RAiO

RA8871M_Lite

User Guide

Sep 15, 2017

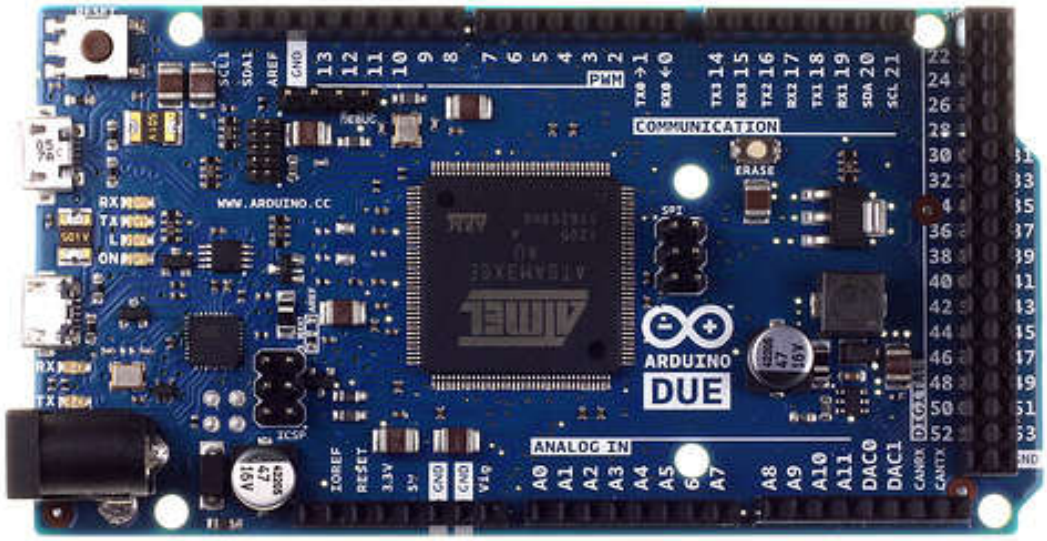
Revise History		
Version	**Date**	**Description**
1.0	2017.09.15	Initial Release

第 1 章 RA8871M_Lite 介紹

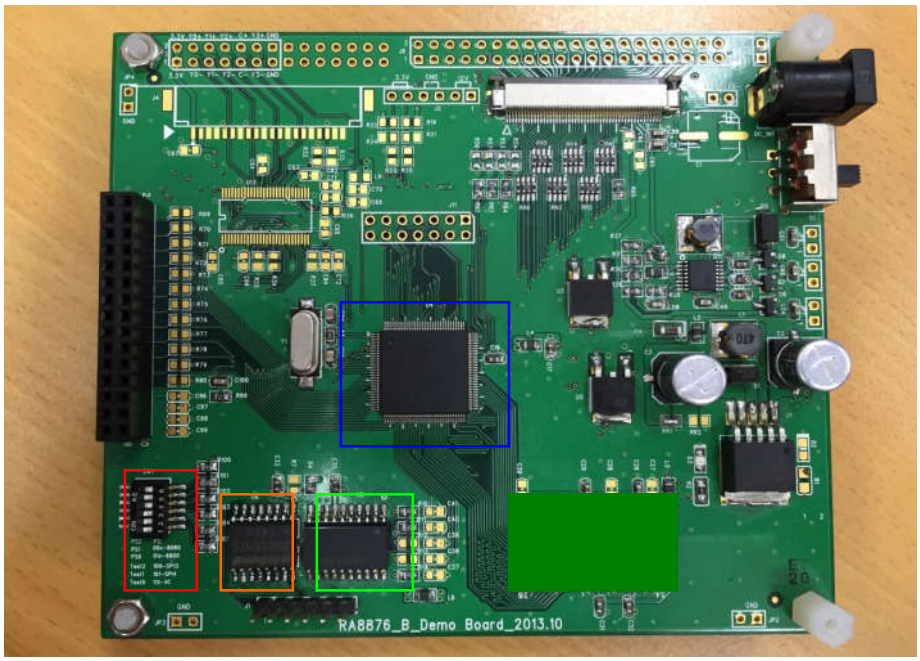
RA8871M_Lite 是基於 Arudino Due 開發板連接 RA8871M 驅動板，與 SD 卡的圖像界面應用接口源代碼提供，可以協助使用者，快速的利用 Arduino Due 開發環境，驅動由瑞佑科技開發的彩色 TFT-LCD 控制器 RA8871M。

硬件需求

1.Arduino Due 開發板

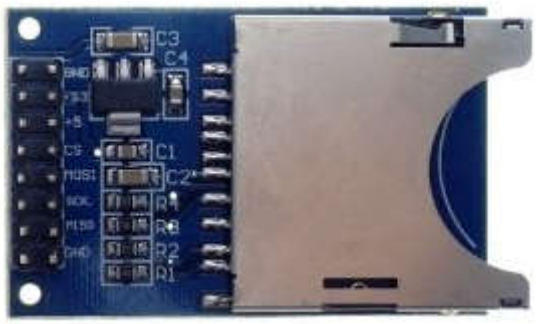


2.RA8871M 驅動板(板載 SPI FLASH，集通字庫 IC)



- RA8871M Chip
- 選擇 SPI interface
- Serial Flash ROM for DMA function
- 集通字庫(Genitop Font ROM)

3.SD 卡轉接板



4.SD 卡(容量 4GB 含以下)

軟件需求

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

Image_Tool_v1.1.0.1 www.raio.com.tw

RA8871M_Lite 特點

RA8871M_Lite 提供 RA8871M TFTLCD 控制器主要的內建功能的應用接口(API)函數，本文所有的演示皆是基於 Arduino Due 開發板 SPI 界面，驅動 RA8871M 顯示 16BPP 色深的圖像到 TFT-LCD。演示的特點包含下列：

初始化(Initialization)

RA8871M 初始化流程。

記憶體規劃與視窗(Memory configuration & Window)

說明如何規劃 RA8871M 內部記憶體(Buffer RAM)和各視窗之間的關係與設定。

圖像(Graphic)

RA8871M 繪圖模式, Arduino Due 寫入彩色圖像數據。

RA8871M 繪圖模式, Arduino Due 寫入使用者自建的 ASCII code 字型文字。

文字(Text)

RA8871M 文字模式，Arduino Due 透過 RA8871M 文字功能寫入 RA8871M 內建 ASCII 字型，並演示 RA8871M 字體放大功能。

搭配支援的集通字庫顯示 ASCII code，BIG5，GB2312 等字型。

幾何繪圖(Geometric Draw)

RA8871M 繪圖模式，Arduino Due 透過 RA8871M 繪圖引擎繪製線，矩形，矩形填滿，圓角矩形，圓角矩形填滿，三角形，三角形填滿,圓形，圓形填滿，橢圓形，橢圓型填滿。

BTE

RA8871M 繪圖模式，Arduino Due 透過 RA8871M BTE 引擎演示：

BTE 記憶體複製。

BTE 記憶體 ROP 邏輯運算與複製。

BTE 記憶體複製與透明色。

Arduino Due 透過 BTE 引擎執行記憶體寫入與 ROP 邏輯運算。

Arduino Due 透過 BTE 引擎執行記憶體寫入與透明色。

Arduino Due 透過 BTE 引擎執行記憶體寫入與顏色擴充。
Arduino Due 透過 BTE 引擎執行記憶體寫入與顏色擴充和透明色。
BTE 圖案填滿。
BTE 圖案填滿與透明色。

DMA

RA8871M 繪圖模式，透過 RA8871M DMA 引擎直接讀取 Serial Flash 內部圖像數據並寫入到 RA8871M 內部記憶體(Buffer RAM)。

PWM

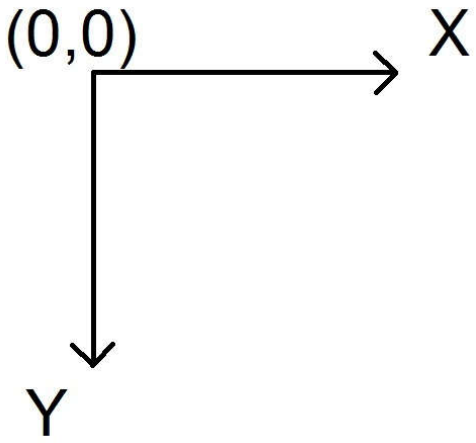
RA8871M PWM 初始設定與頻率計算,責任周期規劃.(需示波器量測)。

Arduino SD

Arduino Due 讀取 SD 卡內圖像數據，然後寫入 RA8871M 內部記憶體(Buffer RAM)。
Arduino Due 讀取 SD 卡內圖像數據，然後透過 BTE 引擎寫入 RA8871M 內部記憶體(Buffer RAM)。

註：

1.本文件的顯示坐標系統：



2.本文使用的顯示屏解析度為 480*272，如需其他解析度參考第 2 章初始化與第 3 章記憶體規劃與視窗。

3.RA8871M_Lite 使用的自定義變數型態如下：

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.接線圖參考附錄 A ：

Figure A-1

Figure A-2

第 2 章 初始化(Initialization)

RA8871M 初始化主要流程如下：

RA8871M 硬件復位(hardware reset)

↓

RA8871M PLL 初始化

↓

RA8871M BufferRAM 初始化

↓

RA8871M 一般設定

↓

RA8871M TFT 時序設定

↓

RA8871M 影像顯示記憶體與視窗初始化設定

↓

RA8871M TFT 顯示開啟

2.1 硬件復位(hardware reset)

begin()

RA8871M 硬件復位程序包含在 begin()函數內。
當 begin()函數返回值為 true，表示硬件復位成功且正確地連接 RA8871M，如返回值為 false，表示連接失敗，檢查 Arduino SPI bus 是否有正確的連接到 RA8871M 驅動板。

2.2 PLL 初始化

ra8871mPllInitial()

函數會參考以下 User_def.h 內定義值，自動完成 PLL 初始化，使用者只要根據顯示的需求設定下列數值。

#define OSC_FREQ 10 // OSC clock frequency, unit: MHz.
#define DRAM_FREQ 120 // SDRAM clock frequency, unit: MHz.
#define CORE_FREQ 100 // Core (system) clock frequency, unit: MHz.
#define SCAN_FREQ TFT_PCLK // Panel Scan clock frequency, unit: MHz.

