Template:Raspberry Pi Guides for SPI e-Paper

From Waveshare Wiki

Hardware connection

If the e-Paper you have is the HAT version which has 40pin GPIO, you can directly attach the e-Paper HAT on Raspberry Pi, otherwise, you can connect your e-Paper to Raspberry Pi by 8pins cable provided.
To connect the e-Paper, you can following the table below

Connect to Raspberry Pi

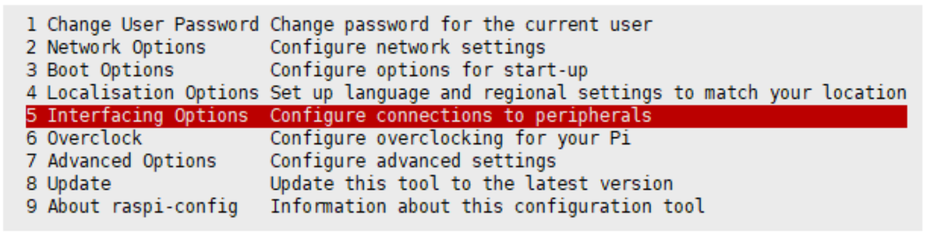
e-Paper	Raspberry Pi	
	BCM2835	Board
VCC	3.3V	3.3V
GND	GND	GND
DIN	MOSI	19
CLK	SCLK	23
CS	CE0	24
DC	25	22
RST	17	11
BUSY	24	18

Enable SPI interface

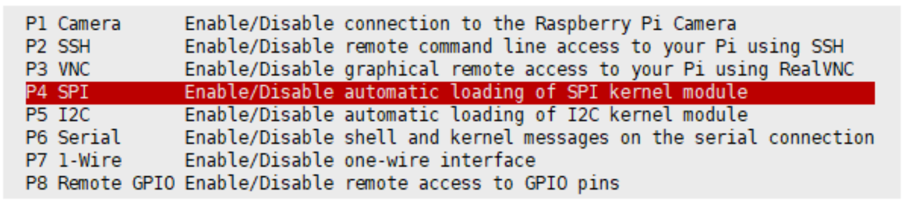
The communication interface of e-Paper is SPI, to use it, we should firstly enable SPI interface of SPI
Open terminal of Raspberry Pi, and open configuration by the following command

```
sudo raspi-config
```

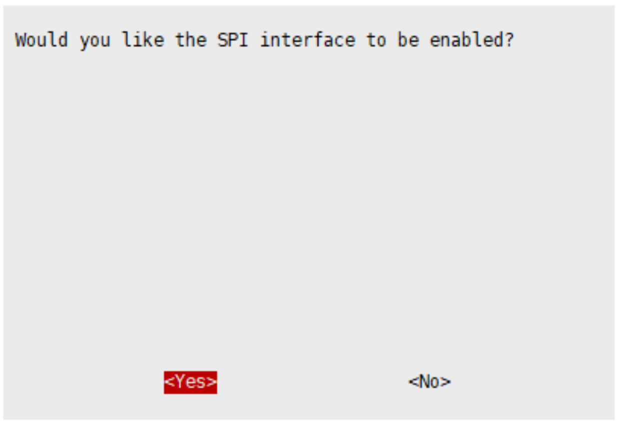
Choose Interfacing Options -> SPI -> Yes



```
1 Change User Password Change password for the current user
2 Network Options      Configure network settings
3 Boot Options         Configure options for start-up
4 Localisation Options Set up language and regional settings to match your location
5 Interfacing Options  Configure connections to peripherals
6 Overclock            Configure overclocking for your Pi
7 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool
```



```
P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial connection
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins
```



```
Would you like the SPI interface to be enabled?
```

Restart Raspberry Pi

```
sudo reboot
```

Install libraries

Open terminal of Raspberry Pi and run the following commands to install corresponding libraries:

- Install BCM2835 libraries

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.60.tar.gz
tar zxvf bcm2835-1.60.tar.gz
cd bcm2835-1.60/
sudo ./configure
sudo make
sudo make check
sudo make install
```

- Install WiringPi libraries

```
sudo apt-get install wiringpi
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
```

- Install Python2 libraries

```
sudo apt-get update
sudo apt-get install python-pip
sudo apt-get install python-pil
sudo apt-get install python-numpy
sudo pip install RPi.GPIO
sudo pip install spidev
```

- Install Python3 libraries

```
sudo apt-get update
sudo apt-get install python3-pip
sudo apt-get install python3-pil
sudo apt-get install python3-numpy
sudo pip3 install RPi.GPIO
sudo pip3 install spidev
```

Download demo codes

Open the terminal of Raspberry Pi and clone demo codes by the following commands:

```
sudo git clone https://github.com/waveshare/e-Paper
```

Examples

C

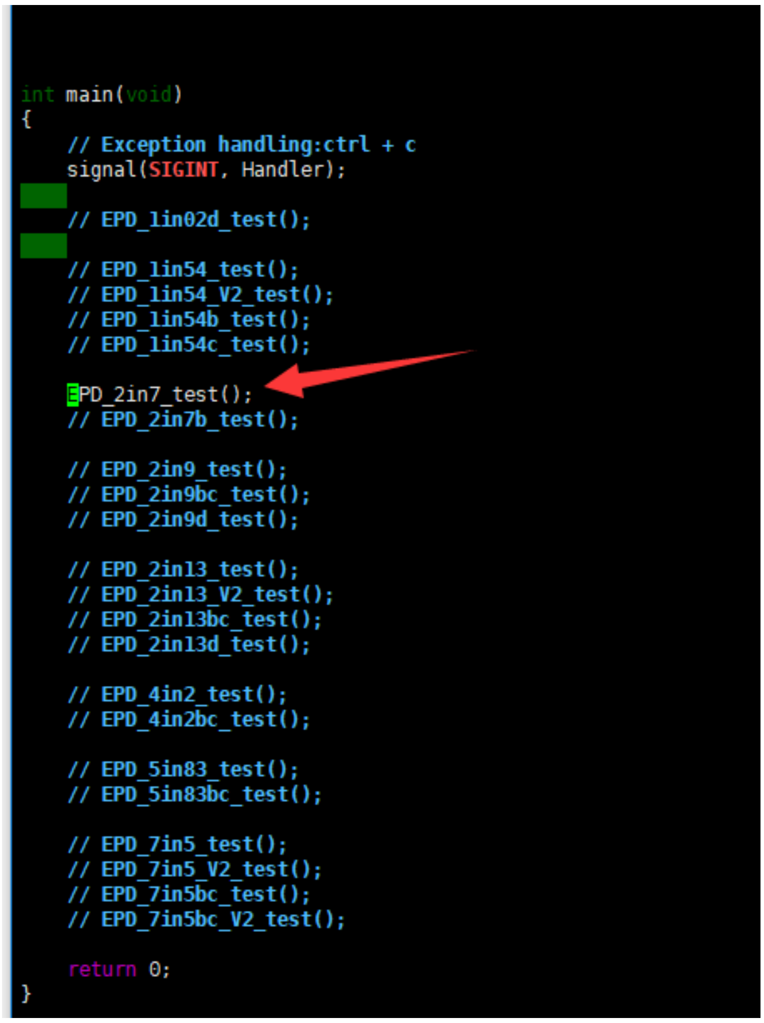
- Enter the folder of C examples

```
cd ~/e-Paper/RaspberryPi\&JetsonNano/
cd c
```

- Modify the main.c file for corresponding e-Paper

```
sudo nano examples/main.c
```

For example, if you want to update a 2.7inch e-Paper/2.7inch e-Paper HAT, you should modify the main.c file, uncomment the line EPD_2in7_test(), comment others, and save.


```
int main(void)
{
    // Exception handling:ctrl + c
    signal(SIGINT, Handler);

    // EPD_1in02d_test();

    // EPD_1in54_test();
    // EPD_1in54_V2_test();
    // EPD_1in54b_test();
    // EPD_1in54c_test();

    EPD_2in7_test();
    // EPD_2in7b_test();

    // EPD_2in9_test();
    // EPD_2in9bc_test();
    // EPD_2in9d_test();

    // EPD_2in13_test();
    // EPD_2in13_V2_test();
    // EPD_2in13bc_test();
    // EPD_2in13d_test();

    // EPD_4in2_test();
    // EPD_4in2bc_test();

    // EPD_5in83_test();
    // EPD_5in83bc_test();

    // EPD_7in5_test();
    // EPD_7in5_V2_test();
    // EPD_7in5bc_test();
    // EPD_7in5bc_V2_test();

