

网站： http://www.starfivetech.com

邮箱：sales@starfivetech.com （销售） support@starfivetech.com（支持）

前言

关于本指南和技术支持信息

关于本手册

本应用说明提供使用昉·星光 2的GPIO pin，通过Python示例程序在LED矩阵MAX7219上点亮赛昉科技徽标的步骤。

修订历史

表 0-1 修订历史

版本	发布说明	修订
1.1	2023/06/08	• 在40-Pin GPIO Header定义 (第 7页)增加注释。 • 在准备软件 (第 9页)中更新安装方式。 • 新增资源下载 (第 16页)和立即购买 (第 17页)章节。
1.0	2022/12/15	首次发布。

注释和注意事项

本指南中可能会出现以下注释和注意事项:

- **提示：**
  建议如何在某个主题或步骤中应用信息。
- **注：**
  解释某个特例或阐释一个重要的点。
- **重要：**
  指出与某个主题或步骤有关的重要信息。
- **警告：**
  表明某个操作或步骤可能会导致数据丢失、安全问题或性能问题。
- **警告：**
  表明某个操作或步骤可能导致物理伤害或硬件损坏。

目录

表格清单 5
插图清单 6
法律声明 ii
前言 iii
1. 产品简介 7
1.1. 40-Pin GPIO Header定义 7
2. 准备 8
2.1. 运行环境要求 8
2.2. 准备硬件 8
2.2.1. 连接硬件 8
2.3. 准备软件 9
3. 执行演示代码 12
4. 演示源代码 14
5. 资源下载 16
6. 立即购买 17

表格清单

表 0-1 修订历史 iii
表 2-1 硬件准备 8
表 2-2 将MAX7219连接到40-Pin GPIO Header上 8

插图清单

图 1-1 40-Pin GPIO Header定义 7
图 2-1 将MAX7219连接到40-Pin GPIO Header上 9
图 3-1 示例输出 13

