StarFive
赛昉科技

使用昉·星光 2的SPI点亮LCD屏幕

Python语言版本
应用说明
版本：1.12
日期：2025/04/27
Doc ID: VisionFive 2-ANCH-006

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。上海赛昉半导体科技有限公司（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： 中国（上海）自由贸易试验区盛夏路61弄张润大厦2号电梯楼层5层（实际楼层4层）06室

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SPI制作带有指定图片的2.4inch LCD显示器的步骤。

修订历史

Table 0-1 修订历史

版本	发布说明	修订
1.12	2025/04/27	修正演示源代码 *(on page 16)*中第二步目录。
1.11	2024/06/18	• 在执行演示代码 *(on page 13)*新增一个步骤。 • 更新了演示源代码 *(on page 16)*。
1.1	2023/06/08	• 在40-Pin GPIO Header定义 *(on page 7)*增加注释。 • 在准备软件 *(on page 10)*中更新安装方式。 • 新增资源下载 *(on page 22)*和立即购买 *(on page 23)*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:

- **Tip:**
  建议如何在某个主题或步骤中应用信息。
- **Note:**
  解释某个特例或阐释一个重要的点。
- **Important:**
  指出与某个主题或步骤有关的重要信息。
- **CAUTION:**
  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。
- **Warning:**
  表明某个操作或步骤可能导致物理伤害或硬件损坏。

Contents

List of Tables 5
List of Figures 6
法律声明 2
前言 3
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 10
3. 执行演示代码 13
4. 演示源代码 16
5. 资源下载 22
6. 立即购买 23

List of Tables

Table 0-1 修订历史 3
Table 2-1 硬件准备 8
Table 2-2 将2.4inch LCD模块连接到40-Pin Header上 8

List of Figures

Figure 1-1 40-Pin GPIO Header定义 7
Figure 2-1 将2.4inch LCD模块连接到40-Pin Header上 10
Figure 3-1 示例输出 14
Figure 3-2 示例输出 15

