2.9inch e-Paper Module

From Waveshare Wiki

Contents

2.9inch e-Paper

296x128, 2.9inch E-Ink display raw panel

2.9inch e-Paper Module

296x128, 2.9inch E-Ink display module, SPI interface

Primary Attribute

Category: OLEDs / LCDs, LCD

Brand: Waveshare

Website

English: Waveshare website (http://www.waveshare.com/product/2.9inch-e-paper-module.htm)

Chinese: 官方中文站点 (http://www.waveshare.net/shop/2.9inch-e-paper-module.htm)

Onboard Interfaces

SPI

Introduction

Note: The raw panel require a driver board, If you are the first time use this e-Paper, we recommend you to buy the HAT version or buy more one driver hat for easy use, otherwise you need to make the driver board yourself. And this instruction is based on the version with PCB or driver board.

296x128, 2.9inch E-Ink display module, SPI interface

More (http://www.waveshare.com/product/2.9inch-e-paper-module.htm)

Video

Related Products

- 4.3inch e-Paper UART Module
- 7.5inch e-Paper HAT
- 7.5inch e-Paper HAT (B)
- 7.5inch e-Paper HAT (C)
- 5.83inch e-Paper HAT
- 5.83inch e-Paper HAT (B)
- 5.83inch e-Paper HAT (C)
- 4.2inch e-Paper Module
- 4.2inch e-Paper Module (B)
- 4.2inch e-Paper Module (C)
- **2.9inch e-Paper Module**
- 2.9inch e-Paper Module (B)
- 2.9inch e-Paper Module (C)
- 2.7inch e-Paper HAT
- 2.7inch e-Paper HAT (B)
- 2.13inch e-Paper HAT
- 2.13inch e-Paper HAT (B)
- 2.13inch e-Paper HAT (C)
- 2.13inch e-Paper HAT (D)
- 1.54inch e-Paper Module
- 1.54inch e-Paper Module (B)
- 1.54inch e-Paper Module (C)
- e-Paper Driver HAT
- E-Paper Shield
- e-Paper ESP8266 Driver Board

Interfaces

VCC	3.3V
GND	GND
DIN	SPI MOSI
CLK	SPI SCK
CS	SPI chip select (Low active)
DC	Data/Command control pin (High for data, and low for command)
RST	External reset pin (Low for reset)
BUSY	Busy state output pin (High for busy)

Working principle

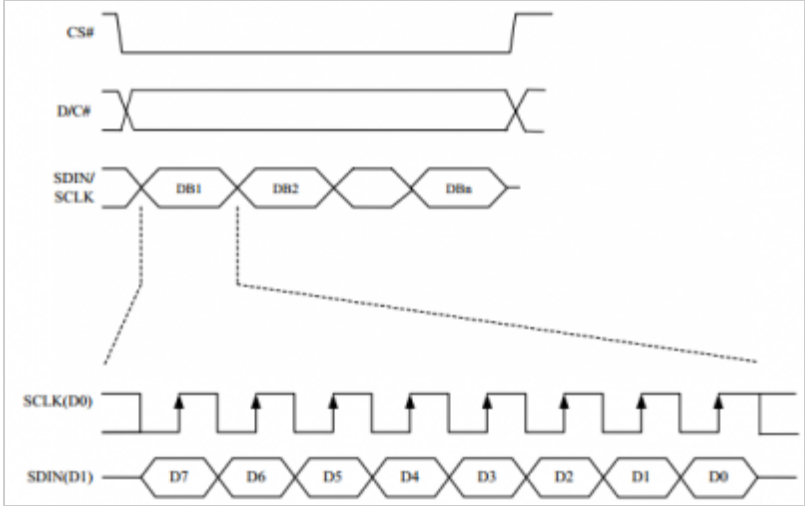
Introduction

This product is an E-paper device adopting the image display technology of Microencapsulated Electrophoretic Display, MED. The initial approach is to create tiny spheres, in which the charged color pigments are suspending in the transparent oil and would move depending on the electronic charge. The E-paper screen display patterns by reflecting the ambient light, so it has no background light requirement. Under sunshine, the E-paper screen still has high visibility with a wide viewing angle of 180 degree. It is the ideal choice for E-reading.