    return 0;
}
```

- Compile codes

```
sudo make clean
sudo make
```

- Try to run the example

```
sudo ./epd
```

Supports type

1.02inch（128×80）：

EPD_1in02d_test()：Example for 1.02inch e-Paper/1.02inch e-Paper Module

1.54inch（1.54inch e-paper c：152×152，others：200×200）：

EPD_1in54_test()： Example for 1.54inch e-paper V1（Balck/White）：This version is stop production which can be bought before 2019-11-22;
EPD_1in54_V2_test()：Example for 1.54inch e-paper V2（Balck/White): This is the current version which can be buy now (2020-07-29). The e-Paper has V2 sticker on the backside.
EPD_1in54b_test()： Example for 1.54inch e-paper B（Black/White/Red）；
EPD_1in54c_test()： Example for 1.54inch e-paper C（Black/White/Red）;br />

2.7inch（264×176）：

EPD_2in7_test()： Example for 2.7inch e-paper（Black/White）；
EPD_2in7b_test()： Example for 2.7inch e-paper B（Black.White/Red）;<be />

2.9inch（296×128）：

EPD_2in9_test(): Example for 2.9inch e-paper（Black/White）；
EPD_2in9bc_test()： Example for 2.9inch e-paper B（Balck/White/Red） and 2.9inch e-paper C（Black/White/Yellow）；
EPD_2in9d_test()： Example for 2.9inch e-paper D（Black/White）；

2.13inch（2.13inch e-Paper：250×122，others：212×104）：

EPD_2in13_test()： Example for 2.13inch e-paper V1（Black/White）, this version is stop production and it can be bought before 019-05-15;
EPD_2in13_V2_test()：Example for 2.13inch e-paper V2（Black/White） This is the current version with sticker V2 on the backside (2020-07-29);
EPD_2in13bc_test()：Example for 2.13inch e-paper B（Black/White/Red）and 2.13inch e-paper C（Blackj/White/Yellow）;
EPD_2in13d_test()： Example for 2.13inch e-paper D（Black/White）;

4.01inch (640x400)

EPD_4in01_test(): Example for the 4.01inch e-Paper HAT (F);

4.2inch（400×300）

EPD_4in2_test()： Example for 4.2inch e-paper（Black/White）;
EPD_4in2bc_test()： Example for 4.2inch e-paper B（Black/White/Red）;

5.65inch (600x448)

EPD_5in65f_test(): Example for for 5.65inch e-Paper F (Seven-color);

5.83inch（600×448）：

EPD_5in83_test()： Example for 5.83inch e-paper（Black/White）;
EPD_5in83bc_test()： Example for 5.83inch e-paper B（Black/White/Red）and 5.83inch e-paper C（Black/White/Yellow）;

7.5inch（V1：640×384，V2：800×480）：

EPD_7in5_test()： Example for 7.5inch e-paper（Black/White）, this version is stop production and it can be bought before 2019-12-07;
EPD_7in5bc_test()： Example for 7.5inch e-paper B（Black/White/Red）and 7.5inch e-paper C（Black/White/Yellow）， 7.5inch e-paper B V1 version is stop production and it can be bought before 2019-12-07;
EPD_7in5_V2_test()： Example for 7.5inch e-paper V2（Black/White）, This is the current version with V2 sticker on the backside (2020-07-29)
EPD_7in5bc_V2_test()：Example for 7.5inch e-paper B V2（Black/White/Red）; This is the current version with V2 sticker on the backside. (2020-07-29);

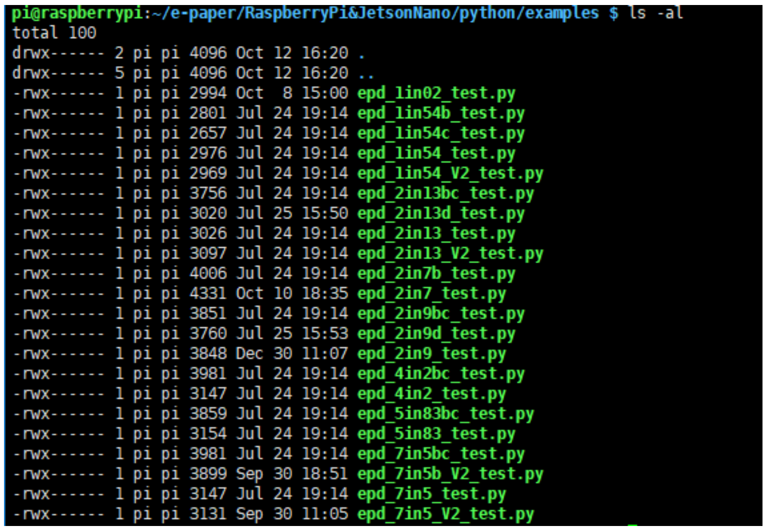
Python

- Enter the folder of python code

```
cd ~/e-Paper/RaspberryPi\&JetsonNano/
cd python/examples
```

- Check the folder, you can see that there are .py files for different e-Paper.

```
ls -al
```



```
pi@raspberrypi:~/e-paper/RaspberryPi&JetsonNano/python/examples $ ls -al
total 100
drwx------ 2 pi pi 4096 Oct 12 16:20 .
drwx------ 5 pi pi 4096 Oct 12 16:20 ..
-rwx------ 1 pi pi 2994 Oct  8 15:00 epd_1in02_test.py
-rwx------ 1 pi pi 2801 Jul 24 19:14 epd_1in54b_test.py
-rwx------ 1 pi pi 2657 Jul 24 19:14 epd_1in54c_test.py
-rwx------ 1 pi pi 2976 Jul 24 19:14 epd_1in54_test.py
-rwx------ 1 pi pi 2969 Jul 24 19:14 epd_1in54_V2_test.py
-rwx------ 1 pi pi 3756 Jul 24 19:14 epd_2in13bc_test.py
-rwx------ 1 pi pi 3020 Jul 25 15:50 epd_2in13d_test.py
-rwx------ 1 pi pi 3026 Jul 24 19:14 epd_2in13_test.py
-rwx------ 1 pi pi 3097 Jul 24 19:14 epd_2in13_V2_test.py
-rwx------ 1 pi pi 4006 Jul 24 19:14 epd_2in7b_test.py
-rwx------ 1 pi pi 4331 Oct 10 18:35 epd_2in7_test.py
-rwx------ 1 pi pi 3851 Jul 24 19:14 epd_2in9bc_test.py
-rwx------ 1 pi pi 3760 Jul 25 15:53 epd_2in9d_test.py
-rwx------ 1 pi pi 3848 Dec 30 11:07 epd_2in9_test.py
-rwx------ 1 pi pi 3981 Jul 24 19:14 epd_4in2bc_test.py
-rwx------ 1 pi pi 3147 Jul 24 19:14 epd_4in2_test.py
-rwx------ 1 pi pi 3859 Jul 24 19:14 epd_5in83bc_test.py
-rwx------ 1 pi pi 3154 Jul 24 19:14 epd_5in83_test.py
-rwx------ 1 pi pi 3981 Jul 24 19:14 epd_7in5bc_test.py
-rwx------ 1 pi pi 3899 Sep 30 18:51 epd_7in5b_V2_test.py
-rwx------ 1 pi pi 3147 Jul 24 19:14 epd_7in5_test.py
-rwx------ 1 pi pi 3131 Sep 30 11:05 epd_7in5_V2_test.py
```

- Run the responding example

