RAiO

RA8876_Lite

User Guide

Apr 22, 2015

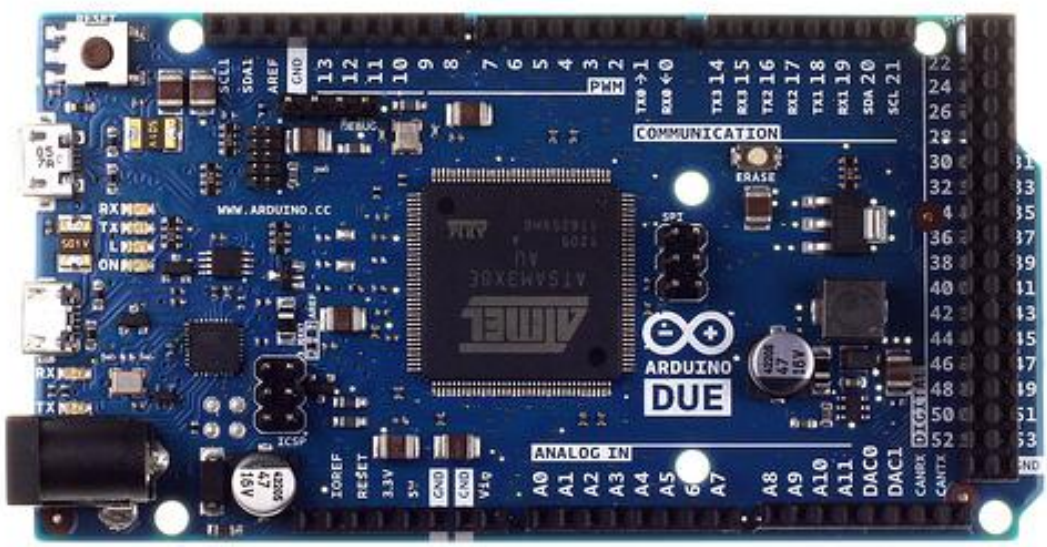
Revise History		
Version	**Date**	**Description**
1.0		Initial Release

Chapter 1 RA8876_Lite introduction

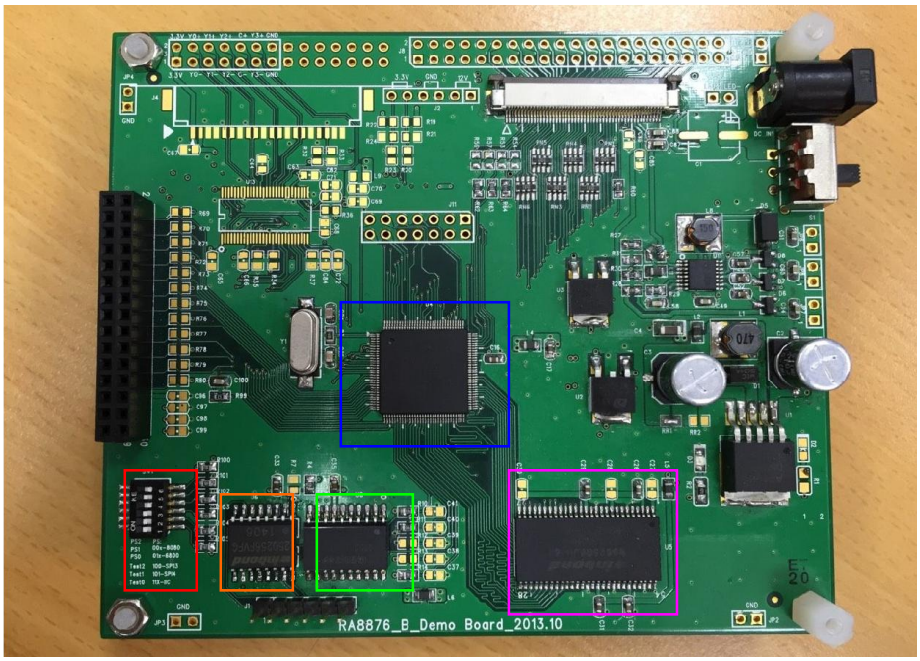
RA8876_Lite provides GUI application source code that based on the Arudino Due development board, and it can be connected to RA8876 driver board and SD card adapter. This document will help users to rapidly realize how to apply the Arduino Due development environment with RA8876 (Parallel RGB output) /RA8877 (LVDS output) for the TFT – LCD solutions.

Hardware requirements

1. Arduino Due development board



2. RA8876 or RA8877's evaluation board (mounted SPI FLASH ROM and Genitop Font ROM IC on board)



- RA8876 Chip
- Select SPI 4 wire interface
- Serial Flash ROM for DMA function
- Genitop Font ROM
- SDRAM

3. SD card adapter

4. SD card (maximum 4GB)

Software requirements

Arduino IDE 1.5.7 http://arduino.cc/en/Main/Software

RA8876 Image_Tool_1.0 www.raio.com.tw

RA8876_Lite features

RA8876_Lite provides application interface (API) that is used for the major built-in functions of RA8876 TFT LCD controller, all demonstration in this document is based on the SPI interface of Arduino Due development board, that it is used with RA8876 for displaying the 16BPP color depth image on the TFT-LCD. The following is the demo features in this document:

Initialization

RA8876's initialized procedures.

Memory configuration & Window

Describe how to configure the external memory of RA8876 which is corresponded to the distinct operating windows.

Graphic

RA8876 is in Graphic Mode, the Arduino Due writes the color image data.

RA8876 is in Graphic Mode, the Arduino Due writes user's customized ASCII fonts.

Text

RA8876 is in Text Mode, the Arduino Due writes built-in the ASCII fonts with RA8876's text function, illustrate the font enlarge function of RA8876.

Display ASCII code, BIG5 and GB2312 fonts. Please note that fonts are provided by the Genitop's Font ROM.

Geometric Draw

RA8876 is in Graphic Mode, the Arduino Due draws line, square, square fill, circle square, circle square fill, triangle, triangle fill, circle, circle fill, ellipse, ellipse fill on the display through the particular functions of RA8876.

BTE

RA8876 is in Graphic Mode, the Arduino Due shows RA8876 BTE functions on the display:

- BTE memory copy
- BTE memory ROP logic operation and copy
- BTE memory copy with chroma key
- Arduino Due executes memory write with ROP logic operation through BTE engine
- Arduino Due executes memory write with chroma key through BTE engine
- Arduino Due executes memory write with color expansion through BTE engine
- Arduino Due executes memory write with color expansion and chroma key through BTE engine
- BTE pattern fill
- BTE pattern fill with chroma key

DMA

RA8876 is in Graphic Mode, the RA8876 reads image data from serial flash directly, and then writes into the external memory of RA8876 through the DMA engine.

PWM

RA8876 PWM initial setup and frequency calculations, duty cycle configure. (Need an oscilloscope to measure the produced frequency)

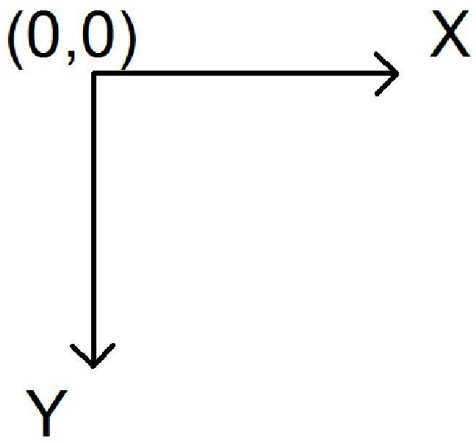
Arduino SD

Arduino Due reads image data from SD card and then writes data into the RA8876 external memory.

Arduino Due reads image data from SD card and then writes into RA8876 external memory through BTE function.

Note:

1. Display coordinate system in this document:



2. The display resolution is 800 * 600 in this document, for other resolutions, please refer to Chapter 2 Initialization and Chapter 3 Memory configuration & Window.

3. RA8876_Lite user defines variables type as following:

typedef	signed char	rs8;
typedef	signed short	rs16;
typedef	signed long	rs32;
typedef	unsigned char	ru8;
typedef	unsigned short	ru16;
typedef	unsigned long	ru32;

4. Circuitry connection please refer to appendix A :

Figure A-1

Figure A-2

Chapter 2 Initialization

RA8876 initial process is as follows:

RA8876 hardware reset

↓

RA8876 PLL initialization

↓

RA8876 SDRAM initialization

↓

RA8876 General setting

↓

RA8876 TFT timing setting

↓

RA8876 Image display memory and windows initialized setting

↓

RA8876 TFT Display on

2.1 Hardware reset

begin()

RA8876 hardware reset program is included in the function begin().
When the function begin() return “true”, indicates hardware reset successful and connect RA8876 or RA8877 correctly, if return “false”, indicates connect fail, please check the Arduino SPI bus is correctly connected to RA8876 driver board or not?

2.2 PLL initialization

ra8876PllInitial()

This PLL initialized subroutine will automatically finish the related initialization works depending on the parameters which defined in the RA8876_Lite.h. So according to their display requirement, users just need to define the parameters as the following.

```
#define OSC_FREQ     10   // OSC clock frequency, unit: MHz.
#define DRAM_FREQ    120  // SDRAM clock frequency, unit: MHz.
#define CORE_FREQ    120  // Core (system) clock frequency, unit: MHz.
#define SCAN_FREQ    40   // Panel Scan clock frequency, unit: MHz.
```

Define	Description
OSC_FREQ	Crystal resonator for RA8876, suggested 10MHz
DRAM_FREQ	SDRAM access clock, suggested 50~160MHz
CORE_FREQ	RA8876 system core clock, suggested 50~130MHz
SCAN_FREQ	TFT driving clock PCLK, refer to LCD SPEC specified PCLK frequency requirements

Note: DRAM_FREQ >= CORE_FREQ
CORE_FREQ >= 2 * SCAN_FREQ

2.3 SDRAM initialization

RA8876 does not have the built-in memory, so RA8876 must expand an external SDRAM as the image operating buffer and display memory.

ra8876SdramInitial()

The function will refer to RA8876_Lite.h define as following, and execute SDRAM initialize automatically, users only need to select one of the following defined SDRAM part number. If the following parts do not meet your device, please refer to the contents of this subroutine and create your own one.

```
#define IS42SM16160D
#define IS42S16320B
#define IS42S16400F
#define M12L32162A
#define M12L2561616A
#define M12L64164A
#define W9825G6JH
#define W9812G6JH
```

```
#define MT48LC4M16A
#define K4S641632N
#define K4S281632K
```

2.4 General setting

According to customer's display requirement, the following registers should be set during executing the initialization for RA8876 or RA8877. The relevant information please refer to RA8876 or RA8877's specification and the bit definition of each register in the RA8876_Lite.h

```
lcdRegWrite(RA8876_CCR);//01h
lcdDataWrite(RA8876_PLL_ENABLE<<7|RA8876_WAIT_NO_MASK<<6|RA8876_KEY_SCAN_DISABLE<<5|RA8876_TFT_OUTPUT24<<3|RA8876_I2C_MASTER_DISABLE<<2|RA8876_SERIAL_IF_ENABLE<<1|RA8876_HOST_DATA_BUS_SERIAL);

lcdRegWrite(RA8876_MACR);//02h
lcdDataWrite(RA8876_DIRECT_WRITE<<6|RA8876_READ_MEMORY_LRTB<<4|RA8876_WRITE_MEMORY_LRTB<<1);

lcdRegWrite(RA8876_ICR);//03h
lcdDataWrite(RA8877_LVDS_FORMAT<<3|RA8876_GRAPHIC_MODE<<2|RA8876_MEMORY_SELECT_IMAGE);

lcdRegWrite(RA8876_MPWCTR);//10h
lcdDataWrite(RA8876_PIP1_WINDOW_DISABLE<<7|RA8876_PIP2_WINDOW_DISABLE<<6|RA8876_SELECT_CONFIG_PIP1<<4|RA8876_IMAGE_COLOCR_DEPTH_16BPP<<2|TFT_MODE);

lcdRegWrite(RA8876_PIPCDEP);//11h
lcdDataWrite(RA8876_PIP1_COLOR_DEPTH_16BPP<<2|RA8876_PIP2_COLOR_DEPTH_16BPP);

lcdRegWrite(RA8876_AW_COLOR);//5Eh
lcdDataWrite(RA8876_CANVAS_BLOCK_MODE<<2|RA8876_CANVAS_COLOR_DEPTH_16BPP);

lcdRegDataWrite(RA8876_BTE_COLR,RA8876_S0_COLOR_DEPTH_16BPP<<5|RA8876_S
```

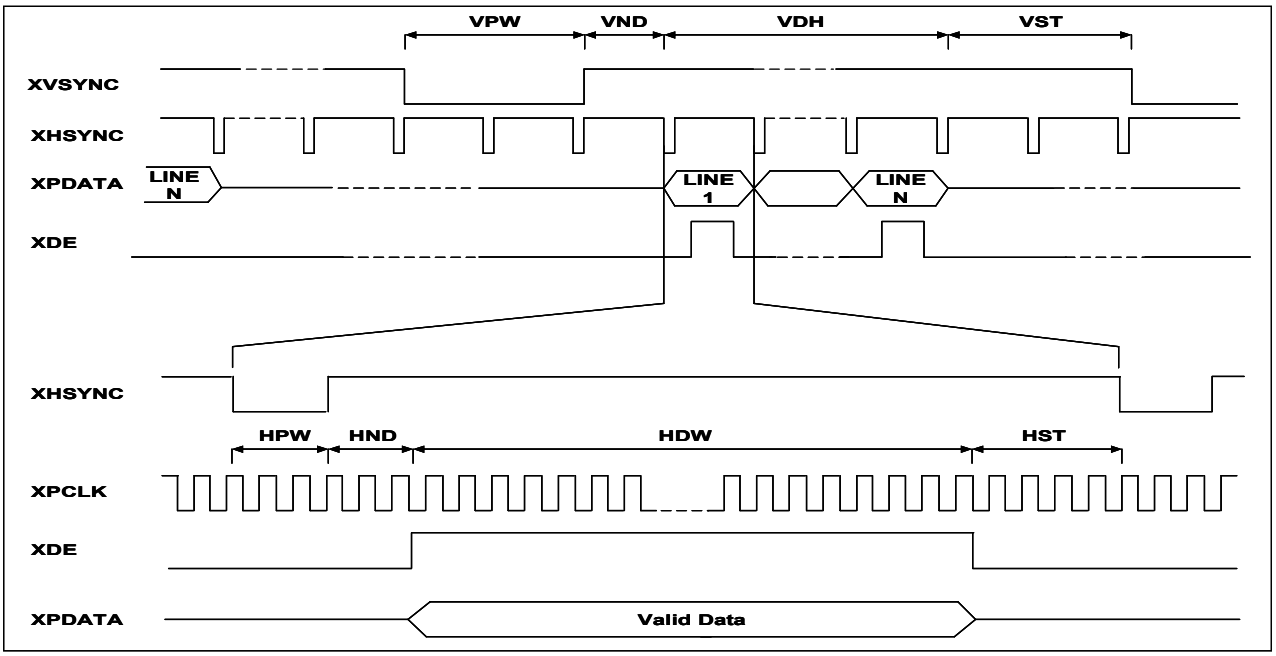
1_COLOR_DEPTH_16BPP<<2|RA8876_S0_COLOR_DEPTH_16BPP);//92h

2.5 TFT timing setting

According to the TFT LCD's datasheet, the relevant timing should be set for RA8876 as below. The following definitions are defined in the RA8876_Lite.h.

```
#define TFT_MODE     0    //0:SYNC_mode(SYNC+DE mode), 1: DE mode
//if sync only mode do not connect DE signal or set XDE_INV to 1
#define XHSYNC_INV    0 // 0:no inversion, 1:inversion
#define XVSYNC_INV    0 // 0:no inversion, 1:inversion
#define XDE_INV       0 // 0:no inversion, 1:inversion
#define XPCLK_INV    1    // 0:no inversion, 1:inversion
#define HPW          8    //
#define HND          38
#define HDW          800
#define HST          16
#define VPW          8
#define VND          15
#define VDH          600
#define VST          12

//use RA8877 need to set LVDS output data format, set to 0 when use RA8876
#define RA8877_LVDS_FORMAT    0       // 0:Format1(VESA format), 1:Format2 =(JEIDA format)
```

