



6inch e-Paper HAT

User Manual

OVERVIEW

This is an E-Ink display HAT for Raspberry Pi, 6inch, 800x600 resolution, with embedded controller IT8951, communicating via USB/SPI/I80/I2C interface.

Due to the advantages like ultra-low power consumption, wide viewing angle, clear display without electricity, it is an ideal choice for applications such as shelf label, industrial instrument, and so on.

FEATURES

- No backlight, keeps displaying last content for a long time even when power down
- Low power consumption, basically power is only required for refreshing
- Compatible with Raspberry Pi Zero/Zero W/Zero WH/2B/3B/3B+
- USB/SPI/I80/I2C interface, for connecting with host boards like Raspberry Pi, etc.
- Comes with development resources and manual (examples for Raspberry Pi/STM32)

SPECIFICATIONS

- Operating voltage: 5V
- Interface: USB/SPI/I80/I2C
- Outline dimension: 138.4mm × 101.8mm × 0.954mm

- Display size: 122.4mm × 90.6mm
- Dot pitch: 0.153mm × 0.151mm
- Resolution: 800 × 600
- Display color: black, white
- Gray scale: 2-16 (1-4 bit)
- Full refresh time: <1s
- Total refresh power: 0.6W(typ.)
- Total standby power: 0.3W(typ.)
- Viewing angle: >170°

SPI PINOUTS

SYMBOL	DESCRIPTION
5V	Power input
GND	Ground
MISO	SPI MISO pin
MOSI	SPI MOSI pin
SCK	SPI SCK pin
CS	SPI chip selection, low active
RST	External reset, low active
HRDY	Busy status output, low active

CONTENT

Overview.....	1
Features.....	1
Specifications	1
SPI Pinouts	2
Notes	4
How to use	5
Working protocol.....	5
With Windows PC.....	5
Working with Raspberry Pi.....	7
Working with STM32 Microprocessor	9
SPI	9
I80	11
Picture Display.....	13

NOTES

1. 6inch e-Paper is big size screen, the glass panel and FPC is fragile, please be careful when use it for developing. we recommend you reinforce the FPC with scotch tape when developing.
2. Do Not hot plug the e-Paper
3. There are two version, one is raw panel and another is HAT version. Driver board (IT8951) is required for raw panel, if you are the first time to buy this e-paper, recommend you choose HAT version which come with the driver board.

HOW TO USE

WORKING PROTOCOL

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.

