StarFive 赛昉科技

Using VisionFive 2 SPI to Support LCD Display

with Python

Application Note

Version: 1.2

Date: 2025/08/06

Doc ID: VisionFive2-ANEN-006

Legal Statements

Important legal notice before reading this documentation.

PROPRIETARY NOTICE

Information in this document is provided "as is," with all faults. Contents may be periodically updated or revised due to the product development. Shanghai StarFive Semiconductor Co., Ltd. (hereinafter "StarFive") reserves the right to make changes without further notice to any products herein.

StarFive expressly disclaims all warranties, representations, and conditions of any kind, whether express or implied, including, but not limited to, the implied warranties or conditions of merchantability, fitness for a particular purpose and non-infringement.

StarFive does not assume any liability rising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages.

Contact Us

Address: Room 506, Building 2, No. 61 Shengxia Rd., China (Shanghai) Pilot Free Trade Zone, Shanghai, 201203, China

Website: http://www.starfivetech.com

Email: sales@starfivetech.com(sales) , support@starfivetech.com(support)

Preface

About this guide and technical support information.

About this document

This application note provides steps to use VisionFive 2's SPI to make a 2.4inch LCD display with specified pictures.

Revision History

Table 0-1 Revision History

Version	Released	Revision
1.2	2025/08/06	Updated the Linux and OS version in Environment Requirements *(on page 8)*. Updated the steps in Preparing Software *(on page 10)*. Updated the steps in Running Demo Code *(on page 13)*.
1.12	2025/04/27	Updated the fiepath as `example` in Running Demo Code *(on page 13)*.
1.11	2024/06/18	• Added a new step in Running Demo Code *(on page 13)*. • Updated the Demo Source Code *(on page 16)*.
1.1	2023/06/08	• Added a note in 40-Pin GPIO Header Definition *(on page 7)*. • Updated the method for installing `VisionFive.gpio` package in Preparing Software *(on page 10)*. • Added Resources *(on page 22)* and Buy Now *(on page 23)* chapters.
1.0	2022/11/30	The first official release.

Notes and notices

The following notes and notices might appear in this guide:

- **Tip:**
  Suggests how to apply the information in a topic or step.
- **Note:**
  Explains a special case or expands on an important point.
- **Important:**
  Points out critical information concerning a topic or step.
- **CAUTION:**
  Indicates that an action or step can cause loss of data, security problems, or performance issues.
- **Warning:**
  Indicates that an action or step can result in physical harm or cause damage to hardware.