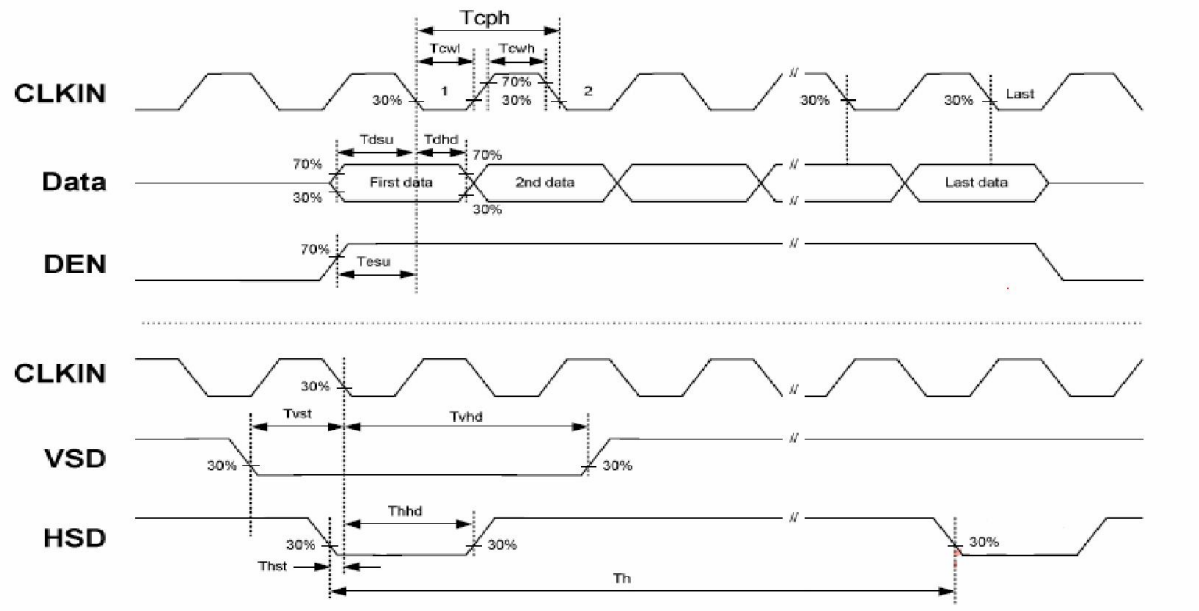
RA8876 Output Timing Reference



The TFT LCD AT080TN52, TFT timing requirements as below:

Item	Symbol	Values			Unit	Remark
		Min.	Typ.	Max.		
Horizontal Display Area	thd	-	800	-	DCLK	
DCLK Frequency	fclk	-	40	50	MHz	
One Horizontal Line	th	862	1056	1200	DCLK	
HS pulse width	thpw	1	-	40	DCLK	
HS Back Porch(Blanking)	thb	46	46	46	DCLK	
HS Front Porch	thfp	16	210	354	DCLK	

Item	Symbol	Values			Unit	Remark
		Min.	Typ.	Max.		
Vertical Display Area	tvd	-	600	-	TH	
VS period time	tv	624	635	700	TH	
VS pulse width	tvpw	1	-	20	TH	
VS Back Porch(Blanking)	tvb	23	23	23	TH	
VS Front Porch	tvfp	1	12	77	TH	



TFT timing initialization setup program:

lcdRegWrite(RA8876_DPCR);//12h
lcdDataWrite(XPCLK_INV<<7|RA8876_DISPLAY_OFF<<6|RA8876_OUTPUT_RGB);

lcdRegWrite(RA8876_PCSR);//13h
lcdDataWrite(XHSYNC_INV<<7|XVSYNC_INV<<6|XDE_INV<<5);

lcdHorizontalWidthVerticalHeight(HDW,VDH);
lcdHorizontalNonDisplay(HND);
lcdHsyncStartPosition(HST);
lcdHsyncPulseWidth(HPW);
lcdVerticalNonDisplay(VND);
lcdVsyncStartPosition(VST);
lcdVsyncPulseWidth(VPW);

2.6 Image display memory initialization setting

Please refer to RA8876_Lite.h's definitions as the following, user need to define the following values:

// define screen resolution
#define SCREEN_WIDTH 800
#define SCREEN_HEIGHT 600

```
// user image memory buffer page configure
// the maximum number of pages depending on the capacity of the SDRAM and what the page
//use of color depth, width, height.
// for example, the paper selects the SDRAM model W9825G6JH its capacity = 32Mbyte
//page_size = 800*600*2byte(16bpp) = 960000byte
//maximum number = 32/0.96 = 33.3
//so maximum configure page is 33 for application
// this article is configure 10 pages to display applications such as the following, the size of
//each page is the same to the display size 800 * 600, 16bpp color depth, that is configure for
//vertical direction and mulit-page buffer application

#define PAGE1_START_ADDR    0
#define PAGE2_START_ADDR    800*600*2
#define PAGE3_START_ADDR    800*600*2*2
#define PAGE4_START_ADDR    800*600*2*3
#define PAGE5_START_ADDR    800*600*2*4
#define PAGE6_START_ADDR    800*600*2*5
#define PAGE7_START_ADDR    800*600*2*6
#define PAGE8_START_ADDR    800*600*2*7
#define PAGE9_START_ADDR    800*600*2*8
#define PAGE10_START_ADDR    800*600*2*9
```

Windows initialization program:

```
displayImageStartAddress(PAGE1_START_ADDR);
displayImageWidth(SCREEN_WIDTH);
displayWindowStartXY(0,0);
canvasImageStartAddress(PAGE1_START_ADDR);
canvasImageWidth(SCREEN_WIDTH);
activeWindowXY(0,0);
activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
```

2.7 TFT display on

After running the RA8876 initialization setting, usually executes writing image data into display memory firstly, then turn the display on. RA8876 TFT LCD timing controller will fetch the image

data from the display windows block of the image display memory and then output to the LCD to display automatically, after turning on the display.

displayOn()

Description:
Display on/off.

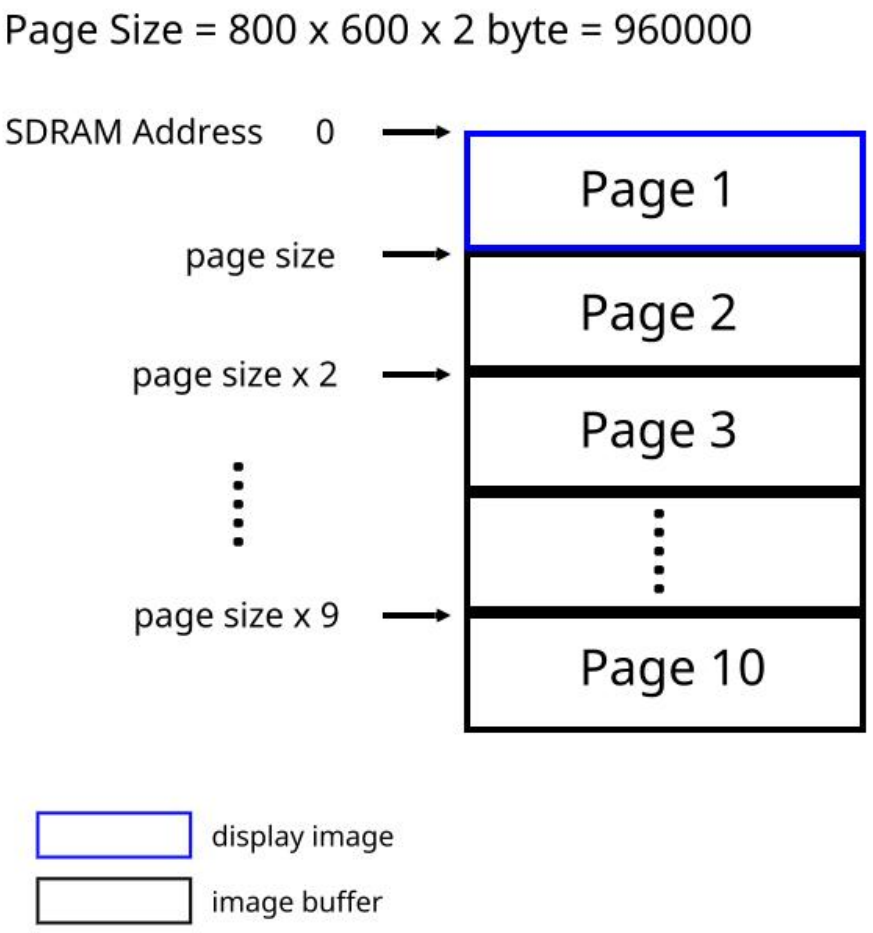
Function prototype:
void displayOn(boolean on);

Parameter	Description
on	= true Display on = false Display off

Chapter 3 Memory Configuration & Window

In this document, the memory be configured to 10 pages, the first page is assigned to image display memory, the others are used for image buffer; for example, update image to image buffer page 2, and then use BTE memory copy function, copy the image data from page2 to page1 image display memory. This method can avoid to lead to the flicker effect or the overlap effect when updating the display data to image display memory directly.

Memory configuration diagram:



The related functions for Memory and Windows are shown as below:

Function	Description
displayImageStartAddress()	Set the start address of the image display memory
displayImageWidth()	Set the width of image display memory
displayWindowStartXY()	Set the display window start point of the upper left corner of the image display memory
canvasImageStartAddress()	Set the start address of the canvas image memory
canvasImageWidth()	Set the width of the canvas image memory
activeWindowXY()	Set the active window start point of the upper left corner of canvas
activeWindowWH()	Set the width and height of the active window

displayImageStartAddress()

Description:

Set the start address of the image display memory.

Function prototype:

void displayImageStartAddress(ru32 addr);

Parameter	Description
addr	Start address of image display memory

Note and example:

Image display memory is the data source of the display window, the start address is recommended to address at 0. In this document, the memory is configured to 10 pages, the first page is assigned for image display memory, the initialization setting is shown as the following:

displayImageStartAddress(PAGE1_START_ADDR);

displayImageWidth()

Description:

Set the width of image display memory.

Function prototype:

void displayImageWidth(ru16 width);

Parameter	Description
width	Width of the image display memory

Note and example:

Width of the image display memory must be set to equal to the page (canvas) width.

Set each page (canvas) width to 800(=SCREEN_WIDTH), so initialization is set as the following:

displayImageWidth(SCREEN_WIDTH);

This function need to set one time only during the initialization.

User can also configure the page (canvas) width> SCREEN_WIDTH

For example:

//configure image display page (canvas) start point to address 0 of the memory, width is 1600,
//height is 600.

displayImageStartAddress(0)

displayImageWidth(1600);

*//start point of the memory address of the next page = 1600*600*2(byte)*

displayWindowStartXY()

Description:

Set the display window start point on the upper left corner of the image display memory.

Function prototype:

void displayWindowStartXY(ru16 x0,ru16 y0);

Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate

Note and example:

Width and height of the display window are referenced to the TFT display timing setting HDW and VDH, user only need to set display window start point of the upper left corner of the image display memory.

Setting is shown as the following:

displayWindowStartXY(0,0);

When width and height of the image display memory page (canvas) > width and height of the LCD resolution, X-axis and Y-axis coordinates can be changed, The minimum X-axis offset is multiples of 4, the minimum offset of the Y-axis is 1.

The corresponding relation between the display window and the current image display memory is like child and parent, the display window (child) is always attached to the current specified image display memory (parent).

The Contents of display window will output to the TFT-LCD display by RA8876 TFT timing controller, after setting displayOn (true).

canvasImageStartAddress()

Description:
Set the start address of the canvas image memory.

Function prototype:
void canvasImageStartAddress(ru32 addr);

Parameter	Description
addr	Start address of the canvas image memory

canvasImageWidth()

Description:
Set the width of the canvas image memory.

Function prototype:
void canvasImageWidth(ru16 width);

Parameter	Description
width	Width of the canvas image memory

Note and example:
With the operations of the Graphic, Text, Draw or DMA, all the display manipulations must be executed in the area of the active window of the current canvas, in this document, the memory is configured to 10 pages, each and all pages can be specified as the current canvas, for

example:

```
// specify the page 1 for the current canvas
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);

// specify the page 2 for the current canvas
ra8876lite.canvasImageStartAddress(PAGE2_START_ADDR);

// specify the page 3 for the current canvas
ra8876lite.canvasImageStartAddress(PAGE3_START_ADDR);
```

activeWindowXY()

Description:

Set the active window start point on the upper left corner of canvas.

Function prototype:

void activeWindowXY(ru16 x0,ru16 y0);

Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate

activeWindowWH()

Description:

Set the width and height of the active window.

Function prototype:

void activeWindowWH(ru16 width,ru16 height);

Parameter	Description
width	Width of the active window
height	height of the active window

Note and example:

With the operations of the Graphic, Text, Draw or DMA, all the display manipulations must be executed in the area of the active window of the current canvas. The corresponding relation between the active window and the current canvas is like child and parent, the active window (child) is always attached to the current canvas (parent).
Active window must be set in the current canvas area.

Chapter 4 Graphic

Function	Description
graphicMode()	Switch to graphics mode or text mode
setPixelCursor()	Set the pixel cursor coordinate
ramAccessPrepare()	Pre instruction of the memory access
putPixel_16bpp()	Draw a pixel at the specified coordinate
putPicture_16bpp()	Specify coordinate and width, height and then write image data
putPicture_16bpp()	Specify coordinate and width, height image data pointer (Byte format)
putPicture_16bpp()	Specify coordinate and width, height image data pointer (Word format)

Note:

Please refer to "*RA8876 Arduino Wire Sketch.jpg"* circuitry connection or appendix Figure A-1.

The image data is converted by using "*Image_Tool_V1.0*" image tool.

graphicMode()

Description:

Option for selecting that RA8876 is worked in the graphics mode or text mode.

Function prototype:

void graphicMode(boolean on);

Parameter	Description
on	= true Set to graphic mode = false Set to Text mode

Note:

The default value for RA8876 is stayed in graphic mode.

setPixelCursor()

Description:

Set the pixel cursor's coordinate.

Function prototype:

void setPixelCursor(ru16 x,ru16 y);

Parameter	Description
x	X-axis coordinate
y	Y-axis coordinate

ramAccessPrepare()

Description:

Pre-instruction for the memory access

Function prototype:

void ramAccessPrepare(void);

Note:

This function must be called before the memory access.

putPixel_16bpp()

Description:

Draw a pixel at the specified coordinate.

Function prototype:

void putPixel_16bpp(ru16 x,ru16 y,ru16 color);

Parameter	Description
x	X-axis coordinate
y	Y-axis coordinate
color	RGB565 data

Note and example:

```
//clean current canvas page1 specified active window to color blue
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

// draw a color red pixel dot to specified coordinate (20, 20) of the current canvas
ra8876lite.setPixelCursor(20,20);
ra8876lite.ramAccessPrepare();
ra8876lite.lcdDataWrite(0x00);//RGB565 LSB data
ra8876lite.lcdDataWrite(0xf8); //RGB565 MSB data

// draw a color white pixel dot to specified coordinate (30, 20) of the current canvas
ra8876lite.setPixelCursor(30,30);
ra8876lite.ramAccessPrepare();
ra8876lite.lcdDataWrite16bbp(COLOR65K_WHITE);//RGB565 16bpp data

// draw a color magenta pixel dot to specified coordinate (40, 30) of the current canvas
ra8876lite.putPixel_16bpp(40,40,COLOR65K_MAGENTA);
```

Screenshot of the example:

putPicture_16bpp()

Description:

Set the start coordinate of the upper left corner width and height for the intended image, after setting the relevant parameters, user is able to proceed with writing image data.

Function prototype:

void putPicture_16bpp(ru16 x,ru16 y,ru16 width, ru16 height);

Parameter	Description
x	Upper left corner X-axis coordinate
y	Upper left corner Y-axis coordinate
width	Image width(horizontal pixel size)
height	Image height(vertical pixel size)

putPicture_16bpp()

Description:

Set the coordinate, width and height of the image and the image data pointer (Byte format), after the previous settings, the function will depend on the data pointer and then starting to automatically write the image data to the specified address which is defined within the current active window of the current canvas.

Function prototype:

void putPicture_16bpp(ru16 x,ru16 y,ru16 width, ru16 height, const unsigned char *data);

Parameter	Description
x	Upper left corner X-axis coordinate
y	Upper left corner Y-axis coordinate
width	Image width(horizontal pixel size)
height	Image height(vertical pixel size)
*data	Byte format image data pointer

Note:

Image data is converted by using "*Image_Tool_V1.0*" image tool.

putPicture_16bpp()

Description:

Set the coordinate, width and height of the image and the image data pointer (Word format), after the previous settings, the function will depend on the data pointer and then starting to automatically write the image data to the specified address which is defined within the current active window of the current canvas.

Function prototype:

void putPicture_16bpp(ru16 x,ru16 y,ru16 width, ru16 height, const unsigned short *data);

Parameter	Description
x	Upper left corner X-axis coordinate
y	Upper left corner Y-axis coordinate
width	Image width(horizontal pixel size)
height	Image height(vertical pixel size)
*data	Word format image data pointer

Note:

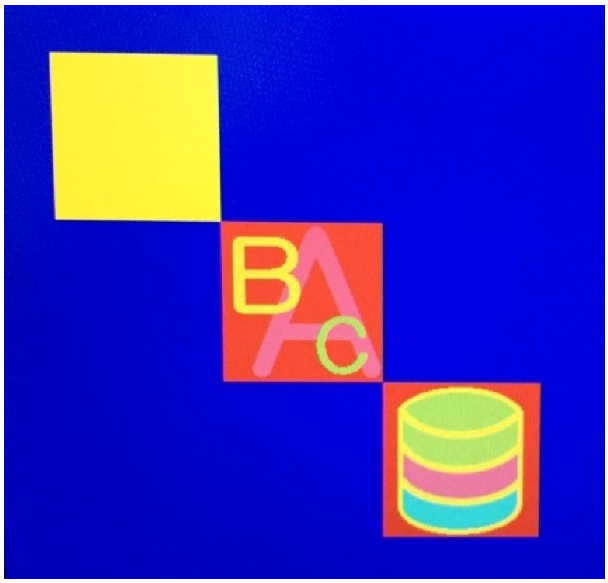
Image data is converted by using "*Image_Tool_V1.0*" image tool.

