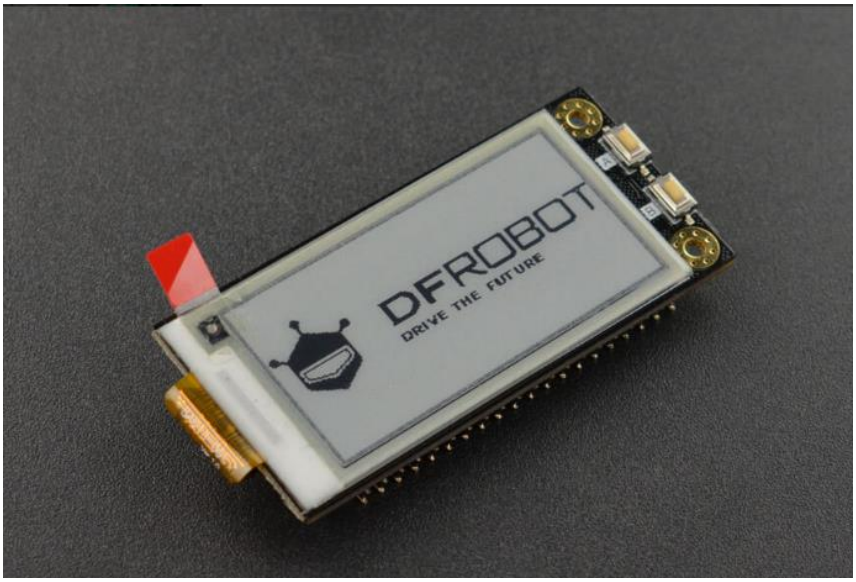


Raspberry Pi e-ink Display Module

Introduction

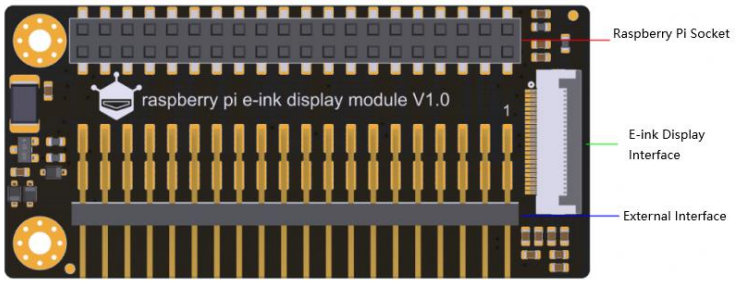
This Raspberry Pi e-ink display module comes with a 2.13" screen with a resolution of 250×122, adopting SPI interface to communicate with Raspberry Pi. It leads out the pins on Raspberry Pi, so there would be no conflicts if you need to use the e-ink display and Raspberry Pi's pins at the same time. The display supports custom font. There are two patch touch switches on the module for expanding more applications. This e-ink display features small size, compact layout, plug and play, and low power consumption. It can provide clear display even in sunlight. When powered off, the e-ink display will continue to display the last screen. Since it will take a long time for the display to refresh fully, we don't recommend using the product in the application that needs frequent refreshing to display data.

⚠ **NOTE**: The e-ink display does not come with fonts chip, but you can download ttf fonts to realize different display effects.

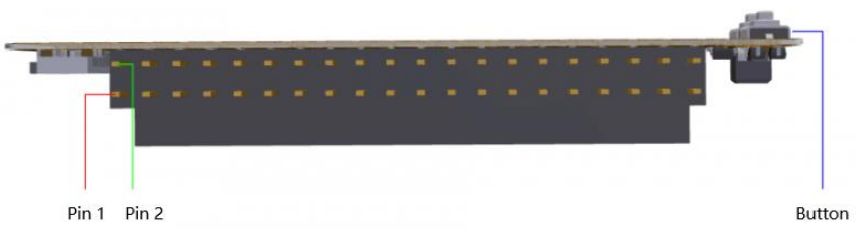
Specification

- Operating Voltage: 3.3V
- Button ×2
  - Button A (Occupy GPIO29, P40 on Raspberry Pi)
  - Button B (Occupy GPIO28, P38 on Raspberry Pi)
- Outline Dimension：66.5 x 31 mm/2.62x1.22”
- Model: GDE0213B1
  - E-ink Display Dimension: 59.2 x 29.2x 1.05mm
  - Dispaly Area Dimension: 48.55 x 23.80mm
- Resolution: 250 x 122 （Logical resolution: 250 × 128 ）
- Communication Interface: SPI
- Display Color: black and white
- Greyscale Level: 2
- Refresh Way: fully or partially refresh
- Refresh Time: 3s for full refresh, 0.5s for partial refresh
- Refresh Power: 26.4mW
- Standby Power: 0.017mW
- Viewing Angle: >170°