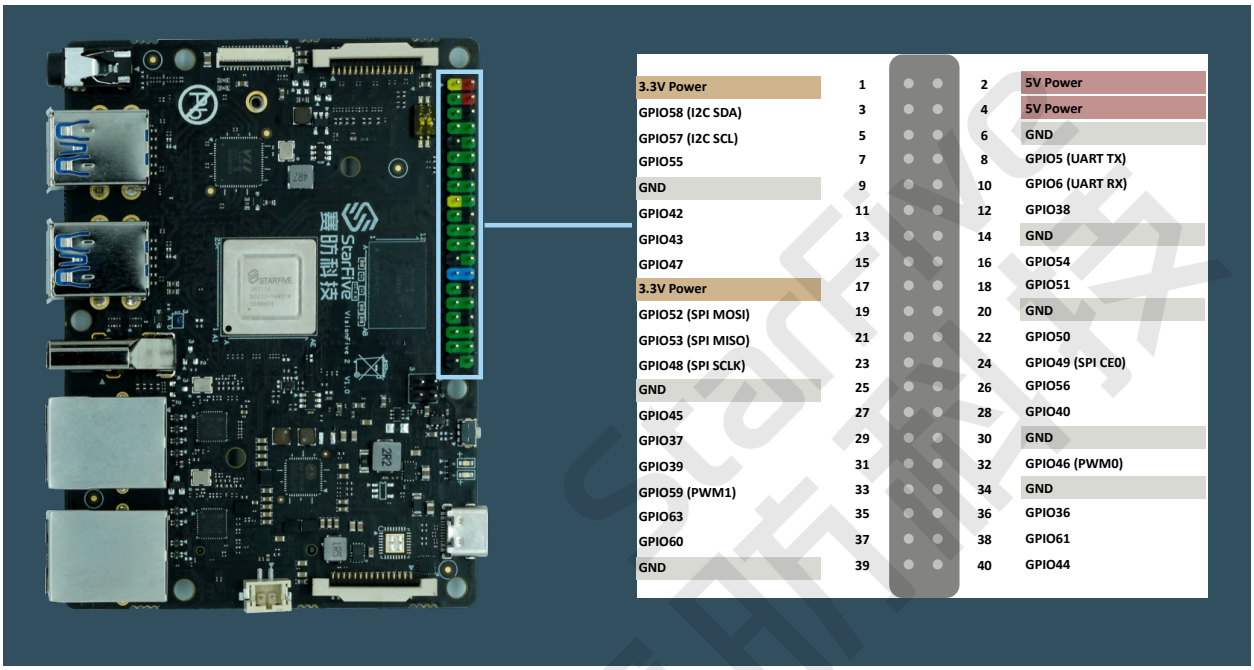
1. 产品简介

本应用说明提供使用昉·星光 2的SPI制作带有指定图片的2.4inch LCD显示器的步骤。

1.1. 40-Pin GPIO Header定义

下图显示了40-pin GPIO Header的位置：

Figure 1-1 40-Pin GPIO Header定义



Note:
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：

Table 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。
SPI LCD		• 2.4英寸LCD模块 • 杜邦线	-

Note:
*: 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LED连接到40-Pin Header上：

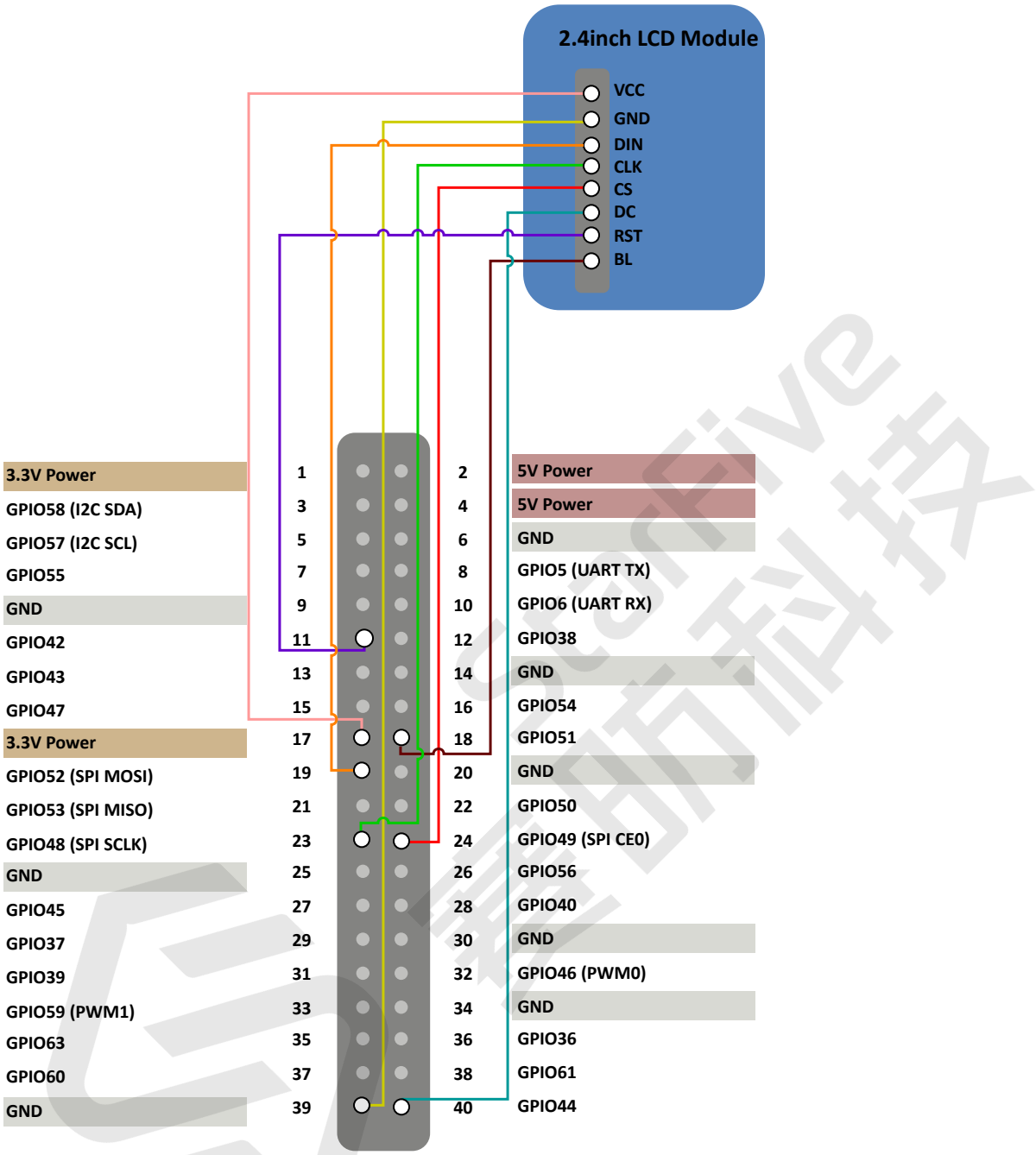
Table 2-2 将2.4inch LCD模块连接到40-Pin Header上