Note and example:

```
//clean current canvas page1 specify active window to color blue
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//write 128*128 image to the specified coordinate of the current canvas
ra8876lite.putPicture_16bpp(50,50,128,128);
 for(i=0;i<16384;i++)
 {
 ra8876lite.lcdDataWrite16bbp(COLOR65K_YELLOW);//RGB565 16bpp data
 }
// write 128*128 image to the specified coordinate of the current canvas
ra8876lite.putPicture_16bpp(50+128,50+128,128,128,pic16bpp_byte);
```

```
// write 128*128 image to the specified coordinate of the current canvas
ra8876lite.putPicture_16bpp(50+128+128,50+128+128,128,128,pic16bpp
  _word);
```

Screenshot of the example:



Additional functions and examples

Function	Description
lcdPutChar8x12()	Draw 8x12 ASCII character
lcdPutString8x12()	Draw 8x12 ASCII string
lcdPutChar16x24()	Draw 16x24 ASCII character
lcdPutString16x24()	Draw 16x24 ASCII string
lcdPutChar32x48()	Draw 32x48 ASCII character
lcdPutString32x48()	Draw 32x48 ASCII string

Note:

Please refer to the file "*RA8876_Lite_Graphic.ino"* for getting the relevant information of the above functions. If user needs the functions for their display requirement, please migrate the needed functions to their own firmware project.

lcdPutChar8x12()

lcdPutChar16x24()

lcdPutChar32x48()

Description:

Show ASCII character at specified coordinate which is located in the current active window of the current canvas.

Function prototype:

void lcdPutChar8x12(unsigned short x,unsigned short y,unsigned short fgcolor, unsigned short bgcolor, boolean bg_transparent, unsigned char code)

void lcdPutChar16x24(unsigned short x, unsigned short y, unsigned short fgcolor, unsigned short bgcolor, boolean bg_transparent, unsigned char code);

void lcdPutChar32x48(unsigned short x, unsigned short y, unsigned short fgcolor, unsigned short bgcolor, boolean bg_transparent, unsigned char code);

Parameter	Description
x	Upper left corner X-axis coordinate
y	Upper left corner Y-axis coordinate
fgcolor	Text foreground color
bgcolor	Text background color
bg_transparent	= ture : select background transparent, =false : select background color
code	ASCII code

lcdPutString8x12()
lcdPutString16x24()
lcdPutString32x48()

Description:

Show ASCII string at specified coordinate which is located in the current active window of the current canvas.

Function prototype:

void lcdPutString8x12(unsigned short x, unsigned short y, unsigned short fgcolor, unsigned short bgcolor, boolean bg_transparent, char *ptr)

void lcdPutString16x24(unsigned short x, unsigned short y, unsigned short fgcolor, unsigned

short bgcolor, boolean bg_transparent, char *ptr)

void lcdPutString32x48(unsigned short x, unsigned short y, unsigned short fgcolor, unsigned short bgcolor, boolean bg_transparent, char *ptr)

Parameter	Description
x	Upper left corner X-axis coordinate
y	Upper left corner Y-axis coordinate
fgcolor	Text foreground color
bgcolor	Text background color
bg_transparent	= ture : select background transparent , =false : select background color
*ptr	String or data pointer

Note and example:

```
//clean current canvas page1 specified active window to color blue
  ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
  ra8876lite.canvasImageWidth(SCREEN_WIDTH);
  ra8876lite.activeWindowXY(0,0);
  ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
  ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

// draw 8*12 ASCII character to specified coordinate in the active window of the current canvas.
#ifdef DEMO_ASCII_8X12
  lcdPutString8x12(0,0,0xFFFF,0x0000,true," !\"#$%&'()*+,-./012345678");
  lcdPutString8x12(0,12,0xFFFF,0x0000,true,"9:;<=>?@ABCDEFGHIJKLMNOPQ");
  lcdPutString8x12(0,24,0xFFFF,0x0000,true,"RSTUVWXYZ[\\]^_`abcdefghij");
  lcdPutString8x12(0,36,0xFFFF,0x0000,true,"klmnopqrstuvwxyz{|}~");
#endif

// draw 16*24 ASCII character to specified coordinate in the active window of the current
//canvas.
#ifdef DEMO_ASCII_16X24
    lcdPutString16x24(0,48,0xFFFF,0x0000,true," !\"#$%&'()*+,-./012345678");
    lcdPutString16x24(0,72,0xFFFF,0x0000,true,"9:;<=>?@ABCDEFGHIJKLMNOPQ");
```

```
   lcdPutString16x24(0,96,0xFFFF,0x0000,true,"RSTUVWXYZ[\\]^_`abcdefghij");
   lcdPutString16x24(0,120,0xFFFF,0x0000,true,"klmnopqrstuvwxyz{|}~");
#endif

// draw 32*48 ASCII character to specified coordinate in the active window of the current
//canvas.

#ifdef DEMO_ASCII_32X48
   lcdPutString32x48(0,144,0xFFFF,0x0000,false," !\"#$%&'()*+,-./012345678");
   lcdPutString32x48(0,192,0xFFFF,0x0000,false,"9:;<=>?@ABCDEFGHIJKLMNOPQ");
   lcdPutString32x48(0,240,0xFFFF,0x0000,false,"RSTUVWXYZ[\\]^_`abcdefghij");

  lcdPutString32x48(0,288,0xFFFF,0x0000,false,"klmnopqrstuvwxyz{|}~");
#endif
```

Screenshot of the example:

Chapter 5 Text and Value

Function	Description
textMode()	Switch to text mode or graphic mode
textColor()	Set the text foreground color and background color
setTextCursor()	Set the text cursor coordinate
setTextParameter1()	Set the text function parameter1
setTextParameter2()	Set the text function parameter2
genitopCharacterRomParameter()	Set the Genitop font function parameter
putString()	Write string to specified coordinate
putDec()	Write decimal value to specified coordinate
putFloat()	Write floating value to specified coordinate
putHex()	Write hexadecimal value to specified coordinate

Note:

Please refer to" *RA8876 Arduino Wire Sketch.jpg*" for the circuitry connection or please refer to the appendix Figure A-1

textMode()

Description:

Option for selecting that RA8876 is worked in the graphics mode or text mode.

Function prototype:

void textMode (boolean on);

Parameter	Description
on	= true Set to text mode = false Set to graphic mode

Note:

It is recommended that set the operating mode of RA8876 back to the graphic mode after each time user finished the text mode operation in text mode.

textColor()

Description:

Set the foreground color and the background color for text.

Function prototype:

void textColor(ru16 foreground_color, ru16 background_color);

Parameter	Description
foreground_color	Color for text foreground
background_color	Color for text background

setTextCursor()

Description:

Set the coordinate for text cursor .

Function prototype:

void setTextCursor(ru16 x, ru16 y);

Parameter	Description
x	X-axis coordinate
y	Y-axis coordinate

setTextParameter1()

Description:

Set the text function's parameter1.

Function prototype:

void setTextParameter1(ru8 source_select, ru8 size_select, ru8 iso_select);

Parameter	Description
source_select	RA8876_SELECT_INTERNAL_CGROM RA8876_SELECT_EXTERNAL_CGROM RA8876_SELECT_USER_DEFINED

size_select	RA8876_CHAR_HEIGHT_16 RA8876_CHAR_HEIGHT_24 RA8876_CHAR_HEIGHT_32
iso_select	RA8876_SELECT_8859_1 RA8876_SELECT_8859_2 RA8876_SELECT_8859_4 RA8876_SELECT_8859_5

setTextParameter2()

Description:

Set the text function's parameter2.

Function prototype:

void setTextParameter2(ru8 align, ru8 chroma_key, ru8 width_enlarge, ru8 height_enlarge);

Parameter	Description
align	RA8876_TEXT_FULL_ALIGN_DISABLE RA8876_TEXT_FULL_ALIGN_ENABLE Full-width font aligment enable bit
chroma_key	RA8876_TEXT_CHROMA_KEY_DISABLE RA8876_TEXT_CHROMA_KEY_ENABLE Text background color transparent enable bit
width_enlarge	RA8876_TEXT_WIDTH_ENLARGEMENT_X1 RA8876_TEXT_WIDTH_ENLARGEMENT_X2 RA8876_TEXT_WIDTH_ENLARGEMENT_X3 RA8876_TEXT_WIDTH_ENLARGEMENT_X4 Text horizontal enlarge select
height_enlarge	RA8876_TEXT_HEIGHT_ENLARGEMENT_X1 RA8876_TEXT_HEIGHT_ENLARGEMENT_X2 RA8876_TEXT_HEIGHT_ENLARGEMENT_X3 RA8876_TEXT_HEIGHT_ENLARGEMENT_X4 Text vertical enlarge select

genitopCharacterRomParameter()

Description:

Set the parameters for Genitop font function.

Function prototype:

void genitopCharacterRomParameter(ru8 scs_select, ru8 clk_div, ru8 rom_select, ru8 character_select, ru8 gt_width);

Parameter	Description
scs_select	RA8876_SERIAL_FLASH_SELECT0 RA8876_SERIAL_FLASH_SELECT1 Select use SPI0 or SPI1
clk_div	RA8876_SPI_DIV2 RA8876_SPI_DIV4 RA8876_SPI_DIV6 RA8876_SPI_DIV8 RA8876_SPI_DIV10 Set Genitop font SPI clock divider
rom_select	RA8876_GT21L16T1W RA8876_GT30L16U2W RA8876_GT30L24T3Y RA8876_GT30L24M1Z RA8876_GT30L32S4W RA8876_GT20L24F6Y RA8876_GT21L24S1W Select Genitop font
character_select	RA8876_GB2312 RA8876_GB12345_GB18030 RA8876_BIG5 RA8876_ASCII RA8876_UNICODE RA8876_UNI_JAPANESE RA8876_JIS0208 RA8876_LATIN_GREEK_CYRILLIC_ARABIC_THAI_HEBREW RA8876_ISO_8859_1_AND_ASCII RA8876_ISO_8859_2_AND_ASCII RA8876_ISO_8859_3_AND_ASCII

	RA8876_ISO_8859_4_AND_ASCII RA8876_ISO_8859_5_AND_ASCII RA8876_ISO_8859_7_AND_ASCII RA8876_ISO_8859_8_AND_ASCII RA8876_ISO_8859_9_AND_ASCII RA8876_ISO_8859_10_AND_ASCII RA8876_ISO_8859_11_AND_ASCII RA8876_ISO_8859_13_AND_ASCII RA8876_ISO_8859_14_AND_ASCII RA8876_ISO_8859_15_AND_ASCII RA8876_ISO_8859_16_AND_ASCII Select font decoder
gt_width	RA8876_GT_FIXED_WIDTH RA8876_GT_VARIABLE_WIDTH_ARIAL RA8876_GT_VARIABLE_FIXED_WIDTH_ROMAN RA8876_GT_BOLD Select font

Note:

RA8876 provides 2 SPI master interfaces are the IF0 and the IF1. It is recommended to use the IF0 for the GENITOP's font ROM, and use the IF1 for the serial flash memory, please refer to the datasheet of RA8876 for the detailed information.

putString()

Description:

Write a string to specified coordinate within the current active window of the current canvas.

Function prototype:

void putString(ru16 x0, ru16 y0, char *str);

Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate
*str	String or data pointer

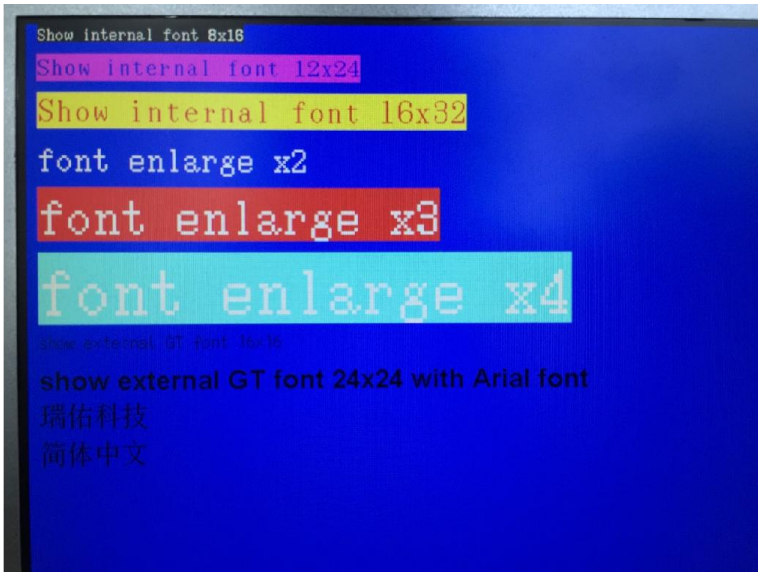
Example:

```
//clean current canvas page1 specified active window to color blue
 ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
 ra8876lite.canvasImageWidth(SCREEN_WIDTH);
 ra8876lite.activeWindowXY(0,0);
 ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
 ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

 //set the text function parameter
 //set the text color
 //write build-in font 8x16 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
  HT_16,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
 ra8876lite.putString(10,0,"Show internal font 8x16");

// set the text function parameter
// set the text color
// write build-in font 12x24 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
  HT_24,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8876lite.textColor(COLOR65K_BLUE,COLOR65K_MAGENTA);
 ra8876lite.putString(10,26,"Show internal font 12x24");

// set the text function parameter
// set the text color
// write build-in font 16x32 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
  HT_32,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
```

```
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8876lite.textColor(COLOR65K_RED,COLOR65K_YELLOW);
 ra8876lite.putString(10,60,"Show internal font 16x32");

// set the text function parameter
// set the text color
// write build-in font and enlarge 2 times to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_16,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X2,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X2);
 ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_RED);
 ra8876lite.putString(10,102,"font enlarge x2");

// set the text function parameter
// set the text color
// write build-in font and enlarge 3 times to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_16,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X3,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X3);
 ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_RED);
 ra8876lite.putString(10,144,"font enlarge x3");

// set the text function parameter
// set the text color
// write build-in font and enlarge 4 times to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
 HT_16,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X4,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X4);
 ra8876lite.textColor(COLOR65K_WHITE, COLOR65K_LIGHTCYAN);
 ra8876lite.putString(10,202,"font enlarge x4");
```

```
// set the text function parameter
// set the Genitop font function parameter
// set the text color
// write string of the Genitop font to specified coordinate
 ra8876lite.setTextParameter1(RA8876_SELECT_EXTERNAL_CGROM,RA8876_CHAR_HEI
GHT_16,RA8876_SELECT_8859_1);//cch
 ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X1);

ra8876lite.genitopCharacterRomParameter(RA8876_SERIAL_FLASH_SELECT0,RA8876_SP
I_DIV4,RA8876_GT30L24T3Y,RA8876_BIG5,RA8876_GT_FIXED_WIDTH);
ra8876lite.textColor(COLOR65K_BLACK,COLOR65K_RED);
ra8876lite.putString(10,276,"show external GT font 16x16");

// set the text function parameter
// set the Genitop font function parameter
// set the text color
// write string of the Genitop font to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_EXTERNAL_CGROM,RA8876_CHAR_HEIG
HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X1);

ra8876lite.genitopCharacterRomParameter(RA8876_SERIAL_FLASH_SELECT0,RA8876_SP
I_DIV4,RA8876_GT30L24T3Y,RA8876_BIG5,RA8876_GT_VARIABLE_WIDTH_ARIAL);
ra8876lite.putString(10,302,"show external GT font 24x24 with Arial font");

ra8876lite.putString(10,336,string1);

ra8876lite.setTextParameter1(RA8876_SELECT_EXTERNAL_CGROM,RA8876_CHAR_HEIG
HT_24,RA8876_SELECT_8859_1);//cch

ra8876lite.genitopCharacterRomParameter(RA8876_SERIAL_FLASH_SELECT0,RA8876_SP
I_DIV4,RA8876_GT30L24T3Y,RA8876_GB2312,RA8876_GT_FIXED_WIDTH);
ra8876lite.putString(10,370,string2);
```

Screenshot of the example:



putDec()

Description:

Write decimal number to specified coordinate within the current active window of the current canvas.