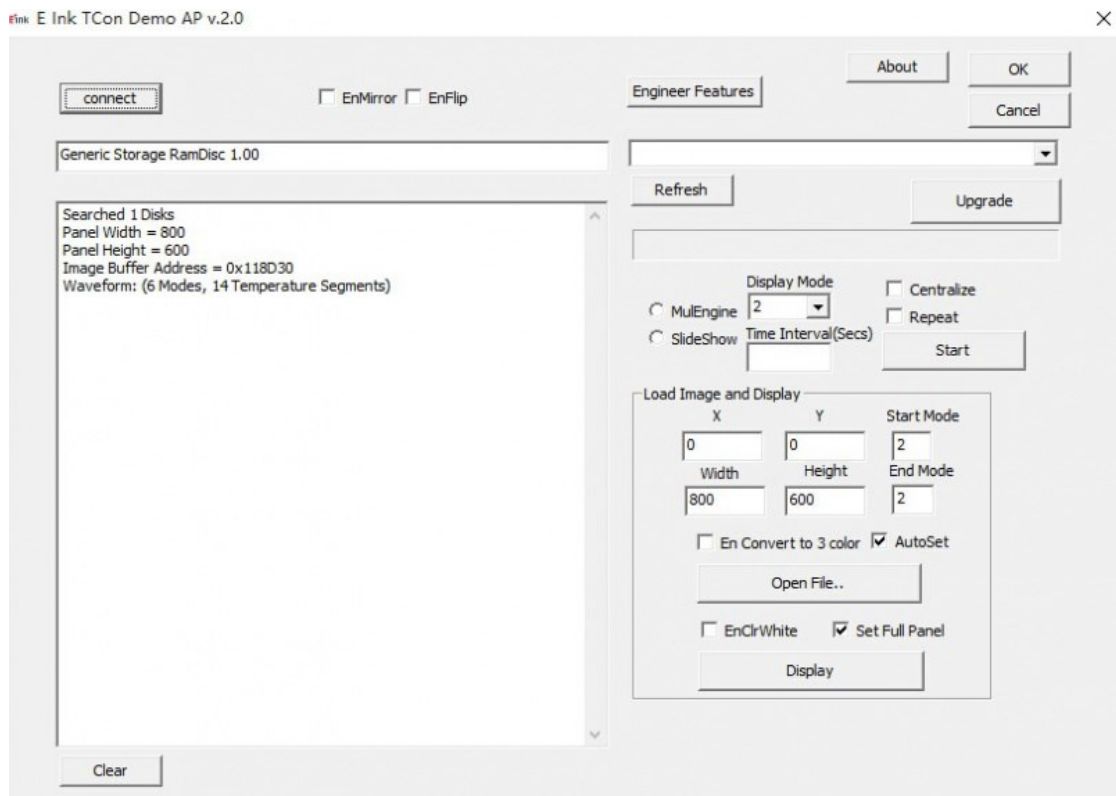
WITH WINDOWS PC

1. Connect e-Paper to driver board.

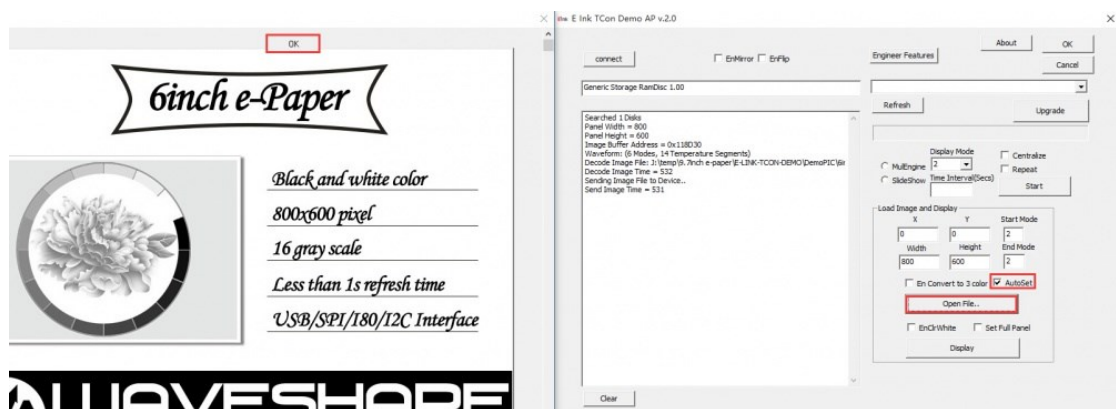


2. Connect Driver board to PC via USB interface

3. Download and open E-LINK-TCOM-DEMO¹ test software
4. Click connect.

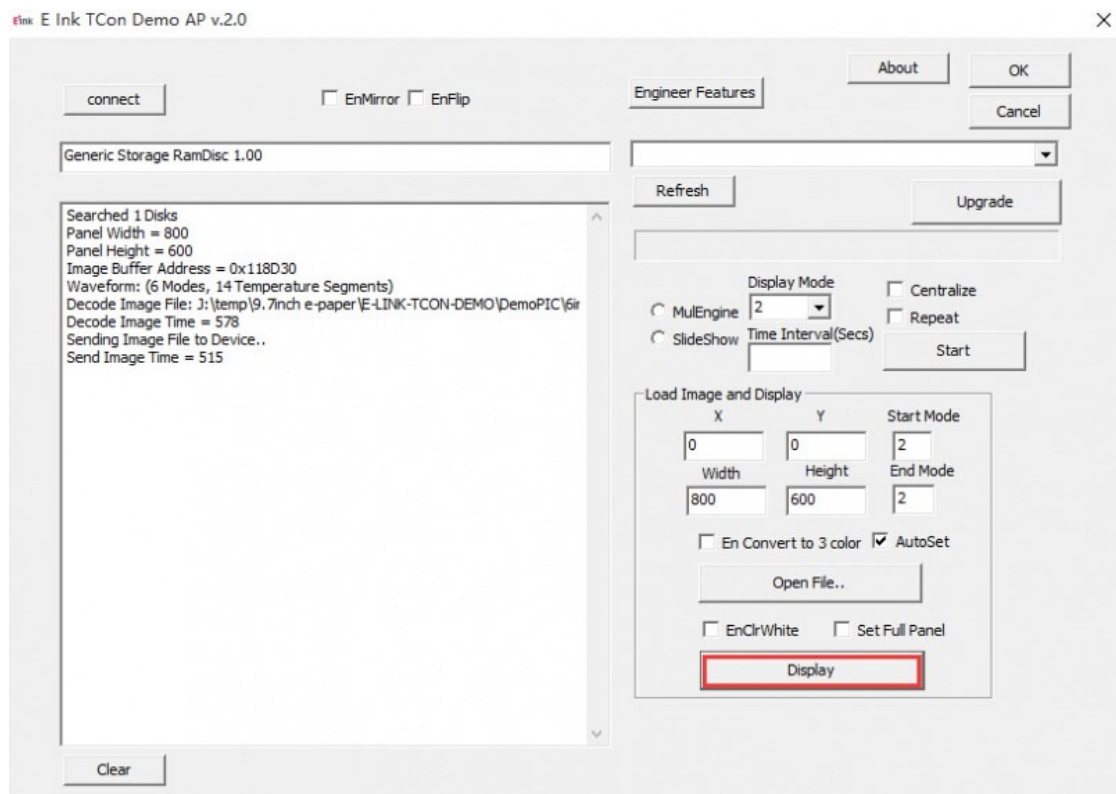


5. Check Option "AutoSet" , and click "Open File" to open one picture. Click "OK" of Browse dialog.



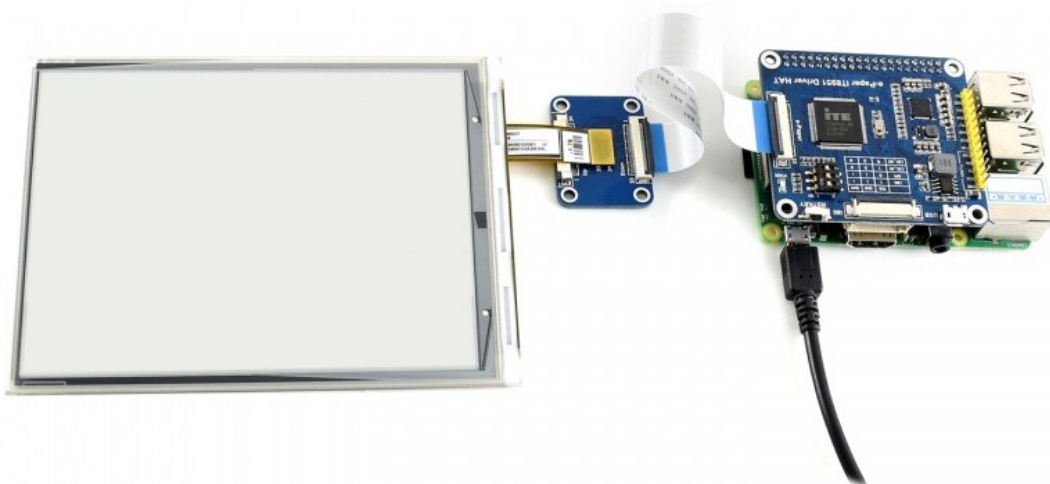
¹ Can be found on wiki

6. Click Display to refresh e-Paper

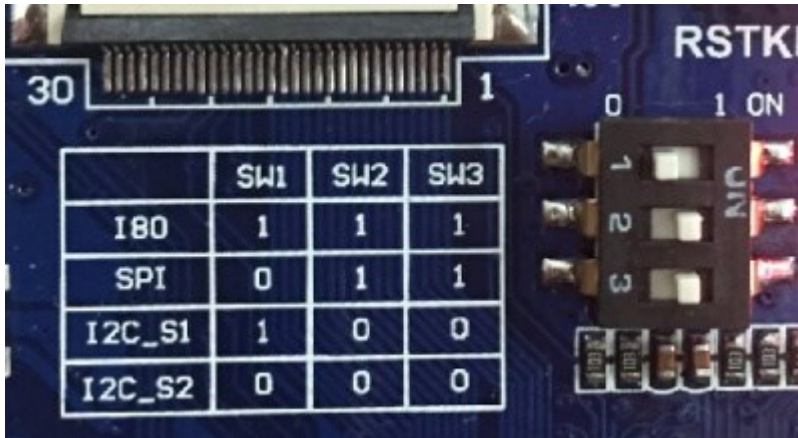


WORKING WITH RASPBERRY PI

1. Connect e-Paper to Driver board and inset the driver board to Raspberry Pi



2. Set the DIP switch to SPI mode



3. Install BCM2835 libraries to your Raspberry Pi, you can download the newest bcm2835 library from its official website

<http://www.airspayce.com/mikem/bcm2835/>

Copy the library you download to Raspberry Pi and install it with commands below. You can also following the instruction on its website above

```
1. tar zxvf bcm2835-1.xx.tar.gz
2. cd bcm2835-1.xx
3. ./configure
4. make
5. sudo make check
6. sudo make install
```

4. Download the demo codes

```
1. git clone https://github.com/waveshare/IT8951.git
2. cd IT8951
3. make
4. sudo ./IT8951 0 0 01.bmp
```

This demo code supports display general BMP pictures directly, if you find that your BMP picture cannot be displayed, please open it on Windows PC with Paint software (Windows APP), save as BMP and try again.

The command `./IT8951 0 0 01.bmp`, the first two parameters is X and Y coordinate of picture's left-top, 01.bmp is the file name of picture

【Note】 For better display, you can try to adjust the VCOM value, the VCOM voltage are different among different panels. The recommend VCOM voltage is stuck in the FPC. For example, if the VCOM voltage= -1.5V, the related VCOM value is 1500



```

10
11 #define CS           8
12 #define HRDY        24
13 #define RESET       17
14 #define VCOM        1500 //e.g. -1.53 = 1530 = 0x5FA
15

```

WORKING WITH STM32 MICROPROCESSOR

Because IT8951 will cost big size of RAM, some of STM32 cannot support without external SDRAM device. So, we here use Open429I as test board, Open429I integrates IS42S16400J (64-MBIT) SDRAM, has full memory to drive the 6inch e-paper.

