2.7inch e-Paper HAT
用户手册

产品概述

- 本品是 2.7 英寸电子墨水屏模块，分辨率为 264x176，带有内部控制器，使用 SPI 接口通信。
- 具有耗低、视角宽、阳光直射下仍可清晰显示等优点，常用于货架标签、工业仪表等显示应用。

特点

- 无需背光，断电可长时间保持最后一屏的显示内容
- 功耗非常低，基本只在刷新时耗电
- 自带 4 个按键，方便扩展应用
- 适用于 Raspberry Pi 2 代 B/3 代 B/Zero/Zero W
- SPI 控制接口，可接入 Raspberry/Arduino/Nucleo 等主控板
- 提供完善的配套资料手册(Raspberry/Arduino/STM32 等示例程序)

版本：V1.2.1，日期：2017 年 9 月 15 日

产品参数

工作电压：	3.3V
通信接口：	3-wire SPI、4-wire SPI
外形尺寸：	85mm × 56mm
显示尺寸：	57.288mm × 38.192mm
点距：	0.217 × 0.217
分辨率：	264 × 176
显示颜色：	黑、白
灰度等级：	2
全局刷新：	6s
刷新功耗：	26.4mW(typ.)
待机功耗：	<0.017mW
可视角度：	>170°

接口说明

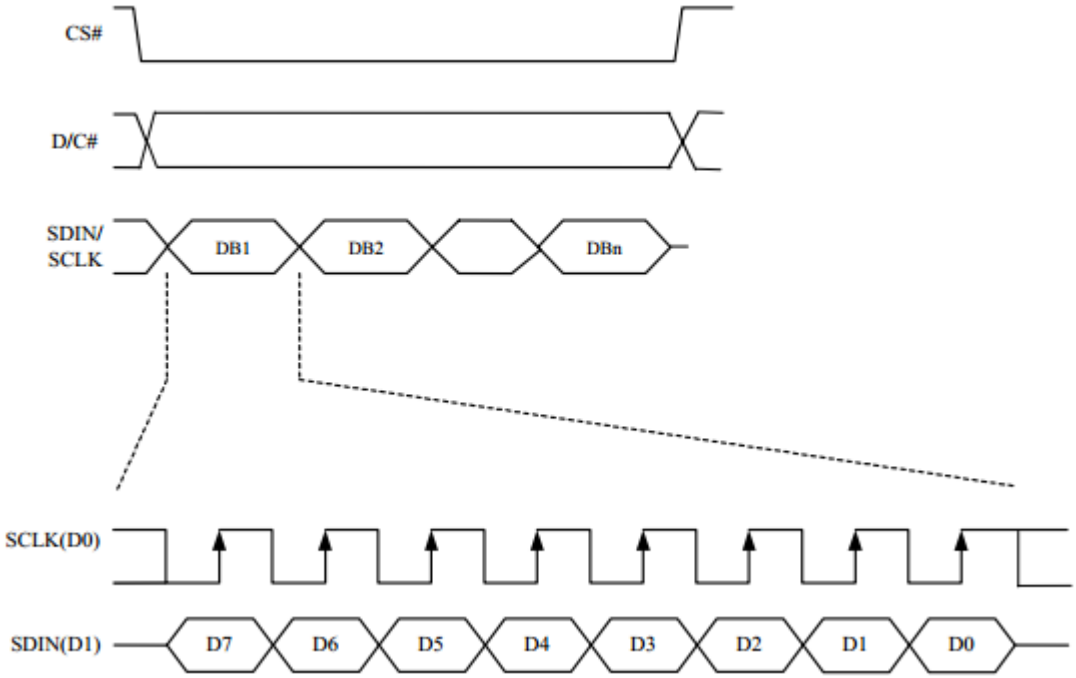
VCC：	3.3V
GND：	GND
DIN：	SPI 通信 MOSI 引脚
CLK：	SPI 通信 SCK 引脚
CS：	SPI 片选引脚（低电平有效）
DC：	数据/命令控制引脚（高电平表示数据，低电平表示命令）
RST：	外部复位引脚（低电平复位）
BUSY：	忙状态输出引脚（低电平表示忙）

工作原理

器件介绍

本产品使用的电子纸采用“微胶囊电泳显示”技术进行图像显示，其基本原理是悬浮在液体中的带电纳米粒子受到电场作用而产生迁移。电子纸显示屏是靠反射环境光来显示图案的，不需要背光，即使是在阳光底下，电子纸显示屏依然清晰可视，可视角度几乎达到了 180° 。因此，电子纸显示屏非常适合阅读。