Function prototype:

void putDec(ru16 x0,ru16 y0,rs32 vaule,ru8 len,const char *flag);

Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate
vaule	Input value -2147483648(-2^31) ~ 2147483647(2^31-1)
len	Minimum display number of bits(1~11)
*flag	= “n” : Display to the right = “-” : Display to the left = “+” : Output sign = “0” : fill 0 at the beginning, not fill space

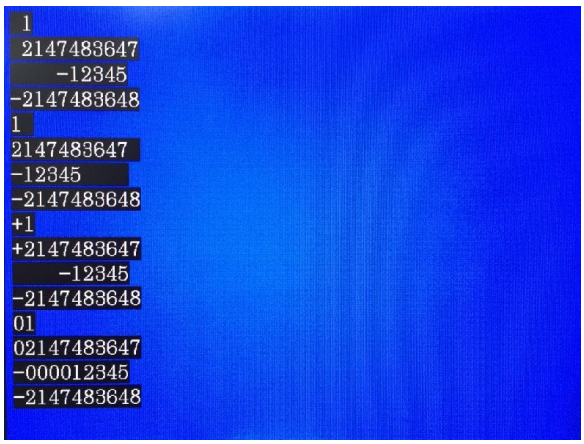
Example:

//clean current canvas page1 specified active window to color blue

ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);

```
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

// set text function parameter
// set text color
//write build-in font 16x32 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_32,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//display value
ra8876lite.putDec(10,10,1,2,"n");
ra8876lite.putDec(10,44,2147483647,11,"n");
ra8876lite.putDec(10,78,-12345,10,"n");
ra8876lite.putDec(10,112,-2147483648,11,"n");

ra8876lite.putDec(10,146,1,2,"-");
ra8876lite.putDec(10,180,2147483647,11,"-");
ra8876lite.putDec(10,214,-12345,10,"-");
ra8876lite.putDec(10,248,-2147483648,11,"-");

ra8876lite.putDec(10,282,1,2,"+");
ra8876lite.putDec(10,316,2147483647,11,"+");
ra8876lite.putDec(10,350,-12345,10,"+");
ra8876lite.putDec(10,384,-2147483648,11,"+");

ra8876lite.putDec(10,418,1,2,"0");
ra8876lite.putDec(10,452,2147483647,11,"0");
ra8876lite.putDec(10,486,-12345,10,"0");
ra8876lite.putDec(10,520,-2147483648,11,"0");
```

Screenshot of the example:



putFloat()

Description:

Write floating value to specified coordinate within the current active window of the current canvas.

Function prototype:

void putFloat (ru16 x0,ru16 y0, double vaule,ru8 len, ru8 precision,const char *flag);

Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate
vaule	Input value (3.4E-38 ~ 3.4E38)
len	Minimum display number of bits (1~11)
precision	The precise number of bits to the right of the decimal point (1~4bits)
*flag	= “n” : Display to the right = “-” : Display to the left = “+” : Output sign = “0” : fill 0 at the beginning, not fill space

Note:

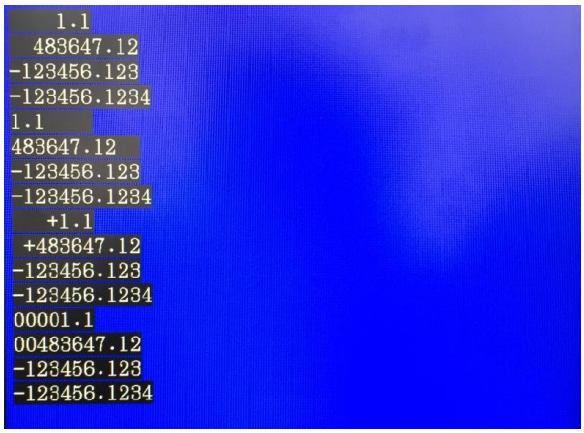
Use a double for getting more precision accuracy.

Example:

//clean current canvas page1 specified active window to color blue

```
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//set text function parameter
//set text color
//write build-in font 16x32 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_32,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//display value
ra8876lite.putFloat(10,10,1.1,7,1,"n");
ra8876lite.putFloat(10,44,483647.12,11,2,"n");
ra8876lite.putFloat(10,78,-12345.123,11,3,"n");
ra8876lite.putFloat(10,112,-123456.1234,11,4,"n");

ra8876lite.putFloat(10,146,1.1234,7,1,"-");
ra8876lite.putFloat(10,180,483647.12,11,2,"-");
ra8876lite.putFloat(10,214,-12345.123,11,3,"-");
ra8876lite.putFloat(10,248,-123456.1234,11,4,"-");

ra8876lite.putFloat(10,282,1.1,7,1,"+");
ra8876lite.putFloat(10,316,483647.12,11,2,"+");
ra8876lite.putFloat(10,350,-12345.123,11,3,"+");
ra8876lite.putFloat(10,384,-123456.1234,11,4,"+");

ra8876lite.putFloat(10,418,1.1,7,1,"0");
ra8876lite.putFloat(10,452,483647.12,11,2,"0");
ra8876lite.putFloat(10,486,-12345.123,11,3,"0");
ra8876lite.putFloat(10,520,-123456.1234,11,4,"0");
```

Screenshot of the example:



putHex()

Description:

Write hexadecimal value to specify coordinate within the current active window of the current canvas.

Function prototype:

void putHex(ru16 x0,ru16 y0,ru32 vaule,ru8 len,const char *flag);

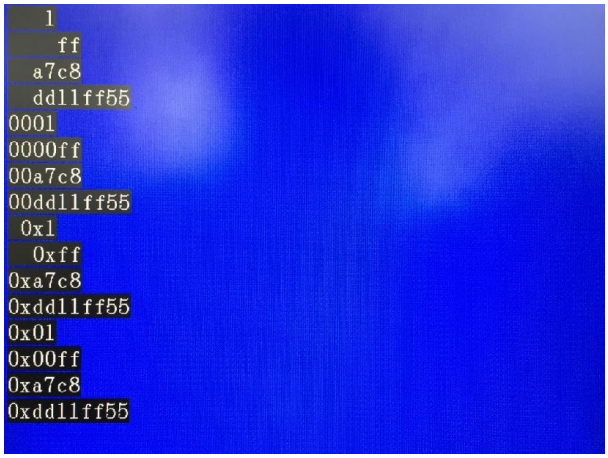
Parameter	Description
x0	Upper left corner X-axis coordinate
y0	Upper left corner Y-axis coordinate
vaule	Input value 0x00000000~0xffffffff
len	Minimum display number of bits (1~10)
*flag	= “n” : Display to the right = “#” : Force output 0x as the beginning = “0” : fill 0 at the beginning, not fill space = “x” : Force output 0x as the beginning，fill 0

Example:

```
//clean current canvas page1 specified active window to color blue
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);
```

```
// set text function parameter
// set text color
//write build-in font 16x32 ASCII string to specified coordinate
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_32,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_DISABLE,
RA8876_TEXT_CHROMA_KEY_DISABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
8876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//display value
ra8876lite.putHex(10,10,1,4,"n");
ra8876lite.putHex(10,44,255,6,"n");
ra8876lite.putHex(10,78,0xa7c8,6,"n");
ra8876lite.putHex(10,112,0xdd11ff55,10,"n");

ra8876lite.putHex(10,146,1,4,"0");
ra8876lite.putHex(10,180,255,6,"0");
ra8876lite.putHex(10,214,0xa7c8,6,"0");
ra8876lite.putHex(10,248,0xdd11ff55,10,"0");

ra8876lite.putHex(10,282,1,4,"#");
ra8876lite.putHex(10,316,255,6,"#");
ra8876lite.putHex(10,350,0xa7c8,6,"#");
ra8876lite.putHex(10,384,0xdd11ff55,10,"#");

ra8876lite.putHex(10,418,1,4,"x");
ra8876lite.putHex(10,452,255,6,"x");
ra8876lite.putHex(10,486,0xa7c8,6,"x");
ra8876lite.putHex(10,520,0xdd11ff55,10,"x");
```

Screenshot of the example:


1
ff
a7c8
dd11ff55
0001
0000ff
00a7c8
00dd11ff55
0x1
0xff
0xa7c8
0xdd11ff55
0x01
0x00ff
0xa7c8
0xdd11ff55

Chapter 6 Geometric Draw

Function	Description
drawLine()	Draw a line
drawSquare()	Draw a square
drawSquareFill()	Draw a square fill
drawCircleSquare()	Draw a circle square
drawCircleSquareFill()	Draw a circle square fill
drawTriangle()	Draw a triangle
drawTriangleFill()	Draw a triangle fill
drawCircle()	Draw a circle
drawCircleFill()	Draw a circle fill
drawEllipse()	Draw a ellipse
drawEllipseFill()	Draw a ellipse fill

Note:

Please refer to" *RA8876 Arduino Wire Sketch.jpg*" for the circuitry connection or please refer to the appendix Figure A-1

drawLine()

Description:

Specify any two points to draw a color line in the active window of the current canvas.

Function prototype:

void drawLine(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
color	Set color(RGB565)

Example:

```
ra8876lite.drawLine(40,40,159,159,COLOR65K_RED);
ra8876lite.drawLine(40,159,159,40,COLOR65K_LIGHTRED);
```

Screenshot of the example:

drawSquare()

Description:

Specify any two points to draw a color square in the active window of the current canvas.

Function prototype:

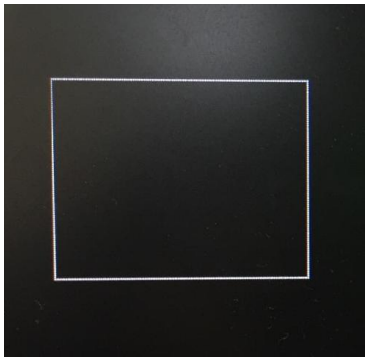
void drawSquare(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
color	Set color(RGB565)

Example:

ra8876lite.drawSquare(200+30, 50, 399-30, 199-50, COLOR65K_GRAYSCALE23);

Screenshot of the example:



drawSquareFill()

Description:

Specify any two points to draw a color square fill in the active window of the current canvas.

Function prototype:

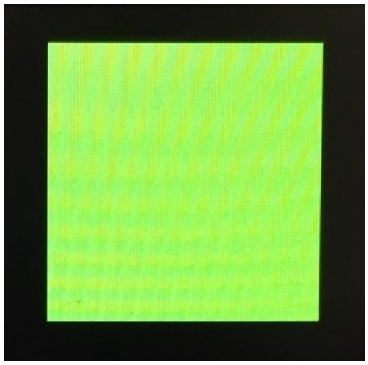
void drawSquareFill(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
color	Set color(RGB565)

Example:

ra8876lite.drawSquareFill(420, 20, 579, 179, COLOR65K_GREEN);

Screenshot of the example:



drawCircleSquare()

Description:

Specify any two points to draw a color circle square in the active window of the current canvas.

Function prototype:

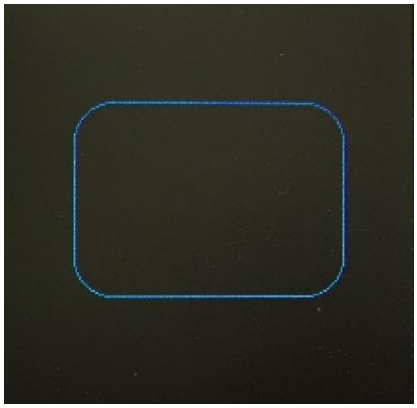
void drawCircleSquare(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 xr, ru16 yr, ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
xr	Horizontal radius of the rounded corner
yr	Vertical radius of the rounded corner
color	Set color(RGB565)

Example:

ra8876lite.drawCircleSquare(600+30,0+50, 799-30, 199-50, 20, 20, COLOR65K_BLUE2);

Screenshot of the example:



drawCircleSquareFill()

Description:

Specify any two points to draw a color circle square fill in the active window of the current

canvas.

Function prototype:

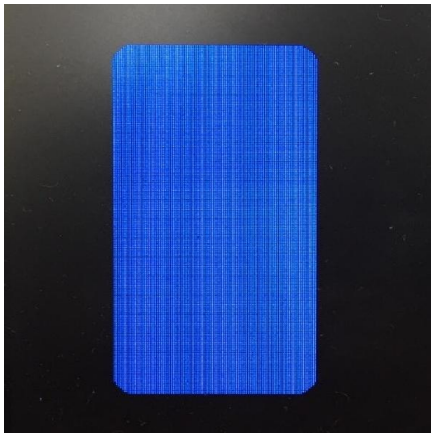
void drawCircleSquareFill(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 xr, ru16 yr, ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
xr	Horizontal radius of the rounded corner
yr	Vertical radius of the rounded corner
color	Set color(RGB565)

Example:

ra8876lite.drawCircleSquareFill(50,200, 149, 399, 10, 10, COLOR65K_BLUE);

Screenshot of the example:



drawTriangle()

Description:

Specify any three points to draw a color triangle in the active window of the current canvas.

Function prototype:

void drawTriangle(ru16 x0,ru16 y0,ru16 x1,ru16 y1,ru16 x2,ru16 y2,ru16 color);

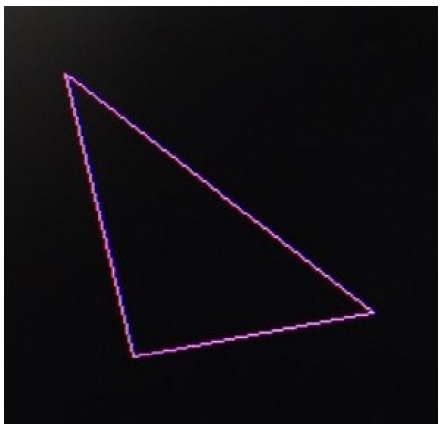
Parameter	Description
x0	X-axis coordinate of point 1

y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
x2	X-axis coordinate of point 3
y2	Y-axis coordinate of point 3
color	Set color(RGB565)

Example:

ra8876lite.drawTriangle(220,250,360,360,250,380,COLOR65K_MAGENTA);

Screenshot of the example:



drawTriangleFill()

Description:

Specify any three points to draw a color triangle fill in the active window of the current canvas.

Function prototype:

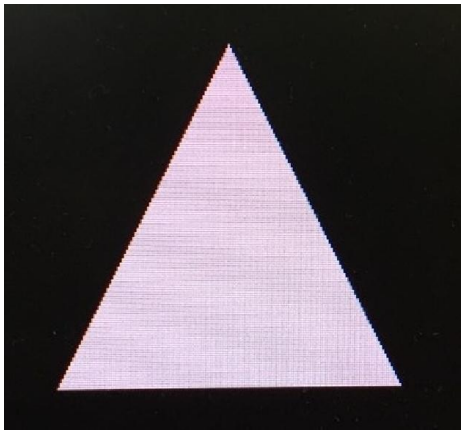
void drawTriangleFill(ru16 x0,ru16 y0,ru16 x1,ru16 y1,ru16 x2,ru16 y2,ru16 color);

Parameter	Description
x0	X-axis coordinate of point 1
y0	Y-axis coordinate of point 1
x1	X-axis coordinate of point 2
y1	Y-axis coordinate of point 2
x2	X-axis coordinate of point 3
y2	Y-axis coordinate of point 3
color	Set color(RGB565)

Example:

ra8876lite.drawTriangleFill(500,220,580,380,420,380,COLOR65K_LIGHTMAGENTA);

Screenshot of the example:



drawCircle()

Description:

Specify any points as a center and define the radius for drawing a color circle in the active window of the current canvas.

Function prototype:

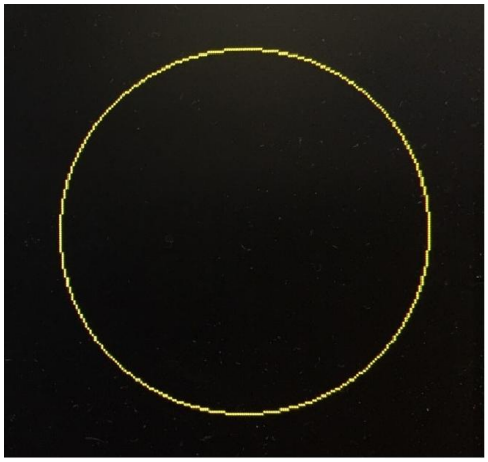
void drawCircle(ru16 x0,ru16 y0,ru16 r,ru16 color);

Parameter	Description
x0	X-axis coordinate of the center
y0	Y-axis coordinate of the center
r	Radius
color	Set color(RGB565)

Example:

ra8876lite.drawCircle(700,300,80,COLOR65K_YELLOW);

Screenshot of the example:



drawCircleFill()

Description:

Specify any points as a center and define the radius for drawing a color filled circle in the active window of the current canvas.

Function prototype:

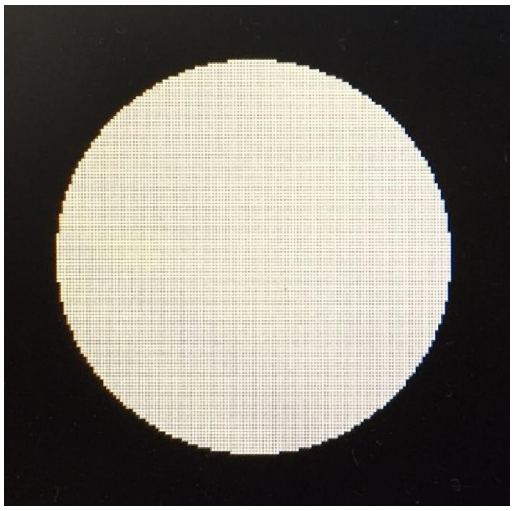
void drawCircleFill(ru16 x0,ru16 y0,ru16 r,ru16 color);

Parameter	Description
x0	X-axis coordinate of the center
y0	Y-axis coordinate of the center
r	Radius
color	Set color(RGB565)

Example:

ra8876lite.drawCircleFill(100,500,60,COLOR65K_LIGHTYELLOW);

Screenshot of the example:



drawEllipse()

Description:

Specify any points as a center and define the horizontal radius and the vertical radius for drawing a color ellipse in the active window of the current canvas.