```
sudo python epd_xxx_test.py
sudo python3 epd_xxx_test.py
```

Supports type

1.02inch（128×80）：

epd_1in02_test.py：Example for 1.02inch e-Paper/1.02inch e-Paper Module

1.54inch（1.54inch e-paper c：152×152，others：200×200）：

epd_1in54_test.py： Example for 1.54inch e-paper V1（Balck/White）：This version is stop production which can be bought before 2019-11-22；
epd_1in54_V2_test.py：Example for 1.54inch e-paper V2（Balck/White): This is the current version which can be buy now (2020-07-29). The e-Paper has V2 sticker on the backside.
epd_1in54b_test.py： Example for 1.54inch e-paper B（Black/White/Red）；
epd_1in54c_test.py： Example for 1.54inch e-paper C（Black/White/Red）;br />

2.7inch（264×176）：

epd_2in7_test.py： Example for 2.7inch e-paper（Black/White）；
epd_2in7b_test.py： Example for 2.7inch e-paper B（Black.White/Red）;<be />

2.9inch（296×128）：

epd_2in9_test.py: Example for 2.9inch e-paper（Black/White）；
epd_2in9bc_test.py： Example for 2.9inch e-paper B（Balck/White/Red） and 2.9inch e-paper C（Black/White/Yellow）；
epd_2in9d_test.py： Example for 2.9inch e-paper D（Black/White）；

2.13inch（2.13inch e-Paper：250×122，others：212×104）：

epd_2in13_test.py： Example for 2.13inch e-paper V1（Black/White）, this version is stop production and it can be bought before 019-05-15;
epd_2in13_V2_test.py：Example for 2.13inch e-paper V2（Black/White） This is the current version with sticker V2 on the backside (2020-07-29)；
epd_2in13bc_test.py：Example for 2.13inch e-paper B（Black/White/Red）and 2.13inch e-paper C（Blackj/White/Yellow）；
epd_2in13d_test.py： Example for 2.13inch e-paper D（Black/White）；

4.01inch (640x400)

epd_4in01f_test.py: Example for 4.01inch e-Paper (Seven-color);

4.2inch（400×300）

epd_4in2_test.py： Example for 4.2inch e-paper（Black/White）；
epd_4in2bc_test.py： Example for 4.2inch e-paper B（Black/White/Red）；

5.65inch (600x448)

epd_5in65f_test.py: Example for 5.65inch e-Paper F (Seven-color);

5.83inch（600×448）：

epd_5in83_test.py： Example for 5.83inch e-paper（Black/White）；
epd_5in83bc_test.py： Example for 5.83inch e-paper B（Black/White/Red）and 5.83inch e-paper C（Black/White/Yellow）；

7.5inch（V1：640×384，V2：800×480）：

epd_7in5_test.py： Example for 7.5inch e-paper（Black/White）, this version is stop production and it can be bought before 2019-12-07;
epd_7in5bc_test.py： Example for 7.5inch e-paper B（Black/White/Red）and 7.5inch e-paper C（Black/White/Yellow），7.5inch e-paper B V1 version is stop production and it can be bought before 2019-12-07;
epd_7in5_V2_test.py： Example for 7.5inch e-paper V2（Black/White）, This is the current version with V2 sticker on the backside (2020-07-29)
epd_7in5bc_V2_test.py：Example for 7.5inch e-paper B V2（Black/White/Red）; This is the current version with V2 sticker on the backside. (2020-07-29);

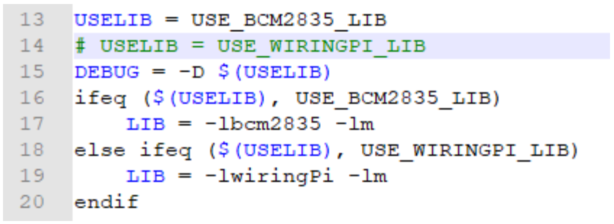
Description of codes (API)

The libraries for Raspberry Pi and Jetson Nano are same. Examples contain three parts, hardware interface, EPD driver and the GUI functions.

C

Hardware interface

Two libraries are used by C example,WiringPi and BCM2835. The codes use wiringPi by default, if you want to use BCM2835, you can modify RaspberryPi&JetsonNano\c\Makefile file, modify lines 13 and 14. Change it as below:



```
USELIB = USE_BCM2835_LIB
# USELIB = USE_WIRINGPI_LIB
DEBUG = -D $(USELIB)
ifeq ($(USELIB), USE_BCM2835_LIB)
    LIB = -lbcm2835 -lm
else ifeq ($(USELIB), USE_WIRINGPI_LIB)
    LIB = -lwiringPi -lm
endif
```

- Data type

```
#define UBYTE   uint8_t
#define UWORD   uint16_t
#define UDOUBLE uint32_t
```

- Init and Exit