定義	說明
OSC_FREQ	連接到 RA8871M 的晶體振盪器時脈,建議為 10MHz
DRAM_FREQ	SDRAM 存取時脈,建議 40~120MHz
CORE_FREQ	RA8871M 系統核心時脈,建議 40~100MHz
SCAN_FREQ	TFT 顯示驅動時脈 PCLK,參考 LCD SPEC 指定的 PCLK 時脈需求

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

通常使用者只需要在 User_def.h 選擇以下的定義
//#define TFT_OUT_320_240
#define TFT_OUT_480_272

2.3 BufferRAM 初始化

RA8871M 有內建記憶體(BufferRAM)做為圖像操作與顯示記憶體。

ra8871mBufferRamInitial ()

函數會參考 User_def.h 內定義值 #define BufferRAM_FREQ，並自動執行 BufferRAM 初始化。

通常使用者只需要在 User_def.h 選擇以下的定義

//#define TFT_OUT_320_240
#define TFT_OUT_480_272

2.4 一般設定

以下寄存器在初始化期間設定，參考 RA8871M 規格書與 Ra8871m_Lite.h 寄存器各 bit 定義,根據你的需求設定下列幾項。

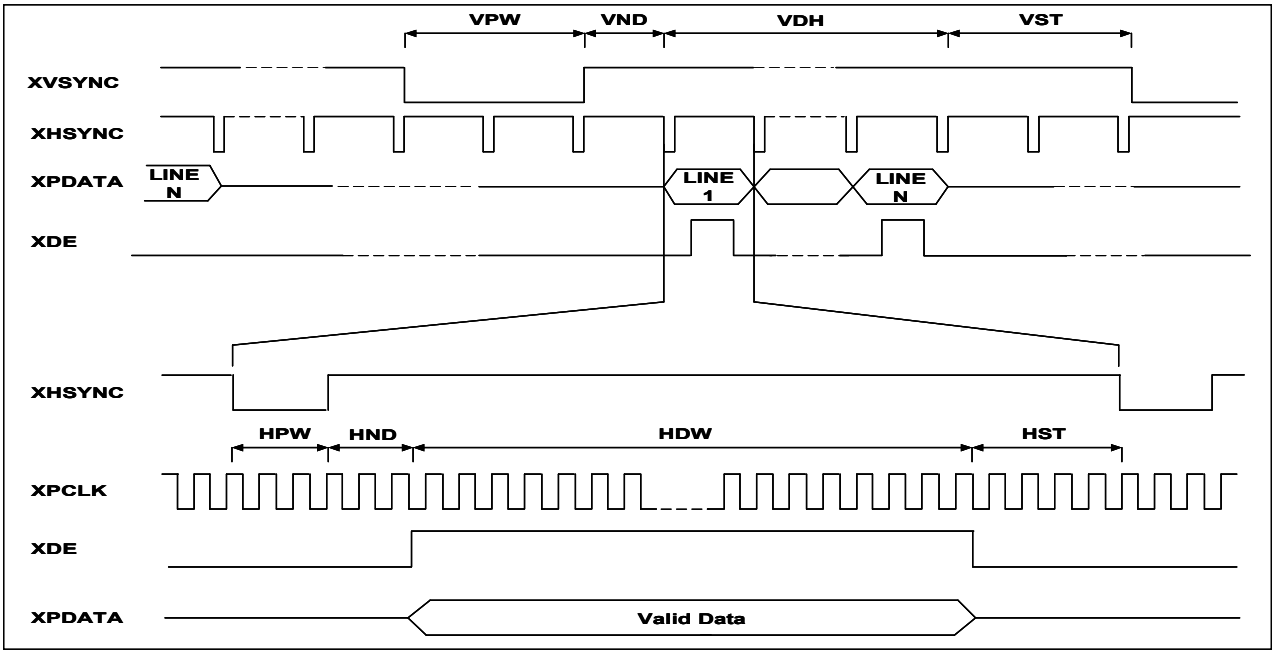
```
lcdRegWrite(RA8871M_CCR);//01h
lcdDataWrite(RA8871M_PLL_ENABLE<<7|RA8871M_WAIT_NO_MASK<<6|RA8871M_KEY_SCAN_DISABLE<<5|RA8871M_TFT_OUTPUT24<<3|RA8871M_I2C_MASTER_DISABLE<<2|RA8871M_SERIAL_IF_ENABLE<<1|RA8871M_HOST_DATA_BUS_SERIAL);

lcdRegWrite(RA8871M_MACR);//02h
lcdDataWrite(RA8871M_DIRECT_WRITE<<6|RA8871M_READ_MEMORY_LRTB<<4|RA8871M_WRITE_MEMORY_LRTB<<1);

lcdRegWrite(RA8871M_ICR);//03h
lcdDataWrite(RA8871M_GRAPHIC_MODE<<2|RA8871M_MEMORY_SELECT_IMAGE);

lcdRegWrite(RA8871M_MPWCTR);//10h
lcdDataWrite(RA8871M_PIP1_WINDOW_DISABLE<<7|RA8871M_PIP2_WINDOW_DISABLE<<6|RA8871M_SELECT_CONFIG_PIP1<<4|RA8871M_IMAGE_COLOCR_DEPTH_16BPP<<2|TFT_MODE);

lcdRegWrite(RA8871M_PIPCDEP);//11h
lcdDataWrite(RA8871M_PIP1_COLOR_DEPTH_16BPP<<2|RA8871M_PIP2_COLOR_DEPTH_16BPP);

lcdRegWrite(RA8871M_AW_COLOR);//5Eh
lcdDataWrite(RA8871M_CANVAS_BLOCK_MODE<<2|RA8871M_CANVAS_COLOR_DEPTH_16BPP);

lcdRegDataWrite(RA8871M_BTE_COLR,RA8871M_S0_COLOR_DEPTH_16BPP<<5|RA8871M_S1_COLOR_DEPTH_16BPP<<2|RA8871M_S0_COLOR_DEPTH_16BPP);//92h
```

2.5 TFT 時序初始化設定

參考以下 User_def.h 內定義值，使用者需參考 LCD 規格書設定以下 TFT 時序數值。

```
#define TFT_PCLK    10    //set PCLK=10MHz

#define TFT_MODE     0    //0:SYNC_mode(SYNC+DE mode), 1: DE mode    //if sync only
                          //mode do not connect DE signal or define XDE_INV = 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           35
#define HDW           480
#define HST           2
#define VPW           2
#define VND           15
#define VDH           272
#define VST           2
```

RA8871M 輸出時序參考



本文選用之 **TFT LCD** 為 **AT043tn24**,要求的 **TFT** 時序如下圖

Item	Symbol	Values			Unit	Remark
		Min.	Typ.	Max.		
Clock cycle	1/tc	-	9.00	15	MHz	
Hsync cycle	$1/f_H$	-	17.14	-	KHz	
Vsync cycle	$1/f_V$	-	59.94	-	Hz	
Horizontal signal	t_h	-	525	-	CLK	Note 1
Horizontal display period	t_{hd}	-	480	-	CLK	
Horizontal Front porch	t_{hf}	2	-	-	CLK	Note 2
Horizontal Pulse width	t_{hp}	2	41	-	CLK	Note 2
Horizontal Back porch	t_{hb}	2	-	-	CLK	Note 2
Vertical cycle	t_v	-	286	-	H	
Vertical display period	t_{vd}	-	272	-	H	
Vertical Front porch	t_{vf}	2	2	-	H	
Vertical Pulse width	t_{vp}	2	10	-	H	
Vertical Back porch	t_{vb}	2	2	-	H	

TFT 時序初始化設定程序：

```
lcdRegWrite(RA8871M_DPCR);//12h
lcdDataWrite(XPCLK_INV<<7|RA8871M_DISPLAY_OFF<<6|RA8871M_OUTPUT_RGB);

lcdRegWrite(RA8871M_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 影像顯示記憶體初始化設定

參考以下 User_def.h 內定義值：

```
//定義顯示屏解析度寬高
#define SCREEN_WIDTH 480
#define SCREEN_HEIGHT 272

/*RA8871M 提供 3 個 512Kbytes 記憶體區塊(BufferRAM)，我們可以規劃為 3 個頁面*/
/*使用者影像記憶體緩存頁面規劃*/
#define PAGE1_START_ADDR    786432    //0C0000h
#define PAGE2_START_ADDR    1835008 //1C0000h
#define PAGE3_START_ADDR    2883584 //2C0000h
```

視窗初始化程序：

```
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 顯示開啟

RA8871M 初始化設定完成後，通常先對顯示記憶體執行寫入圖像數據，然後再將顯示開啟；開啟後，RA8871M TFT LCD 時序控制器，將會自動提取影像顯示記憶體的顯示視窗區塊內的影像數據，並且輸出到 LCD 顯示。

displayOn()

描述：

顯示開啟或關閉.

函數原型：

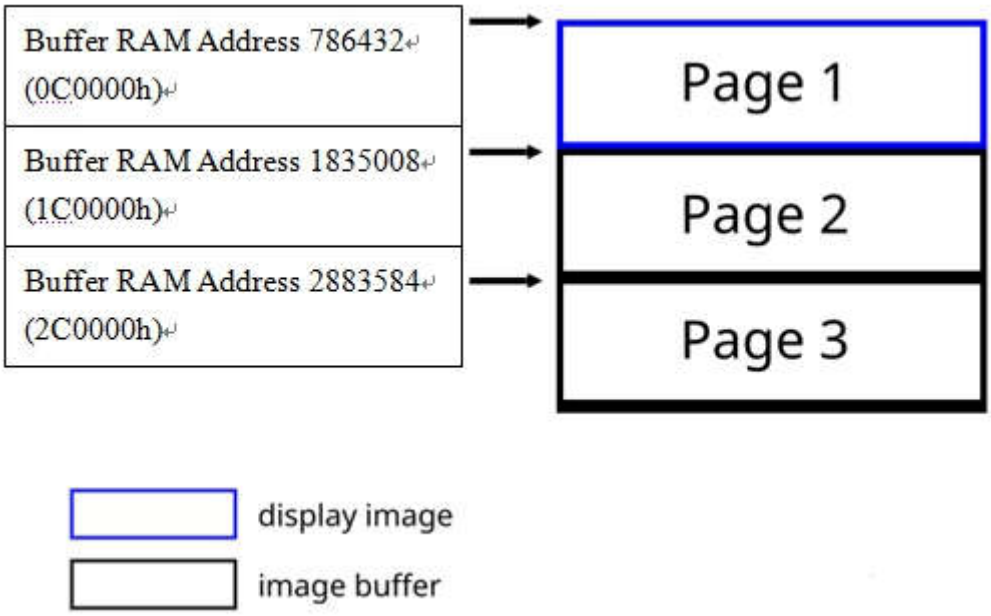
void displayOn(boolean on);

參數	說明
on	= true 顯示開啟 = false 顯示關閉

第 3 章 記憶體規劃與視窗(Memory Configuration & Window)

在本文中，記憶體被分配成 3 頁，第 1 頁分配給影像顯示記憶體，其他頁做為圖像緩存，例如圖像更新到第 2 頁，然後使用 BTE memory copy 功能，將該頁數據複製到第 1 頁影像顯示記憶體，這方法可以避免直接對影像顯示記憶體，執行圖形更新而造成畫面顯示的閃動，疊加效果。

記憶體規劃圖示：



記憶體和視窗相關函數如下表：

函數	說明
displayImageStartAddress()	設定影像顯示記憶體的起始位址
displayImageWidth()	設定影像顯示記憶體的寬度
displayWindowStartXY()	設定顯示視窗在影像顯示記憶體的左上角起始點
canvasImageStartAddress()	設定畫布影像記憶體的起始位址
canvasImageWidth()	設定畫布影像記憶體的寬度
activeWindowXY()	設定活動視窗在畫布上的左上角起始點
activeWindowWH()	設定活動視窗寬高

displayImageStartAddress()

描述：

設定影像顯示記憶體的起始位址。

函數原型：

void displayImageStartAddress(ru32 addr);

參數	說明
addr	影像顯示記憶體的起始位置

附註和範例：

影像顯示記憶體為顯示視窗的數據來源記憶體，建議設定為 0。在本文中，記憶體規劃為 3 個頁面，第 1 個頁面(Buffer RAM address = 786432)則分配給影像顯示記憶體，所以初始化階段設定如下：

displayImageStartAddress(PAGE1_START_ADDR);

displayImageWidth()

描述：

設定影像顯示記憶體的寬度。

函數原型：

void displayImageWidth(ru16 width);

參數	說明
width	影像顯示記憶體的寬度

附註和範例：

影像顯示記憶體的寬度設定必須與頁面(畫布)寬度相等。

將每個頁面(畫布)寬度設定 480(=SCREEN_WIDTH),所以初始化設定如下：

displayImageWidth(SCREEN_WIDTH);

此函數只需要在初始化期間設定一次。

displayWindowStartXY()

描述：

設定顯示視窗在影像顯示記憶體的左上角起始點。

函數原型：

void displayWindowStartXY(ru16 x0,ru16 y0);