Function prototype:

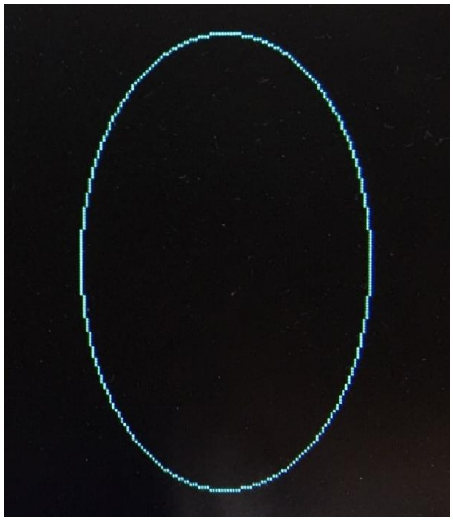
void drawEllipse(ru16 x0,ru16 y0,ru16 xr,ru16 yr,ru16 color);

Parameter	Description
x0	X-axis coordinate of the center
y0	Y-axis coordinate of the center
xr	Horizontal radius
yr	Vertical radius
color	Set color(RGB565)

Example:

ra8876lite.drawEllipse(300,500,50,80,COLOR65K_CYAN);

Screenshot of the example:



drawEllipseFill()

Description:

Specify any points as a center and define the radius for drawing a color filled ellipse in the active window of the current canvas.

Function prototype:

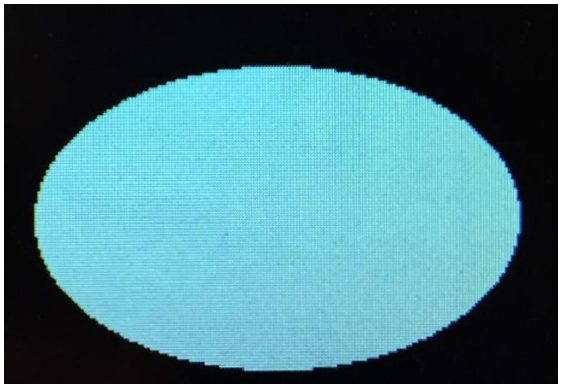
void drawEllipseFill(ru16 x0,ru16 y0,ru16 xr,ru16 yr,ru16 color);

Parameter	Description
x0	X-axis coordinate of the center
y0	Y-axis coordinate of the center
xr	Horizontal radius
yr	Vertical radius
color	Set color(RGB565)

Example:

ra8876lite.drawEllipseFill(500,500,80,50,COLOR65K_LIGHTCYAN);

Screenshot of the example:



Chapter 7 BTE

Block Transfer Engine is a 2D acceleration engine, provides fast memory data transfer with copy and logic operation, chroma key color data ignored, monochrome (1bpp) data convert to color data with color expansion and color expansion with chroma key color, pattern image fill and fill with chroma key color.

Amount of data for color display is huge, if the operation speed of MPU write is not fast enough, you can see the update scan line on the display is shown like a waterfall. Or in the other operation, you need dynamic effects to the display, such as a background image static (such as wallpaper), and the foreground text or image is changed. For this effect in the regular coding case, programmer must re-write the background data and then refresh the foreground text or image data, if we directly execute the current contents of the display memory, it will leaded to the screen flicker cause by updating the background data. If you update foreground text or image data directly without re-write the background data, will result the image overlay, so if you want to get a better display effect, you can take advantage of the BTE function, the image data can be written to the non-display area of the display memory through the MPU interface or DMA interface firstly, and then use BTE memory copy function to duplicate and move the image data to the display memory area, to avoids the bad display effect which is described above.

Color expansion function can convert monochrome data like 0 or 1 to the specified color data, due to the MPU's ROM is limited, typically is under 512Kbyte, if we convert the image data from 16bpp to 1bpp format and store the converted image data into the MPU's ROM, therefore we can reduce the ROM usage of MPU/MCU. For example, users may need 64 * 128 resolution numeric digits 0-9 for display; they can convert the numeric image data to 1bpp data format, and store them in the MPU ROM. If we want to show a color and customized members on the display, use BTE color expansion function, the BTE function will automatically take the image data from the MPU/MCU's ROM, convert the monochrome image data to specified color image data, and write the color image data into the memory of RA8876.

Pattern fill function allows user to use a color image (16bpp) in size 8*8 or 16*16 to fill a specified block.

The detailed information for all of BTE functions, please refer to the description in the following sections, or refer to the datasheet.

Function	Description

bteMemoryCopy()	Memory data copy and move
bteMemoryCopyWithROP()	Memory data copy and move with logic operation
bteMemoryCopyWithChromakey()	Memory data copy and move with chroma key color ignore
bteMpuWriteWithROP()	MPU write data with logic operation(included data pointer ,Byte format)
bteMpuWriteWithROP()	MPU write data with logic operation(included data pointer, Word format)
bteMpuWriteWithROP()	MPU write data with logic operation
bteMpuWriteWithChromaKey()	MPU write data with chroma key color ignore(included data pointer ,Byte format)
bteMpuWriteWithChromaKey()	MPU write data with chroma key color ignore(included data pointer, Word format)
bteMpuWriteWithChromaKey()	MPU write data with chroma key color ignor
bteMpuWriteColorExpansion()	MPU write data with color expansion(included data pointer)
bteMpuWriteColorExpansion()	MPU write data with color expansion
bteMpuWriteColorExpansionWith ChromaKey()	MPU write data with color expansion and chroma key color ignore (included data pointer)
bteMpuWriteColorExpansionWith ChromaKey()	MPU write data with color expansion and chroma key color ignore
btePatternFill()	Pattern image fill
btePatternFillWithChromaKey()	Pattern image fill with chroma key color ignore

Note:

Please refer to" *RA8876 Arduino Wire Sketch.jpg*" for the circuitry connection or please refer to the appendix Figure A-1

bteMemoryCopy()

Description:

Perform memory data copy means that duplicate the memory data from the specified memory source to the specified memory destination, the memory data moving range is specified within the current canvas or is specified between two canvases.

Function prototype:

void bteMemoryCopy(ru32 s0_addr, ru16 s0_image_width, ru16 s0_x, ru16 s0_y, ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 copy_width, ru16

copy_height);

Parameter	Description
s0_addr	Start address memory of the source 0 canvas
s0_image_width	Width of the image memory of the source 0 canvas
s0_x	Source 0 image X-axis coordinate of the canvas
s0_y	Source 0 image Y-axis coordinate of the canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
copy_width	Image width for copy
copy_height	Image height for copy

Note:

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Pic16bpp_word.bmp

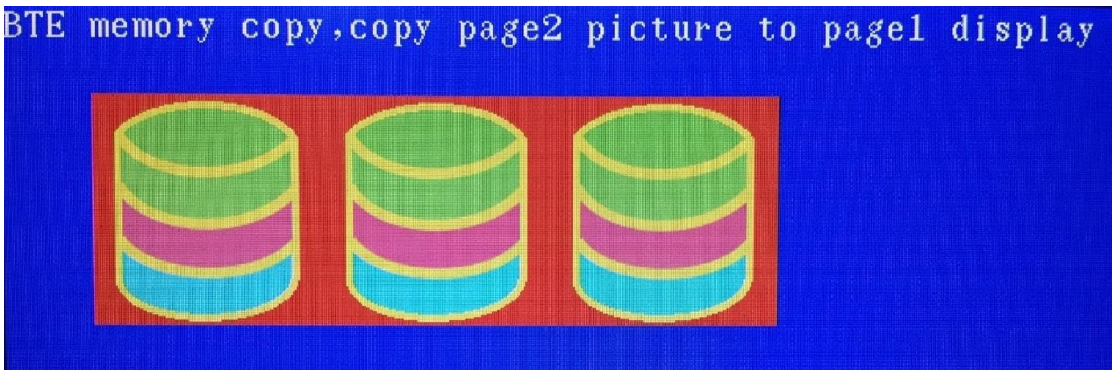
Before performing the following example, we will need an image data source, so user should prepare a converted 16bpp image data file (such as pic16bpp_word.h) and then include the relevant header files to the main programming project.

Example:

```
//clean current canvas page1 specified active window to color blue
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1,
COLOR65K_BLUE);

//clean current canvas page2 specified active window to color red
```

```
ra8876lite.canvasImageStartAddress(PAGE2_START_ADDR);
ra8876lite.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_RED);

//write image data to current canvas page2 specified position
ra8876lite.putPicture_16bpp(50,50,128,128,pic16bpp_word);

//write string to current canvas page1 specified position
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.putString(0,0,"BTE memory copy,copy page2 picture to page1 display");
//copy image data from page2 canvas(source) and written to page1 canvas (destination)
ra8876lite.bteMemoryCopy(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_
ADDR,SCREEN_WIDTH, 50,50,128,128);

ra8876lite.bteMemoryCopy(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_
ADDR,SCREEN_WIDTH, (50+128),50,128,128);

ra8876lite.bteMemoryCopy(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_
ADDR,SCREEN_WIDTH, (50+128+128),50,128,128);
```

Screenshot of the example:



bteMemoryCopyWithROP()

Description:

Perform the memory data copy with ROP function means that duplicate the memory data from specified memory source to the specified memory destination with the ROP logic operation, the memory moving range is specified within the current canvas or is specified between two canvases.

Function prototype:

void bteMemoryCopy WithROP (ru32 s0_addr, ru16 s0_image_width, ru16 s0_x, ru16 s0_y, ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 copy_width, ru16 copy_height, ru8 rop_code);

Parameter	Description
s0_addr	Start address of the memory of the source 0 canvas
s0_image_width	Width of the image memory of the source 0 canvas
s0_x	Source 0 image X-axis coordinate of the canvas
s0_y	Source 0 image Y-axis coordinate of the canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
copy_width	Image width for copy
copy_height	Image height for copy
rop_code	Select of the logic operation RA8876_BTE_ROP_CODE_0 (Blackness) RA8876_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8876_BTE_ROP_CODE_2 ~S0・S1 RA8876_BTE_ROP_CODE_3 ~S0 RA8876_BTE_ROP_CODE_4 S0・~S1 RA8876_BTE_ROP_CODE_5 ~S1

	RA8876_BTE_ROP_CODE_6 S0^S1 RA8876_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8876_BTE_ROP_CODE_8 S0・S1 RA8876_BTE_ROP_CODE_9 ~ (S0^S1) RA8876_BTE_ROP_CODE_10 S1 RA8876_BTE_ROP_CODE_11 ~S0+S1 RA8876_BTE_ROP_CODE_12 S0 RA8876_BTE_ROP_CODE_13 S0+~S1 RA8876_BTE_ROP_CODE_14 S0+S1 RA8876_BTE_ROP_CODE_15 (Whiteness)

Note:

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Pic16bpp_word.bmp

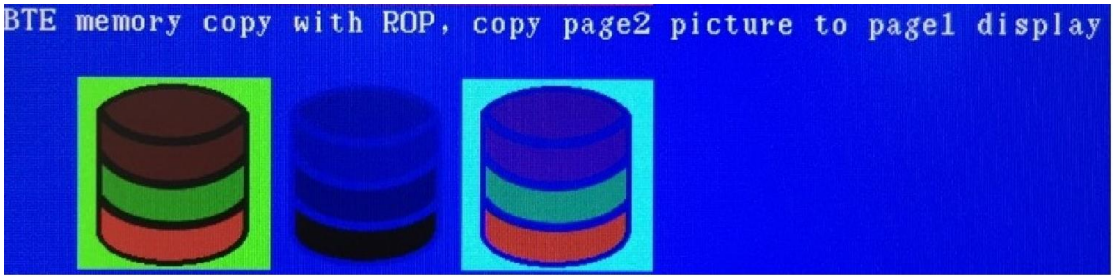
Before performing the following example, we will need an image data source, so user should prepare a converted 16bpp image data file (such as pic16bpp_word.h) and then include the relevant header files to the main programming project.

Example:

//write string to current canvas page1 specified position

```
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.putString(0,178,"BTE memory copy with ROP, copy page2 picture to page1 display");
//copy image data from page2 canvas(source) and logic operation with page1
//canvas(destination) and then written to page1 canvas (destination)
ra8876lite.bteMemoryCopyWithROP(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1
_START_ADDR,SCREEN_WIDTH,50,228,
PAGE1_START_ADDR,SCREEN_WIDTH,50,228,128,128,RA8876_BTE_ROP_CODE_1);

ra8876lite.bteMemoryCopyWithROP(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1
_START_ADDR,SCREEN_WIDTH,(50+128),228,PAGE1_START_ADDR,SCREEN_WIDTH,(5
0+128),228,128,128,RA8876_BTE_ROP_CODE_2);

ra8876lite.bteMemoryCopyWithROP(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1
_START_ADDR,SCREEN_WIDTH,(50+128+128),228,PAGE1_START_ADDR,SCREEN_WID
TH,(50+128+128),228,128,128,RA8876_BTE_ROP_CODE_3);
```

Screenshot of the example:



bteMemoryCopyWithChromaKey()

Description:

Perform the memory data copy with chroma key function, the chroma key means that RA8876 will ignore the indicated background data and the memory data copy function will move the front ground display data from the specified memory source to the specified memory destination. The memory moving range is specified within the current canvas or is specified between the two canvases.

Function prototype:

void bteMemoryCopyWithChromaKey(ru32 s0_addr, ru16 s0_image_width, ru16 s0_x, ru16 s0_y, ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 copy_width, ru16

copy_height, ru16 chromakey_color);

Parameter	Description
s0_addr	Start address of the memory of the source 0 canvas
s0_image_width	Width of the image memory of the source 0 canvas
s0_x	Source 0 image X-axis coordinate of the canvas
s0_y	Source 0 image Y-axis coordinate of the canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
copy_width	Image width for copy
copy_height	Image height for copy
chromakey_color	Data of chroma key color

Note:

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Pic16bpp_word.bmp

Before performing the following example, we will need an image data source, so user should prepare a converted 16bpp image data file (such as pic16bpp_word.h) and then include the relevant header files to the main programming project.

Example:

//write string to current canvas page1 specified position

ra8876lite.putString(0,356,"BTE memory copy with ChromaKey, copy page2 picture to page1 display");

//copy image data from page2 canvas(source) and then written to page1 canvas (destination)

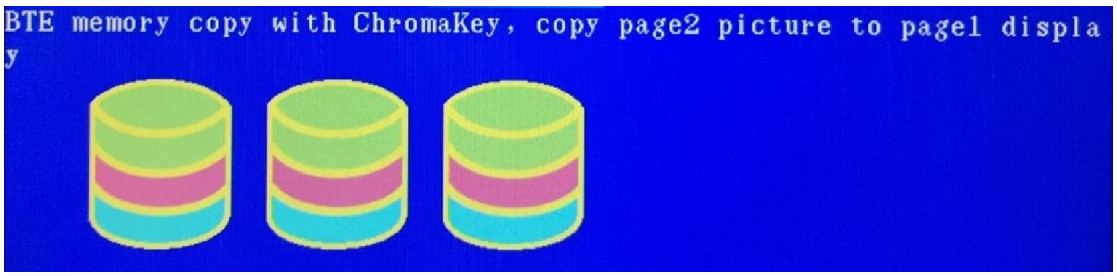
//with chroma key color ignore.

ra8876lite.bteMemoryCopyWithChromakey(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_ADDR,SCREEN_WIDTH,50,406,128,128,0xf800);

ra8876lite.bteMemoryCopyWithChromakey(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_ADDR,SCREEN_WIDTH,50+128,406,128,128,0xf800);

ra8876lite.bteMemoryCopyWithChromakey(PAGE2_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,406,128,128,0xf800);

Screenshot of the example:



bteMpuWriteWithROP()

Description:

For this function, the image data written by MCU will be regard as the source 0, these data will be performed the logic operation with the source1 image data, and then the results of the operations will be moved into the specified memory destination.

Function prototype:

void bteMpuWriteWithROP(ru32 s1_addr,ru16 s1_image_width,ru16 s1_x,ru16 s1_y,ru32 des_addr,ru16 des_image_width,ru16 des_x,ru16 des_y,ru16 width,ru16 height,ru8 rop_code,const unsigned char *data);

void bteMpuWriteWithROP(ru32 s1_addr,ru16 s1_image_width,ru16 s1_x,ru16 s1_y,ru32 des_addr,ru16 des_image_width,ru16 des_x,ru16 des_y,ru16 width,ru16 height,ru8 rop_code,const unsigned short *data);

void bteMpuWriteWithROP(ru32 s1_addr,ru16 s1_image_width,ru16 s1_x,ru16 s1_y,ru32 des_addr,ru16 des_image_width,ru16 des_x,ru16 des_y,ru16 width,ru16 height,ru8 rop_code);

Parameter	Description
s1_addr	Start address of the memory of the source 1 canvas
s1_image_width	Width of the image memory of the source 1 canvas
s1_x	Source 1 image X-axis coordinate of the canvas
s1_y	Source 1 image Y-axis coordinate of the canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
rop_code	Select of the logic operation RA8876_BTE_ROP_CODE_0 (Blackness) RA8876_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8876_BTE_ROP_CODE_2 ~S0・S1 RA8876_BTE_ROP_CODE_3 ~S0 RA8876_BTE_ROP_CODE_4 S0・~S1 RA8876_BTE_ROP_CODE_5 ~S1 RA8876_BTE_ROP_CODE_6 S0^S1 RA8876_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8876_BTE_ROP_CODE_8 S0・S1 RA8876_BTE_ROP_CODE_9 ~ (S0^S1) RA8876_BTE_ROP_CODE_10 S1 RA8876_BTE_ROP_CODE_11 ~S0+S1 RA8876_BTE_ROP_CODE_12

	S0 RA8876_BTE_ROP_CODE_13 S0+~S1 RA8876_BTE_ROP_CODE_14 S0+S1 RA8876_BTE_ROP_CODE_15 (Whiteness)
*data	Data pointer (Byte or Word format)

Note:

Function of BTE with MPU data write related, S0(Source0) = MPU data write.