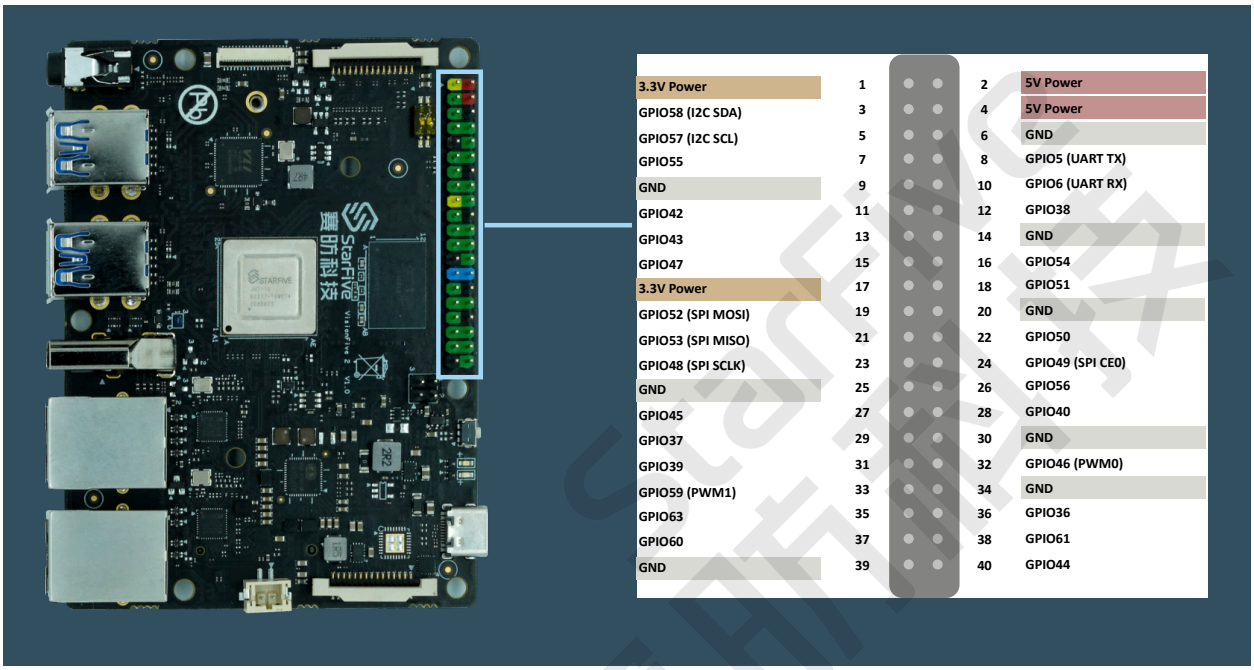
1. 产品简介

本应用说明提供使用昉·星光 2的GPIO pin，通过Python示例程序在LED矩阵MAX7219上点亮赛昉科技徽标的步骤。

1.1. 40-Pin GPIO Header定义

下图显示了40-pin GPIO Header的位置：

图 1-1 40-Pin GPIO Header定义



注：
功能复用pin脚已初始化，不可作为通用GPIO使用。

2. 准备

在执行演示程序之前，务必确认已准备好以下项目：

2.1. 运行环境要求

该演示运行环境要求如下：

- Linux内核版本：Linux 5.15
- 操作系统：Debian 12
- 硬件版本：昉·星光 2
- SoC：昉·惊鸿7110

2.2. 准备硬件

在执行演示程序之前，请务必准备以下硬件：

表 2-1 硬件准备

类型	M/O*	项目	注释
通用	M	昉·星光 2 单板计算机	-
通用	M	• 容量不低于32 GB的Micro-SD卡 • Micro-SD卡读卡器 • 计算机（Windows/Mac OS/Linux） • USB转串口转换器（3.3 V I/O，带线） • 以太网电缆 • 电源适配器（5 V/ 3 A） • USB Type-C数据线	上述项目用于将Debian OS烧录到Micro-SD上。
GPIO演示 (LED矩阵)	M	MAX7219串行点阵显示模块（带5路母对母杜邦电缆）	-