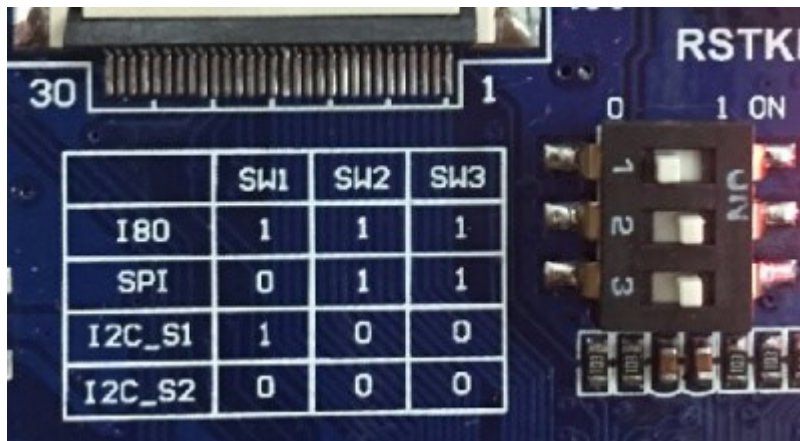
Working with STM32, you can use SPI, I80 or I2C interface. SPI is simple and need a few of GPIO, its speed can also meet the requirement of most applications. I80 is also simple and fast, however, it need to use lots of GPIO. I2C is every slow, which we don't recommend.

SPI

1. Hardware connection

IT8951	STM32	Description
5V	5V	5V Power input
GND	GND	Ground
MISO	PE13	MISO Pin of SPI
MOSI	PE14	MOSI Pin of SPI
SCK	PE12	Clock Pin of SPI
CS	PE11	Chip select (Low active)
RST	PC5	Reset (Low for reset)
HRDY	PA7	BUSY state output (Low for busy)

- Set the DIP switch to SPI Mode

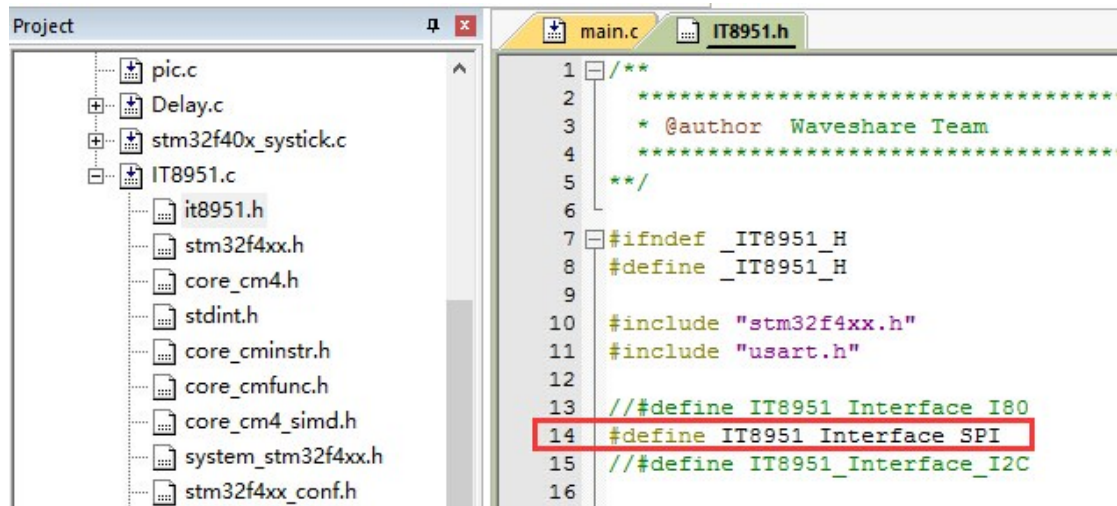


- Download demo code from wiki

Open the project with keil :Open429I-C-IT8951-Demo\Project\9.7-IT8951\MDK-ARM\Project.uvproj

Compile it, then open IT8951.h, check if SPI mode is enabled. Compile it again and download to your board. After downloading, the The information will be printed as

below (115200, 8N1)



```

1  /**
2  * *****
3  * @author Waveshare Team
4  * *****
5  */
6
7  #ifndef _IT8951_H
8  #define _IT8951_H
9
10 #include "stm32f4xx.h"
11 #include "usart.h"
12
13 // #define IT8951 Interface I80
14 #define IT8951 Interface SPI
15 // #define IT8951_Interface_I2C
16

```

```

SYSCLK: 180M
HCLK: 180M
PCLK1: 45M
PCLK2: 90M
IT8951 Example
Panel(W, H) = (800, 600)
Image Buffer Address = 118D30
FW Version = SWv 0.1.
LUT Version = M641
IT8951DisplayExample 01
IT8951HostAreaPackedPixelWrite01
IT8951HostAreaPackedPixelWrite02
IT8951DisplayExample 02
IT8951DisplayExample 03
IT8951HostAreaPackedPixelWrite01
IT8951HostAreaPackedPixelWrite02
IT8951HostAreaPackedPixelWrite01