Communication protocol

Note: Different from the traditional SPI protocol, the data line from the slave to the master is hidden since the device only has display requirement.

- CS is slave chip select, when CS is low, the chip is enabled.
- DC is data/command control pin, when DC = 0, write command, when DC = 1, write data.
- SCLK is the SPI communication clock.
- SDIN is the data line from the master to the slave in SPI communication.



SPI timing

SPI communication has data transfer timing, which is combined by CPHA and CPOL.

1. CPOL determines the level of the serial synchronous clock at idle state. When CPOL = 0, the level is Low. However, CPOL has little effect to the transmission.
2. CPHA determines whether data is collected at the first clock edge or at the second clock edge of serial synchronous clock; when CPHL = 0, data is collected at the first clock edge.

- There are 4 SPI communication modes. SPI0 is commonly used, in which CPHL = 0, CPOL = 0.

As you can see from the figure above, data transmission starts at the first falling edge of SCLK, and 8 bits of data are transferred in one clock cycle. In here, SPI0 is in used, and data is transferred by bits, MSB first.

How to use

Working with Raspberry Pi

Installing libraries required

If you want to connect your E-paper screen to Raspberry Pi, you should install some necessary libraries, or else the Demo (click to download) below may work improperly. For more information about how to install the Raspberry Pi libraries, please visit the website: Libraries Installation for RPi.

You can find the detailed presentation about the installations of libraries wiringPi, bcm2835 and python.

Hardware connection

Here is the connection between Raspberry Pi 3B and E-paper.

e-Paper	Raspberry Pi 3B
3.3V	3.3V
GND	GND
DIN	MOSI
CLK	SCLK
CS	CE0
DC	Pin22/GPIO25
RST	Pin11/GPIO17
BUSY	Pin18/GPIO24

Expected result

1. After the corresponding libraries installed, you can copy the relative programs into your Raspberry Pi, and then enter the corresponding file.
   - **BCM2835:** Execute the command: `make`, to compile the code and generate a file `epd`. Execute the command: `sudo ./epd`, the program will run.
   - **WringPi:** Execute the command: `make`, to compile the code and generate a file `epd`. Execute the command: `sudo ./epd`, the program will run.
   - **Python:** Execute the command: `sudo python main.py`
2. The screen displays strings and shapes after whole screen refresh.
3. The screen displays images and the time after partial screen refresh. This demonstrates the partial refreshing capability.

Working with Arduino

Hardware connection

e-Paper	UNO PLUS (3.3V)
3.3V	3V3
GND	GND
DIN	D11
CLK	D13
CS	D10
DC	D9
RST	D8
BUSY	D7

Expected result

1. Copy the files from the directory arduino/libraries of the demo package to documents/arduino/libraries, where can be specified by Arduino IDE --> File --> Preferences --> Sketchbook location.
2. Click the button **Upload** to compile and upload the program to your Arduino board.
3. The screen displays strings and shapes after whole screen refresh.
4. The screen displays images and the time after partial screen refresh. This demonstrates the partial refreshing capability.

Working with the STM32 development board

Here we use the development board XNUCLEO-F103RB. The Demo is base on the library HAL.

Hardware connection

Here is the hardware connection between the development board XNUCLEO-F103RB and E-paper:

e-Paper	XNUCLEO-F103RB
3.3V	3V3
GND	GND
DIN	PA7
CLK	PA5
CS	PB6
DC	PC7
RST	PA9
BUSY	PA8

Expected result

1. Open the Keil project (MDK-ARM/epd-demo.uvprojx)
2. Click **Build** to compile the project.
3. Click **Download** to download the program to the target board.
4. The screen displays strings and shapes after whole screen refresh.

5. The screen displays images and the time after partial screen refresh. This demonstrates the partial refreshing capability.

Code analysis

Here, we will analyze the driving code and take the demos for Raspberry Pi based on WiringPi library as examples.

Hardware interface function

The functions of drive code like DigitalWrite, DigitalRead, SendCommand, SenData and DelayMs call the interface functions which are provided by hardware device (epdif.h, epdif.c, epdif.cpp) to respectively implements the functions that Control IO Level, Read IO Level, Send SPI Command, Send SPI Data and Delay For Millisecond. If you want to port the demo code, you need to implement all the interfaces of epdif (e-paper display interface) according to the corresponding hardware device.