Overview



Pinout



Tutorial

Preparation

- Raspberry Pi ×1
- Raspberry Pi e-ink Display Module ×1

Steps

1. Plug the e-ink display into Raspberry Pi, and enabe the SPI interface of the Raspberry Pi.

2. Install DFR_RPi_Display library.

To make the e-ink display work properly, we have to install function library. Click to download **DFRobot_RPi_Display** library. The library provides the driving program of e-ink display and sample code. The library only supports Python at present. We will provide more libraries under users' requirement later.

3. Download the freetype library.

Make sure that your Raspberry Pi is connected to network before downloading. Open the Raspberry Pi terminal and input the following command to download the library.

```
python -m pip install freetype-py
```

4. Run the example program

Enter the examples files under decompress path, and run the example program.

```
cd DFRobot_RPi_Display/examples/dfrobot_epaper
   python xxx.py
```

There are 5 example programs in the file: demo_bitmap, demographics, demo_multi_lingual and so on. Replace the xxx with the name of the example program when running the command.

Examples

1. Example program to dispaly image: demo_bitmap.py

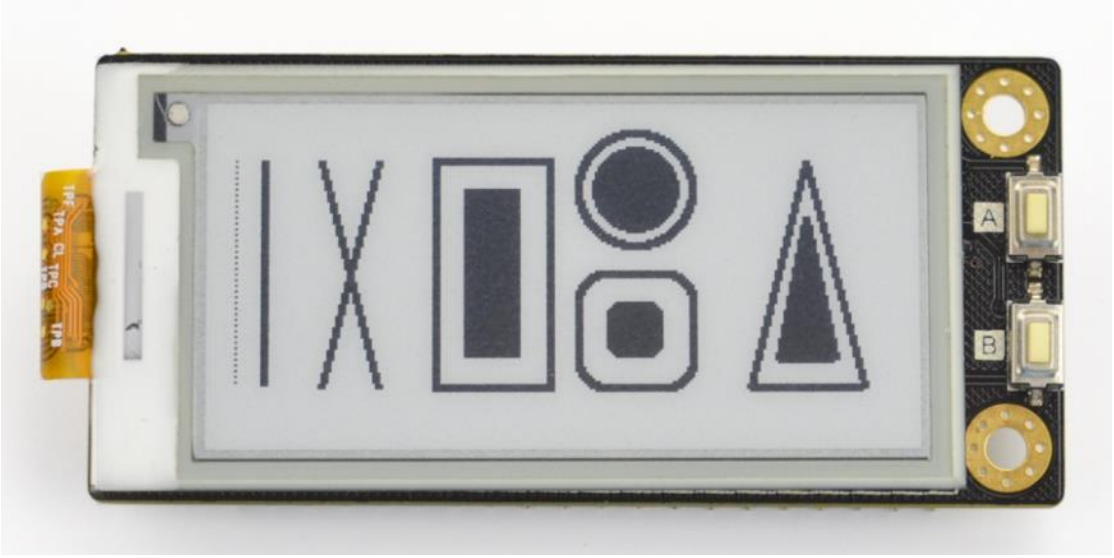
The screen supports monochrome bitmap, change the file name and file path in demo_bitmap.py to display your own picture.

Result:

2. Example program to display geometrical figure: demo_graphics.py

The example programs provide the interface functions of line, rectangle, circle, triangle and so on.