參數	說明
x0	左上角 X 軸座標
y0	左上角 Y 軸座標

附註和範例：

顯示視窗的寬度和高度是參考 TFT 時序設定 HDW 和 VDH，使用者只要設定顯示視窗位於影像顯示記憶體左上角起始點。

設定如下：

displayWindowStartXY(0,0);

顯示視窗與當前影像顯示記憶體為子父關係，顯示視窗(子)永遠依附在當前指定影像顯示記憶體(父)。

顯示視窗的內容會由 RA8871M TFT 時序控制器輸出到 LCD 上顯示，當設定 displayOn(true)。

canvasImageStartAddress()

描述：

設定畫布影像記憶體的起始位置

函數原型：

void canvasImageStartAddress(ru32 addr);

參數	說明
addr	畫布影像記憶體的起始位置

canvasImageWidth()

描述：

設定畫布影像記憶體的寬度

函數原型：

void canvasImageWidth(ru16 width);

參數	說明
width	畫布影像記憶體的寬度

附註和範例：

圖像(Graphic) ，文字(Text) ，繪圖(Draw) ，DMA 等操作,都必須在當前畫布的活動視窗(active window)區域內執行，本文中記憶體規劃為 3 個頁，每一個頁都可以指定為當前畫布，例如：

```
//指定第 1 頁為當前畫布
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);

//指定第 2 頁為當前畫布
ra8871m.canvasImageStartAddress(PAGE2_START_ADDR);

//指定第 3 頁為當前畫布
ra8871m.canvasImageStartAddress(PAGE3_START_ADDR);
```

activeWindowXY()

描述：

設定活動視窗在畫布上的左上角起始點

函數原型：

void activeWindowXY(ru16 x0,ru16 y0);

參數	說明
x0	左上角 X 軸座標
y0	左上角 Y 軸座標

activeWindowWH()

描述：

設定活動視窗區塊寬高

函數原型：

void activeWindowWH(ru16 width,ru16 height);

參數	說明
width	活動視窗區塊寬
height	活動視窗區塊高

附註和範例：

圖像(Graphic)，文字(Text)，繪圖(Draw)，DMA 等操作，都必須在當前畫布的活動視窗(active window)區塊內執行。

活動視窗與當前畫布為子與父關係，活動視窗為(子)永遠依附在當前畫布(父) 。

活動視窗設定必須在當前畫布區域內。

第 4 章 圖像(Graphic)

函數	說明
graphicMode()	切換圖形模式或文字模式
setPixelCursor()	設定畫素游標座標
ramAccessPrepare()	記憶體存取預指令
putPixel_16bpp()	指定座標繪製一個像素點
putPicture_16bpp()	指定座標與寬高然後寫入圖像數續
putPicture_16bpp()	指定座標與繪製圖像寬高與圖像數據指針(Byte 格式)
putPicture_16bpp()	指定座標與繪製圖像寬高與圖像數據指針(Word 格式)

註：

參考 RA8871M Arduino Wire Sketch.jpg 接線圖或附錄 Figure A-1。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

graphicMode()

描述：

切換圖形模式或文字模式。

函數原型：

void graphicMode(boolean on);

參數	說明
on	= true 設定為圖像模式 = false 設定為文字模式

註：

RA8871M 初始化設定缺省為圖形模式。

setPixelCursor()

描述：

設定畫素游標座標。

函數原型：

void setPixelCursor(ru16 x,ru16 y);

參數	說明
x	X 軸座標
y	Y 軸座標

ramAccessPrepare()

描述：

記憶體存取預指令。

函數原型：

void ramAccessPrepare(void);

附註：

記憶體存取前必須調用此函數。

putPixel_16bpp()

描述：

在指定座標繪製一個像素點。

函數原型：

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

參數	說明
x	X 軸座標
y	Y 軸座標
color	RGB565 數據

附註和範例：

```
//清除當前畫布(canvas) page1 指定的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
```

```
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1,
COLOR65K_BLUE);

//在當前畫布(canvas)指定座標位置(20,20)繪製一個紅色像素點
ra8871m.setPixelCursor(20,20);
ra8871m.ramAccessPrepare();
ra8871m.lcdDataWrite(0x00);//RGB565 LSB data
ra8871m.lcdDataWrite(0xf8); //RGB565 MSB data

//在當前畫布(canvas)指定座標位置(30,20)繪製一個白色像素點
ra8871m.setPixelCursor(30,30);
ra8871m.ramAccessPrepare();
ra8871m.lcdDataWrite16bbp(COLOR65K_WHITE);//RGB565 16bpp data

//在當前畫布(canvas)指定座標位置(40,30)繪製一個紫紅色像素點
ra8871m.putPixel_16bpp(40,40,COLOR65K_MAGENTA);
```

範例顯示截圖：

putPicture_16bpp()

描述：

設定左上角起始座標與圖像寬高,設定完成後，使用者可以接續寫入圖像數據。

函數原型：

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

參數	說明
x	左上角 X 軸座標
y	左上角 Y 軸座標
width	圖像寬(水平像素尺寸)
height	圖像高(垂直像素尺寸)

putPicture_16bpp()

描述：

設定座標與圖像寬高與圖像數據指針(Byte 格式)，設定完後，函數會參考數據指針並自動寫入圖像數據到當前畫布的活動視窗內指定坐標。

函數原型：

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

參數	說明
x	左上角 X 軸座標
y	左上角 Y 軸座標
width	圖像寬(水平像素尺寸)
height	圖像高(垂直像素尺寸)
*data	Byte 格式圖像數據指針

註：

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

putPicture_16bpp()

描述：

設定座標與圖像寬高與圖像數據指針(Word 格式)，設定完後，函數會參考數據指針並自動寫入圖像數據到當前畫布的活動視窗內指定坐標。

函數原型：

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

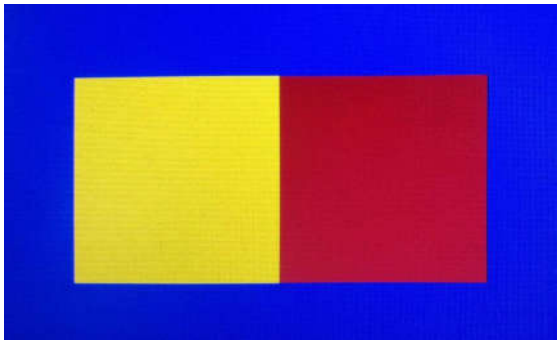
參數	說明
x	左上角 X 軸座標
y	左上角 Y 軸座標
width	圖像寬(水平像素尺寸)
height	圖像高(垂直像素尺寸)
*data	Word 格式圖像數據指針

註：
圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

附註和範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
 ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
 ra8871m.canvasImageWidth(SCREEN_WIDTH);
 ra8871m.activeWindowXY(0,0);
 ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
 ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1,
 COLOR65K_BLUE);


//在當前畫布(canvas)指定座標位置填入 128*128 圖像
ra8871m.putPicture_16bpp(50,50,128,128);
 for(i=0;i<16384;i++)
 {
 ra8871m.lcdDataWrite16bbp(COLOR65K_YELLOW);//RGB565 16bpp data
 }
ra8871m.putPicture_16bpp(50+128,50,128,128);
    for(i=0;i<16384;i++)
  {
    ra8871m.lcdDataWrite16bbp(COLOR65K_BROWN);//RGB565 16bpp data
  }
```

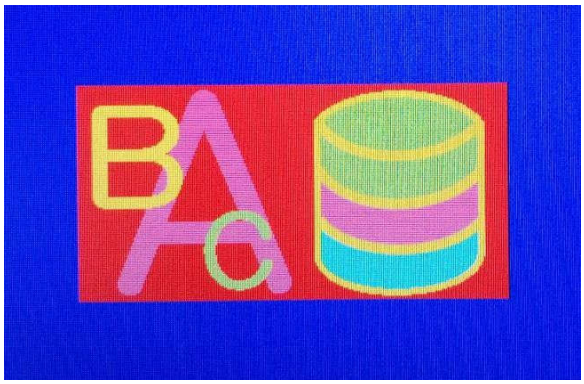
範例顯示截圖：



//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1,
COLOR65K_BLUE);

//在當前畫布(canvas)指定座標位置填入 128*128 圖像
ra8871m.putPicture_16bpp(50,50,128,128,pic16bpp_byte);
ra8871m.putPicture_16bpp(50+128,50,128,128,pic16bpp_word);

範例顯示截圖：



額外的涵數與範例

函數	說明
lcdPutChar8x12()	繪製 8x12 ASCII 字元
lcdPutString8x12()	繪製 8x12 ASCII 字串
lcdPutChar16x24()	繪製 16x24 ASCII 字元
lcdPutString16x24()	繪製 16x24 ASCII 字串
lcdPutChar32x48()	繪製 32x48 ASCII 字元
lcdPutString32x48()	繪製 32x48 ASCII 字串

註：

請參考 User_def.h 文件和設定 DEMO_ASCII_8X12 、DEMO_ASCII_16X24 、DEMO_ASCII_32X48 定義為 1 或 0 如以下，增加 8x12、16x24、32x48 尺寸字的支援。

```
#define DEMO_ASCII_8X12 1
#define DEMO_ASCII_16X24 1
#define DEMO_ASCII_32X48 1

#ifdef DEMO_ASCII_8X12
#include "ascii_table_8x12.h"
#endif

#ifdef DEMO_ASCII_16X24
#include "ascii_table_16x24.h"
#endif

#ifdef DEMO_ASCII_32X48
#include "ascii_table_32x48.h"
#endif
```

lcdPutChar8x12()
lcdPutChar16x24()
lcdPutChar32x48()

描述：

在當前畫布的活動視窗內指定座標繪製 ASCII 字元。

函數原型：

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);

參數	說明
x	左上角 X 軸座標
y	左上角 Y 軸座標
fgcolor	文字前景色
bgcolor	文字背景色
bg_transparent	= ture ： 選擇背景透明 ， =false ： 選擇背景色
code	ASCII 碼

lcdPutString8x12()
lcdPutString16x24()
lcdPutString32x48()

描述：

在當前畫布與活動視窗內指定座標位置繪製 ASCII 字串。

函數原型：

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)

參數	說明
x	左上角 X 座標
y	左上角 Y 座標
fgcolor	文字前景色
bgcolor	文字背景色
bg_transparent	= ture：選擇背景透明 ， =false：選擇背景色
*ptr	字串或數據指針

附註和範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
  ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
  ra8871m.canvasImageWidth(SCREEN_WIDTH);
  ra8871m.activeWindowXY(0,0);
  ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
  ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//在當前畫布的活動視窗內指定座標位置填入 8x12 ASCII 字串
#ifdef DEMO_ASCII_8X12
  ra8871m.lcdPutString8x12(0,0,0xFFFF,0x0000,true," !\"#$%&'()*+,-./012345678");
  ra8871m.lcdPutString8x12(0,12,0xFFFF,0x0000,true,"9:;<=>?@ABCDEFGHIJKLMNOPQ");
  ra8871m.lcdPutString8x12(0,24,0xFFFF,0x0000,true,"RSTUVWXYZ[\\]^_`abcdefghij");
  ra8871m.lcdPutString8x12(0,36,0xFFFF,0x0000,true,"klmnopqrstuvwxyz{|}~");
#endif

//在當前畫布的活動視窗內指定座標位置填入 16x24 ASCII 字串
#ifdef DEMO_ASCII_16X24
  ra8871m.lcdPutString16x24(0,78,0xFFFF,0x0000,true,"012345");
  ra8871m.lcdPutString16x24(0,102,0xFFFF,0x0000,true,"Hello!");
#endif

//在當前畫布的活動視窗內指定座標位置填入 32x48 ASCII 字串
#ifdef DEMO_ASCII_32X48
  ra8871m.lcdPutString32x48(0,140,0xFFFF,0x0000,false,"012345");
  ra8871m.lcdPutString32x48(0,190,0xFFFF,0x0000,false,"Hello!");
#endif
```