Note that Raspberry Pi uses hardware chip select while transmitting SPI data. So we needn't set the CS pin to LOW before transmitting data, and the code will set it automatically while transmitting. However, for Arduino and STM32, etc. you need to explicitly set the CS pin to LOW with codes to start the SPI transmission of module.

Send Commands and Data (SendCommand and SendData)

SendCommand and SendData are used to send commands and data to module respectively. What the difference between them is that, D/C pin is set to LOW for sending commands and HIGH for sending data. If the D/C pin is LOW, the data transmitted from SPI interface to module will be recognized as commands and executed. If the D/C pin is HIGH, the data will be recognized as normal data. Generally, normal data will follow the command, works as parameter or image data.

Reset (Reset)

Module will reset if RST pin is LOW. It is used to restart the module after powered on or awakened. After restarting, you need to initialize module with initialization function (Init) for working properly.

Initialization (Init)

Init has 3 effects:

1. Set the arguments at power up.
2. Awaken the module from deep sleep.
3. Set the mode to Full update or Partial update.

Process of initialization: reset --> driver output control --> booster soft start control --> write VCOM register --> set dummy line period --> set gate time --> data entry mode setting --> look-up table setting

Configuration of LUT table(SetLut)

Look-up table is used to set the update mode of the module. This table is provided by us but it may be different among different batches. If the table changed, we will update the demo code as soon as possible.

Set the frame memory (SetFrameMemory)

SetFrameMemory is used to write image data to the memory.

- Process:

  Set the area size (see the function SetMemoryArea) --> set the start point (see the function SetMemoryPointer) --> send the command *Write RAM* --> start image data transfer.

- The module has two memory areas. Once DisplayFrame is invoked, the following action of SetFrameMemory will set the other memory area, e.g. to set all the two memory areas, the process is: SetFrameMemory --> DisplayFrame --> SetFrameMemory --> DisplayFrame, i.e. set and update twice.
- The data from SPI interface is first saved into the memory and then updated if the module received the update command.
- About the image to be sent: 1 byte = 8 pixels, doesn't support Gray scale (Can only display black and white). A bit set stands for a white pixel, otherwise a bit reset stands for a black pixel.

For example：

```
0xC3: 8 pixels □□■■■■□□
0x00: 8 pixels ■■■■■■■■
0xFF: 8 pixels □□□□□□□□
0x66: 8 pixels ■□□■■□□■
```

Display a Frame (DisplayFrame)

DisplayFrame is used to display the data from the frame memory.

Note:

- The module has two memory areas. Once DisplayFrame is invoked, the following function SetFrameMemory will set the other memory area, e.g. to set all the two memory areas, the process is: SetFrameMemory --> DisplayFrame --> SetFrameMemory --> DisplayFrame, i.e. set and update twice.
- The data from SPI interface is first saved into the memory and then updated if the module received the update command.
- The module will flicker during full update.
- The module won't flicker during partial update, however, it may retain a "ghost image" of the last page.

Sleep mode (Sleep)

Sleep can make the module go into sleep mode to reduce the consumption.

If you want to wake up the module from sleep mode, you need to give a LOW pulse to RST pin. Then maybe you need to reconfigure the parameter of power (According to the batches, some of them need to reconfigure, some needn't). So if you want to wake up module, you had better use the Init function instead of Reset. Reset function and relative commands will be executed while executing the Init function.

Private function: Set the memory area (SetMemoryArea)

SetMemoryArea is used to specify the memory area, the arguments are the start/end points. Because 1 byte = 8 pixels of the image data to be sent, the x coordinates must be the multiple of 8, or else the last 3 bits will be ignored.

Private function: Set the memory pointer (SetMemoryPointer)

SetMemoryPointer is used to set the start point of the following image to be sent. Because 1 byte = 8 pixels of the image data to be sent, the x coordinates must be the multiple of 8, or else the last 3 bits will be ignored.

How to display an image

There are two ways to display pictures. One is display directly and other is indirectly.