Result:



3. demo_multi_Lingual.py

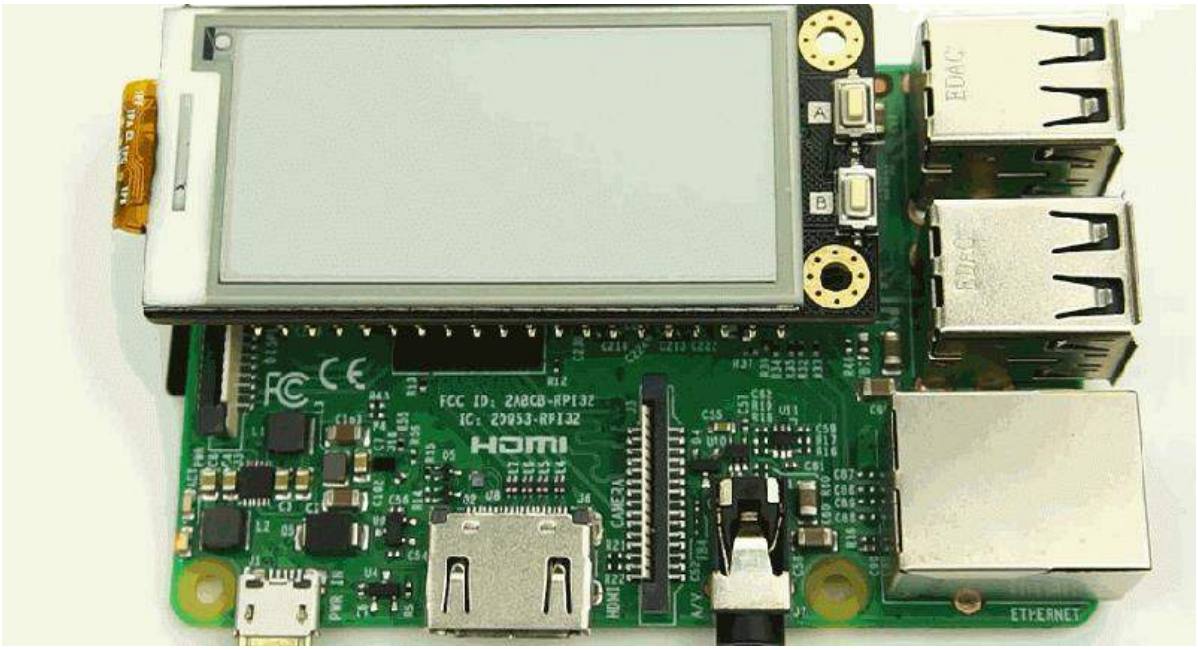
The e-ink screen supports multilingual display, just need to download the ttf library into examples/dfrobot_epaper.

Result:

4. demo_print.py

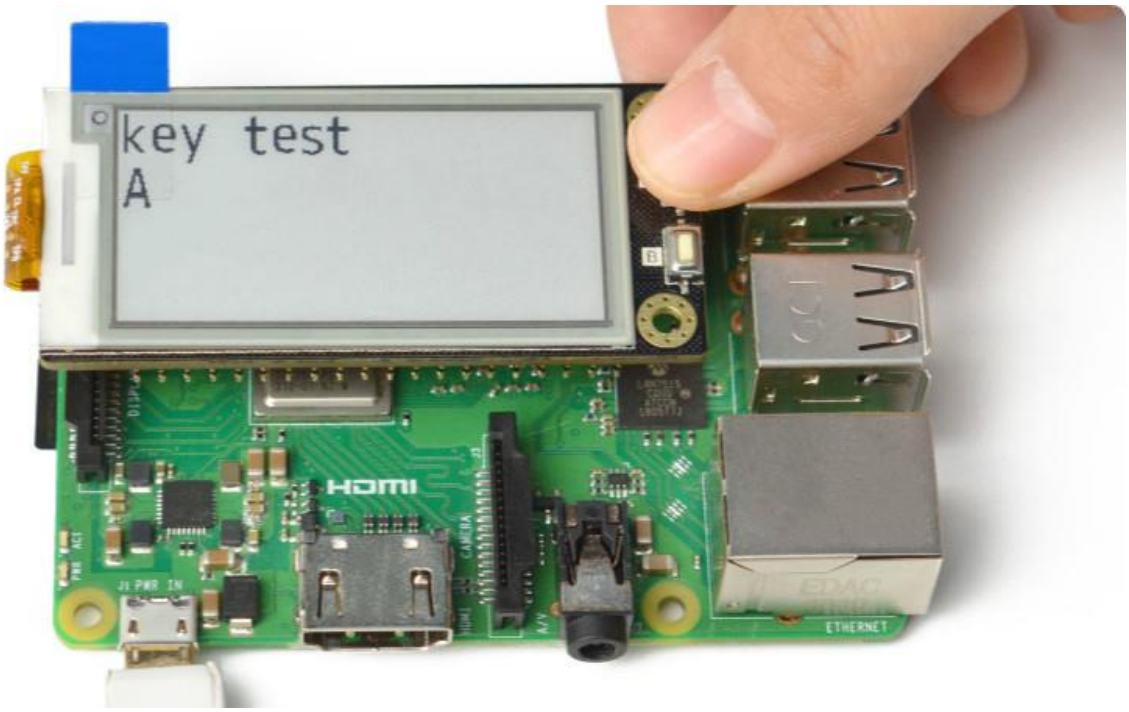
The screen can be refreshed fully or partially, and the font is adjustable.