範例顯示截圖:

第 5 章 文字(Text)和數值(Value)

函數	說明
textMode()	切換文字模式或圖像模式
textColor()	設定文字前景色與背景色
setTextCursor()	設定文字游標座標
setTextParameter1()	設定文字功能參數 1
setTextParameter2()	設定文字功能參數 2
genitopCharacterRomParameter()	設定集通字庫文字功能參數
putString()	指定座標位置寫入字串
putDec()	指定座標位置寫入十進制數值
putFloat()	指定座標位置寫入浮點數數值
putHex()	指定座標位置寫入十六進制數值

註：

參考 RA8871M Arduino Wire Sketch.jpg 接線圖或附錄 Figure A-1。

textMode()

描述：

切換文字模式或圖像模式。

函數原型：

void textMode (boolean on);

參數	說明
on	= true 設定為文字模式 = false 設定為圖像模式

註：

建議進行文字功能操作時，設定為文字模式，操作完成後，設定返回圖像模式。

textColor()

描述：

設定文字前景色與背景色。

函數原型：

void textColor(ru16 foreground_color, ru16 background_color);

參數	說明
foreground_color	文字前景色
background_color	文字背景色

setTextCursor()

描述：

設定文字座標位置。

函數原型：

void setTextCursor(ru16 x, ru16 y);

參數	說明
x	X 軸座標
y	Y 軸座標

setTextParameter1()

描述：

設定文字功能參數 1。

函數原型：

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

參數	說明
source_select	RA8871M_SELECT_INTERNAL_CGROM RA8871M_SELECT_EXTERNAL_CGROM RA8871M_SELECT_USER_DEFINED
size_select	RA8871M_CHAR_HEIGHT_16 RA8871M_CHAR_HEIGHT_24

	RA8871M_CHAR_HEIGHT_32
iso_select	RA8871M_SELECT_8859_1 RA8871M_SELECT_8859_2 RA8871M_SELECT_8859_4 RA8871M_SELECT_8859_5

setTextParameter2()

描述：

設定文字功能參數 2。

函數原型：

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

參數	說明
align	RA8871M_TEXT_FULL_ALIGN_DISABLE RA8871M_TEXT_FULL_ALIGN_ENABLE 對齊全型字開啟位元
chroma_key	RA8871M_TEXT_CHROMA_KEY_DISABLE RA8871M_TEXT_CHROMA_KEY_ENABLE 文字背景色透明開啟位元
width_enlarge	RA8871M_TEXT_WIDTH_ENLARGEMENT_X1 RA8871M_TEXT_WIDTH_ENLARGEMENT_X2 RA8871M_TEXT_WIDTH_ENLARGEMENT_X3 RA8871M_TEXT_WIDTH_ENLARGEMENT_X4 文字水平放大選擇
height_enlarge	RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1 RA8871M_TEXT_HEIGHT_ENLARGEMENT_X2 RA8871M_TEXT_HEIGHT_ENLARGEMENT_X3 RA8871M_TEXT_HEIGHT_ENLARGEMENT_X4 文字垂直放大選擇

genitopCharacterRomParameter()

描述：

設定集通字庫文字功能參數。

函數原型：

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

參數	說明
scs_select	RA8871M_SERIAL_FLASH_SELECT0 RA8871M_SERIAL_FLASH_SELECT1 選擇使用 SPI0 或 SPI1
clk_div	RA8871M_SPI_DIV2 RA8871M_SPI_DIV4 RA8871M_SPI_DIV6 RA8871M_SPI_DIV8 RA8871M_SPI_DIV10 設定集通字庫用 SPI clock 除頻
rom_select	RA8871M_GT21L16T1W RA8871M_GT30L16U2W RA8871M_GT30L24T3Y RA8871M_GT30L24M1Z RA8871M_GT30L32S4W RA8871M_GT20L24F6Y RA8871M_GT21L24S1W 選擇集通字庫
character_select	RA8871M_GB2312 RA8871M_GB12345_GB18030 RA8871M_BIG5 RA8871M_ASCII RA8871M_UNICODE RA8871M_UNI_JAPANESE RA8871M_JIS0208 RA8871M_LATIN_GREEK_CYRILLIC_ARABIC_THAI_HEBREW RA8871M_ISO_8859_1_AND_ASCII RA8871M_ISO_8859_2_AND_ASCII RA8871M_ISO_8859_3_AND_ASCII RA8871M_ISO_8859_4_AND_ASCII RA8871M_ISO_8859_5_AND_ASCII

	RA8871M_ISO_8859_7_AND_ASCII RA8871M_ISO_8859_8_AND_ASCII RA8871M_ISO_8859_9_AND_ASCII RA8871M_ISO_8859_10_AND_ASCII RA8871M_ISO_8859_11_AND_ASCII RA8871M_ISO_8859_13_AND_ASCII RA8871M_ISO_8859_14_AND_ASCII RA8871M_ISO_8859_15_AND_ASCII RA8871M_ISO_8859_16_AND_ASCII 選擇字型解碼器
gt_width	RA8871M_GT_FIXED_WIDTH RA8871M_GT_VARIABLE_WIDTH_ARIAL RA8871M_GT_VARIABLE_FIXED_WIDTH_ROMAN RA8871M_GT_BOLD 選擇字體

註：

建議 serial IF0 連接到集通字庫，serial IF1 連接到一般的 serial flash。

RA8871M 支援多個集通字庫型號，詳細參考 RA8871M DataSheet。

putString()

描述：

在當前畫布的活動視窗內的指定座標位置寫入字串。

函數原型：

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

參數	說明
x0	左上角 X 軸座標
y0	左上角 Y 軸座標
*str	字串或數據指針

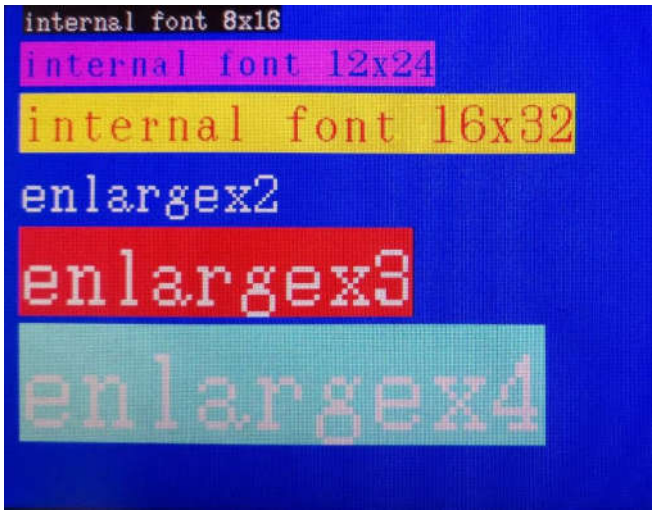
範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
  ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
  ra8871m.canvasImageWidth(SCREEN_WIDTH);
```

```
 ra8871m.activeWindowXY(0,0);
 ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
 ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

 //設定文字功能參數
 //設定文字顏色
 //在指定座標位置顯示內建字型 8x16 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_H
  EIGHT_16,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
 ra8871m.putString(10,0,"internal font 8x16");

//設定文字功能參數
//設定文字顏色
//在指定座標位置顯示內建字型 12x24 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
 IGHT_24,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8871m.textColor(COLOR65K_BLUE,COLOR65K_MAGENTA);
 ra8871m.putString(10,20,"internal font 12x24");

//設定文字功能參數
//設定文字顏色
//在指定座標位置顯示內建字型 16x32 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
 IGHT_32,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
 ra8871m.textColor(COLOR65K_RED,COLOR65K_YELLOW);
 ra8871m.putString(10,48,"internal font 16x32");
```

```
//設定文字功能參數
//設定文字顏色
//在指定座標位置顯示內建字型 2 倍放大
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_16,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,RA8871M_TEXT_CHROMA_KEY_ENABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X2,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X2);
 ra8871m.textColor(COLOR65K_WHITE,COLOR65K_RED);
 ra8871m.putString(10,84,"enlarge x2");

//設定文字功能參數
//設定文字顏色
//在指定座標位置顯示內建字型 3 倍放大
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_16,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X3,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X3);
 ra8871m.textColor(COLOR65K_WHITE,COLOR65K_RED);
 ra8871m.putString(10,120,"enlarge x3");

//設定文字功能參數
//設定文字顏色
//在指定座標位置顯示內建字型 4 倍放大
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
 IGHT_16,RA8871M_SELECT_8859_1);//cch
 ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X4,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X4);
 ra8871m.textColor(COLOR65K_WHITE, COLOR65K_LIGHTCYAN);
 ra8871m.putString(10,172,"enlarge x4");
```

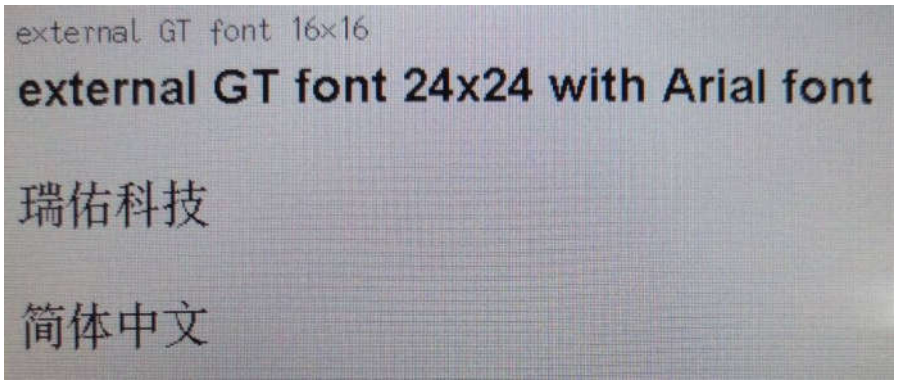
範例顯示截圖：




```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
 ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
 ra8871m.canvasImageWidth(SCREEN_WIDTH);
 ra8871m.activeWindowXY(0,0);
 ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
 ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//設定文字功能參數
//設定集通字庫參數
//設定文字顏色
//在指定座標位置顯示集通字型字串
ra8871m.setTextParameter1(RA8871M_SELECT_EXTERNAL_CGROM,RA8871M_CHAR_H
EIGHT_16,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);

ra8871m.genitopCharacterRomParameter(RA8871M_SERIAL_FLASH_SELECT0,RA8871M_
SPI_DIV4,RA8871M_GT30L24T3Y,RA8871M_BIG5,RA8871M_GT_FIXED_WIDTH);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.putString(10,10,"external GT font 16x16");