```
void DEV_Module_Init(void);
void DEV_Module_Exit(void);
```

Note: The Init() and Exit() function are used to configure GPIOs . EPD enter sleep mode after Exit() function is used, and the consumption of e-Paper should be 0 in sleep mode if the PCB is Rev2.1 version.

- GPIO Read/Write

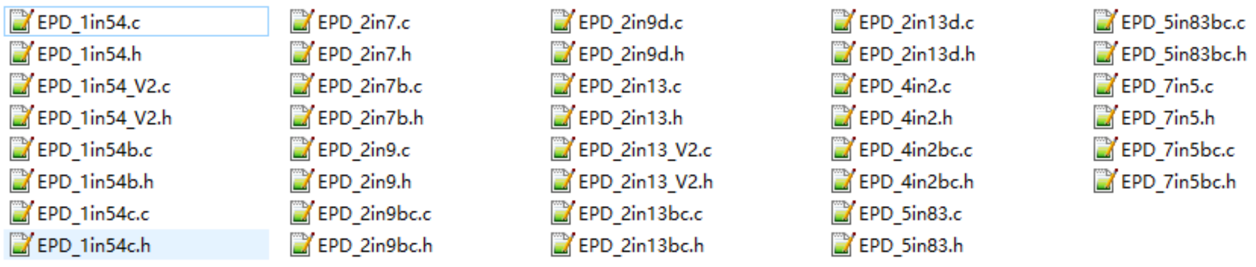
```
void DEV_Digital_Write(UWORD Pin, UBYTE Value);
UBYTE DEV_Digital_Read(UWORD Pin);
```

- SPI transmit data

```
void DEV_SPI_WriteByte(UBYTE Value);
```

EPD driver

The driver file are saved under RaspberryPi&JetsonNano\c\lib\e-Paper



EPD_1in54.c	EPD_2in7.c	EPD_2in9d.c	EPD_2in13d.c	EPD_5in83bc.c
EPD_1in54.h	EPD_2in7.h	EPD_2in9d.h	EPD_2in13d.h	EPD_5in83bc.h
EPD_1in54_V2.c	EPD_2in7b.c	EPD_2in13.c	EPD_4in2.c	EPD_7in5.c
EPD_1in54_V2.h	EPD_2in7b.h	EPD_2in13.h	EPD_4in2.h	EPD_7in5.h
EPD_1in54b.c	EPD_2in9.c	EPD_2in13_V2.c	EPD_4in2bc.c	EPD_7in5bc.c
EPD_1in54b.h	EPD_2in9.h	EPD_2in13_V2.h	EPD_4in2bc.h	EPD_7in5bc.h
EPD_1in54c.c	EPD_2in9bc.c	EPD_2in13bc.c	EPD_5in83.c	
EPD_1in54c.h	EPD_2in9bc.h	EPD_2in13bc.h	EPD_5in83.h	

- Initial EPD

For 1.54inch e-Paper, 1.54inch e-Paper V2, 2.13inch e-Paper, 2.13inch e-Paper V2, 2.13inch e-Paper (D), 2.9inch e-Paper and 2.9inch e-Paper (D), partial refresh is supported. You can set Mode = 0 for full refresh and Mode =1 for partial refresh

```
void EPD_xxx_Init(UBYTE Mode);
```

For other e-Paper

```
void EPD_xxx_Init(void);
```

- Trasmite one frame of e-Paper and display

For black/white e-Paper

```
void EPD_xxx_Display(UBYTE *Image);
```

For three-color e-Paper

```
void EPD_xxx_Display(const UBYTE *blackimage, const UBYTE *ryimage);
```

There are exception:
For 2.13inch e-Paper (D) and 2.13inch e-Paper (D), if you want to do partial refresh, you should use function

```
void EPD_2IN13D_DisplayPart(UBYTE *Image);
void EPD_2IN9D_DisplayPart(UBYTE *Image);
```

For 1.54inch e-Paper V2 and 2.13inch e-Paper V2. you should first display static background (base image) and then dynamicly display (display Part) when partial refresh.

```
void EPD_1IN54_V2_DisplayPartBaseImage(UBYTE *Image);
void EPD_1IN54_V2_DisplayPart(UBYTE *Image);
void EPD_2IN13_V2_DisplayPart(UBYTE *Image);
void EPD_2IN13_V2_DisplayPartBaseImage(UBYTE *Image);
```

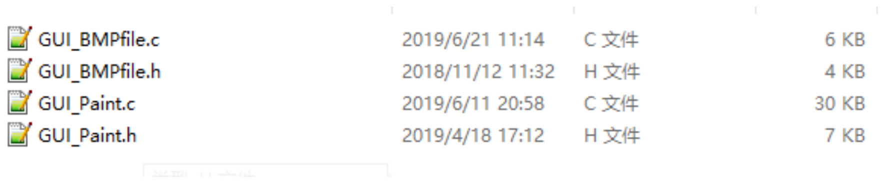
- Sleep mode