2.4英寸LCD模块	40-Pin GPIO Header	
	Pin Number	Pin Name
VCC	17	3.3V 电压
GND	39	GND

Table 2-2 将2.4inch LCD模块连接到40-Pin Header上 (continued)

2.4英寸LCD模块	40-Pin GPIO Header	
	Pin Number	Pin Name
DIN	19	GPIO52 (SPI MOSI)
CLK	23	GPIO48 (SPI SCLK)
CS	24	GPIO49 (SPI CE0)
DC	40	GPIO44
RST	11	GPIO42
BL	18	GPIO51

Figure 2-1 将2.4inch LCD模块连接到40-Pin Header上



2.3. 准备软件

确认按照以下步骤进行操作:

Note:
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：

Note:
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`2.4inch_LCD_demo`所在的目录：

    a. 执行以下命令安装依赖：

    ```
    pip install pillow
    ```

    b. 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
    ```

    Note:

    实际输出取决于应用的安装方式。

    c. 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
    ```

    d. 执行以下命令进入`sample-code`目录：

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

    e. 执行以下命令，以进入测试代码`2.4inch_LCD_demo`所在的目录：

    ```
    cd ./lcddemo/example/
    ```

2. 在`example`目录下，执行以下命令以运行演示代码：

    ```
    sudo python 2.4inch_LCD_demo
    ```

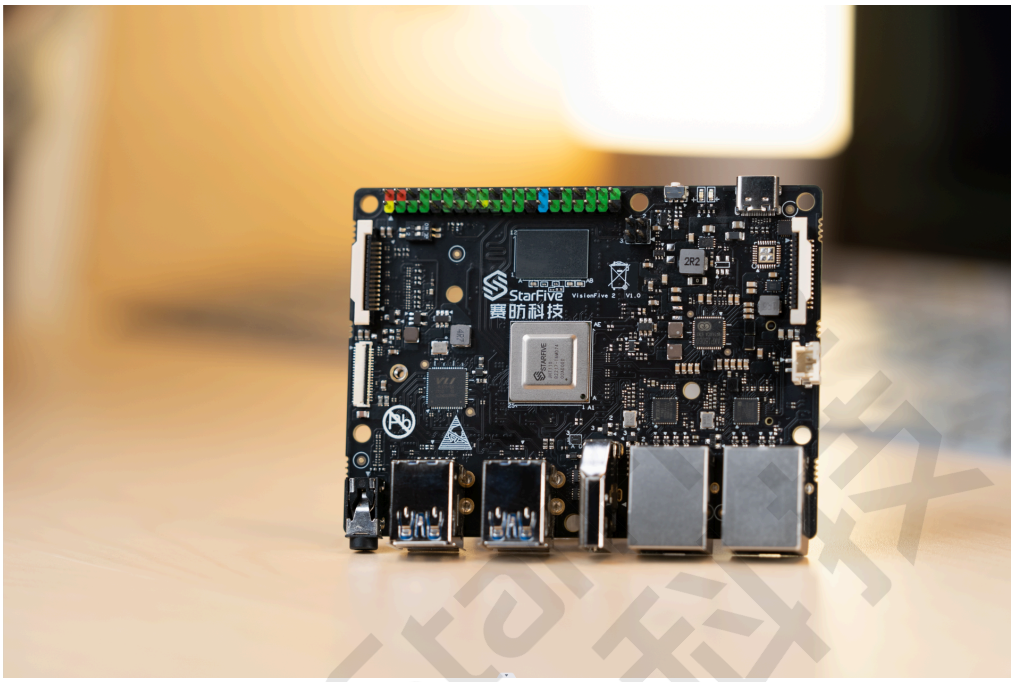
    或者，您也可以执行以下命令：

    ```
    sudo python3 2.4inch_LCD_demo
    ```

    结果：

- 在2.4英寸LCD显示器上：
  - 首先，以下带有赛昉科技徽标的图片将会显示两秒钟。

Figure 3-1 示例输出



  - 然后依次显示以下两个官方示例图。

Figure 3-2 示例输出

◦ 终端输出如下:

```
-----------lcd demo-----------
Set SPI mode successfully
spi mode: 0x0
bits per word: 8
max speed: 40000000 Hz(40000 kHz)
2022-07-04 16:40:40
2022-07-04 16:40:41
2022-07-04 16:40:41
2022-07-04 16:40:42
2022-07-04 16:40:42
2022-07-04 16:40:43
2022-07-04 16:40:44
2022-07-04 16:40:44
2022-07-04 16:40:45
```