//設定文字功能參數
//設定集通字庫參數
```

```
//設定文字顏色
//在指定座標位置顯示集通字型字串
ra8871m.setTextParameter1(RA8871M_SELECT_EXTERNAL_CGROM,RA8871M_CHAR_HEIGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);

ra8871m.genitopCharacterRomParameter(RA8871M_SERIAL_FLASH_SELECT0,RA8871M_SPI_DIV4,RA8871M_GT30L24T3Y,RA8871M_BIG5,RA8871M_GT_VARIABLE_WIDTH_ARIAL);
ra8871m.putString(10,30,"external GT font 24x24 with Arial font");

ra8871m.putString(10,90,string1);

ra8871m.setTextParameter1(RA8871M_SELECT_EXTERNAL_CGROM,RA8871M_CHAR_HEIGHT_24,RA8871M_SELECT_8859_1);//cch

ra8871m.genitopCharacterRomParameter(RA8871M_SERIAL_FLASH_SELECT0,RA8871M_SPI_DIV4,RA8871M_GT30L24T3Y,RA8871M_GB2312,RA8871M_GT_FIXED_WIDTH);
ra8871m.putString(10,150,string2);
```

範例顯示截圖：



putDec()

描述：

在當前畫布的活動視窗內的指定座標位置寫入十進制數值。

函數原型：

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

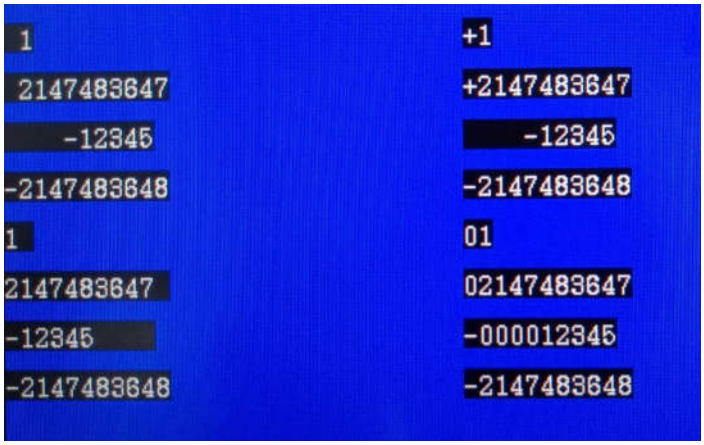
參數	說明
x0	左上角 X 軸座標
y0	左上角 Y 軸座標
vaule	輸入數值 -2147483648(-2^31) ~ 2147483647(2^31-1)
len	最小顯示位元數(1~11)
*flag	= “n” ： 顯示靠右 = “-” ： 顯示靠左 = “+” ： 輸出正負號 = “0” ： 在開頭處補 0,非補空白

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//設定文字功能參數
//設定文字顏色
//在指定座標位置繪製內建字型 16x32 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_32,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//顯示數值
ra8871m.putDec(0,10,1,2,"n");
ra8871m.putDec(0,36,2147483647,11,"n");
ra8871m.putDec(0,62,-12345,10,"n");
ra8871m.putDec(0,88,-2147483648,11,"n");
```

```
ra8871m.putDec(0,114,1,2,"-");
ra8871m.putDec(0,140,2147483647,11,"-");
ra8871m.putDec(0,166,-12345,10,"-");
ra8871m.putDec(0,192,-2147483648,11,"-");

ra8871m.putDec(SCREEN_WIDTH/2,10,1,2,"+");
ra8871m.putDec(SCREEN_WIDTH/2,36,2147483647,11,"+");
ra8871m.putDec(SCREEN_WIDTH/2,62,-12345,10,"+");
ra8871m.putDec(SCREEN_WIDTH/2,88,-2147483648,11,"+");

ra8871m.putDec(SCREEN_WIDTH/2,114,1,2,"0");
ra8871m.putDec(SCREEN_WIDTH/2,140,2147483647,11,"0");
ra8871m.putDec(SCREEN_WIDTH/2,166,-12345,10,"0");
ra8871m.putDec(SCREEN_WIDTH/2,192,-2147483648,11,"0");
```

範例顯示截圖：



putFloat()

描述：

在當前畫布的活動視窗內的指定座標位置寫入浮點數數值。

函數原型：

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

參數	說明

x0	左上角 X 軸座標
y0	左上角 Y 軸座標
vaule	輸入數值(3.4E-38 ~ 3.4E38)
len	最小顯示位元數(1~11)
precision	小數點右邊精準位元數(1~4 位)
*flag	= “n” ：顯示靠右 = “-” ：顯示靠左 = “+” ：輸出正負號 = “0” ：在開頭處補 0,非補空白

註：

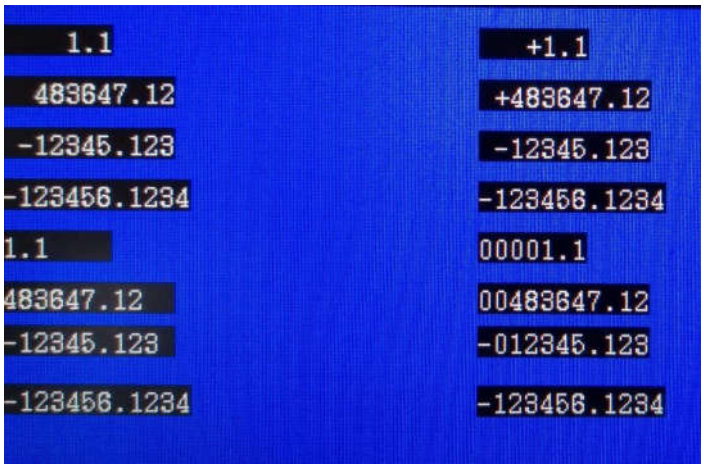
為了得到更高得精準度,這裡使用了 double。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//設定文字功能參數
//設定文字顏色
//在指定座標位置繪製內建字型 16x32 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
IGHT_32,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//顯示數值
ra8871m.putFloat(0,10,1.1,7,1,"n");
ra8871m.putFloat(0,36,483647.12,11,2,"n");
ra8871m.putFloat(0,62,-12345.123,11,3,"n");
ra8871m.putFloat(0,88,-123456.1234,11,4,"n");

ra8871m.putFloat(0,114,1.1234,7,1,"-");
```

```
ra8871m.putFloat(0,140,483647.12,11,2,"-");
ra8871m.putFloat(0,162,-12345.123,11,3,"-");
ra8871m.putFloat(0,192,-123456.1234,11,4,"-");

ra8871m.putFloat(SCREEN_WIDTH/2,10,1.1,7,1,"+");
ra8871m.putFloat(SCREEN_WIDTH/2,36,483647.12,11,2,"+");
ra8871m.putFloat(SCREEN_WIDTH/2,62,-12345.123,11,3,"+");
ra8871m.putFloat(SCREEN_WIDTH/2,88,-123456.1234,11,4,"+");

ra8871m.putFloat(SCREEN_WIDTH/2,114,1.1,7,1,"0");
ra8871m.putFloat(SCREEN_WIDTH/2,140,483647.12,11,2,"0");
ra8871m.putFloat(SCREEN_WIDTH/2,162,-12345.123,11,3,"0");
ra8871m.putFloat(SCREEN_WIDTH/2,192,-123456.1234,11,4,"0");
```

範例顯示截圖：



putHex()

描述：

在當前畫布的活動視窗內的指定座標位置寫入十六進制數值。

函數原型：

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

參數	說明
x0	左上角 X 軸座標
y0	左上角 Y 軸座標

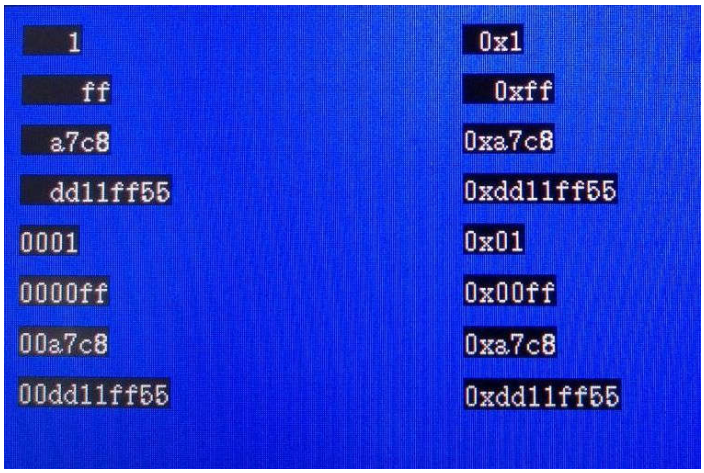
vaule	輸入數值 0x00000000~0xffffffff
len	最小顯示位元數(1~10)
*flag	= “n” ：顯示靠右 = “#” ：強制輸出 0x 作為開頭 = “0” ：在開頭處補 0,非補空白 = “x” ：強制輸出 0x 作為開頭，補 0

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//設定文字功能參數
//設定文字顏色
//在指定座標位置繪製內建字型 16x32 ASCII 字串
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
IGHT_32,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_DISABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

//顯示數值
ra8871m.putHex(10,10,1,4,"n");
ra8871m.putHex(10,36,255,6,"n");
ra8871m.putHex(10,62,0xa7c8,6,"n");
ra8871m.putHex(10,88,0xdd11ff55,10,"n");

ra8871m.putHex(10,114,1,4,"0");
ra8871m.putHex(10,140,255,6,"0");
ra8871m.putHex(10,166,0xa7c8,6,"0");
ra8871m.putHex(10,192,0xdd11ff55,10,"0");

ra8871m.putHex(SCREEN_WIDTH/2,10,1,4,"#");
```

```
ra8871m.putHex(SCREEN_WIDTH/2,36,255,6,"#");
ra8871m.putHex(SCREEN_WIDTH/2,62,0xa7c8,6,"#");
ra8871m.putHex(SCREEN_WIDTH/2,88,0xdd11ff55,10,"#");

ra8871m.putHex(SCREEN_WIDTH/2,114,1,4,"x");
ra8871m.putHex(SCREEN_WIDTH/2,140,255,6,"x");
ra8871m.putHex(SCREEN_WIDTH/2,166,0xa7c8,6,"x");
ra8871m.putHex(SCREEN_WIDTH/2,192,0xdd11ff55,10,"x");
```

範例顯示截圖：



第 6 章 幾何繪圖(Geometric Draw)

函數	說明
drawLine()	繪製線
drawSquare()	繪製矩形
drawSquareFill()	繪製矩形填滿
drawCircleSquare()	繪製圓角矩形
drawCircleSquareFill()	繪製圓角矩形填滿
drawTriangle()	繪製三角型
drawTriangleFill()	繪製三角型填滿
drawCircle()	繪製圓形
drawCircleFill()	繪製圓形填滿
drawEllipse()	繪製橢圓形
drawEllipseFill()	繪製橢圓形填滿

註：

參考 RA8871M Arduino Wire Sketch.jpg 接線圖或附錄 Figure A-1。

drawLine()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定兩點繪製線。

函數原型：

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

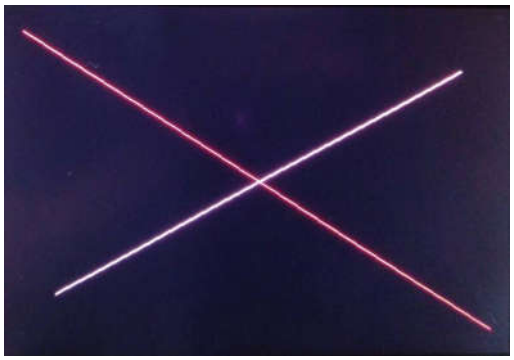
參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
color	設定顏色(RGB565)

範例：

ra8871m.drawLine(20,20,SCREEN_WIDTH-20,SCREEN_HEIGHT-20,COLOR65K_RED);

```
//
ra8871m.foreGroundColor16bpp(COLOR65K_LIGHTRED);
ra8871m.drawLine(SCREEN_WIDTH-50,50,50,SCREEN_HEIGHT-50);
```

範例顯示截圖：



drawSquare()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定兩點繪製矩形。

函數原型：

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

參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
color	設定顏色(RGB565)