Result:



5. Example program with keys

Result:



⚠**Attention:** if you want to alter the customized font and language, you have to download the GPL ttf file from the internet, place it to /DFR0bot_RPi_Display/display_extension, then call the file in the programs.

Pins Expansion

The expanded pins of the e-ink display match the ones on the Raspberry Pi, refer to the Pinout.

Please note that SPI, P20 and P21 of Raspberry Pi are occupied by the E-ink display.

Calling Functions

- epaper.clear(self, color)

```
@Function description: clear the screen, remove the content in buffer display.

@Parameter      color: color. Value epaper.WHITE, epaper.BLACK
```

- epaper.flush(self, mode)

```
@Function description: select refresh mode
@Parameter       mode：refresh mode
          epaper.FULL：refresh fully
          epaper.PART：refresh partially
```

Calling fucntions about geometrical figure

- epaper.setLineWidth(self, w)

```
@Function description: set the line width of the figure
@parameter      w: line width
```

- epaper.pixel(self, x, y, color)

```
@Fucntion description：draw a point
@（parameter1, parameter2）   （x, y)：position of the point（row, column）
@parameter3     color：color of the point. Value epaper.BLACK, epaper.WHITE
```

- epaper.VLine(self, x, y, h, color)

```
@Function description: draw a vertical line
@（parameter1, parameter2）  (x, y): starting point of the vertical line.
@ parameter3    h：height of vertical line
@ parameter4    color：line color, value epaper.BLACK, epaper.WHITE.
```

- epaper.HLine(self, x, y, w, color)

```
@Function description: draw a horizontal line
@（parameter1, parameter2）  (x, y): starting point of the horizontal line
@parameter3     w：width of horizontal line
```

```
@parameter4     color: line color, value epaper.BLACK, epaper.WHITE.
```

- epaper.line(self, x, y, x1, y1, color)

```
- @Function description：draw a line arbitrarily
- @（parameter1，parameter2）  （x, y): starting point of the line
- @（parameter3，parameter4）  （x1, y1): ending point of the line
```

```
@ parameter 4    color: line color, value epaper.BLACK, epaper.WHITE.
```

- epaper.triangle(self, x, y, x1, y1, x2, y2, color)

```
- @Function description: draw a triangle
- @（parameter1，parameter2）  （x, y): the position of the first apex of the triangle
- @（parameter3，parameter4）  （x1, y1)：the position of the second apex of the triangle
- @（parameter5，parameter6）  （x2, y2): the position of the third apex of the triangle
```

```
@parameter7     color：triangle color, value epaper.BLACK, epaper.WHITE.
```

- epaper.fillTriangle(self, x, y, x1, y1, x2, y2, color)

```
- @Fucnton description: draw a filled triangle
- @（parameter1，parameter2）  （x, y): the position of the first apex of the triangle
- @（parameter3，parameter4）  （x1, y1)：the position of the second apex of the triangle
- @（parameter5，parameter6）  （x2, y2): the position of the third apex of the triangle
```

```
@parameter7     color：triangle color, value epaper.BLACK, epaper.WHITE.
```

- epaper.rect(self, x, y, w, h, color)

```
- @Function description: draw a rectangle
```

- @（parameter1，parameter2） （x, y): the position of the upper-left apex of the rectangle
- @paramter3 w: rectangle width
- @parameter4 h：rectangle height

```
@parameter5 color：rectangle color value epaper.BLACK, epaper.WHITE.
```

- epaper.fillRect(self, x, y, w, h, color)