该输出表示:

- SPI模式设置成功
- SPI模式
- 上述三张图片显示的日期和时间

4. 演示源代码

本演示中的资源代码仅作为参考。

`2.4inch_LCD_demo.py`:

```
#!/usr/bin/python
"""
Please make sure the 2.4inch LCD Moudle is connected to the correct pins.
The following table describes how to connect the 2.4inch LCD Module to the 40-pin header.
-------------------------------------------------
__2.4inch LCD Module___Pin Number_____Pin Name
    VCC                   17           3.3 V Power
    GND                   39             GND
    DIN                   19           SPI MOSI
    CLK                   23           SPI SCLK
    CS                    24           SPI CE0
    DC                    40            GPIO44
    RST                   11            GPIO42
    BL                    18            GPIO51
-------------------------------------------------
"""

import os
import sys
import time
import logging
from PIL import Image

sys.path.append("..")

import VisionFive.boardtype as board_t
from lib import LCD2inch4_lib

"""
Demo modification ans new function description
-------------------------------------------------------------
  I.   add the clear() function to fill LCD screen with white
  II.  give a hexadecimal value of white
  III. cycle through multiple pictures
-------------------------------------------------------------
"""

WHITE = 0xFF


def main():
    print("-----------lcd demo-------------")

    # Determining cpu Type: 1 means visionfive1; 2 means visionfive 2
    vf_t = board_t.boardtype()
    if vf_t == 1:
        SPI_DEVICE = "/dev/spidev0.0"
    elif vf_t == 2:
        SPI_DEVICE = "/dev/spidev1.0"
    else:
        print("This module can only be run on a VisionFive board!")
        return 0

    """The initialization settings of 2inch and 2.4inch are distinguished"""
    disp = LCD2inch4_lib.LCD_2inch4(11, 40, SPI_DEVICE)
    # disp.lcd_init()
    disp.lcd_init_2inch4()

    disp.lcd_clear(WHITE)

    if vf_t == 1:
        image = Image.open("./visionfive.bmp")
    elif vf_t == 2:
        image = Image.open("./visionfive2.png")
```

```
    else:
        return

    disp.lcd_ShowImage(image, 0, 0)
    time.sleep(2)

    """add the part of displaying pictures circularly"""
    while True:
        try:
            print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(time.time())))

            """rotate the picture 90 degrees anticlockwise"""
            """to keep consistent with the display direction of other pictures"""
            image = Image.open("./LCD_2inch4_parrot.bmp")
            image = image.transpose(Image.Transpose.ROTATE_90)
            disp.lcd_ShowImage(image, 0, 0)
            time.sleep(0.5)

            image = Image.open("./LCD_2inch.jpg")
            disp.lcd_ShowImage(image, 0, 0)
            time.sleep(0.5)

            if vf_t == 1:
                image = Image.open("./visionfive.bmp")
            elif vf_t == 2:
                image = Image.open("./visionfive2.png")
            else:
                return
            disp.lcd_ShowImage(image, 0, 0)
            time.sleep(0.5)

        except KeyboardInterrupt:
            break

    print("Exit demo!")


if __name__ ==
```