範例：

```
ra8871m.drawSquare(20, 20, SCREEN_WIDTH-20, SCREEN_HEIGHT-20,
COLOR65K_GRAYSCALE23);
```

drawSquareFill()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定兩點繪製矩形填滿。

函數原型：

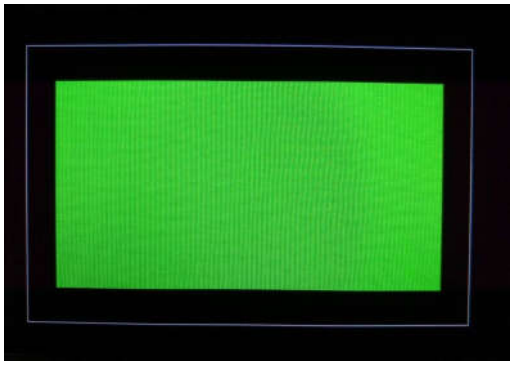
void drawSquareFill(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 color);

參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
color	設定顏色(RGB565)

範例：

ra8871m.drawSquareFill(50,50,SCREEN_WIDTH-50,SCREEN_HEIGHT-50,
COLOR65K_GREEN);

範例顯示截圖：



drawCircleSquare()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定兩點繪製圓角矩形。

函數原型：

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

參數	說明

x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
xr	圓角水平半徑
yr	圓角垂直半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawCircleSquare(20,20,SCREEN_WIDTH-20, SCREEN_HEIGHT-20, 20, 20, COLOR65K_BLUE2);

drawCircleSquareFill()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定兩點繪製圓角矩形填滿。

函數原型：

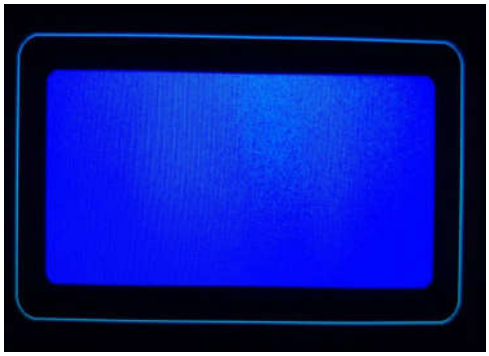
void drawCircleSquareFill(ru16 x0, ru16 y0, ru16 x1, ru16 y1, ru16 xr, ru16 yr, ru16 color);

參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
xr	圓角水平半徑
yr	圓角垂直半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawCircleSquareFill(50,50,SCREEN_WIDTH-50, SCREEN_HEIGHT-50, 10, 10, COLOR65K_BLUE);

範例顯示截圖：



drawTriangle()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定三點繪製三角型。

函數原型：

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

參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標
x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
x2	第 3 點 X 軸座標
y2	第 3 點 Y 軸座標
color	設定顏色(RGB565)

範例：

ra8871m.drawTriangle(160,20,300,200,50,220,COLOR65K_MAGENTA);

drawTriangleFill()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定三點繪製三角型填滿。

函數原型：

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

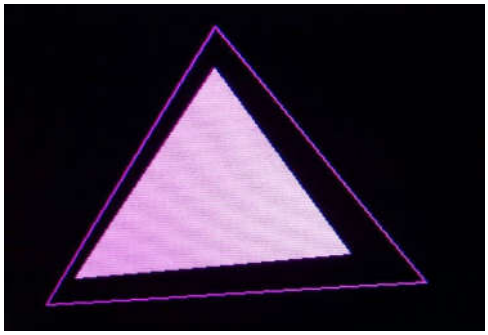
參數	說明
x0	第 1 點 X 軸座標
y0	第 1 點 Y 軸座標

x1	第 2 點 X 軸座標
y1	第 2 點 Y 軸座標
x2	第 3 點 X 軸座標
y2	第 3 點 Y 軸座標
color	設定顏色(RGB565)

範例：

ra8871m.drawTriangleFill(160,50,250,180,70,200,COLOR65K_LIGHTMAGENTA);

範例顯示截圖：



drawCircle()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定中心點繪製圓。

函數原型：

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

參數	說明
x0	中心點 X 軸座標
y0	中心點 Y 軸座標
r	半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawCircle(SCREEN_WIDTH/2,SCREEN_HEIGHT/2,80,COLOR65K_YELLOW);

drawCircleFill()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定中心點繪製圓填滿。

函數原型：

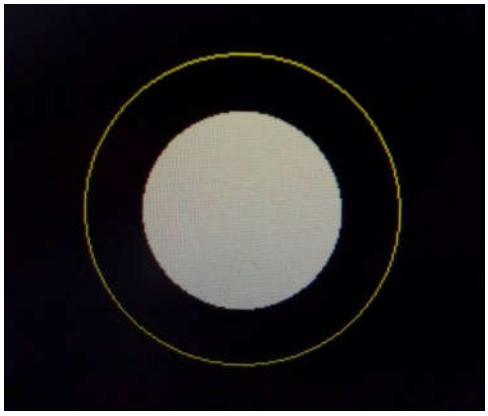
void drawCircleFill(ru16 x0,ru16 y0,ru16 r,ru16 color);

參數	說明
x0	中心點 X 軸座標
y0	中心點 Y 軸座標
r	半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawCircleFill(SCREEN_WIDTH/2,SCREEN_HEIGHT/2,50,COLOR65K_LIGHTYELLOW);

範例顯示截圖：



drawEllipse()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定中心點繪製橢圓。

函數原型：

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

參數	說明
x0	中心點 X 軸座標
y0	中心點 Y 軸座標
xr	水平半徑

yr	垂直半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawEllipse(SCREEN_WIDTH/2,SCREEN_HEIGHT/2,50,80,COLOR65K_CYAN);

drawEllipseFill()

描述：

在當前畫布(canvas)的活動視窗(active window)內的任意指定中心點繪製橢圓填滿。

函數原型：

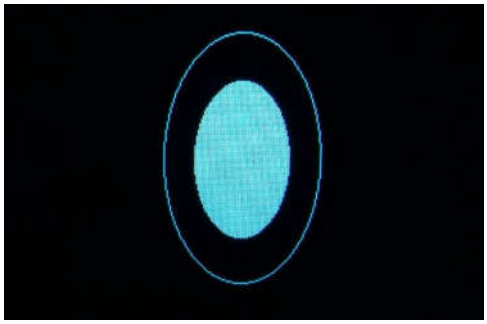
void drawEllipseFill(ru16 x0,ru16 y0,ru16 xr,ru16 yr,ru16 color);

參數	說明
x0	中心點 X 軸座標
y0	中心點 Y 軸座標
xr	水平半徑
yr	垂直半徑
color	設定顏色(RGB565)

範例：

ra8871m.drawEllipseFill(SCREEN_WIDTH/2,SCREEN_HEIGHT/2,30,50,COLOR65K_LIGHTCYAN);

範例顯示截圖：



第 7 章 BTE

Block Transfer Engine 是個 2D 加速功能引擎，提供了快速的記憶體數據傳送的複製和邏輯運算，數據的透明色忽略，單色(1bpp)數據轉換彩色數據的顏色擴充和顏色擴充與透明色，樣版圖像填充和填充與透明色等功能。

彩色顯示的數據量是巨大的，如果 MPU 寫入的速度不夠快，就可以看到顯示畫面數據更新時像瀑布一樣的掃描線，又或者需要顯示一個背景圖是靜態(如桌布) ，前景的文字或圖像是變動的動態的效果，這樣的效果就必須重新寫入背景圖後再寫入前景的文字或圖像，如果直接在當前的顯示記憶體執行，就會因為更新背景圖導致畫面出現閃動，如果不重新寫入背景圖就直接更新前景的文字或圖像，則會造成圖像的疊加，所以想要得到良好的顯示效果，就可以透過 BTE 功能的協助，使用者可以先將圖像數據透過 MPU 介面或 DMA 介面的方式寫入到非顯示區的記憶體，然後再使用 BTE memory copy 的功能將數據複製搬移到顯示區的記憶體，就可以避免上述的不良效果。

顏色擴充的功能可以把 0 或 1 的數據轉換成指定的彩色數據，由於 MPU 的內建 ROM 是有限制的，通常小於 512Kbyte，如果把 16bpp 圖像數據轉換成 1bpp 的格式存放到 MPU ROM，就可以減少 16 倍的數據量，例如使用者可能需要 64*128 尺寸的數字 0~9 用於顯示，就可以將這些數字影像轉換成 1bpp 格式數據，存放到 MPU ROM，然後透過 BTE color expansion 功能把數字影像寫入到記憶體。

樣版填充的功能讓使用者可以使用尺寸 8*8 或 16*16 的彩色圖像(16bpp)，快速的填充指定的記憶體區塊。
關於相關詳細的 BTE 功能， 請參考以下的章節的說明，或者是參考 RA8871M 的規格書。

函數	說明
bteMemoryCopy()	記憶體數據複製搬移
bteMemoryCopyWithROP()	記憶體數據複製搬移與邏輯運算
bteMemoryCopyWithChromaKey()	記憶體數據複製搬移並忽略透明色
bteMpuWriteWithROP()	MPU 寫入數據與記憶體邏輯運算(含數據指針，Byte 格式)
bteMpuWriteWithROP()	MPU 寫入數據與記憶體邏輯運算(含數據指針，Word 格式)
bteMpuWriteWithROP()	MPU 寫入數據與記憶體邏輯運算
bteMpuWriteWithChromaKey()	MPU 寫入數據並忽略透明色(含數據指針，Byte 格式)
bteMpuWriteWithChromaKey()	MPU 寫入數據並忽略透明色(含數據指

	針，Word 格式)
bteMpuWriteWithChromaKey()	MPU 寫入數據並忽略透明色
bteMpuWriteColorExpansion()	MPU 寫入數據並執行顏色擴充(含數據指針)
bteMpuWriteColorExpansion()	MPU 寫入數據並執行顏色擴充
bteMpuWriteColorExpansionWithChromaKey()	MPU 寫入數據並執行顏色擴充與忽略透明色(含數據指針)
bteMpuWriteColorExpansionWithChromaKey()	MPU 寫入數據並執行顏色擴充與忽略透明色
btePatternFill()	樣板圖像填充
btePatternFillWithChromaKey()	樣板圖像填充與忽略透明色

註：

參考 RA8871M Arduino Wire Sketch.jpg 接線圖或附錄 Figure A-1。

bteMemoryCopy()

描述：

執行畫布內或畫布與畫布間的記憶體數據複製與搬移。

函數原型：

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);

參數	說明
s0_addr	來源畫布的記憶體起始地址
s0_image_width	來源畫布的圖像記憶體寬度
s0_x	來源圖像在畫布上的 X 軸座標
s0_y	來源圖像在畫布上的 Y 軸座標
des_addr	目的地畫布的記憶體起始地址
des_image_width	目的地畫布的記憶體影像寬度
des_x	目的地圖像在畫布上的 X 軸座標
des_y	目的地圖像在畫布上的 Y 軸座標
copy_width	複製的圖像寬度
copy_height	複製的圖像高度

註：

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

Pic16bpp_word.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pic16bpp_word.h)，並且包含到程式內。

範例：

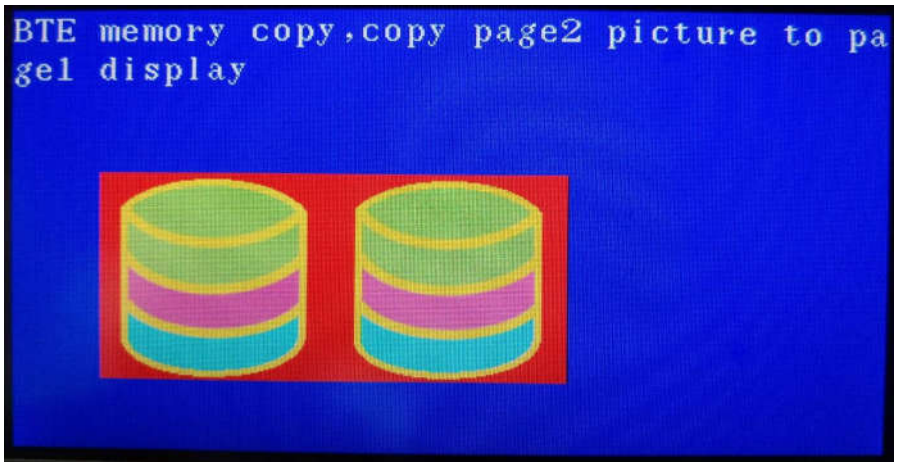
```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

//清除當前畫布(canvas) page2 的活動視窗(active window)為紅色
ra8871m.canvasImageStartAddress(PAGE2_START_ADDR);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_RED);

//寫入圖像數據至當前畫布 page2 指定位置
ra8871m.putPicture_16bpp(0,0 ,128,128,pic16bpp_word);

//寫入字串至當前畫布(canvas) page1 指定位置
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
RA8871M_TEXT_CHROMA_KEY_ENABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
```

ra8871m.putString(0,0,"BTE memory copy,copy page2 picture to page1 display");