S1 (Source1) can be set the same with Des (destination).

User can continuously write the image data after calling the function which without pointer.

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Pic16bpp_byte.bmp

Pic16bpp_word.bmp

Before performing the following example, we will need an image data source, so user should prepare the converted 16bpp image data files (such as pic16bpp_byte.h and pic16bpp_word.h) and then include the relevant header files to the main programming project.

Example:

//clean current canvas page1 specified active window to color blue

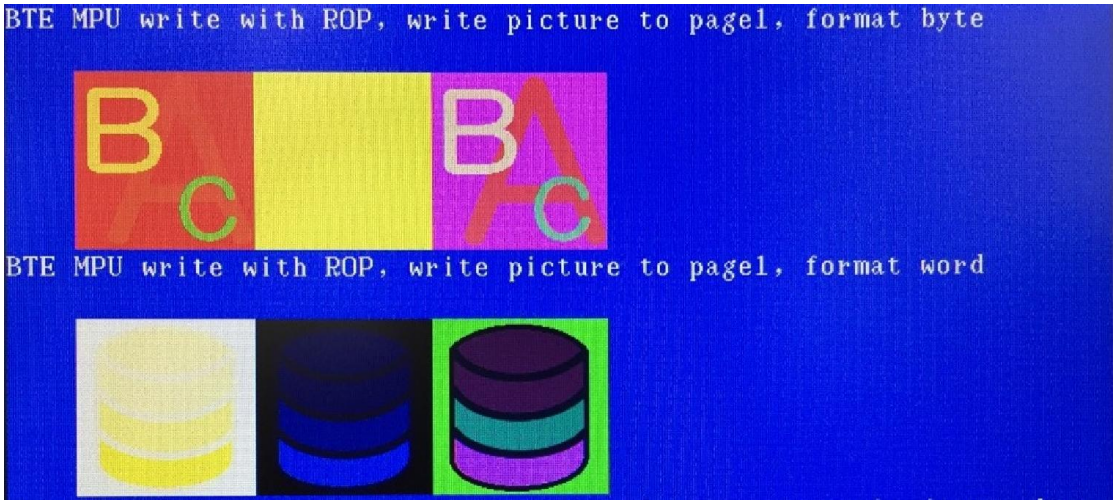
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);

ra8876lite.canvasImageWidth(SCREEN_WIDTH);

```
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//write string to current canvas page1 specified position
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.putString(0,0,"BTE MPU write with ROP, write picture to page1, format byte");

//MPU(Source0) written data to destination canvas(Destination) through BTE engine after
//execute logic operation with specified block of canvas(Source1).
ra8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50,50,PAGE1_ST
ART_ADDR,SCREEN_WIDTH,50,50,128,128,RA8876_BTE_ROP_CODE_4,pic16bpp_byte);

ra8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,50,PAGE
1_START_ADDR,SCREEN_WIDTH,50+128,50,128,128,RA8876_BTE_ROP_CODE_5,pic16b
pp_byte);

ra8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,50,P
AGE1_START_ADDR,SCREEN_WIDTH,50+128+128,50,128,128,RA8876_BTE_ROP_CODE
_6,pic16bpp_byte);

ra8876lite.putString(0,178,"BTE MPU write with ROP, write picture to page1, format word");

raA8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50,228,PAGE1_
START_ADDR,SCREEN_WIDTH,50,228,128,128,RA8876_BTE_ROP_CODE_7,pic16bpp_wo
rd);

ra8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,228,PAG
E1_START_ADDR,SCREEN_WIDTH,50+128,228,128,128,RA8876_BTE_ROP_CODE_8,pic1
6bpp_word);
```

ra8876lite.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,228,PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,228,128,128,RA8876_BTE_ROP_CODE_9,pic16bpp_word);

Screenshot of the example:



bteMpuWriteWithChromaKey()

Description:

MPU write data to the destination with the chroma key function.

Function prototype:

void bteMpuWriteWithChromaKey(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 chromakey_color, const unsigned char *data);

void bteMpuWriteWithChromaKey(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 chromakey_color, const unsigned short *data);

void bteMpuWriteWithChromaKey(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 chromakey_color);

Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas

des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
chromakey_color	Data of chroma key color
*data	Data pointer

Note:

User can continuously write the image data after calling the function which without pointer. Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Pic16bpp_byte.bmp

Pic16bpp_word.bmp

Before performing the following example, we will need an image data source, so user should prepare the converted 16bpp image data files (such as pic16bpp_byte.h and pic16bpp_word.h) and then include the relevant header files to the main programming project.

Example:

```
//write string to current canvas page1 specified position
ra8876lite.putString(0,356,"BTE MPU write with Chroma Key, write picture to page1, format byte,word");

// MPU write data to destination canvas(page1) through BTE with chroma key color ignore.
   ra8876lite.bteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,
```

50,406,128,128,0xf800,pic16bpp_byte);
ra8876lite.bteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,
50+128,406,128,128,0xf800,pic16bpp_word);

Screenshot of the example:

bteMpuWriteColorExpansion()

Description:
MPU writes 1bpp data to the specified block of destination canvas through using BTE color expansion.

Function prototype:
void bteMpuWriteColorExpansion(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 foreground_color, ru16 background_color, const unsigned char *data);

void bteMpuWriteColorExpansion(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 foreground_color, ru16 background_color);

Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
foreground_color	Foreground color
background_color	Background color
*data	Data pointer(Byte format)

Note:

User can continuously write the image data after calling the function which without pointer. Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Bw.bmp

Before performing the following example, we will need an 1bpp image data source, so user should prepare a converted 1bpp image data file (such as bw.h) and then include the relevant header file to the main programming project.

Example:

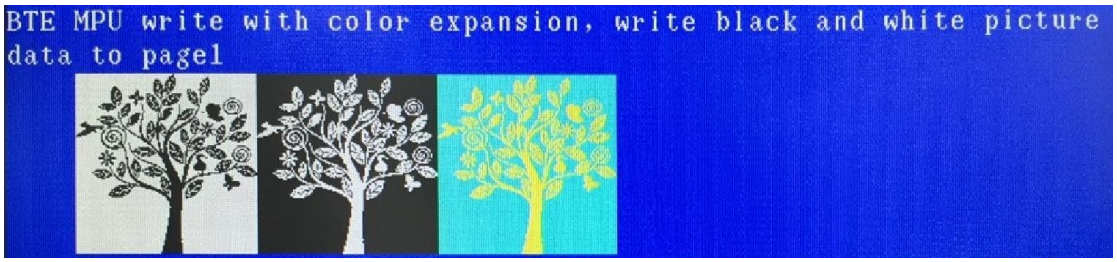
```
//clean current canvas page1 specify active window to color blue
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//write string to current canvas page1 specified position
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8
876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.putString(0,0,"BTE MPU write with color expansion, write black and white picture
data to page1");

// MPU written 1bpp data to specified block of destination canvas through BTE after excute
//color expansion.
ra8876lite.bteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50,50,128,1
```

28,COLOR65K_BLACK,COLOR65K_WHITE,bw);

ra8876lite.bteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,50,128,128,COLOR65K_WHITE,COLOR65K_BLACK,bw);

ra8876lite.bteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,50,128,128,COLOR65K_YELLOW,COLOR65K_CYAN,bw);

Screenshot of the example:



bteMpuWriteColorExpansionWithChromaKey()

Description:

MPU writes 1bpp data to the specified block of destination canvas through using BTE color expansion with chroma key function.

Function prototype:

void bteMpuWriteColorExpansionWithChromaKey(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 foreground_color, ru16 background_color, const unsigned char *data);

void bteMpuWriteColorExpansionWithChromaKey(ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height, ru16 foreground_color, ru16 background_color);

Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write

height	Image height for write
foreground_color	Foreground color
background_color	Background color
*data	Data pointer(Byte format)

Note:

The foreground_color and the background_color must be set to the different color data.

User can continuously write the image data after calling the function which without pointer.

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Bw.bmp

Before performing the following example, we will need an 1bpp image data source, so user should prepare a converted 1bpp image data file (such as bw.h) and then include the relevant header file to the main programming project.

Example:

//write string to current canvas page1 specified position

ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

ra8876lite.putString(0,178,"BTE MPU write with color expansion with chroma key, write black and white picture data to page1");

//MPU written 1bpp data to specified block of destination canvas through BTE after execute color //expansion with chroma key

ra8876lite.bteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,50,228,128,128,COLOR65K_BLACK,COLOR65K_WHITE,bw);

ra8876lite.bteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,228,128,128,COLOR65K_WHITE,COLOR65K_BLACK,bw);

ra8876lite.bteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,50+128+128,228,128,128,COLOR65K_YELLOW,COLOR65K_BLACK,bw);

Screenshot of the example:

btePatternFill()

Description:

Use an indicated pattern to fill the specified block of the canvas.

Function prototype:

void btePatternFill(ru8 p8x8or16x16, ru32 s0_addr, ru16 s0_image_width, ru16 s0_x, ru16 s0_y, ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height);

Parameter	Description
p8x8or16x16	Pattern size select, 0 = 8*8, 1=16*16
s0_addr	Start address of the memory of the pattern image source 0 canvas
s0_image_width	Width of the image memory of the pattern image source 0 canvas
s0_x	Pattern image X-axis coordinate of the source 0 canvas
s0_y	Pattern image Y-axis coordinate of the source 0 canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for fill
height	Image height for fill

Note:

The indicated pattern must be pre-write to the specified address of memory by user.

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

pattern6.bmp

pattern11.bmp

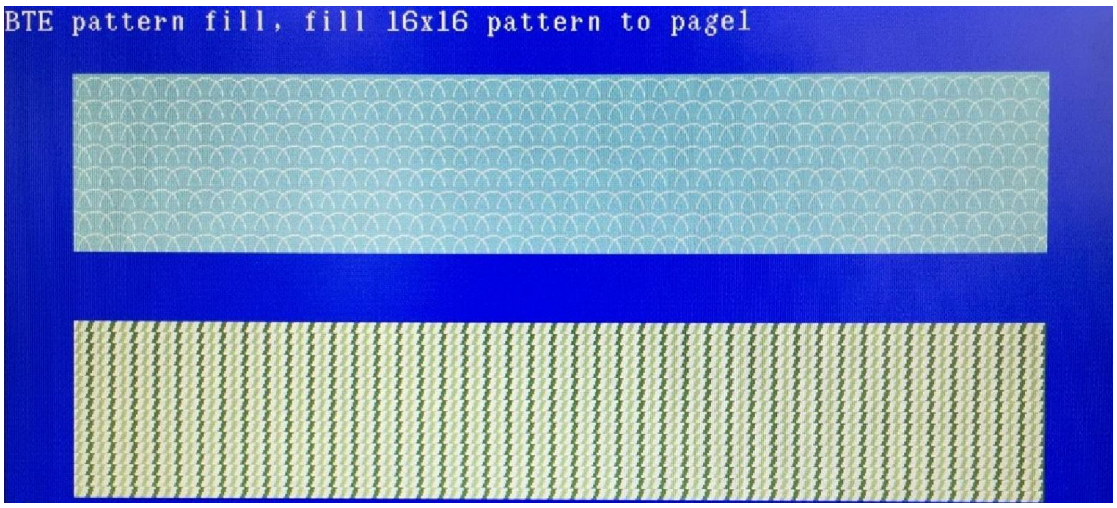
Before performing the following example, we will need an image data source, so user should prepare the converted 16bpp image data files (such as pattern6.h and pattern11.h) and then include the relevant header files to the main programming project.

Example:

```
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//write picture to pattern1 ram
ra8876lite.canvasImageStartAddress(PATTERN1_RAM_START_ADDR);
ra8876lite.canvasImageWidth(16);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(16,16);
ra8876lite.putPicture_16bpp(0,0,16,16,pattern6);

//write picture to pattern2 ram
ra8876lite.canvasImageStartAddress(PATTERN2_RAM_START_ADDR);
ra8876lite.putPicture_16bpp(0,0,16,16,pattern11);

//write picture to pattern3 ram
ra8876lite.canvasImageStartAddress(PATTERN3_RAM_START_ADDR);
ra8876lite.putPicture_16bpp(0,0,16,16,bug1);

//set canvas and active window back
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
```

```
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);

ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIGHT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA8876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.putString(0,0,"BTE pattern fill, fill 16x16 pattern to page1");

ra8876lite.btePatternFill(1,PATTERN1_RAM_START_ADDR,16,0,0,PAGE1_START_ADDR,SCREEN_WIDTH, 50,50,700,128);

ra8876lite.btePatternFill(1,PATTERN2_RAM_START_ADDR,16,0,0,PAGE1_START_ADDR,SCREEN_WIDTH, 50,228,700,128);
```

Screenshot of the example:



btePatternFillWithChromaKey()

Description:

Use an indicated pattern with chroma key function to fill specified block of the canvas.

Function prototype:

void btePatternFill(ru8 p8x8or16x16, ru32 s0_addr, ru16 s0_image_width, ru16 s0_x, ru16 s0_y, ru32 des_addr, ru16 des_image_width, ru16 des_x, ru16 des_y, ru16 width, ru16 height , ru16 chromakey_color);

Parameter	Description
p8x8or16x16	Pattern size select, 0 = 8*8, 1=16*16
s0_addr	Start address of the memory of the pattern image source 0 canvas
s0_image_width	Width of the image memory of the pattern image source 0 canvas
s0_x	Pattern image X-axis coordinate of the source 0 canvas
s0_y	Pattern image Y-axis coordinate of the source 0 canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for fill
height	Image height for fill
chromakey_color	Data of the chroma key color

Note:

The indicated pattern must be pre-write to the specified address of memory by user.

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

Reference picture:

Bug1.bmp

Before performing the following example, we will need an image data source, so user should prepare a converted 16bpp image data file (such as bug1.h) and then include the relevant header files to the main programming project.

Example:

ra8876lite.putString(0,356,"BTE pattern fill with chroma key, fill with chroma key 16x16 pattern to page1");

ra8876lite.btePatternFillWithChromaKey(1,PATTERN3_RAM_START_ADDR,16,0,0,PAGE1_S

TART_ADDR,SCREEN_WIDTH, 50,406,700,128,0xe8e4);

Screenshot of the example:

Chapter 8 DMA

RA8876 provides the DMA function, DMA function can read image data from serial flash of the RA8876 expanded and written to specified block of the canvas quickly, external expansion of serial flash provides space to store user image data, the amount of the color image data is huge, built-in ROM of low-end MPU usually less than 512Kbyte, can store a small amount of image data only, clock of the low-end MPU is usually less than 50MHz, If writing huge amounts of data need to spend a long time, so user can choose to use DMA function, to program the image data into the Serial Flash first, then use DMA function performs fast image access.

Function	Description
setSerialFlash4BytesMode()	Set serial flash to 4Bytes mode
dma_24bitAddressBlockMode()	DMA read 24bit serial flash, block mode
dma_32bitAddressBlockMode()	DMA read 32bit serial flash, block mode

Note:

Please refer to" *RA8876 Arduino Wire Sketch.jpg*" for the circuitry connection or please refer to the appendix Figure A-1

Regarding the serial flash programming, please refer to "*ArduinoDue_SpiFlashProgramWithSdCard"* demonstration and explanation.

Before performing all of the demonstrated examples in this chapter, user has to pre-program the file "*ALL_Pic.bin*" into the serial flash memory. The file "*ALL_Pic.bin*" is stored in the folder "file2sdcard " of the demonstrated project "*ArduinoDue_SpiFlashProgramWithSdCard* ".

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

setSerialFlash4BytesMode()

Description:

When using the 32bit address serial flash memory, user must call the function "setSerialFlash4BytesMode()" first for setting the serial flash memory as 4Bytes mode.

Function prototype:

void setSerialFlash4BytesMode(ru8 scs_select);

Parameter	Description
scs_select	Select serial IF0 or serial IF1

Note:

RA8876 provides 2 SPI master interfaces are the IF0 and the IF1. It is recommended to use the

IF0 for the GENITOP's font ROM, and use the IF1 for the serial flash memory (as image data source for DMA function), please refer to the datasheet of RA8876 for the detailed information.

dma_24bitAddressBlockMode()

Description:

Read the image data from a 24bit address serial flash memory via the specified serial I/F, and the write them into the specified memory block of the current canvas.