LCD2inch4_lib.py:

```
import os
import sys
import time
import logging
import VisionFive.spi as spi
import VisionFive.gpio as gpio
import numpy as np
from PIL import Image, ImageDraw, ImageFont


class LCD_2inch4:
    width = 240
    height = 320

    def __init__(self, rst_pin, dc_pin, dev):
        gpio.setmode(gpio.BOARD)

        self.rstpin = rst_pin
        self.dcpin = dc_pin
        self.spidev = dev
        spi.getdev(self.spidev)

        """Reset the maximum clock frequency of communication"""
        """The display speed of the picture is positively correlated with the clock frequency"""
        # spi.setmode(500000, 0, 8)
        spi.setmode(40000000, 0, 8)
        gpio.setup(self.rstpin, gpio.OUT)
        gpio.setup(self.dcpin, gpio.OUT)

    def __del__(self):
        spi.freedev()
```

```
    """add a short delay for each change of electrical level"""

    def lcd_reset(self):
        gpio.output(self.rstpin, gpio.HIGH)
        time.sleep(0.01)
        gpio.output(self.rstpin, gpio.LOW)
        time.sleep(0.01)
        gpio.output(self.rstpin, gpio.HIGH)
        time.sleep(0.01)

    def lcd_spisend(self, data):
        spi.transfer(data)

    def lcd_sendcmd(self, cmd):
        gpio.output(self.dcpin, gpio.LOW)
        spi.transfer(cmd)

    def lcd_senddata(self, data):
        gpio.output(self.dcpin, gpio.HIGH)
        spi.transfer(data)

    """write multiple bytes"""

    def lcd_sendnbytes(self, data):
        gpio.output(self.dcpin, gpio.HIGH)
        spi.write(data)

    """common registers' initialization of 2.4inch LCD module"""

    def lcd_init_2inch4(self):
        self.lcd_reset()

        self.lcd_sendcmd(0x11)  # sleep out

        self.lcd_sendcmd(0xCF)  # Ppower Control B
        self.lcd_senddata(0x00)
        self.lcd_senddata(0xC1)
        self.lcd_senddata(0x30)

        self.lcd_sendcmd(0xED)  # Power on sequence control
        self.lcd_senddata(0x64)
        self.lcd_senddata(0x03)
        self.lcd_senddata(0x12)
        self.lcd_senddata(0x81)

        self.lcd_sendcmd(0xE8)  # Driver Timing Control A
        self.lcd_senddata(0x85)
        self.lcd_senddata(0x00)
        self.lcd_senddata(0x79)

        self.lcd_sendcmd(0xCB)  # Power Control A
        self.lcd_senddata(0x39)
        self.lcd_senddata(0x2C)
        self.lcd_senddata(0x00)
        self.lcd_senddata(0x34)
        self.lcd_senddata(0x02)

        self.lcd_sendcmd(0xF7)  # Pump ratio control
        self.lcd_senddata(0x20)

        self.lcd_sendcmd(0xEA)  # Driver Timing Control B
        self.lcd_senddata(0x00)
        self.lcd_senddata(0x00)

        self.lcd_sendcmd(0xC0)  # Power Control 1
        self.lcd_senddata(0x1D)  # VRH[5:0]

        self.lcd_sendcmd(0xC1)  # Power Control 2
        self.lcd_senddata(0x12)  # SAP[2:0],BT[3:0]

        self.lcd_sendcmd(0xC5)  # VCOM Control 1
```

```
        self.lcd_senddata(0x33)
        self.lcd_senddata(0x3F)

        self.lcd_sendcmd(0xC7)  # VCOM Control 2
        self.lcd_senddata(0x92)

        self.lcd_sendcmd(0x3A)  # COLMOD:Pixel Format Set
        self.lcd_senddata(0x55)

        self.lcd_sendcmd(0x36)  # Memory Access Control
        self.lcd_senddata(0x08)

        self.lcd_sendcmd(0xB1)  # Frame Rate Control(In Normal Mode/Full Colors)
        self.lcd_senddata(0x00)
        self.lcd_senddata(0x12)

        self.lcd_sendcmd(0xB6)  # Display Function Control
        self.lcd_senddata(0x0A)
        self.lcd_senddata(0xA2)

        self.lcd_sendcmd(0x44)  # Set_Tear_Scanline
        self.lcd_senddata(0x02)

        self.lcd_sendcmd(0xF2)  # 3Gamma Function Disable
        self.lcd_senddata(0x00)

        self.lcd_sendcmd(0x26)  # Gamma curve selected
        self.lcd_senddata(0x01)

        self.lcd_sendcmd(0xE0)  # Set Gamma
        self.lcd_senddata(0x0F)
        self.lcd_senddata(0x22)
        self.lcd_senddata(0x1C)
        self.lcd_senddata(0x1B)
        self.lcd_senddata(0x08)
        self.lcd_senddata(0x0F)