//複製 page2 畫布(來源)的圖像數據寫入 page1 畫布(目的地)
ra8871m.bteMemoryCopy(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,PAGE1_START_ADDR,SCREEN_WIDTH, 50,100,128,128);

ra8871m.bteMemoryCopy(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,PAGE1_START_ADDR,SCREEN_WIDTH, 50+128,100,128,128);

範例顯示截圖：



bteMemoryCopyWithROP()

描述：
執行畫布內或畫布與畫布間的記憶體數據複製與邏輯運算搬移。

函數原型：
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);

參數	說明
s0_addr	來源記憶體起始地址
s0_image_width	來源記憶體影像寬度
s0_x	來源 X 軸座標位置
s0_y	來源 Y 軸座標位置

des_addr	目的地記憶體起始地址
des_image_width	目的地記憶體影像寬度
des_x	目的地 X 軸座標位置
des_y	目的地 Y 軸座標位置
copy_width	複製的圖像寬度
copy_height	複製的圖像高度
rop_code	邏輯運算選擇碼 RA8871M_BTE_ROP_CODE_0 (Blackness) RA8871M_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8871M_BTE_ROP_CODE_2 ~S0・S1 RA8871M_BTE_ROP_CODE_3 ~S0 RA8871M_BTE_ROP_CODE_4 S0・~S1 RA8871M_BTE_ROP_CODE_5 ~S1 RA8871M_BTE_ROP_CODE_6 S0^S1 RA8871M_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8871M_BTE_ROP_CODE_8 S0・S1 RA8871M_BTE_ROP_CODE_9 ~ (S0^S1) RA8871M_BTE_ROP_CODE_10 S1 RA8871M_BTE_ROP_CODE_11 ~S0+S1 RA8871M_BTE_ROP_CODE_12 S0 RA8871M_BTE_ROP_CODE_13 S0+~S1 RA8871M_BTE_ROP_CODE_14 S0+S1

	RA8871M_BTE_ROP_CODE_15 (Whiteness)

註：
圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

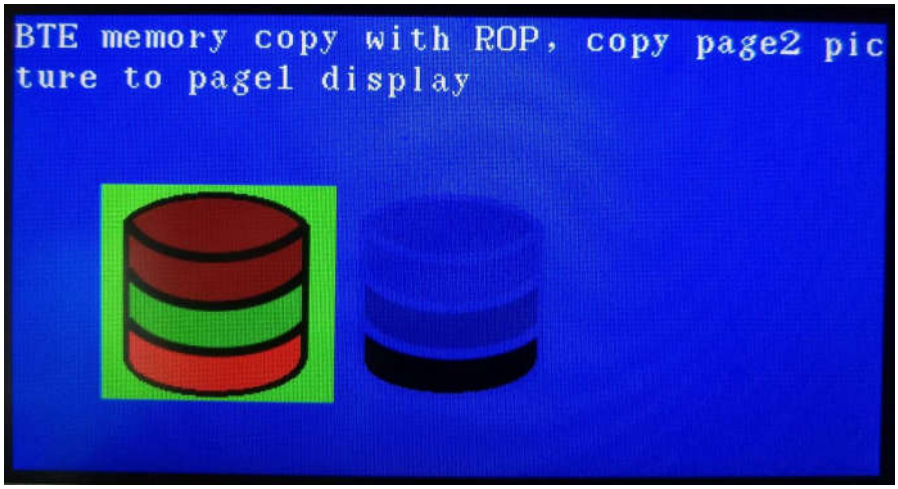
參考圖片：
Pic16bpp_word.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pic16bpp_word.h)，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.putString(0,0,"BTE memory copy with ROP, copy page2 picture to page1 display");

//複製 page2 畫布(來源)的圖像數據與 page1 畫布(目的地)數據運算後寫入 page1 畫布目的地
ra8871m.bteMemoryCopyWithROP(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,PAGE1_ST
ART_ADDR,SCREEN_WIDTH,50,100,PAGE1_START_ADDR,SCREEN_WIDTH,
50,100,128,128,RA8871M_BTE_ROP_CODE_1);

ra8871m.bteMemoryCopyWithROP(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,PAGE1_ST
ART_ADDR,SCREEN_WIDTH,(50+128),100,PAGE1_START_ADDR,SCREEN_WIDTH,
(50+128),100,128,128,RA8871M_BTE_ROP_CODE_2);
```

範例顯示截圖：



bteMemoryCopyWithChromaKey()

描述：

執行畫布內或畫布與畫布間的記憶體數據複製搬移忽略透明色。

函數原型：

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);

參數	說明
s0_addr	來源記憶體起始地址
s0_image_width	來源記憶體影像寬度
s0_x	來源 X 軸座標位置
s0_y	來源 Y 軸座標位置
des_addr	目的地記憶體起始地址
des_image_width	目的地記憶體影像寬度
des_x	目的地 X 軸座標位置
des_y	目的地 Y 軸座標位置
copy_width	複製的圖像寬度
copy_height	複製的圖像高度
chromakey_color	透明色數據

註：

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

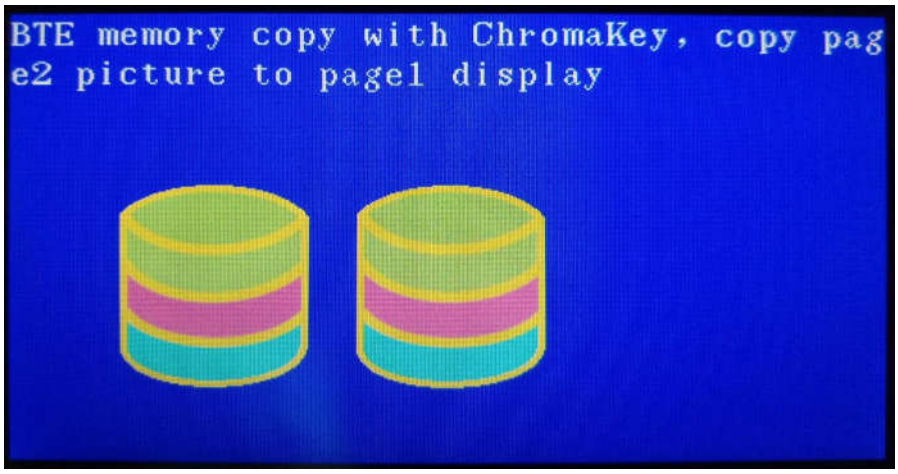
Pic16bpp_word.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pic16bpp_word.h)，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.putString(0,0,"BTE memory copy with ChromaKey, copy page2 picture to page1
display");


//複製 page2 畫布(來源)的圖像數據並忽略透明色數據寫入 page1 畫布目的地指定位置
ra8871m.bteMemoryCopyWithChromaKey(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,
PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,128,0xf800);

ra8871m.bteMemoryCopyWithChromaKey(PAGE2_START_ADDR,SCREEN_WIDTH,0,0,
PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,128,128,0xf800);
```

範例顯示截圖：



bteMpuWriteWithROP()

描述：

MPU(Source0)透過 BTE 與畫布(Source1)指定區塊執行邏輯運算後寫入目的地畫布(Destination)。

函數原型：

```
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);
```

參數	說明
s1_addr	Source1 記憶體起始地址
s1_image_width	Source1 記憶體影像寬度
s1_x	Source1 X 軸座標
s1_y	Source1 Y 軸座標
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度

des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	寫入的圖像寬度
height	寫入的圖像高度
rop_code	邏輯運算選擇碼 RA8871M_BTE_ROP_CODE_0 (Blackness) RA8871M_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8871M_BTE_ROP_CODE_2 ~S0・S1 RA8871M_BTE_ROP_CODE_3 ~S0 RA8871M_BTE_ROP_CODE_4 S0・~S1 RA8871M_BTE_ROP_CODE_5 ~S1 RA8871M_BTE_ROP_CODE_6 S0^S1 RA8871M_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8871M_BTE_ROP_CODE_8 S0・S1 RA8871M_BTE_ROP_CODE_9 ~ (S0^S1) RA8871M_BTE_ROP_CODE_10 S1 RA8871M_BTE_ROP_CODE_11 ~S0+S1 RA8871M_BTE_ROP_CODE_12 S0 RA8871M_BTE_ROP_CODE_13 S0+~S1 RA8871M_BTE_ROP_CODE_14 S0+S1 RA8871M_BTE_ROP_CODE_15 (Whiteness)

*data	數據指針(Byte 或 Word 格式)

註：

BTE 與 MPU 寫入數據相關的功能 S0(Source0) = MPU 寫入數據。

S1(Source1)設定可以與 Des(destination)相同。

無指針函數，調用後使用者可以繼續寫入影像數據。

圖像數據使用 Image_Tool_v1.1.0.10 圖像工具轉換。

參考圖片：

Pic16bpp_byte.bmp

Pic16bpp_word.bmp

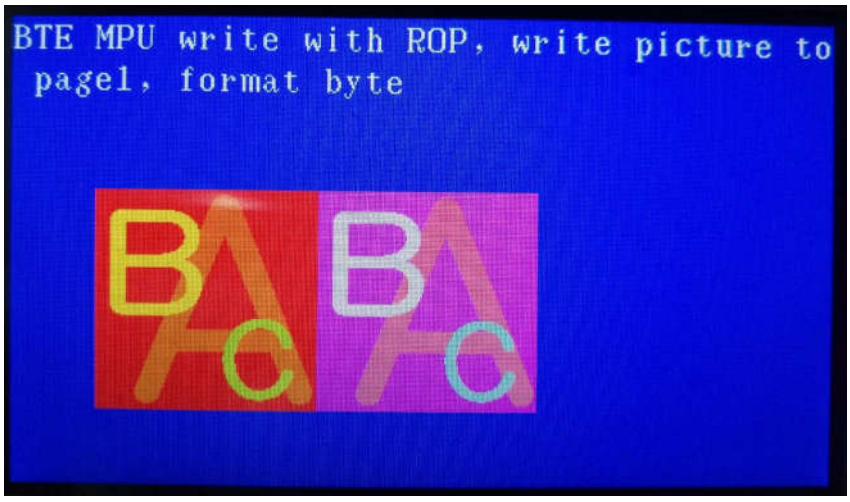
以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pic16bpp_byte.h，pic16bpp_word.h) ，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
```

```
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
RA8871M_TEXT_CHROMA_KEY_ENABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"BTE MPU write with ROP, write picture to page1, format byte");