Function prototype:

void dma_24bitAddressBlockMode(ru8 scs_selct, ru8 clk_div, ru16 x0, ru16 y0, ru16 width, ru16 height, ru16 picture_width, ru32 addr);

Parameter	Description
scs_selct	RA8876_SERIAL_FLASH_SELECT0 RA8876_SERIAL_FLASH_SELECT1 Select serial IF0 or serial IF1
clk_div	RA8876 _SPI_DIV2 RA8876 _SPI_DIV4 RA8876 _SPI_DIV6 RA8876 _SPI_DIV8 RA8876 _SPI_DIV10 Select SPI clock divider
x0	X-axis coordinate of the current canvas
y0	Y-axis coordinate of the current canvas
width	Width of the DMA block
height	Height of the DMA block
picture_width	Image width of the Serial Flash
addr	Image data start address of the Serial Flash

Example:

DMA function can be executed to read the entire image data or read the partial block data of the image, and then write the data into the specified memory block of the current canvas.

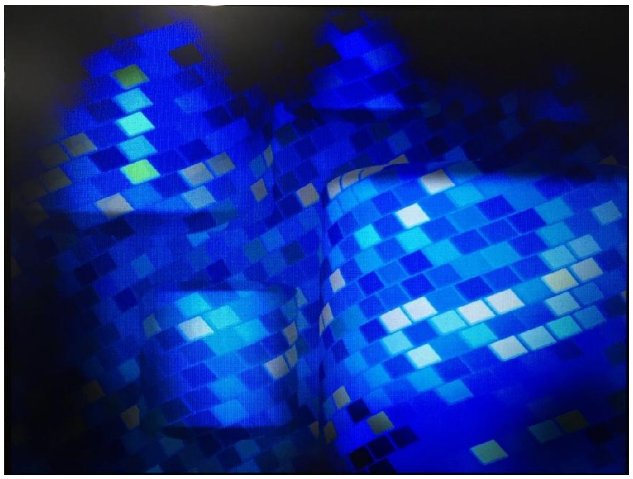
Example, the entire image data read and write:

//set current canvas

//clean current canvas page1 specify active window to color blue
//DMA reads image data from Serial Flash and writes to specified block of the current canvas

```
const int WP2=960000;

ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_BLUE);

//demo 24bit address Serial Flash DMA function
ra8876lite.dma_24bitAddressBlockMode(RA8876 _SERIAL_FLASH_SELEC1,RA8876
_SPI_DIV2,0,0,800,600,800,WP2);
```

Screenshot of the example:



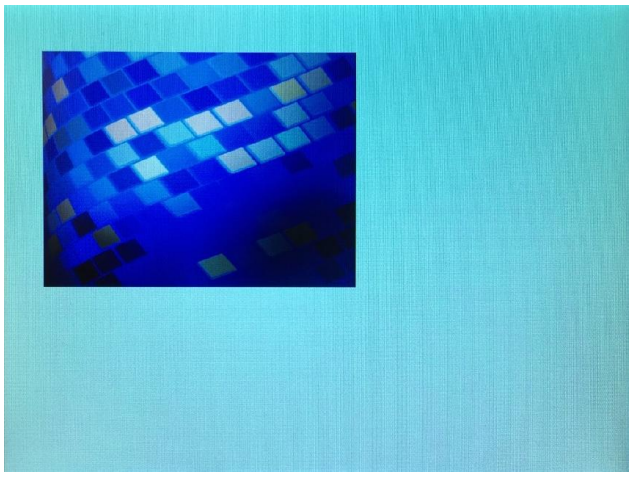
Example, the partial block data read and write of the image:

```
//demo 24bit address serial flash DMA partial
//set current canvas
// clean current canvas page1 specify active window to color light cyan
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
```

ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_LIGHTCYAN);

//DMA read partial block (400 ,300) to (799,599) of the "WP2" from serial flash, then written to the //specified block of the current canvas.
// x0 = 50
// y0 = 60
// width = 400
// height = 300
// picture_width = 800
*// addr = WP2+(800*2*300)+(400*2)=960000+480000+800 = 1440800*
ra8876lite.dma_24bitAddressBlockMode(RA8876_SERIAL_FLASH_SELECT1,RA8876_SPI_DIV2,50,60,400,300,800,1440800);

Screenshot of the example:



dma_32bitAddressBlockMode()

Description:
Read the image data from a 32bit address serial flash memory via the specified serial I/F, and the write them into the specified memory block of the current canvas.

Function prototype:
void dma_32bitAddressBlockMode(ru8 scs_selct, ru8 clk_div, ru16 x0, ru16 y0, ru16 width, ru16 height, ru16 picture_width, ru32 addr);

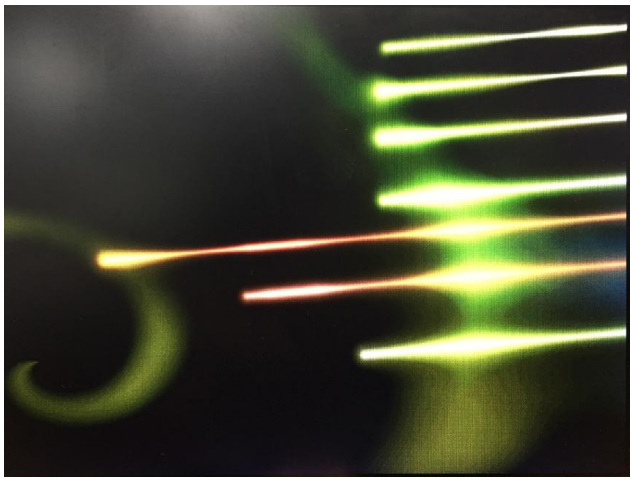
Parameter	Description
scs_selct	RA8876_SERIAL_FLASH_SELECT0

	RA8876_SERIAL_FLASH_SELECT1 Select serial IF0 or serial IF1
clk_div	RA8876_SPI_DIV2 RA8876_SPI_DIV4 RA8876_SPI_DIV6 RA8876_SPI_DIV8 RA8876_SPI_DIV10 Select SPI clock divider
x0	X-axis coordinate of the current canvas
y0	Y-axis coordinate of the current canvas
width	Width of the DMA block
height	Height of the DMA block
picture_width	Image width of the serial flash
addr	Image data start address of the serial flash

Example:

```
const int WP10=8640000;
//DMA demo 32bit address
//when using the 32bit address serial flash, must be setting serial flash to 4Bytes mode
//only needs set one times after power on
ra8876lite.setSerialFlash4BytesMode(1);
while(1)
{
//set current canvas
// clean current canvas page1 specify active window to color light cyan
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.drawSquareFill(0, 0, 799, 599, COLOR65K_LIGHTCYAN);

//DMA read image data from Serial Flash and write to specified block of the current canvas
ra8876lite.dma_32bitAddressBlockMode(RA8876_SERIAL_FLASH_SELECT1,RA8876_SPI_
DIV2,0,0,800,600,800,WP10);
delay(2000);
}
```

Screenshot of the example:



Chapter 9 PWM

Function	Description
pwm_Prescalar()	Set Prescalar
pwm_ClockMuxReg()	PWM frequency divider and the PWM pin function selection
pwm_Configuration()	Setting and start PWM function
pwm0_ClocksPerPeriod()	Setting amount of the each duty cycle clock for PWM0
pwm0_Duty()	PWM0 duty cycle
pwm1_ClocksPerPeriod()	Setting amount of the each duty cycle clock for PWM1
pwm1_Duty()	PWM1 duty cycle

Please refer to" *RA8876 Arduino Wire Sketch.jpg*" for the circuitry connection or please refer to the appendix Figure A-1

pwm_Prescalar()

Description:

Set prescalar.

Function prototype:

void pwm_Prescalar(ru8 Prescalar);

Parameter	Description
Prescalar	RA8876_PRESCALAR

Note:

Base frequency of the PWM0 and PWM1 = Core_Freq / (Prescalar + 1)

pwm_ClockMuxReg()

Description:

It is used for decided the PWM frequency divider and the PWM pin function selection

Function prototype:

void pwm_ClockMuxReg(ru8 pwm1_clk_div, ru8 pwm0_clk_div, ru8 xpwm1_ctrl, ru8 xpwm0_ctrl);

Parameter	Description

pwm1_clk_div	PWM1 base frequency divider setting RA8876_PWM_TIMER_DIV1 RA8876_PWM_TIMER_DIV2 RA8876_PWM_TIMER_DIV4 RA8876_PWM_TIMER_DIV8
pwm0_clk_div	PWM0 base frequency divider setting RA8876_PWM_TIMER_DIV1 RA8876_PWM_TIMER_DIV2 RA8876_PWM_TIMER_DIV4 RA8876_PWM_TIMER_DIV8
xpwm1_ctrl	PWM1 pin function selection RA8876_XPWM1_OUTPUT_ERROR_FLAG RA8876_XPWM1_OUTPUT_PWM_TIMER1 RA8876_XPWM1_OUTPUT_OSC_CLK
xpwm0_ctr	PWM0 pin function selection RA8876_XPWM0_GPIO_C7 RA8876_XPWM0_OUTPUT_PWM_TIMER0 RA8876_XPWM0_OUTPUT_CORE_CLK

pwm_Configuration()

Description:

Set and start PWM function

Function prototype:

void pwm_Configuration(ru8 pwm1_inverter, ru8 pwm1_auto_reload, ru8 pwm1_start,ru8 pwm0_dead_zone, ru8 pwm0_inverter, ru8 pwm0_auto_reload,ru8 pwm0_start);

Parameter	Description
pwm1_inverter	PWM1 output inverter off or on RA8876_PWM_TIMER1_INVERTER_OFF RA8876_PWM_TIMER1_INVERTER_ON
pwm1_auto_reload	PWM1 output one shot or auto reload RA8876_PWM_TIMER1_ONE_SHOT RA8876_PWM_TIMER1_AUTO_RELOAD
pwm1_start	PWM1 stop or start RA8876_PWM_TIMER1_STOP

	RA8876_PWM_TIMER1_START
pwm0_dead_zone	PWM0 dead zone disable or enable RA8876_PWM_TIMER0_DEAD_ZONE_DISABLE RA8876_PWM_TIMER0_DEAD_ZONE_ENABLE
pwm0_inverter	PWM0 output inverter off or on RA8876_PWM_TIMER0_INVERTER_OFF RA8876_PWM_TIMER0_INVERTER_ON
pwm0_auto_reload	PWM0 output one shot or auto reload RA8876_PWM_TIMER0_ONE_SHOT RA8876_PWM_TIMER0_AUTO_RELOAD
pwm0_start	PWM0 stop or start RA8876_PWM_TIMER0_STOP RA8876_PWM_TIMER0_START

pwm0_ClocksPerPeriod()
pwm1_ClocksPerPeriod()

Description:
The function “pwm0_ClocksPerPeriod()” sets the clock amount of each duty cycle of the PWM0.
The function “pwm1_ClocksPerPeriod()” sets the clock amount of each duty cycle of the PWM1.

Function prototype:
```
void pwm0_ClocksPerPeriod(ru16 clocks_per_period);
void pwm1_ClocksPerPeriod(ru16 clocks_per_period);
```

Parameter	Description
clocks_per_period	Amount of the each duty cycle clock (1~65535)

Note:
Another meaning for the setting is PWM resolution, for example, the setting is 1000, then the duty cycle range can be adjusted from 0 to 1000.

pwm0_Duty()
pwm1_Duty()

Description:

"**pwm0_Duty()**" is the duty cycle setting for PWM0.
"**pwm1_Duty()**" is the duty cycle setting for PWM1.

Function prototype:

void pwm0_Duty(ru16 duty);
void pwm1_Duty(ru16 duty);

Parameter	Description
duty	Value of the duty cycle

Note:

Duty cycle's duty range is decided by clocks_per_period setting value.

Example:

```
//pwm demo please measure by oscilloscope
ra8876lite.pwm_Prescalar(RA8876_PRESCALAR); //if core_freq = 120MHz, pwm base clock =
                                            //120/(3+1) = 30MHz

ra8876lite.pwm_ClockMuxReg(RA8876_PWM_TIMER_DIV4,RA8876_PWM_TIMER_DIV4,RA
  8876_XPWM1_OUTPUT_PWM_TIMER1,RA8876_XPWM0_OUTPUT_PWM_TIMER0);
//pwm timer clock = 30 MHz /4 = 7.5MHz

ra8876lite.pwm0_ClocksPerPeriod(1024); // pwm0 = 7.5MHz/1024 = 7.3KHz
ra8876lite.pwm0_Duty(10);//pwm0 set 10/1024 duty

ra8876lite.pwm1_ClocksPerPeriod(256);   // pwm1 = 7.5MHz/256 = 29.2KHz
ra8876lite.pwm1_Duty(5); //pwm1 set 5/256 duty

ra8876lite.pwm_Configuration(RA8876_PWM_TIMER1_INVERTER_ON,RA8876_PWM_TIME
  R1_AUTO_RELOAD,RA8876_PWM_TIMER1_START,RA8876_PWM_TIMER0_DEAD_ZON
  E_DISABLE ,RA8876_PWM_TIMER0_INVERTER_ON,RA8876_PWM_TIMER0_AUTO_REL
  OAD,RA8876_PWM_TIMER0_START);
```

Chapter 10 Arduino SD

In this section, we use a SD card as image data source for RA8876 and it connected with Arduino Due board. So before we use this kind of application, user needs to prepare the converted image file (such as ***.bin) and store the "***.bin" file into the SD card via PC. If the image file has stored into the SD card already, and then RA8876 is able to get the image data from the SD card through Arduino Due's access.

Function	Description
sdCardShowPicture16bpp()	Read image data with specified filename from SD card and written to specified location of the current canvas
sdCardShowPicture16bppBteMpuWriteWithROP()	Read image data with specified filename from SD card, and then written to specified location of the destination canvas through BTE MPU write with logic operation.
sdCardShowPicture16bppBteMpuWriteWithChromaKey()	Read image data with specified filename from SD card, and then written to specified location of the destination canvas through BTE MPU write with chroma key color ignore.
sdCardShowPicture16bppBteMpuWriteColorExpansion()	Read (1bpp) image data with specified filename from SD card, and then written to specified location of the destination canvas through BTE MPU write with color expansion.
sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey()	Read (1bpp) image data with specified filename from SD card, and then written to specified location of the destination canvas through BTE MPU write with color expansion and chroma key color ignore.

Note:

These subroutines are additionally provided, it is not included in RA8876_Lite.cpp, if user needs the relevant application, please refer to" *RA8876_Lite_Arduino_SD.ino"*, and copy the needed functions to your own programming project.

The circuitry connection between Arduino board, SD card and RA8876, please refer to" *RA8876ArduinoDueSD Wire Sketch.jpg"* or appendix Figure A-2.

Image data is converted by using the "*Image_Tool_V1.0*" image tool.

sdCardShowPicture16bpp()

Description:

Read the image data of the specified file from SD card, and then write the image data on the location of the specified canvas.

Function prototype:

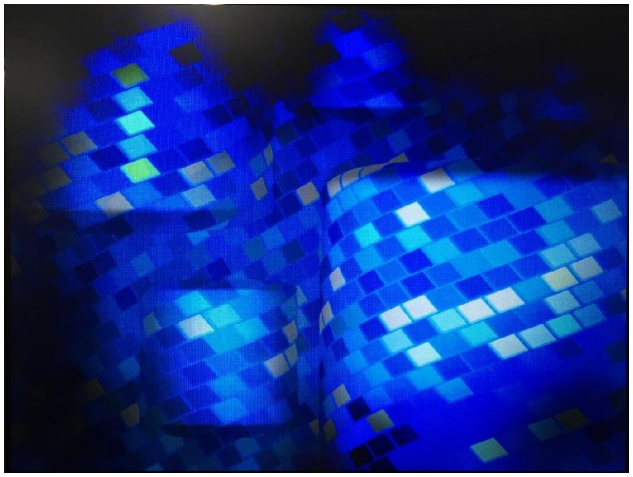
void sdCardShowPicture16bpp(unsigned short x, unsigned short y, unsigned short width, unsigned short height, char *filename);

Parameter	Description
x	X-axis coordinate
y	Y-axis coordinate
width	Width of the image
height	Height of the image
*filename	Image filename

Example:

```
ra8876lite.canvasImageStartAddress(PAGE1_START_ADDR);
ra8876lite.canvasImageWidth(SCREEN_WIDTH);
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);

sdCardShowPicture16bpp(0,0,800,600,"wp2.bin");
```

Screenshot of the example:



sdCardShowPicture16bppBteMpuWriteWithROP()

Description:

Read the image data of the specified file from SD card, and then write the image data on the destination of the specified canvas through the BTE MPU write with ROP function.