通信协议



CS 用于从机片选。仅当 CS 为低电平时，模块才会工作。

DC 用于模块的数据/命令控制。当 DC 为低电平时，接收到的数据会被当做指令执行。

SCLK 用于 SPI 通信时钟。

SDIN 用于数据传输，主发从收。

对于 SPI 通信而言，数据是有传输时序的，即时钟相位（CPHA）与时钟极性(CPOL)的组合：

1) CPOL 的高低决定串行同步时钟的空闲状态电平，CPOL = 0，为低电平。CPOL 对传输影响不大；
2) CPHA 的高低决定串行同步时钟是在第一时钟跳变沿还是第二个时钟跳变沿数据被采集，当 CPHL = 0，在第一个跳变沿进行数据采集；

这两者组合成为四种 SPI 通信方式，本模块工作于 CPHL = 0，CPOL = 0。

当 SCLK 第一个下降沿时开始传输数据，一个时钟周期传输 8bit 数据，使用 SPI0，按位传输,高位在前,低位在后。

使用方法

用于树莓派

安装必要的函数库

墨水屏工作于树莓派上面的时候，需要安装必要的函数库（wiringPi、bcm2835、python 库），否则以下的示例程序可能无法正常工作。安装方法详见：

http://www.waveshare.net/wiki/Pioneer600_Datasheets

硬件连接

以下为树莓派 BCM 管脚编码硬件连接（树莓派三代 B）：

e-Paper	树莓派 3 代 B
3.3V	3.3V
GND	GND
DIN	MOSI
CLK	SCLK
CS	CE0
DC	25 (BCM)
RST	17 (BCM)
BUSY	24 (BCM)

预期结果

1) 安装好相应的库后，把对应程序复制至树莓派中，进入对应目录中：
   - **bcm2835：**执行命令：make，将会编译代码，生成一个名为 epd 的执行文件。执行命令：sudo ./epd，程序将会运行。
   - **wiringpi：**执行命令：make，将会编译代码，生成一个名为 epd 的执行文件。执行命令：sudo ./epd，程序将会运行。
   - **python：**执行命令：sudo python main.py
2) 屏幕会显示图像。

   注意：本模块不支持局部刷新，刷新需要约 6 秒钟，请耐心等待。

用于 Arduino

硬件连接

硬件连接到开发板 UNO PLUS：

e-Paper	Arduino
3.3V	3.3V
GND	GND
DIN	D11
CLK	D13
CS	D10
DC	D9
RST	D8
BUSY	D7

预期结果

1) 把示例程序包中 arduino/libraries 目录下的文件复制到 documents\arduino\libraries，该位置可以通过 Arduino IDE → File → Preferences → Sketchbook location 指定。
2) 点击 Upload 上传工程。
3) 屏幕会显示图像。

注意：本模块不支持局部刷新，刷新需要约 6 秒钟，请耐心等待。

用于 STM32 开发板

- 本例程使用的开发板主控为 STM32F103ZE。
- 本例程基于 HAL 库，因此可以使用 STM32CubeMX 把示例程序移植到其他 STM 芯片上。
- 本例程在 Keil v5 环境下编译通过。

硬件连接

硬件连接到 STM32F103ZE：

e-Paper	STM32F103ZE
3.3V	3.3V
GND	GND
DIN	PA7 (MOSI)
CLK	PA5 (SCK)
CS	PA4
BUSY	PA3
DC	PA2
RST	PA1

预期结果

1) 打开位于 MDK-ARM 目录下的 Keil 工程（epd-demo.uvprojx）。
2) 点击 Build 编译工程。
3) 点击 Download，把工程写入到芯片中。
4) 开发板复位之后，屏幕会显示图像。

   注意：本模块不支持局部刷新，刷新需要约 6 秒钟，请耐心等待。

代码分析

下面以基于树莓派 WiringPi 的函数库的示例程序为例分析驱动代码。

硬件接口函数

驱动代码中的这些函数：DigitalWrite，DigitalRead，SendCommand，SendData，DelayMs 调用硬件设备所提供的接口函数（epdif.h、epdif.c 或 epdif.cpp），分别实现这些功能：IO 电平控制，IO 电平读取，发送 SPI 命令，发送 SPI 数据，毫秒延迟。用户如需移植示例程序，那么需要根据目标硬件，实现 epdif（e-paper display interface）文件中的所有接口。