```

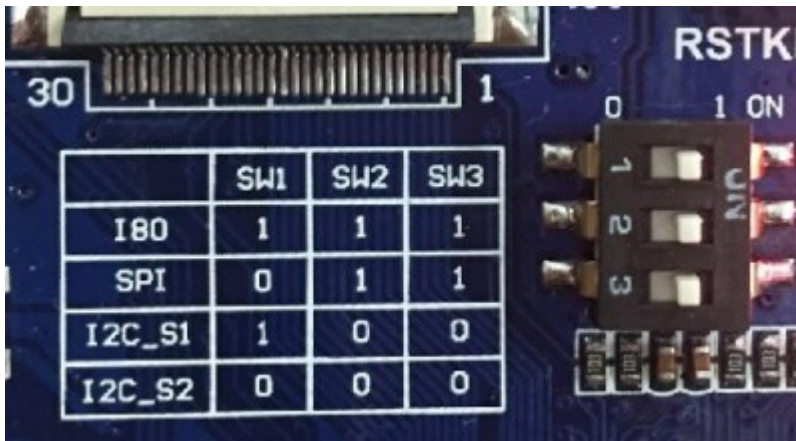
180

1. Hardware connection

IT8951	STM32	Description
VCC	5V	5V Power input
GND	GND	Ground
DBUS0~DBUS15	PB0~PB15	Data pins
HWE	PC1	Write enable (Low active)

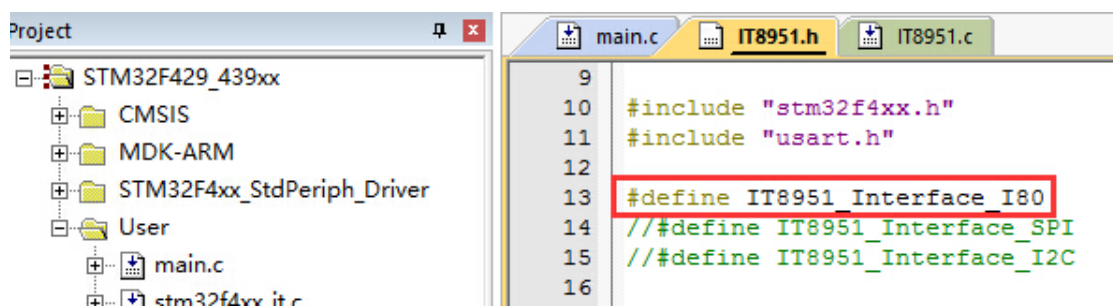
D/C	PC7	Data/Command (Low for command)
CSEL	PC6	Chip select (Low active)
HRD	PC3	Read enable (Low for active)
RST	PC0	Reset (Low for reset)
BUSY	PA7	Busy state output (Low for busy)

- Set the switch to I80 mode



- Download the demo code from wiki

Open project and change set the interface to I80. Modify the interface definition, compile and download it to STM32 board



- After downloading, e-Paper is refreshing image and corresponding information are printed to serial port:

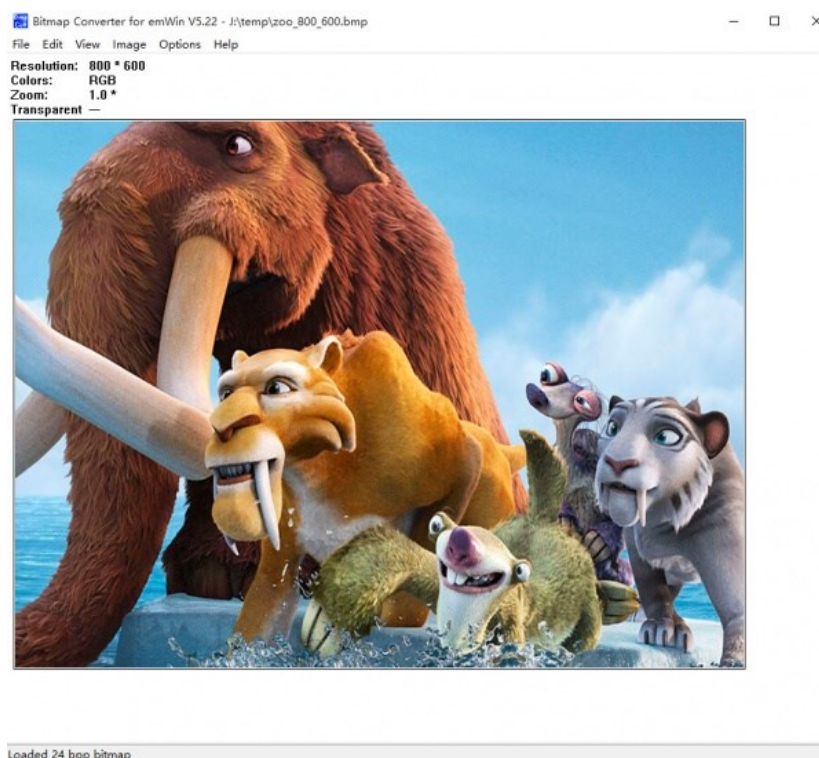
```
SYSCLK:180M
HCLK:180M
PCLK1:45M
PCLK2:90M
IT8951 Example
Panel(W,H) = (800,600)
Image Buffer Address = 118D30
FW Version = SWv_0.1.
LUT Version = M641
IT8951DisplayExample 01
IT8951HostAreaPackedPixelWrite01
IT8951HostAreaPackedPixelWrite02
IT8951DisplayExample 02
IT8951DisplayExample 03
IT8951HostAreaPackedPixelWrite01
IT8951HostAreaPackedPixelWrite02
IT8951HostAreaPackedPixelWrite01
```