注：
*: M：必须。O：可选

2.2.1. 连接硬件

以下表格和图片描述了如何将SMAX7219连接到40-Pin GPIO Header上：

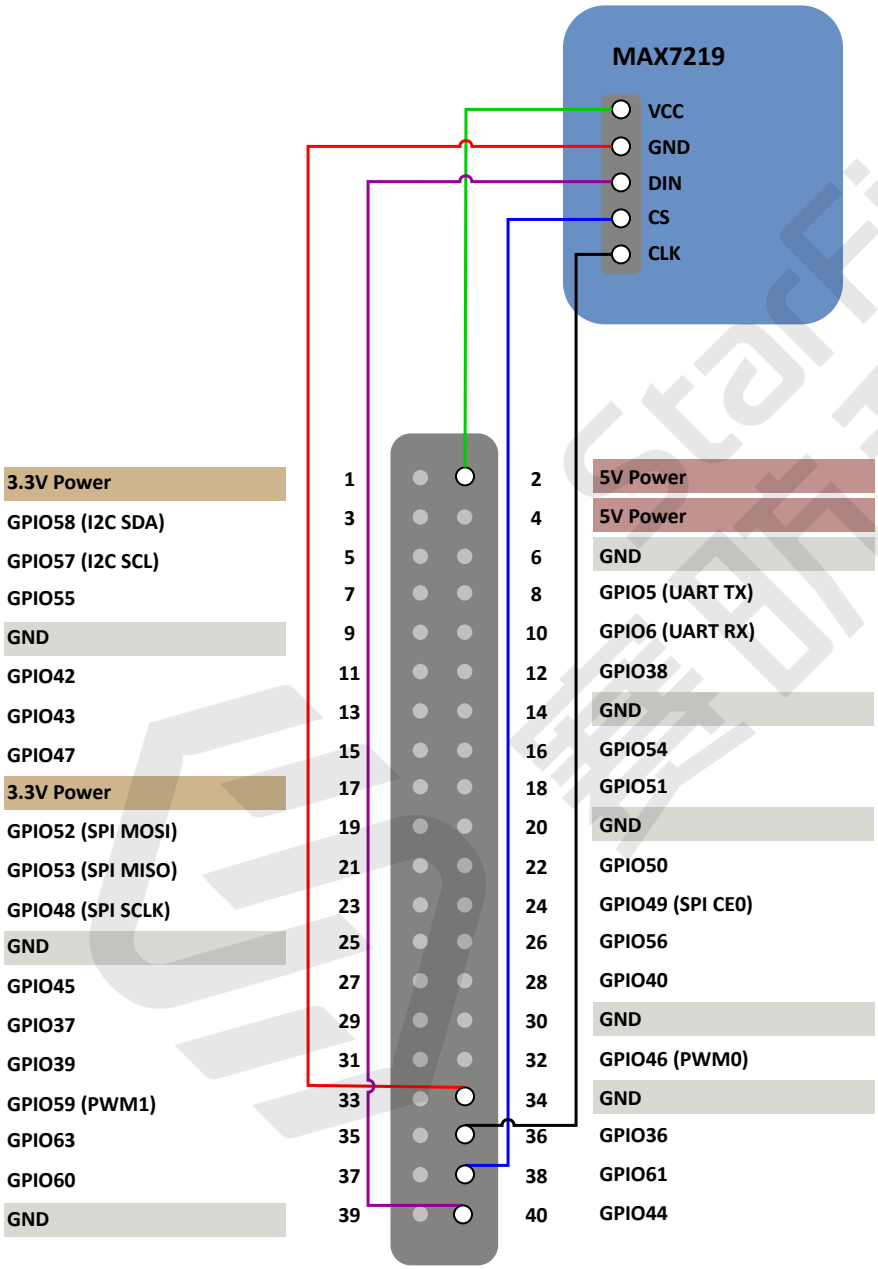
表 2-2 将MAX7219连接到40-Pin GPIO Header上

MAX7219	40-Pin GPIO Header	
	Pin Number	Pin Name
VCC	2	5V 电压
GND	34	GND

表 2-2 将MAX7219连接到40-Pin GPIO Header上 (续)

MAX7219	40-Pin GPIO Header	
	Pin Number	Pin Name
DIN	40	GPIO44
CS	38	GPIO61
CLK	36	GPIO36

图 2-1 将MAX7219连接到40-Pin GPIO Header上



2.3. 准备软件

确认按照以下步骤进行操作:

注：
该Python应用`VisionFive.gpio`适用于昉·星光单板计算机、昉·星光 2和昉·惊鸿7110 EVB。

1. 按照《昉·星光 2单板计算机快速参考手册》中的“将*OS*烧录到*Micro-SD*”章节，将Debian OS烧录到Micro-SD卡上。
2. 登录Debian并确保昉·星光 2已联网。有关详细说明，请参阅《昉·星光 2单板计算机快速参考手册》中“通过以太网使用*SSH*”或“使用*USB*转串口转换器”章节。
3. 在Debian上扩展分区，请参见《昉·星光 2单板计算机快速参考手册》中“扩展分区”章节。
4. 执行以下命令，在Debian系统上安装PIP：

```
apt-get install python3-pip
```

5. 在昉·星光 2 Debian上执行`pip`命令，以安装`VisionFive.gpio`包：

注：
由于pypi.org官网尚不支持上传RISC-V平台的`whl`安装包，不能直接使用`pip install VisionFive.gpio`命令在线安装，因此请按照以下步骤安装`VisionFive.gpio`包。

a. 执行以下命令，安装依赖包：

```
apt install libxml2-dev libxslt-dev
python3 -m pip install requests wget bs4
```

b. 执行以下命令，运行安装脚本`Install_VisionFive_gpio.py`：

```
python3 Install_VisionFive_gpio.py
```

安装脚本代码如下：

```
import requests
import wget
import sys
import os
from bs4 import BeautifulSoup

def parse_data(link_addr, class_type, key_str):
    req = requests.get(url=link_addr)
    req.encoding = "utf-8"
    html=req.text
    soup = BeautifulSoup(req.text,features="html.parser")
    package_version = soup.find(class_type,class_=key_str)
    dd = package_version.text.strip()
    data = dd.split()
    return data

def parse_link(link_addr, class_type, key_str):
    req = requests.get(url=link_addr)
    req.encoding = "utf-8"
    html=req.text
    soup = BeautifulSoup(req.text,features="html.parser")
    search_data = soup.find(class_type,class_=key_str)
    search_data_2 = search_data.find("a")
    dl_link_get = search_data_2.get("href")
    return dl_link_get

def get_dl_addr_page():
    link_address = "https://pypi.org/project/VisionFive.gpio/#history"
    key_str = "release__version"
    class_key = "p"
    data_get = parse_data(link_address, class_key, key_str)
    latest_version = data_get[0]

    dl_addr_page
 = "https://pypi.org/project/VisionFive.gpio/{}/#files".format(latest_version)

    return dl_addr_page
```

```
def get_dl_addr_of_latest_version(link_addr):
    key_str = "card file__card"
    class_key = "div"
    addr_get = parse_link(link_addr, class_key, key_str)

    return addr_get

def main():
    dl_addr_p =  get_dl_addr_page()
    whl_dl_addr = get_dl_addr_of_latest_version(dl_addr_p)

    whl_name = whl_dl_addr.split("/")[-1]
    whl_name_suffix = os.path.splitext(whl_name)[-1]
    whl_name_prefix = os.path.splitext(whl_name)[0]
    whl_name_prefix_no_platform = whl_name_prefix[0: len(whl_name_prefix) - 3]
    new_platform = "linux_riscv64"

    rename_whl_name = "{}{}{}".format(whl_name_prefix_no_platform, new_platform,
 whl_name_suffix)

    wget.download(whl_dl_addr, out=rename_whl_name)

    os.system("pip install " + rename_whl_name)
    os.system("rm -rf " + rename_whl_name)


if __name__ == '__main__':
    sys.exit(main())
```