值得注意的是，在实现 SPI 数据传输的时候，树莓派使用硬件片选，因此传输数据之前不需要在代码中设置 CS 管脚为低电平，而程序在传输 SPI 数据的时候就会自动置 CS 管脚为低电平。然而，对于 Arduino 和 STM32 等设备，用户需要通过在代码中显式设置 CS 管脚为低电平以启用模块的 SPI 传输。

发送命令（SendCommand）和发送数据（SendData）

SendCommand 和 SendData 分别用于向模块发送命令和发送数据，两者区别在于，发送命令的时候，设置 D/C 管脚为低电平，此时通过 SPI 接口发送到模块的数据会当成命令执行。而发送数据的时候，设置 D/C 管脚为高电平，此时通过 SPI 接口发送到模块的数据只是普通的数据，通常是命令后面跟着的参数或者图像数据。

复位（Reset）

RST 为低电平时，模块会复位。用于模块上电或者进入睡眠模式之后，重启模块。重启模块之后，可能还需要通过初始化函数（Init）对模块进行初始化，模块才能正常工作。

初始化（Init）

Init 用于模块上电之后，对模块参数进行配置，或者模块进入睡眠模式之后，唤醒模块。模块初始化流程：reset → power setting → booster soft start → power optimization → power on → panel setting → PLL control → VCM DC setting。

LUT 表设置（SetLut）

模块存储了 look-up table，该表由我们提供，可能因不同批次有所不同。如有改动，我们将会尽快更新示例程序。

显示一帧图像（DisplayFrame）

DisplayFrame 用于给模块发送一帧图像，然后让屏幕刷新以显示图像。

流程：resolution setting → VCM DC setting → VCOM and data interval setting → 发送命令 data start transmission 1 → 发送全白的图像数据（发送 5808 次 0xFF）→发送命令 data start transmission 2 → 发送用于显示的图像数据 → 屏幕刷新。

注意：

- 发送用于显示的图像数据之前，必须发送一帧全白的图像数据（发送 5808 次 0xFF）。
- 1 byte = 8 pixels，不支持灰度显示（只能显示黑色和白色）。Bit 置位的时候代表该像素为白色，反之为黑色/红色。

  例：

  0xC3：8 个像素□□■■■■□□；

  0x00：8 个像素■■■■■■■■；

  0xFF：8 个像素□□□□□□□□；

  0x66：8 个像素■□□■■□□■。
- 本模块不支持局部刷新，每次刷新需要约 6 秒钟，请耐心等待。

睡眠模式（Sleep）

Sleep 可以让模块进入睡眠模式以降低功耗。

进入睡眠模式之后，如需唤醒模块，需要给 RST 复位管脚一个低电平的脉冲。然后可能还需要对重新配置电源参数（不同批次可能有所不同，有些需要重新配置，有些不需要）。因此如需唤醒模块，建议使用 Init 初始化函数，而非 Reset 函数。Init 函数执行过程中会执行 Reset 函数以及相关的初始化命令。

显示图片

有直接和间接两种方法可以让模块显示图片，

直接显示图片：通过库函数直接读取图片的数据进行解码，并转换成对应的数组发送给模块。关于该方法的实现，请参见树莓派 Python 库示例程序（示例程序包中没有直接显示图片的 C 程序）。

间接显示图片：用电脑把图片转换成对应的数组，然后把该数组以.c 文件的形式直接嵌入到程序中。本节将会讲解如何把一张图片转换成对应的数组。

1) 打开 Windows 系统自带的画图工具，新建图片，像素设置成 176x264。
2) 由于模块只能显示两阶的灰度（仅有黑白两色），因此在把图片转换成数组之前，必须转化成单色位图（File → Save as → BMP picture → Monochrome Bitmap）。
   示例程序包中包含一张用于演示的单色位图图片（raspberrypi/python/monocolor.bmp）。
3) 使用 Image2Lcd.exe 软件生成图片所对应的数组（.c 文件）。使用该软件打开图片，设置对应参数：
   - 输出数据类型为：C 语言数组
   - 扫描模式：水平扫描
   - 输出灰度：单色（即两阶）
   - 最大宽度和高度：176 和 264
   - 不勾选“包含图像头数据”
   - 勾选“颜色翻转”（图片中的白色会转换成 1，黑色会转换成 0）
4) 点击“保存”，就会生成对应的.c 文件。将对应的数组复制到工程中，程序调用这个数组进行显示即可。