        self.lcd_senddata(0x48)
        self.lcd_senddata(0xB8)
        self.lcd_senddata(0x34)
        self.lcd_senddata(0x05)
        self.lcd_senddata(0x0C)
        self.lcd_senddata(0x09)
        self.lcd_senddata(0x0F)
        self.lcd_senddata(0x07)
        self.lcd_senddata(0x00)

        self.lcd_sendcmd(0xE1)  # Set Gamma
        self.lcd_senddata(0x00)
        self.lcd_senddata(0x23)
        self.lcd_senddata(0x24)
        self.lcd_senddata(0x07)
        self.lcd_senddata(0x10)
        self.lcd_senddata(0x07)
        self.lcd_senddata(0x38)
        self.lcd_senddata(0x47)
        self.lcd_senddata(0x4B)
        self.lcd_senddata(0x0A)
        self.lcd_senddata(0x13)
        self.lcd_senddata(0x06)
        self.lcd_senddata(0x30)
        self.lcd_senddata(0x38)
        self.lcd_senddata(0x0F)
        self.lcd_sendcmd(0x29)  # Display on

    def lcd_setPos(self, Xstart, Ystart, Xend, Yend):
        self.lcd_sendcmd(0x2A)
        self.lcd_senddata(Xstart >> 8)
        self.lcd_senddata(Xstart & 0xFF)
        self.lcd_senddata((Xend - 1) >> 8)
        self.lcd_senddata((Xend - 1) & 0xFF)
        self.lcd_sendcmd(0x2B)
        self.lcd_senddata(Ystart >> 8)
```

```
        self.lcd_senddata(Ystart & 0xFF)
        self.lcd_senddata((Yend - 1) >> 8)
        self.lcd_senddata((Yend - 1) & 0xFF)
        self.lcd_sendcmd(0x2C)

    def lcd_clear(self, color):
        """Clear contents of image buffer"""

        _buffer = [color] * (self.width * self.height * 2)

        self.lcd_setPos(0, 0, self.width, self.height)
        gpio.output(self.dcpin, gpio.HIGH)

        """modify the original single byte write to multi byte write"""
        # for i in range(0,len(_buffer)):
        #    self.lcd_spisend(_buffer[i])
        self.lcd_sendnbytes(_buffer)

    def lcd_ShowImage(self, Image, Xstart, Ystart):
        """Set buffer to value of Python Imaging Library image."""
        """Write display buffer to physical display"""
        imwidth, imheight = Image.size

        if imwidth == self.height and imheight == self.width:
            img = np.asarray(Image)
            pix = np.zeros((self.width, self.height, 2), dtype=np.uint8)
            # RGB888 >> RGB565
            pix[..., [0]] = np.add(
                np.bitwise_and(img[..., [0]], 0xF8), np.right_shift(img[..., [1]], 5)
            )
            pix[..., [1]] = np.add(
                np.bitwise_and(np.left_shift(img[..., [1]], 3), 0xE0),
                np.right_shift(img[..., [2]], 3),
            )
            pix = pix.flatten().tolist()

            self.lcd_sendcmd(
                0x36
            )  # define read/write scanning direction of frame memory
            self.lcd_senddata(0x78)
            self.lcd_setPos(0, 0, self.height, self.width)

            gpio.output(self.dcpin, gpio.HIGH)

            """modify the original single byte write to multi byte write"""
            # for i in range(0,len(pix),1):
            #    self.lcd_spisend(pix[i])
            self.lcd_sendnbytes(pix)
        else:
            img = np.asarray(Image)
            pix = np.zeros((imheight, imwidth, 2), dtype=np.uint8)

            pix[..., [0]] = np.add(
                np.bitwise_and(img[..., [0]], 0xF8), np.right_shift(img[..., [1]], 5)
            )
            pix[..., [1]] = np.add(
                np.bitwise_and(np.left_shift(img[..., [1]], 3), 0xE0),
                np.right_shift(img[..., [2]], 3),
            )

            pix = pix.flatten().tolist()

            self.lcd_sendcmd(0x36)
            self.lcd_senddata(0x08)
            self.lcd_setPos(0, 0, self.width, self.height)

            gpio.output(self.dcpin, gpio.HIGH)

            """modify the original single byte write to multi byte write"""
            # for i in range(0,len(pix)):
```

```
            #    self.lcd_spisend(pix[i])
            self.lcd_sendnbytes(pix)
```

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