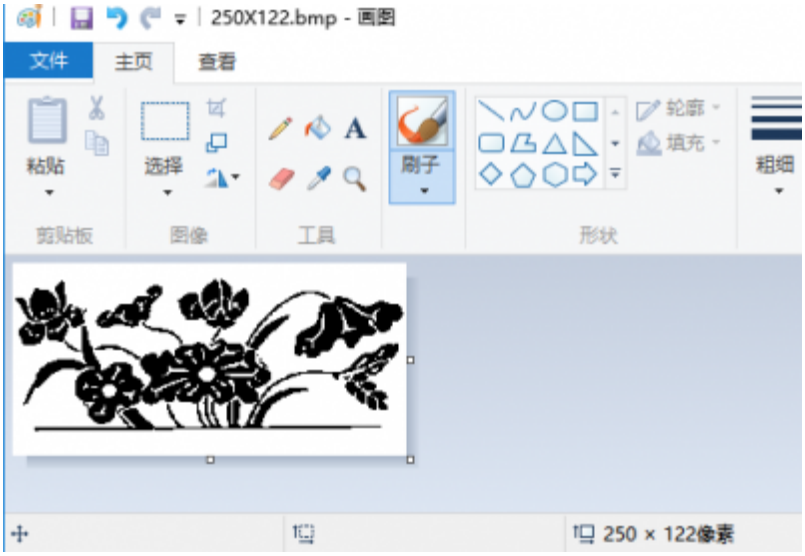
Display directly: Read the data of pictures with library functions, and decode. Then convert it to arrays and send to module. About how to implement it, you can refer to the python examples of Raspberry Pi. (The C demo doesn't display pictures directly)

Display indirectly: Converting pictures to relative arrays on PC and save as c file. Then you can use the c file on your project. This chapter we will talk about how to convert a picture to array.

1. Open a picture with drawing tool comes with Windows system, create a new image, and set the pixel to 128x296.
2. Because this module can only display two gray level (Only black and white), we need to convert picture to monochrome bitmap before converting it to array. That is, File --> BMP picture --> Monochrome Bitmap.

   > There is a monochrome bitmap on examples pack for demonstration (raspberrypi/python/monocolor.bmp).

3. Use Image2Lcd.exe software to generate corresponding array for picture (.c file). Open picture with this software, set the parameters:
   - Output data type: C language array
   - Scanning mode: vertical scanning
   - Output gray: single color (gray level of two)
   - Maximum width and height: 128 and 296
   - Include the data of Image Header: Don't check
   - Inverse color: Check (Check: the white on image will be converted to 1, and black is converted to 0)
4. Click **Save**, to generate .c file. Copy the corresponding array into your project, and you can display picture by calling this array.



Resources

Documentation

- Schematic

Demo code

- Demo code

Code shared from users

Thanks to Martin, who share his code which was modified for particle photon wifi dev kit

- 2.9inch_e-Paper_Module_code_particle

Datasheets

- 1.54inch_e-Paper_Datasheet.pdf
- 2.13inch_e-Paper_Datasheet.pdf
- 2.9inch_e-Paper_Datasheet.pdf

Relate applications

- WAVESHARE EPAPER AND A RASPBERRYPI (https://www.instructables.com/id/Waveshare-EPaper-and-a-RaspberryPi/)

FAQ

Question:

1. Why the e-Paper cant work with Arduino?

Answer: [Collapse]

The I/O level of Arduino is 5V, and the e-Paper should be driven with 3V3. If your Arduino cant drive the e-Paper successfully, please try to convert the level to 3.3V

You can also try to connect the Vcc pin to the 5V of Arduino to see whether the e-Paper works, but we recommend you not to use 5V for a long time.

Question:

2. Why does the color of e-Paper look a little black or grey?

Answer: [Collapse]

You can try to change the value of Vcom on demo codes.

Question:

3. Three-color e-paper looks more red/yellow than the picture on website?

Answer: [Collapse]

Because of different batch, some of them have aberration. Store the e-Paper right side up will reduce it. And if the e-Paper didn't be refreshed for long time, it will become more and more red/yellow. Please use the demo code to refresh the e-paper for several times in this case.

Support

Contact your seller (fast response and most recommended)

or send emails to **service@waveshare.com** (not fast enough but please be patient) for help.

Our working time: 09:00-18:00 (**UTC+8** Monday to Saturday)