Contents

List of Tables

List of Figures

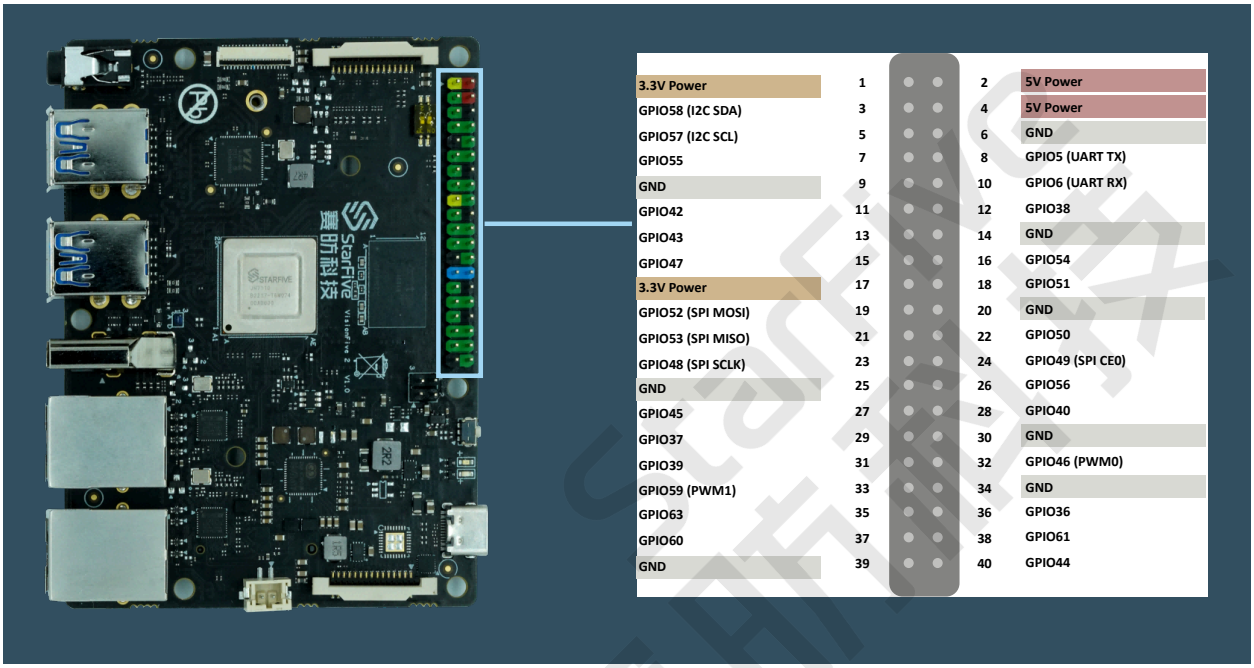
1. Introduction

This application note provides steps to use VisionFive 2's SPI to make a 2.4inch LCD display with specified pictures.

1.1. 40-Pin GPIO Header Definition

The following figure shows the location of the 40-pin header on VisionFive 2.

Figure 1-1 40-Pin GPIO Header Definition



Note:
The multiplexed pin has been initialized and cannot be used as a general GPIO.

2. Preparation

Before executing the demo program, make sure you prepare the following:

2.1. Environment Requirements

The environment requirements are as follows:

- Linux Kernel: Linux 6.6
- OS: Debian 13
- SBC: VisionFive 2
- SoC: JH-7110

2.2. Preparing Hardware

Prepare the following hardware items before running the demo code:

Table 2-1 Hardware Preparation

Type	M/O*	Item	Notes
General	M	VisionFive 2 Board	-
General	M	• 32 GB (or more) micro-SD card • Micro-SD card reader • Computer (Windows/Mac OS/Linux) • USB to serial converter (3.3 V I/O) • Ethernet cable • Power adapter (5 V / 3 A) • USB Type-C Cable	These items are used for flashing Debian OS into a Micro-SD card.
SPI LCD		• 2.4inch LCD Module • Dupont Line	-

Note:
*: M: Mandatory, O: Optional

2.2.1. Hardware Setup

The following table and figure describe how to connect LCD to the 40-pin header:

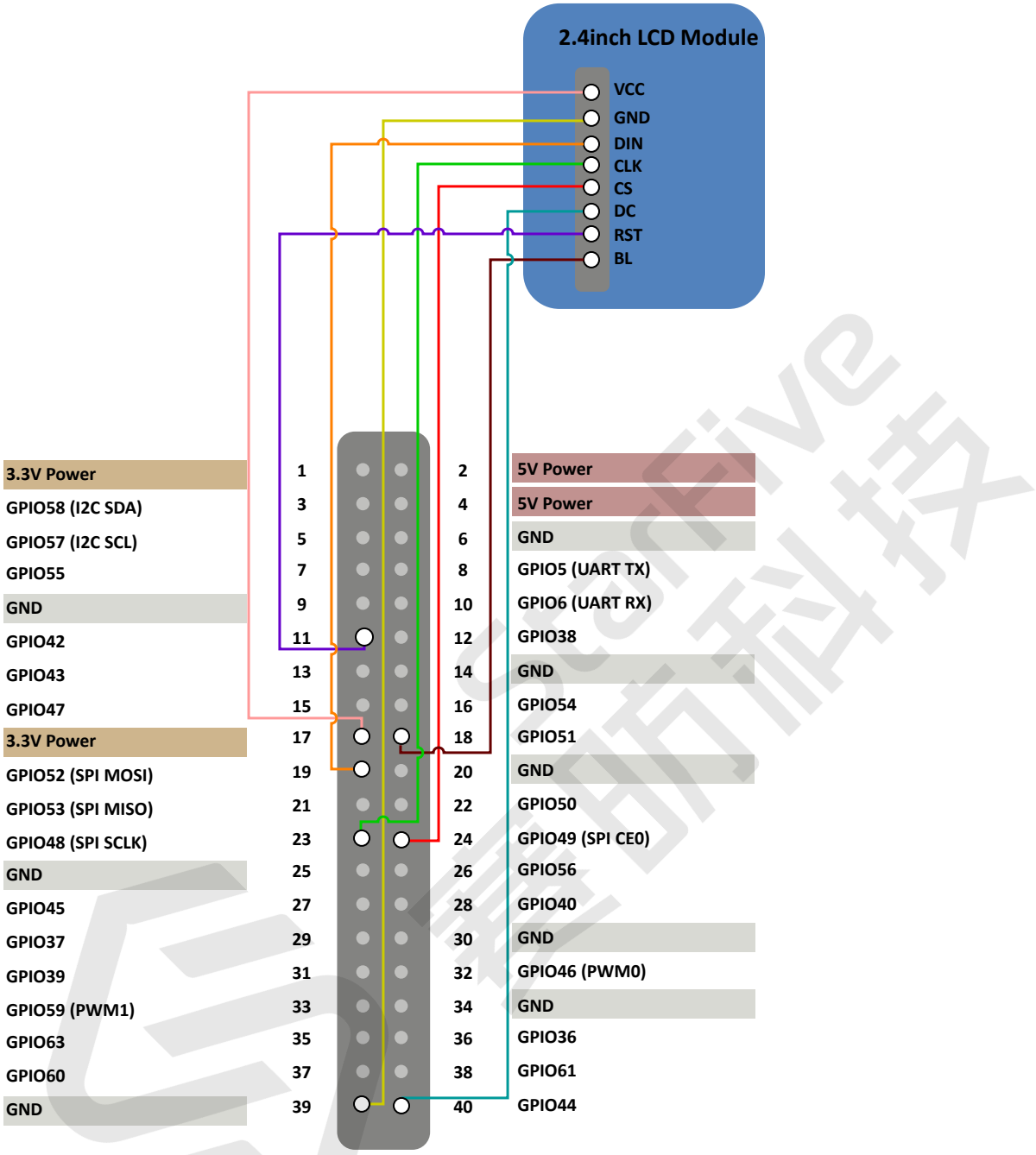
Table 2-2 Connect the 2.4inch LCD to the 40-Pin Header