PICTURE DISPLAY

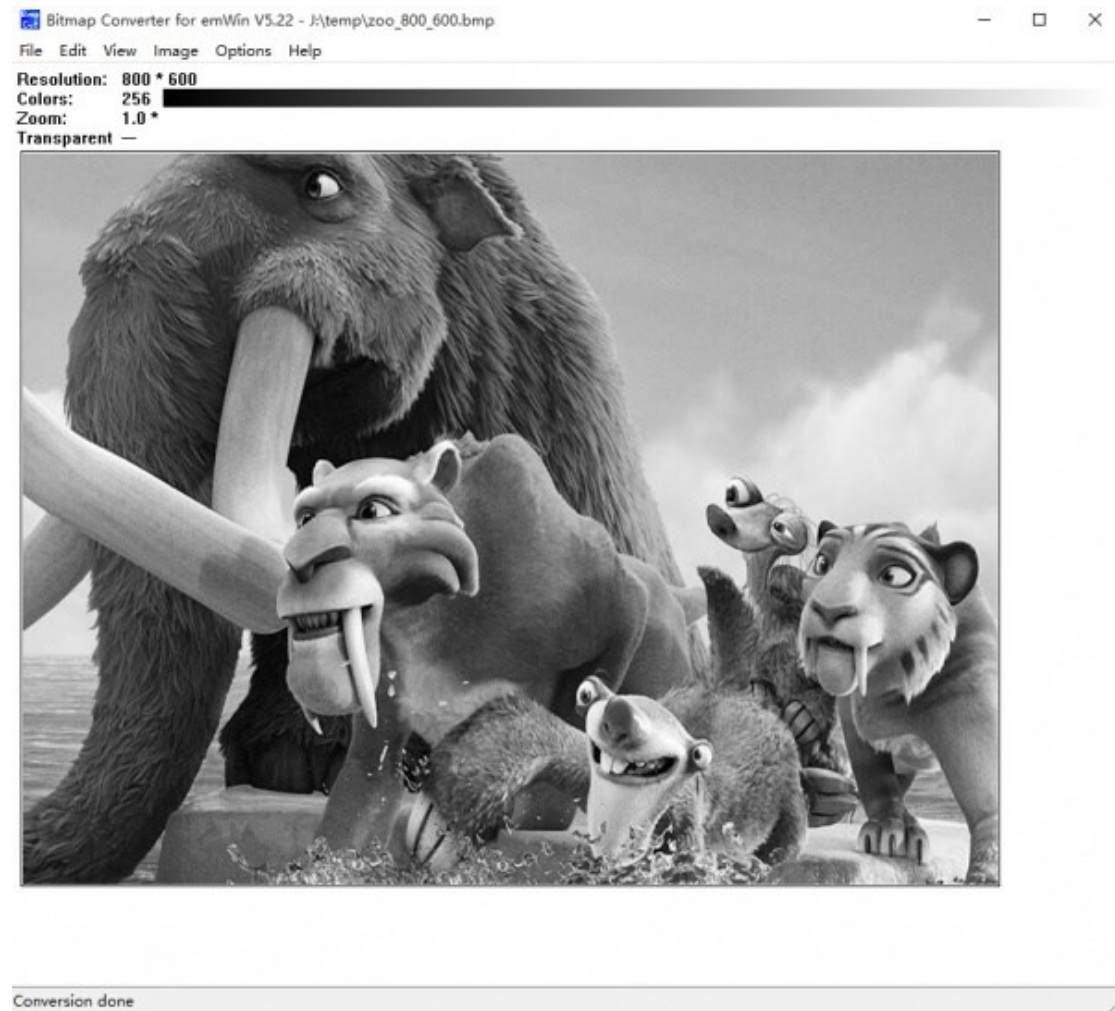
For easy porting our demo code, we display picture with data matrix instead of file system.

We should first convert BMP picture to data matrix (arrays), and use it in demo code.

- 1) Prepare a BMP image, resize the picture to 1200*825 (the resolution of this e-Paper)
- 2) Open [BMP convert software](#), Click File->Open..-> to open the picture as below:

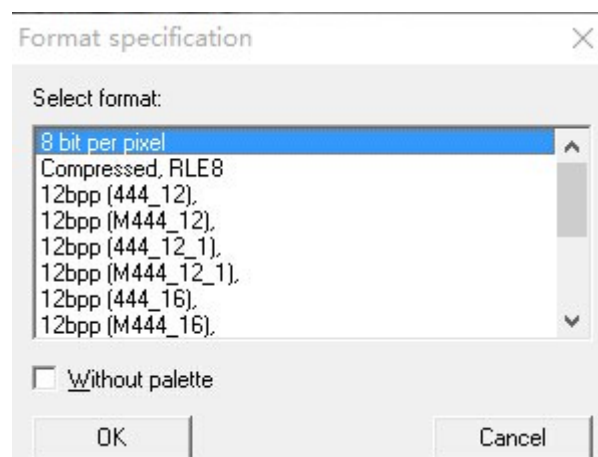


3) Click Image -> Convert to -> Gray256(8 BPP)



4) Click File -> Save As... -> Choose ".c" bitmap file (*.c) -> input file name and click Save.