- @Function description: draw a filled rectangle
- @（parameter1，parameter2） （x, y): the position of the upper-left apex of the rectangle
- @paramter3 w: rectangle width
- @parameter4 h：rectangle height

```
@parameter5 color：rectangle color value epaper.BLACK, epaper.WHITE.
```

- epaper.circleHelper(self, x, y, r, quadrant, color)

- @Function description：draw an arc in a quadrant
- @（paramter1，paramter2） （x, y): the position of the certer point of the circle
- @parameter3 r：radius of the circle
- @parameter4 quadrant：quadrant. For value, refer to the define related to quadrant in class.

```
@paramter5 color：color，value epaper.BLACK, epaper.WHITE.
```

- epaper.fillCircleHelper(self, x, y, r, quadrant, color)

- @Function description：draw a sector in a quadrant
- @（parameter1，parameter2） （x, y): the position of the center point of the circle
- @parameter3 r：radius of the circle
- @parameter4 quadrant：quadrant. Value 1,2,3,4

```
@parameter5 color：sector color, value epaper.BLACK, epaper.WHITE.
```

- epaper.circle(self, x, y, r, color)

- @Function description：draw a circle
- @（parameter1，parameter2） （x, y): the position of the center point
- @parameter3 r: radius of the circle

@parameter4 color：color of the circle. Value epaper.BLACK, epaper.WHITE。

- epaper.fillCircle(self, x, y, r, color)

- @Function description: draw a filled circle
- @（parameter1，parameter2） （x, y): the position of the center point
- @parameter3 r：radius of the circle

@parameter4 color：color of the circle, value epaper.BLACK, epaper.WHITE.

- epaper.roundRect(self, x, y, w, h, r, color)

- @Function description：draw a rounded rectangle
- @（parameter1，parameter2） （x, y): the crossing point of the extension lines of adjacent sides at the upper-left corner.
- @parameter3 w: the width of the rounded rectangle
- @parameter4 h：the height of the rounded rectangle
- @parameter5 r：the radius of the rounded rectangle

@parameter6 color：the color of the rounded rectangle, value epaper.BLACK, epaper.WHITE.

- epaper.fillRoundRect(self, x, y, w, h, r, color)

- @Function description：draw a filled rounded rectangle
- @（parameter1，parameter2） （x, y）：
- @parameter3 w：width
- @parameter4 h：height
- @parameter5 r：radius of the circle

@parameter6 color：color, value epaper.BLACK, epaper.WHITE。

Calling functions about bitmap display

- epaper.setBitmapSize(self, size)

```
@Function display：set the size of the bitmap
```

```
@parameter      size：size of the bitmap that is going to be magnified. value 1,2,...the bitmap is doubled in the screen by default
```

- epaper.setBitmapFmt(self, fmt)

```
@Function description: set the scan mode of the bitmap
@parameter      fmt:scan mode.The default one is  BITMAP_TBMLLR in screen.
          BITMAP_TBMLLR：scan from top to bottom. The high byte in array will be printed at left,low byte, right. For example, the first byte 0xf0 in the bitmap will be printed like this:[*] [*] [*] [*] [] [] [] []
          BITMAP_TBMRLL：scan from top to bottom, right to left
          BITMAP_BTMLLR：scan from bottom to top, left to right
          BITMAP_BTMRLL：scan from bottom to top, right to left
          BITMAP_LRMTLB：scan from left to right, print from top to bottom
          BITMAP_LRMBLT：scan from left to right, bottom to top
          BITMAP_RLMTLB：scan from right to left, top to bottom
          BIMTAP_RLMBLT：scan from right to left, bottom to top
```

```
          BITMAP_UNKNOW：unknown bitmap scan
```

- epaper.bitmap(self, x, y, bitmap, w, h, color, background)

```
@Function description：draw bitmap displayed in the form of array.
@（parameter1，parameter2）  （x, y): the upper-left vertices of the displayed bitmap
@parameter3     bitmap：array
@parameter4     w：width
@parameter5     h：height
@parameter6     color：color, black by default
```

```
@parameter7     background：background color, white by default
```

- epaper.bitmapFile(self, x, y, path)

```
@Function description：display the file of the bitmap
```

- @（parameter1，parameter2） （x, y): the upper-left vertices of the displayed bitmap

@parmaeter3 path：bitmap file path

Calling functions about font display

- epaper.setExFonts(self, obj)

- @Function description：create the class of font expansion

@parameter obj：one class of FreetypeHelper or other class meeting the requirement.