3. 执行演示代码

执行以下操作，以在昉·星光 2的Debian系统上运行演示代码：

1. 找到测试代码`LED_Matrix.py`所在的目录：

    a. 执行以下命令以获取`VisionFive.gpio`所在的目录：

    ```
    pip show VisionFive.gpio
    ```

    示例结果：

    ```
    Location: /usr/local/lib64/python3.9/site-packages
    ```

    注：

    实际输出取决于应用的安装方式。

    b. 如前一步输出中所示，执行以下操作进入目录`/usr/local/lib64/python3.9/site-packages`：

    ```
    cd /usr/local/lib64/python3.9/site-packages
    ```

    c. 执行以下命令进入`sample-code`目录：

    ```
    cd ./VisionFive/sample-code/
    ```

2. 在`sample-code`目录下，执行以下命令以运行演示代码：

    ```
    sudo python LED_Matrix.py
    ```

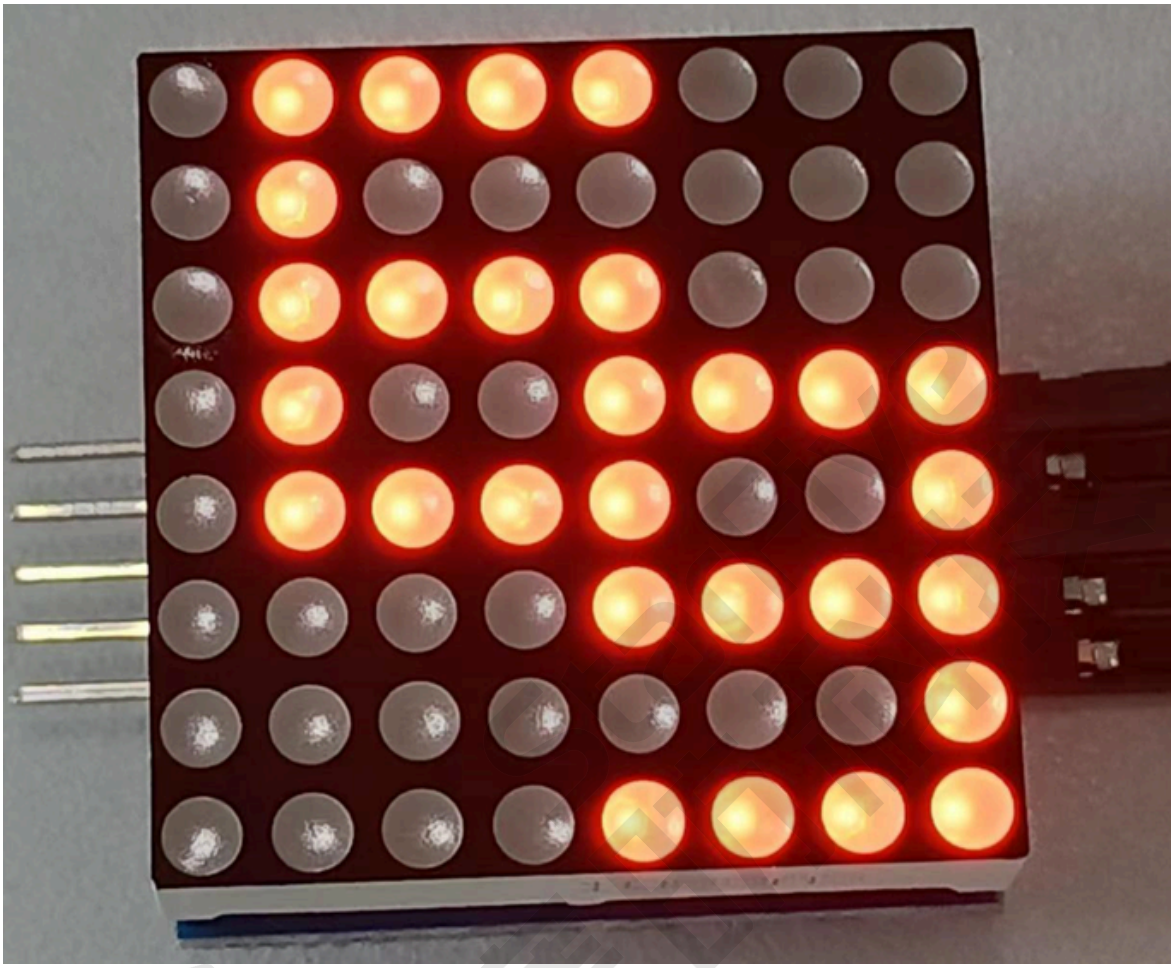
    或者，您也可以执行以下命令：

    ```
    sudo python3 LED_Matrix.py
    ```

    结果：

Led矩阵模块显示出赛昉科技徽标。

图 3-1 示例输出



4. 演示源代码

本演示中的资源代码仅作为参考。

`LED_Matrix.py`:

```
'''
Please make sure the LED Dot Matrix is connected to the correct pins.
The following table describes how to connect LED Dot Matrix to the 40-pin header.
----------------------------------------
___MAX7219_______Pin Number_____Pin Name
    VCC             2           5V Power
    GND             34           GND
    DIN             40          GPIO0
    CS              38          GPIO2
    CLK             36          GPIO4
----------------------------------------
'''

import VisionFive.gpio as GPIO
import sys
import time

DIN = 0
CS  = 2
CLK = 4
#Configure the direction of DIN, CS, and CLK as out.
GPIO.setup(DIN, GPIO.OUT)
GPIO.setup(CS, GPIO.OUT)
GPIO.setup(CLK, GPIO.OUT)

#Display logo data.
buffer = ['01111000', '01000000', '01111000', '01001111', '01111001', '00001111', '000000001', '00001111']

#LED turn off data.
buffer_off = ['0', '0', '0', '0', '0', '0', '0', '0']

def sendbyte(bytedata):
    for bit in range(0, 8):
        if ((bytedata << bit) & 0x80):
            GPIO.output(DIN, GPIO.HIGH)
        else:
            GPIO.output(DIN, GPIO.LOW)