5) Choose 8 bit per pixel, click OK. A C file will be saved to your PC



6) Add the C file to keil project, detect unusable information

```

1. /*****
   *****/
2. * SEGGER Microcontroller GmbH & Co. KG *
3. * Solutions for real time microcontroller applications *
4. * www.segger.com *
5. *****/
6. * *
7. * C-file generated by *
8. * *
9. * Bitmap Converter for emWin V5.22. *
10. * Compiled Jul 4 2013, 12:18:24 *
11. * (c) 1998 - 2013 Segger Microcontroller GmbH & Co. KG *
12. * *
13. *****/
14. * *
15. * Source file: zoo_800_600 *
16. * Dimensions: 800 * 600 *
17. * NumColors: 256 *
18. * *
19. *****/
20. */
21.
22. #include <stdlib.h>
23.
24. #include "GUI.h"
25.
26. #ifndef GUI_CONST_STORAGE
27. #define GUI_CONST_STORAGE const
28. #endif
29.
30. extern GUI_CONST_STORAGE GUI_BITMAP bmzoo_800_600;
31.

```

```

32.  /*****
      *****/
33.  *
34.  * Palette
35.  *
36.  * Description
37.  * The following are the entries of the palette table.
38.  * The entries are stored as a 32-bit values of which 24
      bits are
39.  * actually used according to the following bit mask:
      0xBBGGRR
40.  *
41.  * The lower 8 bits represent the Red component.
42.  * The middle 8 bits represent the Green component.
43.  * The highest 8 bits represent the Blue component.
44.  */
45.  static GUI_CONST_STORAGE GUI_COLOR _Colorszoo_800_600[] =
      {
46.  0x000000, 0x010101, 0x020202, 0x030303,
47.  0x040404, 0x050505, 0x060606, 0x070707,
48.  0x080808, 0x090909, 0x0A0A0A, 0x0B0B0B,
49.  0x0C0C0C, 0x0D0D0D, 0x0E0E0E, 0x0F0F0F,
50.  0x101010, 0x111111, 0x121212, 0x131313,
51.  0x141414, 0x151515, 0x161616, 0x171717,
52.  0x181818, 0x191919, 0x1A1A1A, 0x1B1B1B,
53.  0x1C1C1C, 0x1D1D1D, 0x1E1E1E, 0x1F1F1F,
54.  0x202020, 0x212121, 0x222222, 0x232323,
55.  0x242424, 0x252525, 0x262626, 0x272727,
56.  0x282828, 0x292929, 0x2A2A2A, 0x2B2B2B,
57.  0x2C2C2C, 0x2D2D2D, 0x2E2E2E, 0x2F2F2F,
58.  0x303030, 0x313131, 0x323232, 0x333333,
59.  0x343434, 0x353535, 0x363636, 0x373737,
60.  0x383838, 0x393939, 0x3A3A3A, 0x3B3B3B,
61.  0x3C3C3C, 0x3D3D3D, 0x3E3E3E, 0x3F3F3F,
62.  0x404040, 0x414141, 0x424242, 0x434343,
63.  0x444444, 0x454545, 0x464646, 0x474747,