```
void EPD_xxx_Sleep(void);
```

To wake up module, you should set hardware reset (re power on it) or call the init function.

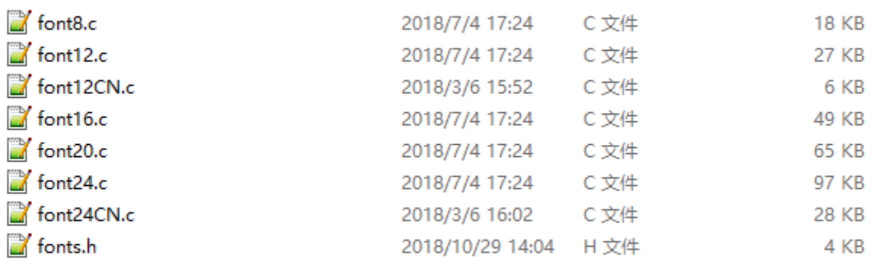
GUI functions

GUI files can be found in RaspberryPi&JetsonNano\c\lib\GUI\GUI_Paint.c(.h) directory



GUI_BMPfile.c	2019/6/21 11:14	C 文件	6 KB
GUI_BMPfile.h	2018/11/12 11:32	H 文件	4 KB
GUI_Paint.c	2019/6/11 20:58	C 文件	30 KB
GUI_Paint.h	2019/4/18 17:12	H 文件	7 KB

The fonts can be found in RaspberryPi&JetsonNano\c\lib\Fonts directory



font8.c	2018/7/4 17:24	C 文件	18 KB
font12.c	2018/7/4 17:24	C 文件	27 KB
font12CN.c	2018/3/6 15:52	C 文件	6 KB
font16.c	2018/7/4 17:24	C 文件	49 KB
font20.c	2018/7/4 17:24	C 文件	65 KB
font24.c	2018/7/4 17:24	C 文件	97 KB
font24CN.c	2018/3/6 16:02	C 文件	28 KB
fonts.h	2018/10/29 14:04	H 文件	4 KB

Create an image buffer

```
void Paint_NewImage(UBYTE *image, UWORD Width, UWORD Height, UWORD Rotate, UWORD Color)
```

- Image: the Image buffer
- Width: width of the image
- Height: Height of the image
- Rotate: Rotate angle
- Color: Color of the image

Select image buffer

```
void Paint_SelectImage(UBYTE *image)
```

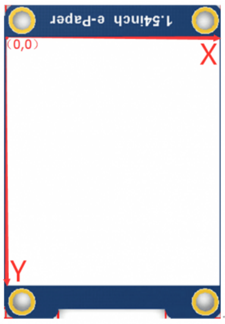
- The image buffer, it is a pointer of image buffer's first address

Rotate image

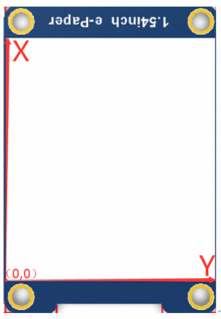
This function should be used after Paint_SelectImage()

```
void Paint_SetRotate(UWORD Rotate)
```

- Rotate: The angle rotated. It should be ROTATE_0, ROTATE_90, ROTATE_180, ROTATE_270
- Note: For different orientation, the position of the first pixel is different, here we take 1.54inch as example



SPI-epaper-C-90.png



Mirroring

```
void Paint_SetMirroring(UBYTE mirror)
```

- mirror: The type of mirroring. (MIRROR_NONE, MIRROR_HORIZONTAL、MIRROR_VERTICAL、MIRROR_ORIGIN)

Set Pixel

This function is used to set the position and types of the pixel

```
void Paint_SetPixel(UWORD Xpoint, UWORD Ypoint, UWORD Color)
```

- Xpoint: The x-axis coordination of pixel
- Ypoint: The y-axis coordination of pixel
- Color: The color of the pixel

Clear

This function is used to clear the e-Paper

```
void Paint_Clear(UWORD Color)
```

- Color: The color of the display

Clear window

This function is used to clear a partial area

```
void Paint_ClearWindows(UWORD Xstart, UWORD Ystart, UWORD Xend, UWORD Yend, UWORD Color)
```

- Xstart: The x-axis coordination of the start point
- Ystart: The y-axis coordination of the start point
- Xend: The x-axis coordination of the end point
- Yend: The y-axis coordination of the end point
- Color: The color of the windows

Draw point

This function is used to draw point.

```
void Paint_DrawPoint(UWORD Xpoint, UWORD Ypoint, UWORD Color, DOT_PIXEL Dot_Pixel, DOT_STYLE Dot_Style)
```

- Xpoint: The x-axis coordination of point
- Ypoint: The y-axis coordination of point
- Dot_Pixel: The size of the point

```
                  typedef enum {
                          DOT_PIXEL_1X1  = 1,    // 1 x 1
                          DOT_PIXEL_2X2  ,               // 2 X 2
                          DOT_PIXEL_3X3  ,               // 3 X 3
                          DOT_PIXEL_4X4  ,               // 4 X 4
                          DOT_PIXEL_5X5  ,               // 5 X 5
                          DOT_PIXEL_6X6  ,               // 6 X 6
                          DOT_PIXEL_7X7  ,               // 7 X 7
                          DOT_PIXEL_8X8  ,               // 8 X 8
                 } DOT_PIXEL;
```

- Dot_Style: The style of the point

```
                 typedef enum {
                    DOT_FILL_AROUND  = 1,
                    DOT_FILL_RIGHTUP,
                 } DOT_STYLE;
```

Drawn Line

```
void Paint_DrawLine(UWORD Xstart, UWORD Ystart, UWORD Xend, UWORD Yend, UWORD Color, LINE_STYLE Line_Style , LINE_STYLE Line_Style)
```

This function is used to draw a line

- Xstart: The start x-axis coordination of the line
- Ystart: The start y-axis coordination of the line
- Xend: The end x-axis coordination of the line
- Yend: The end y-axis coordination of the line
- Line_width: The width of the line

```
                typedef enum {
                        DOT_PIXEL_1X1  = 1,    // 1 x 1
                        DOT_PIXEL_2X2  ,               // 2 X 2
                        DOT_PIXEL_3X3  ,               // 3 X 3
                        DOT_PIXEL_4X4  ,               // 4 X 4
                        DOT_PIXEL_5X5  ,               // 5 X 5
                        DOT_PIXEL_6X6  ,               // 6 X 6
                        DOT_PIXEL_7X7  ,               // 7 X 7
                        DOT_PIXEL_8X8  ,               // 8 X 8
                } DOT_PIXEL;
```

- Line_style: The style of the line

```
                typedef enum {
                        LINE_STYLE_SOLID = 0,
                        LINE_STYLE_DOTTED,
                } LINE_STYLE;
```

Draw rectangle

Draw a rectangle from (Xstart, Ystart) to (Xend, Yend).