        #Configure the voltage level of CLK as high.
        GPIO.output(CLK, GPIO.HIGH)
        #Configure the voltage level of CLK as low.
        GPIO.output(CLK, GPIO.LOW)


def WriteToReg(regaddr, bytedata):
    #Configure the voltage level of cs as high.
    GPIO.output(CS, GPIO.HIGH)
    #Configure the voltage level of led_pin as low.
    GPIO.output(CS,GPIO.LOW)
    GPIO.output(CLK, GPIO.LOW)
    sendbyte(regaddr)
    sendbyte(bytedata)
    GPIO.output(CS, GPIO.HIGH)

def WriteALLReg():
    time.sleep(0.1)
    for i in range(0, 8):
        #Write data to register address.Finally the LED matrix displays StarFive logo.
        WriteToReg(i+1, int(buffer[i], 2))
    time.sleep(5)

    #Display logo.
    for i in range(0, 10):
        for j in range(0, 8):
```

```
            #Write data to the register address.Finally turn off the LED matrix.
            WriteToReg(i+1, int(buffer_off[i], 2))
        time.sleep(0.1)
        for j in range(0, 8):
            #Write data to the register address.Finally the LED matrix displays with StarFive logo.
            WriteToReg(i+1, int(buffer[i], 2))
        time.sleep(0.1)

def initData():
    WriteToReg(0x09, 0x00) #Set the decode mode.
    WriteToReg(0x0a, 0x03) #Set the brightness.
    WriteToReg(0x0b, 0x07) #Set the scan limitation.
    WriteToReg(0x0c, 0x01) #Set the power mode.
    WriteToReg(0x0f, 0x00)

def main():
    initData()
    while True:
        WriteALLReg()

if __name__ == "__main__":
    sys.exit(main())
```

5. 资源下载

点击本栏找到所有的代码下载资源。

本页包括所有赛昉科技提供的代码下载资源。

- RVspace Wiki
- 应用中心
- 文档中心
- 技术论坛
- 昉·星光 2 GitHub代码仓
- 昉·星光 2 Debian操作系统下载
- 代码下载（赛昉科技官方GitHub页面）
- 所有开源技术文档

6. 立即购买

点击本栏获取在线购买链接和配件购买链接。

购买单板计算机

点击以下页面，您可以找到所在地区的经销商，或覆盖全球的销售渠道，以购买昉·星光 2单板计算机。

- 购买昉·星光2开发板

购买配件

点击以下页面，您可以找到所有昉·星光 2单板计算机已验证适配的配件及其购买链接。

- 购买配件