- epaper.setTextCursor(self, x, y)

- @Functon description：set initial position of the font display

@（parameter1，parameter2） （x, y): the position of the cursor

- epaper.setTextFormat(self, size, color, background, intervalRow = 2, intervalCol = 0)

- @Function description：set font format
- @parameter1 size：font size. Value 1,2,3,...
- @parameter2 color: font color. Value epaper.BLACK, epaper.WHITE.
- @parameter3 background: font background color, white by default
- @parameter4 intervalRow：row-spacing, default space: 2

@parameter5 intervalCol：column-spacing, default space: 0

- epaper.setDisLowerLimite(self, limite)

- @Function description：set font weight

@parameter1 limite: range 0~255

- epaper.setExFontsFmt(self, width, height)

```
- @Function : set the width and height of the expansion font
- @parameter1     width : width
```

```
@parameter2     height : height
```

- epaper.printStr(self, c)

```
- @Function description : display a character string
```

```
@parameter      c : character string
```

- epaper.printStrLn(self, c)

```
- @Function description : display a character string and feed lines.
```

```
@parameter      c : character string
```

Compatibility Test

MCU	**Work Well**
Raspberry Pi 3B	√
Raspberry Pi 3B+	√
Raspberry Pi Zero W	√
aspberry Pi 2B+	√

FAQ

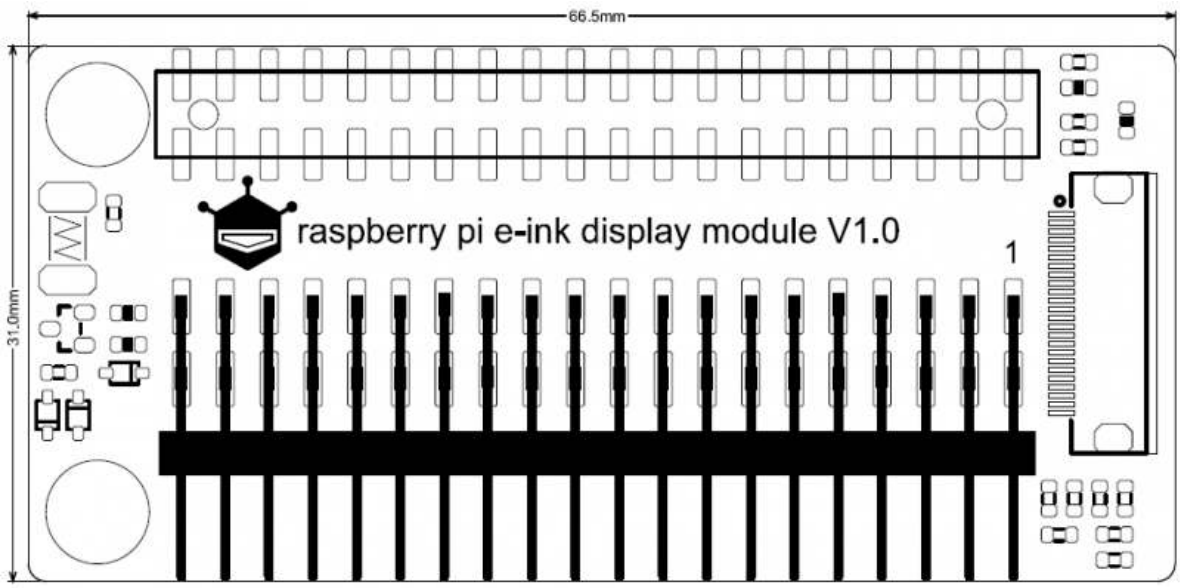
1. Is it normal that the screen flashes for a long time when refreshing?

Although E-ink display is a very popular type of electronic paper display technology, due to the imaging principle, there are also some defects including screen splash and afterimage issue. After repeatedly turning the page, the ink will remain, which causes the afterimage problem. To deal with that, manufactures designed the full refresh function, however, another issue occurred: the screen will flashes for a while when refreshing fully.

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**

Dimension Diagram

- Pin-spacing: 2.54mm
- Board Size: 66.5mm × 31 mm
- Thickness: 1.6mm



More Documents

- Schematic Diagram
- GDEH0213B1 V3.1 Product Specification