```
void Paint_DrawRectangle(UWORD Xstart, UWORD Ystart, UWORD Xend, UWORD Yend, UWORD Color, DOT_PIXEL Line_width, DRAW_FILL Draw_Fill)
```

- Xstart: Start coordinate of X-axes of the rectangle
- Ystart: Start coordinate of Y-axes of the rectangle
- Xend: End coordinate of X-end of the rectangle
- Yend: End coordinate of Y-end of the rectangle
- Color: color of the rectangle
- Line_width: The width of edges, 8 sides are available;

```
                typedef enum {
                        DOT_PIXEL_1X1  = 1,    // 1 x 1
                        DOT_PIXEL_2X2  ,               // 2 X 2
                        DOT_PIXEL_3X3  ,               // 3 X 3
                        DOT_PIXEL_4X4  ,               // 4 X 4
                        DOT_PIXEL_5X5  ,               // 5 X 5
                        DOT_PIXEL_6X6  ,               // 6 X 6
                        DOT_PIXEL_7X7  ,               // 7 X 7
                        DOT_PIXEL_8X8  ,               // 8 X 8
                } DOT_PIXEL;
```

- Draw_Fill: set the rectangle full or empty.

```
typedef enum {
        DRAW_FILL_EMPTY = 0,
        DRAW_FILL_FULL,
} DRAW_FILL;
```

Draw character (ASCII)

Set(Xstart Ystart) as letf-top point, draw a ASCII character.

```
void Paint_DrawChar(UWORD Xstart, UWORD Ystart, const char Ascii_Char, sFONT* Font, UWORD Color_Foreground, UWORD Color_Background)
```

Parameter:

- Xstart: X coordinate of the left-top pixel of character;
- Ystart: Y coordinate of the left-top pixel of character;
- Ascii_Char：Ascii character;
- Font: 5 fonts are available;

    font12：7*12

font16：11*16
font20：14*20
font24：17*24

- Color_Foreground: color of character;
- Color_Background: color of background;

Draw String

Set point (Xstart Ystart) as the left-top pixel, draw a string.

```
void Paint_DrawString_EN(UWORD Xstart, UWORD Ystart, const char * pString, sFONT* Font, UWORD Color_Foreground, UWORD Color_Background)
```

Parameters:

- Xstart: X coordinate of left-top pixel of characters;
- Ystart: Y coordinate of left-top pixel of characters;
- pString：Pointer of string
- Font: 5 fonts are available:

font8：5*8
font12：7*12
font16：11*16
font20：14*20
font24：17*24

- Color_Foreground: color of string
- Color_Background: color of the background

Draw Chinese characters

this function is used to draw Chinese fonts based ON GB2312 fonts.

```
void Paint_DrawString_CN(UWORD Xstart, UWORD Ystart, const char * pString, cFONT* font, UWORD Color_Foreground, UWORD Color_Background)
```

Parameters:

- Xstart: Coordinate of left-top pixel of characters;
- Ystart: Coordinate of left-top pixel of characters;
- pString：Pointer of string;
- Font: GB2312 fonts：

font12CN：11*21(ascii)，16*21 (Chinese)
font24CN：24*41(ascii)，32*41 (Chinese)

- Color_Foreground: color of string
- Color_Background: color of the background

Draw number

Draw a string of numbers, (Xstart, Ystart) is the left-top pixel.

```
void Paint_DrawNum(UWORD Xpoint, UWORD Ypoint, int32_t Nummber, sFONT* Font, UWORD Color_Foreground, UWORD Color_Background)
```

Parameter:

- Xstart: X coordinate of left-top pixel;
- Ystart: Y coordicate of left-to pixel;
- Nummber: the numbers displayed. the numbers are saved in int format, the maximum is 2147483647;
- Font: 5 fonts are available：

font8：5*8
font12：7*12
font16：11*16
font20：14*20
font24：17*24

- Color_Foreground: color of font;
- Color_Background: color of background;

Draw image

Send image data of BMP file to buffer

```
void Paint_DrawBitMap(const unsigned char* image_buffer)
```

Parameters:

- image_buffer: adrress of image data in buffer

Read local bmp picture and write it to buffer

Linux platform like Jetson Nano and Raspberry Pi support to directly operate bmp pictures Raspberry Pi & Jetson Nano：RaspberryPi&JetsonNano\c\lib\GUI\GUI_BMPfile.c(.h)

```
UBYTE GUI_ReadBmp(const char *path, UWORD Xstart, UWORD Ystart)
```

Parameters:

- path：The path of BMP pictures
- Xstart: X coordination of left-top of picture, default 0;
- Ystart: Y coordination of left-top of picture, default 0;

Testing Code

In the above part, we describe the tree structures of Linux codes, here we talk about the testing code for user. Raspberry Pi & Jetson Nano: RaspberryPi&JetsonNano\c\examples. The codes in examples are testing code, you can modify the definition in main.c file for different types of e-Paper. For example, if you want to test 2.13inch e-paper, you need to delete the "//" symbol on line 32. Use 5.65inch e-Paper as example, you need to change the line:

```
//EPD_5in65f_test();
```

to

```
EPD_5in65f_test();
```

Then compile it again and run

```
make clean
make
sudo ./epd
```

Python (Can be used for Jetson nano and Raspberry Pi)

It is compatible with python2.7 and python3
python is easy to use than c codes
Raspberry Pi and Jetson Nano：RaspberryPi&JetsonNano\python\lib\

epdconfigu.py

- Initialize module and exit handle：

```
def module_init()
def module_exit()
```

Note:
1. The functions are used to set GPIP before and after driving e-Paper
2. If the board you have is printed with Rev2.1, module enter low-ultra mode after Module_Exit(). (as we test, the current is about 0 in this mode);

- GPIO Read/Write：

```
def  digital_write(pin, value)
def  digital_read(pin)
```

- SPI Write data:

```
def spi_writebyte(data)
```

epdxxx.py(xxx is the type of e-Paper)

- Initailize e-paper: this function should be used at the begining. It can also be used to wake up e-Paper from Sleep mode.

```
#For 2.13inch e-Paper、2.9inch e-Paper
def init(self, update) #  Choose lut_full_update or lut_partial_update
#Other type
def init(self)
```

- Clear e-paper: This function is used to clear e-Paper to white;

```
def Clear(self)
def Clear(self, color) # Some types of e-Paper should use this function to clear screen
```

- Convert image to arrays

```
def getbuffer(self, image)
```

- Transmit one frame of image data and display

```
#For two-color e-paper
def display(self, image)

#Because that controllers of 2.13inch e-paper are updated, when partial refresh, they should first use displayPartBaseImage() to display static
background, then use displayPart() to dynamaticlly display.
def displayPartBaseImage(self, image)
def displayPart(self, image)
```

- Enter sleep mode

```
def sleep(self)
```

epd_xxx_test.py(xxx is type of e-paper)

python examples are saved in directory：
Raspberry Pi和Jetson Nano：RaspberryPi&JetsonNano\python\examples\
If the python installed in your OS is python2, you should run examples like below：

```
sudo python epd_2in13_V2_test.py
```

If it is python3, the commands should be:

```
sudo python3 epd_2in13-V2_test.py
```

Orientation

To rotate the display, you can use transpose function like blackimage = blackimage.transpose(Image.ROTATE_270):

```
blackimage = blackimage.transpose(Image.ROTATE_270)
redimage = redimage.transpose(Image.ROTATE_270)
#Support ROTATE_90, ROTATE_180, ROTATE_270
```

GUI

Python has a powerful PIL library (http://effbot.org/imagingbook), which can be used directly to drawing figures. Here we use it for drawing

- Install the library firstly

```
sudo apt-get install python3-pil
```

Import the library

```
from PIL import Image,ImageDraw,ImageFont
```

Image: library; ImageDraw: drawing function; ImageFont: fonts

- Set image buffer for drawing.

```
image = Image.new('1', (epd.width, epd.height), 255)  # 255: clear the frame
```

The first parameter is the depth of color, 1 means 2 grayscale. The second parameter is a tuple of image size. The third parameter is color of the image, 0 is black and 255 is white.

- Create an image object.

```
draw = ImageDraw.Draw(image)
```

- Draw rectangle

```
draw.rectangle((0, 10, 200, 34), fill = 0)
```

The first parameter is a tuple of coordination. 0, 10 is the top-left point of rectangle, 200, 34) is the right-bottom point. fille = 0 set the filled color to black.

- Draw line

```
draw.line((16, 60, 56, 60), fill = 0)
```

The first parameter is a type of coordination, 16, 60 is the begin point, 200, 34 is the end point. fill=0 set the line to black

- Draw circle

```
draw.arc((90, 60, 150, 120), 0, 360, fill = 0)
```

This function is used to draw a encircle of a square. The first parameter is a tuple of coordination of the square. the degree of the circle is 0 to 360 °, fille=0 set the circle to black.
If the figure is not square according to the coordination, you will get an ellipse.。

Besides the arc function, you can also use the chord function for drawing solid circle.

```
draw.chord((90, 130, 150, 190), 0, 360, fill = 0)
```

The first parameter is the coordination of the enclosing rectangle. The second and third parameters are the beginning and end degrees of the circle. The fourth parameter is the fill color of the circle.

- Character

You can directkt import ImageFont model for drawing characters:

```
font = ImageFont.truetype(os.path.join(picdir, 'Font.ttc'), 24)
```

You can use the fonts of Windows or other fonts which is in ttc format.

To draw English character, you can directly use the fonts; for Chinese character, you need to add a symbol u:

```
draw.text((8, 12), 'hello world', font = font, fill = 255)
draw.text((8, 36), u'电子墨水屏', font = font, fill = 0)
```

The first parameter is a tuple of coordination of character, the second parameter is the font and las one is set the color.

- Read local picture

```
image = Image.open(os.path.join(picdir, '2in13-v2.bmp'))
```

The parameter is the path of picture.

- Other functions.

For more information about the PIL library, please refer to http://effbot.org/imagingbook.