Function prototype:

void sdCardShowPicture16bppBteMpuWriteWithROP(unsigned long s1_addr, unsigned short s1_image_width, unsigned short s1_x, unsigned short s1_y, unsigned long des_addr, unsigned short des_image_width, unsigned short des_x, unsigned short des_y, unsigned short width, unsigned short height, unsigned char rop_code, char *filename);

Parameter	Description
s1_addr	Start address of the memory of the source 1 canvas
s1_image_width	Width of the image memory of the source 1 canvas
s1_x	Source 1 image X-axis coordinate of the canvas
s1_y	Source 1 image Y-axis coordinate of the canvas
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
rop_code	Select of the logic operation RA8876_BTE_ROP_CODE_0 (Blackness) RA8876_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8876_BTE_ROP_CODE_2 ~S0・S1 RA8876_BTE_ROP_CODE_3 ~S0 RA8876_BTE_ROP_CODE_4 S0・~S1

	RA8876_BTE_ROP_CODE_5 ~S1 RA8876_BTE_ROP_CODE_6 S0^S1 RA8876_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8876_BTE_ROP_CODE_8 S0・S1 RA8876_BTE_ROP_CODE_9 ~ (S0^S1) RA8876_BTE_ROP_CODE_10 S1 RA8876_BTE_ROP_CODE_11 ~S0+S1 RA8876_BTE_ROP_CODE_12 S0 RA8876_BTE_ROP_CODE_13 S0+~S1 RA8876_BTE_ROP_CODE_14 S0+S1 RA8876_BTE_ROP_CODE_15 (Whiteness)
*filename	Image filename

Note:

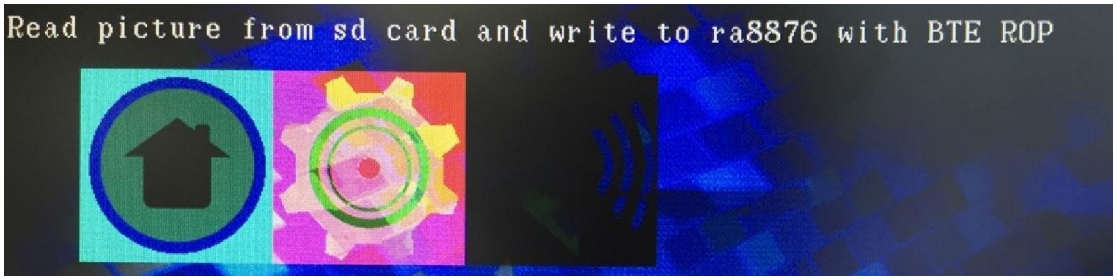
Regarding the related MPU data write functions of BTE, the S0 (source0) can be regarded as the MPU write data and the S1 (Source1) can be set as the destination.

Example:

```
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
  HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
  RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
  8876_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8876lite.putString(0,10,"Read picture from sd card and write to RA8876 with BTE ROP");
```

```
sdCardShowPicture16bppBteMpuWriteWithROP(PAGE1_START_ADDR, SCREEN_WIDTH,
  50, 50, PAGE1_START_ADDR, SCREEN_WIDTH, 50, 50, 128,
  128,RA8876_BTE_ROP_CODE_3,"home.bin");

sdCardShowPicture16bppBteMpuWriteWithROP(PAGE1_START_ADDR, SCREEN_WIDTH,
  50+128, 50, PAGE1_START_ADDR, SCREEN_WIDTH, 50+128, 50, 128,
  128,RA8876_BTE_ROP_CODE_6,"appli.bin");

sdCardShowPicture16bppBteMpuWriteWithROP(PAGE1_START_ADDR,
  SCREEN_WIDTH,50+128+128,50,PAGE1_START_ADDR,SCREEN_WIDTH, 50+128+128,
  50, 128, 128,RA8876_BTE_ROP_CODE_8,"sound.bin");
```

Screenshot of the example:



sdCardShowPicture16bppBteMpuWriteWithChromaKey()

Description:

Read the image data of the specified file from SD card, and then write the image data on the destination of the specified canvas through the BTE MPU write with chroma key function.

Function prototype:

```
void sdCardShowPicture16bppBteMpuWriteWithChromaKey(unsigned long des_addr ,
  unsigned short des_image_width, unsigned short des_x, unsigned short des_y, unsigned short
  width, unsigned short height, unsigned short chromakey_color, char *filename);
```

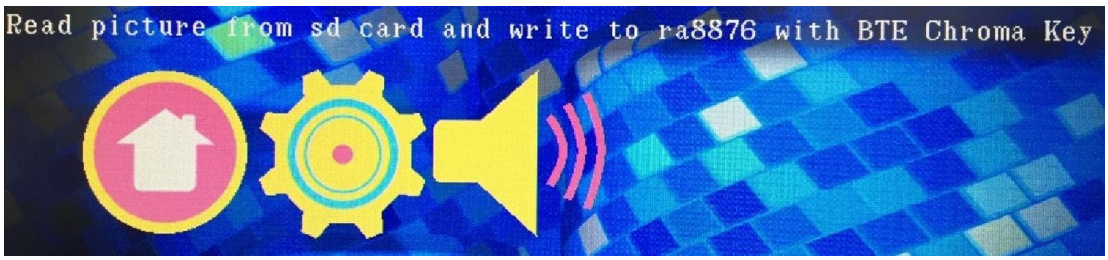
Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas

width	Image width for write
height	Image height for write
chromakey_color	Data of chroma key color
* filename	Image filename

Example:

```
ra8876lite.putString(0,50+128+10,"Read picture from sd card and write to RA8876 with BTE
  Chroma Key");

sdCardShowPicture16bppBteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WI
DTH,50, 50+128+50,128,128,0xf800,"home.bin");

sdCardShowPicture16bppBteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WI
  DTH,50+128, 50+128+50,128,128,0xf800,"appli.bin");

sdCardShowPicture16bppBteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WI
  DTH,50+128+128,50+128+50,128,128,0xf800,"sound.bin");
```

Screenshot of the example:



sdCardShowPicture16bppBteMpuWriteColorExpansion()

Description:

Read the image data (1bpp) of the specified file from SD card, and then write the image data on the destination of the specified canvas through the BTE MPU write with color expansion function.

Function prototype:

```
void sdCardShowPicture16bppBteMpuWriteColorExpansion(unsigned long des_addr, unsigned
  short des_image_width, unsigned short des_x, unsigned short des_y, unsigned short width,
```

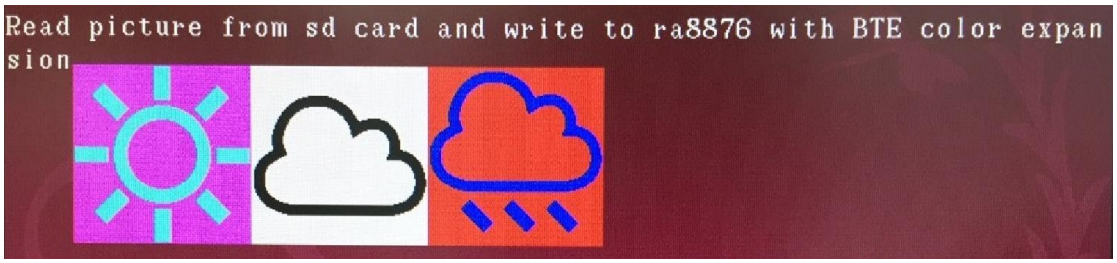
unsigned short height, unsigned short foreground_color, unsigned short background_color, char *filename);

Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
foreground_color	Foreground color
background_color	Background color
* filename	Image filename

Example:

```
sdCardShowPicture16bpp(0,0,800,600,"wp23.bin");
ra8876lite.activeWindowXY(0,0);
ra8876lite.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8876lite.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8876lite.setTextParameter1(RA8876_SELECT_INTERNAL_CGROM,RA8876_CHAR_HEIG
 HT_24,RA8876_SELECT_8859_1);//cch
ra8876lite.setTextParameter2(RA8876_TEXT_FULL_ALIGN_ENABLE,
 RA8876_TEXT_CHROMA_KEY_ENABLE,RA8876_TEXT_WIDTH_ENLARGEMENT_X1,RA
 8876_TEXT_HEIGHT_ENLARGEMENT_X1);

ra8876lite.putString(0,10,"Read picture from sd card and write to RA8876 with BTE color
 expansion");

sdCardShowPicture16bppBteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WI
 DTH,50,50,128,128,COLOR65K_CYAN,COLOR65K_MAGENTA,"sun.bin");

sdCardShowPicture16bppBteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WI
 DTH,50+128,50,128,128,COLOR65K_BLACK,COLOR65K_WHITE,"cloud.bin");

sdCardShowPicture16bppBteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WI
 DTH,50+128+128, 50, 128, 128,COLOR65K_BLUE,COLOR65K_RED,"rain.bin");
```

Screenshot of the example:



sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey()

Description:

Read the image data (1bpp) of the specified file from SD card, and then write the image data on the destination of the specified canvas through the BTE MPU write with color expansive and chroma key function.

Function prototype:

void sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey (unsigned long des_addr, unsigned short des_image_width, unsigned short des_x, unsigned short des_y, unsigned short width, unsigned short height, unsigned short foreground_color, unsigned short background_color, char *filename);

Parameter	Description
des_addr	Start address of the memory of the destination canvas
des_image_width	Width of the image memory of the destination canvas
des_x	Destination image X-axis coordinate of the canvas
des_y	Destination image Y-axis coordinate of the canvas
width	Image width for write
height	Image height for write
foreground_color	Foreground color
background_color	Background color
* filename	Image filename

foreground_color and background_color must be set to different color data.

Example:

ra8876lite.putString(0,50+128+10,"Read picture from sd card and write to RA8876 with BTE color expansion with chroma key");

```
sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADD
  R,SCREEN_WIDTH, 50, 50+128+50+10, 128,
  128,COLOR65K_WHITE,COLOR65K_BLACK,"sun.bin");

sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADD
  R,SCREEN_WIDTH, 50+128, 50+128+50+10, 128,
  128,COLOR65K_WHITE,COLOR65K_BLACK,"cloud.bin");

sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADD
  R,SCREEN_WIDTH,50+128+128,50+128+50+10,128,128,COLOR65K_WHITE,COLOR65K_
  BLACK,"rain.bin");
```

Screenshot of the example:

Appendix A

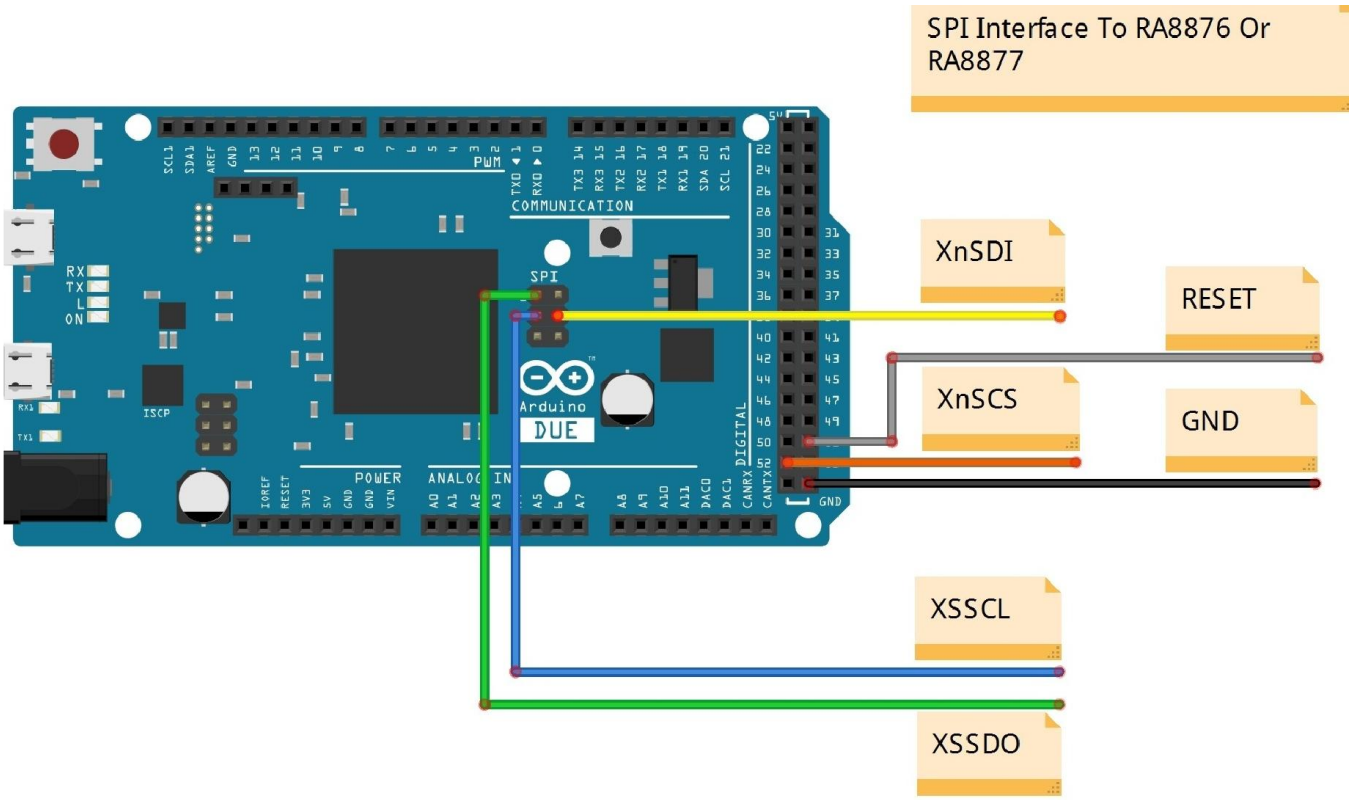


Figure A-1

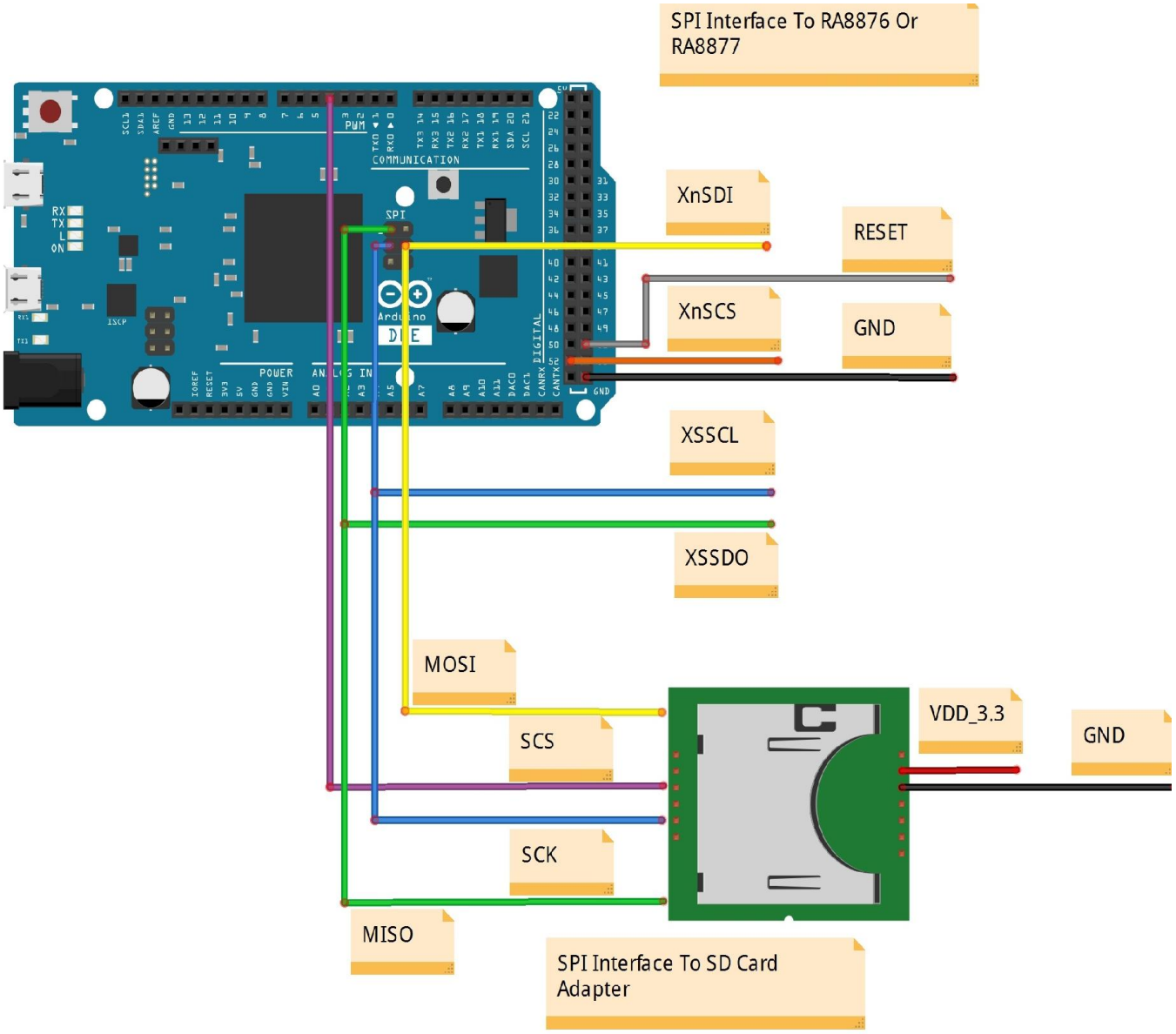


Figure A-2

End