```


64.	0x484848,	0x494949,	0x4A4A4A,	0x4B4B4B,
65.	0x4C4C4C,	0x4D4D4D,	0x4E4E4E,	0x4F4F4F,
66.	0x505050,	0x515151,	0x525252,	0x535353,
67.	0x545454,	0x555555,	0x565656,	0x575757,
68.	0x585858,	0x595959,	0x5A5A5A,	0x5B5B5B,
69.	0x5C5C5C,	0x5D5D5D,	0x5E5E5E,	0x5F5F5F,
70.	0x606060,	0x616161,	0x626262,	0x636363,
71.	0x646464,	0x656565,	0x666666,	0x676767,
72.	0x686868,	0x696969,	0x6A6A6A,	0x6B6B6B,
73.	0x6C6C6C,	0x6D6D6D,	0x6E6E6E,	0x6F6F6F,
74.	0x707070,	0x717171,	0x727272,	0x737373,
75.	0x747474,	0x757575,	0x767676,	0x777777,
76.	0x787878,	0x797979,	0x7A7A7A,	0x7B7B7B,
77.	0x7C7C7C,	0x7D7D7D,	0x7E7E7E,	0x7F7F7F,
78.	0x808080,	0x818181,	0x828282,	0x838383,
79.	0x848484,	0x858585,	0x868686,	0x878787,
80.	0x888888,	0x898989,	0x8A8A8A,	0x8B8B8B,
81.	0x8C8C8C,	0x8D8D8D,	0x8E8E8E,	0x8F8F8F,
82.	0x909090,	0x919191,	0x929292,	0x939393,
83.	0x949494,	0x959595,	0x969696,	0x979797,
84.	0x989898,	0x999999,	0x9A9A9A,	0x9B9B9B,
85.	0x9C9C9C,	0x9D9D9D,	0x9E9E9E,	0x9F9F9F,
86.	0xA0A0A0,	0xA1A1A1,	0xA2A2A2,	0xA3A3A3,
87.	0xA4A4A4,	0xA5A5A5,	0xA6A6A6,	0xA7A7A7,
88.	0xA8A8A8,	0xA9A9A9,	0xA9A9A9,	0xABABAB,
89.	0xACACAC,	0xADADAD,	0xAEAEAE,	0xAFAFAF,
90.	0xB0B0B0,	0xB1B1B1,	0xB2B2B2,	0xB3B3B3,
91.	0xB4B4B4,	0xB5B5B5,	0xB6B6B6,	0xB7B7B7,
92.	0xB8B8B8,	0xB9B9B9,	0xBABABA,	0BBBBBBB,
93.	0xBCBCBC,	0BDBDBD,	0BEBEBE,	0BFBFBF,
94.	0xC0C0C0,	0xC1C1C1,	0xC2C2C2,	0xC3C3C3,
95.	0xC4C4C4,	0xC5C5C5,	0xC6C6C6,	0xC7C7C7,
96.	0xC8C8C8,	0xC9C9C9,	0xCACACA,	0CBCBCB,
97.	0xCCCCCC,	0CDCDCD,	0CECECE,	0CFCFCF,
98.	0xD0D0D0,	0xD1D1D1,	0xD2D2D2,	0xD3D3D3,
99.	0xD4D4D4,	0xD5D5D5,	0xD6D6D6,	0xD7D7D7,

```

100. 0xD8D8D8, 0xD9D9D9, 0xDADADA, 0xDBDBDB,
101. 0xDCDCDC, 0xDDDDDD, 0xDEDEDE, 0xDFDFDF,
102. 0xE0E0E0, 0xE1E1E1, 0xE2E2E2, 0xE3E3E3,
103. 0xE4E4E4, 0xE5E5E5, 0xE6E6E6, 0xE7E7E7,
104. 0xE8E8E8, 0xE9E9E9, 0xEAEAEA, 0xEBEBEB,
105. 0xECECEC, 0xEDEDED, 0xEEEEEE, 0xEF EF EF,
106. 0xF0F0F0, 0xF1F1F1, 0xF2F2F2, 0xF3F3F3,
107. 0xF4F4F4, 0xF5F5F5, 0xF6F6F6, 0xF7F7F7,
108. 0xF8F8F8, 0xF9F9F9, 0xFAFAFA, 0xFBFBFB,
109. 0xFCFCFC, 0xFD FDFD, 0xFEFEFE, 0xFFFF FFF
110. };
111.
112. static GUI_CONST_STORAGE GUI_LOGPALETTE _Palzoo_800_600 =
    {
113. 256, // Number of entries
114. 0, // No transparency
115. &_Colorszoo_800_600[0]
116. };
117.
118. GUI_CONST_STORAGE GUI_BITMAP bmzoo_800_600 = {
119. 800, // xSize
120. 600, // ySize
121. 800, // BytesPerLine
122. 8, // BitsPerPixel
123. _aczoo_800_600, // Pointer to picture data (indices)
124. &_Palzoo_800_600 // Pointer to palette
125. };

```

7) Modify the codes

```
1. static GUI_CONST_STORAGE unsigned char _aczoo_800_600[] = {
```

To (You can change the name of the array to every one you like)

```
1. const unsigned char zoo_800_600[] = {
```

8) modify the IT8951.c:

```
1. extern const unsigned char zoo_800_600[];
2. void IT8951DisplayExample3()
```

```
3. {
4. IT8951LdImgInfo stLdImgInfo;
5. IT8951AreaImgInfo stAreaImgInfo;
6. TWord width = gstI80DevInfo.usPanelW;
7. TWord high = gstI80DevInfo.usPanelH;
8. TDWord i;
9.
10. for (i = 0;i < width*high;i++)
11. {
12. gpFrameBuf[i] = zoo_800_600[i];
13. }
14.
15. IT8951WaitForDisplayReady();
16.
17. //Setting Load image information
18. stLdImgInfo.ulStartFBAddr = (TDWord) gpFrameBuf;
19. stLdImgInfo.usEndianType = IT8951_LDIMG_L_ENDIAN;
20. stLdImgInfo.usPixelFormat = IT8951_8BPP;
21. stLdImgInfo.usRotate = IT8951_ROTATE_0;
22. stLdImgInfo.ulImgBufBaseAddr = gulImgBufAddr;
23. //Set Load Area
24. stAreaImgInfo.usX = 0;
25. stAreaImgInfo.usY = 0;
26. stAreaImgInfo.usWidth = width;
27. stAreaImgInfo.usHeight = high;
28.
29. IT8951HostAreaPackedPixelWrite(&stLdImgInfo,
    &stAreaImgInfo); //Display function 2
30. IT8951DisplayArea(0,0, gstI80DevInfo.usPanelW,
    gstI80DevInfo.usPanelH, 2);
```

【Note】 The instruction herein only show you how to use the demo codes we provide.

If you need further features, please create your own code by referring to resources which are provided in wiki.