2.4inch LCD Module	40-Pin GPIO Header	
	Pin Number	Pin Name
VCC	17	3.3V Power
GND	39	GND

Table 2-2 Connect the 2.4inch LCD to the 40-Pin Header (continued)

2.4inch LCD Module	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 Connect the 2.4inch LCD to the 40-Pin Header



2.3. Preparing Software

Make sure the following procedures are performed:

Note:
The python project, `VisionFive.gpio`, is applicable for VisionFive, VisionFive 2 and JH-7110 EVB.

1. Flash Debian OS into a Micro-SD card as described in the *Flashing Fedora OS to a Micro-SD Card* section in the *VisionFive 2 Single Board Computer Quick Start Guide*.
2. Log into the Debian and make sure VisionFive 2 is connected to the Internet. For detailed instructions, refer to the Using SSH over Ethernet or *Using a USB to Serial Converter* section in the *VisionFive 2 Single Board Computer Quick Start Guide*.
3. Extend the partition on Debian as described in *Extend Partition* in the *VisionFive 2 Single Board Computer Quick Start Guide*.
4. Execute the following command to install and create a Python3 Virtual Environment on Debian:

```
sudo apt install python3-venv
python3 -m venv myvenv
```

Note:

You may rename “myvenv” according to your preference.

5. Execute the `pip` command on VisionFive 2 Debian to install the `VisionFive.gpio` package:

Note:
Due to the fact that `pypi.org` official website does not yet support uploading `whl` installation packages for the RISC-V platform, so it cannot directly execute `python3 -m pip install VisionFive.gpio` command to install online.

Please follow the steps below to install the `VisionFive.gpio` package.

a. Execute the following command to install dependent package within the newly created virtual environment:

```
sudo apt install libxml2-dev libxslt-dev
source ./myvenv/bin/activate
python3 -m pip install requests wget bs4
```

b. Execute the following command to run the installation script `Install_VisionFive_gpio.py`:

```
python3 Install_VisionFive_gpio.py
```

The installation script codes are as follows:

```
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):
    version_list = []
    req = requests.get(url=link_addr)
    req.encoding = "utf-8"
    html = req.text
    soup = BeautifulSoup(req.text, features="html.parser")
    search_data = soup.find_all(class_type, class_=key_str)
    for i in range(0, len(search_data)):
        search_data[i] = search_data[i].find("a").get("href)
        version_list.append(search_data[i].split("cp")[-1].split("-")[0])
```

```
    python_version = sys.version
    python_version = python_version.split(".")[0] + python_version.split(".")[1]

    for i in range(0, len(search_data)):
        if python_version == version_list[i]:
            return search_data[i]

    return search_data[0]

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())
```

c. (Optional) Exit the Python3 Virtual Environment.

```
deactivate
```

3. Running Demo Code

To run the demo code, perform the following on VisionFive 2 Debian:

1. Locate to the directory where the test code, `2.4inch_LCD_demo`, exists:

    a. Execute the following command to install dependency:

    ```
    python3 -m pip install pillow
    ```

    b. Execute the following command to get the directory where `VisionFive.gpio` exists:

    ```
    python3 -m pip show VisionFive.gpio
    ```

    Result:

    ```
    Location: /home/user/myvenv/lib/python3.11/site-packages
    ```

    Note:

    The actual output depends on how the application is installed.

    c. Execute the following to enter the directory, for example, `/home/user/myvenv/lib/python3.11/site-packages` as indicated in the previous step output:

    ```
    cd /home/user/myvenv/lib/python3.11/site-packages
    ```

    d. Execute the following command to enter the `sample-code` directory:

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

    e. Execute the following command to enter the directory where the test code, `2.4inch_LCD_demo`, exists:

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

2. Under the `example` directory, execute the following command to execute the demo code:

    ```
    sudo python 2.4inch_LCD_demo
    ```

    Alternatively, you can execute the following command:

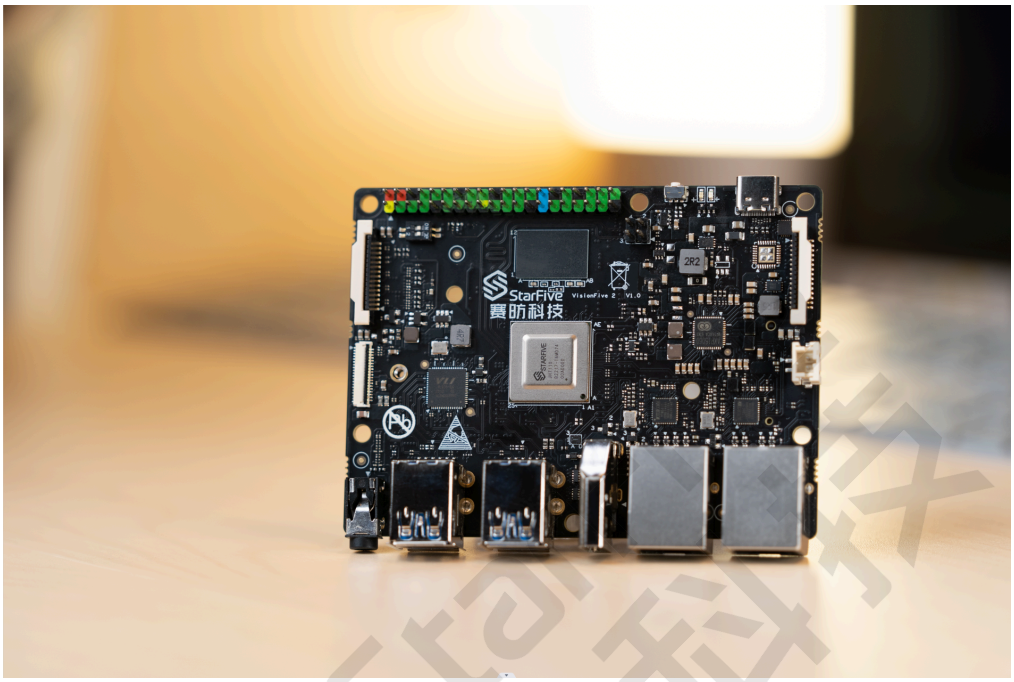
    ```
    sudo python3 2.4inch_LCD_demo
    ```

    Result:

◦ On the 2.4inch LCD:

▪ First, the following picture with the StarFive logo will be displayed for two seconds.

Figure 3-1 Example Output



▪ Then the following two official example figures will be displayed in turn.

Figure 3-2 Example Output

◦ The terminal output is as the following.

```
-----------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
```

The output indicates:

- that the SPI mode is set successfully
- the SPI mode
- the date and time when all the above three figures are displayed

3. (Optional) Exit the Python3 Virtual Environment.

```
deactivate
```

4. Demo Source Code

The Python source code of this demo is provided for reference purposes only.

`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. Resources

Click on this tab to find all SBC relevant resources.

StarFive provides the following resources to guide you through an extraordinary experience on using the VisionFive 2 SBC.

- RVspace Wiki
- Application Center
- Documentation Center
- Technical Forum
- VisionFive 2 GitHub Repository
- VisionFive 2 Debian OS Download
- Code download
- View All PDF Documents

6. Buy Now

Click on this tab to find all the online shops and compatible accessories.

Buy SBC

Use the following page to find your nearest sales channel or the global channels for purchasing a VisionFive 2 Single Board Computer (SBC).

- Buy VisionFive 2

Buy Parts

Use the following page to find the parts that are tested as compatible with VisionFive 2.

- Buy Accessory