// MPU(Source0)寫入數據透過 BTE 引擎與來源畫布(page1)指定位置數據運算
//後寫入目的地畫布(page1)
ra8871m.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50,100,PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,128,RA8871M_BTE_ROP_CODE_4,pic16bpp_byte);
ra8871m.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,128,128,RA8871M_BTE_ROP_CODE_6,pic16bpp_byte);
```

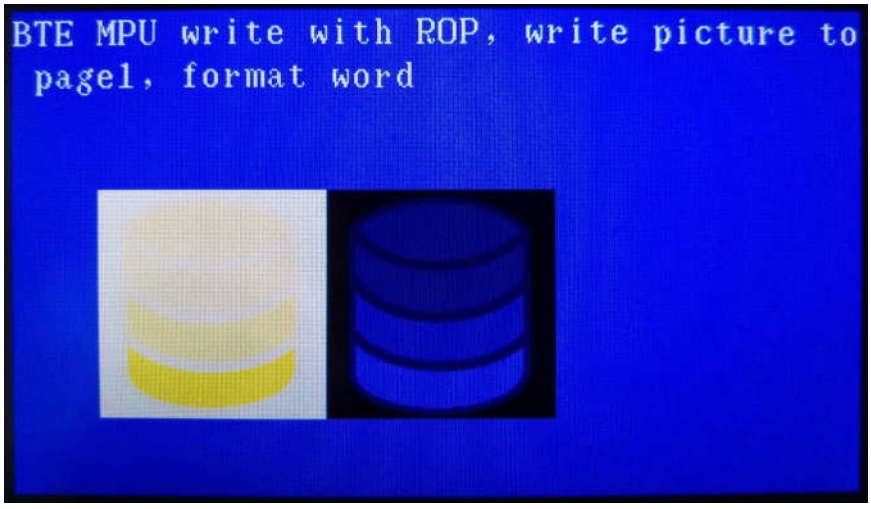
範例顯示截圖：




```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);
```

```
//寫入字串至當前畫布 page1 指定位置
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.putString(0,0,"BTE MPU write with ROP, write picture to page1, format word");

// MPU(Source0)寫入數據透過 BTE 引擎與來源畫布(page1)指定位置數據運算
//後寫入目的地畫布(page1)
ra8871m.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50,100,PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,128,RA8871M_BTE_ROP_CODE_7,pic16bpp_word);

ra8871m.bteMpuWriteWithROP(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,128,128,RA8871M_BTE_ROP_CODE_8,pic16bpp_word);
```

範例顯示截圖：



bteMpuWriteWithChromaKey()

描述：

MPU 透過 BTE 與透明色數據忽略寫入數據到目的地畫布。

函數原型：

```
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);

參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	寫入的圖像寬度
height	寫入的圖像高度
chromakey_color	透明色數據
*data	數據指針

註：

無指針函數,調用後使用者可以繼續寫入影像數據。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

Pic16bpp_byte.bmp

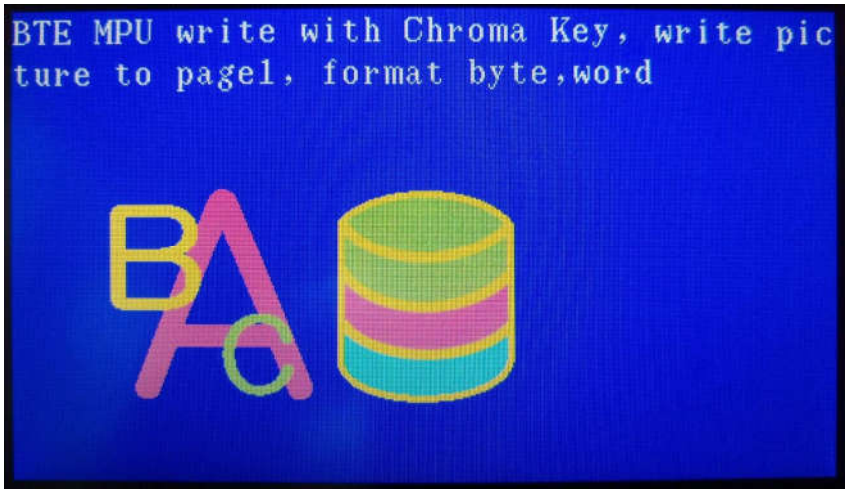
Pic16bpp_word.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pic16bpp_byte.h，pic16bpp_word.h)，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.putString(0,0,"BTE MPU write with Chroma Key, write picture to page1, format byte,word");

// MPU 寫入數據透過 BTE 透明色數據忽略後寫入目的地畫布(page1).
ra8871m.bteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,
50,100,128,128,0xf800,pic16bpp_byte);
ra8871m.bteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,
50+128,100,128,128,0xf800,pic16bpp_word);
```

範例顯示截圖：



bteMpuWriteColorExpansion()

描述：

MPU 寫入 1bpp 圖像數據透過 BTE 顏色擴展後寫入至目的地畫布指定區塊。

函數原型：

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);

參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	寫入的圖像寬度
height	寫入的圖像高度
foreground_color	指定前景色
background_color	指定背景色
*data	數據指針(Byte 格式)

註：

無指針函數,調用後使用者可以繼續寫入影像數據。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

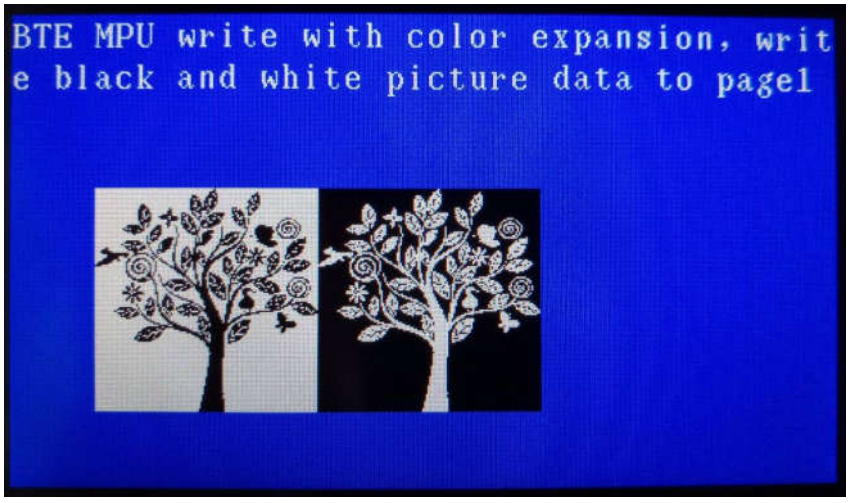
Bw.bmp

以下範例使用者必須預先轉換圖片檔為 1bpp 格式(bw.h)，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
IGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
RA8871M_TEXT_CHROMA_KEY_ENABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"BTE MPU write with color expansion, write black and white picture data
to page1");

//MPU 寫入 1bpp 圖像數據透過 BTE 執行顏色擴展後寫入至目的地畫布
ra8871m.bteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,
128,COLOR65K_BLACK,COLOR65K_WHITE,bw);
ra8871m.bteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,
128,128,COLOR65K_WHITE,COLOR65K_BLACK,bw);
```

範例顯示截圖：



bteMpuWriteColorExpansionWithChromaKey()

描述：

MPU 寫入 1bpp 圖像數據透過 BTE 顏色擴展和忽略背景色寫入至目的地畫布指定位置。

函數原型：

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);

參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	寫入的圖像寬度
height	寫入的圖像高度
foreground_color	指定轉換前景色
background_color	指定轉換背景色
*data	數據指針(Byte 格式)

註：

foreground_color 和 background_color 必須設定為不同顏色數據。

無指針函數，調用後使用者可以繼續寫入影像數據。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

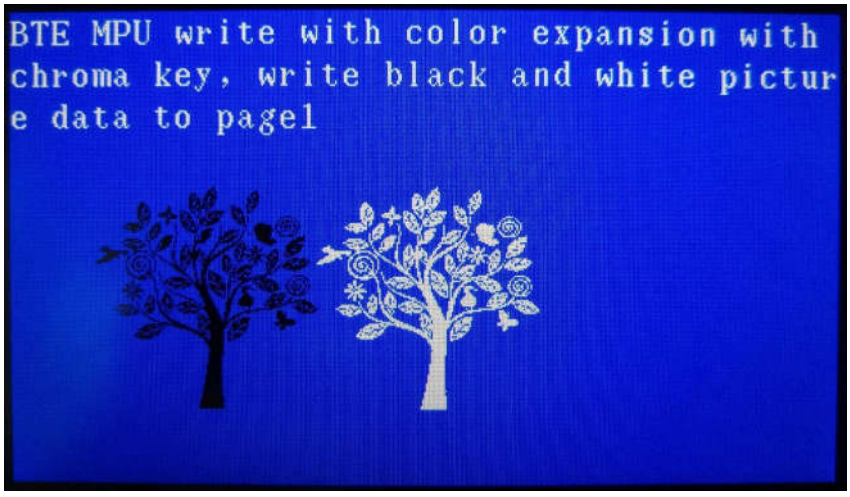
Bw.bmp

以下範例使用者必須預先轉換圖片檔為 1bpp 格式(bw.h)，並且包含到程式內。

範例：

```
//清除當前畫布(canvas) page1 的活動視窗(active window)為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//寫入字串至當前畫布 page1 指定位置
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.putString(0,0,"BTE MPU write with color expansion with chroma key, write black and white picture data to page1");

//MPU 寫入 1bpp 圖像數據透過 BTE 執行顏色擴展後忽略背景色寫入至目的地
//畫布指定位置
ra8871m.bteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,128,COLOR65K_BLACK,COLOR65K_WHITE,bw);

ra8871m.bteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,128,128,COLOR65K_WHITE,COLOR65K_BLACK,bw);
```

範例顯示截圖：



btePatternFill()

描述：

使用樣版圖像填滿指定畫布區塊。

函數原型：

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);

參數	說明
p8x8or16x16	選擇樣版尺寸, 0 = 8*8, 1=16*16
s0_addr	樣版圖像來源(畫布)記憶體起始地址
s0_image_width	樣版圖像影像(畫布)記憶體寬度
s0_x	樣版來源影像 X 軸坐標
s0_y	樣版來源影像 Y 軸坐標
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體寬度
des_x	目的地影像 X 軸起始座標
des_y	目的地影像 Y 軸起始座標
width	要填滿的區塊寬
height	要填滿的區塊高

註：

樣版圖像是預先寫入到使用者指定的記憶體位址。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

pattern6.bmp

pattern11.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(pattern6.h，pattern11.h),並且包含到程式內。

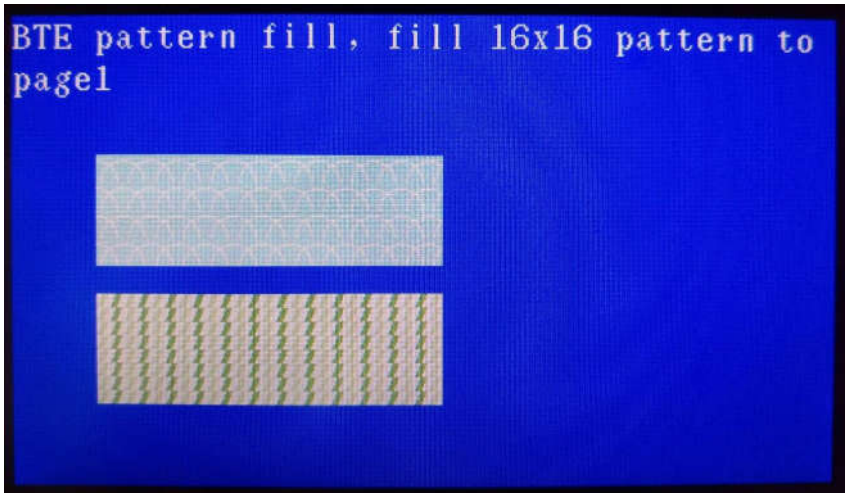
範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
```

```
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

//寫入圖片到指定的 pattern1 記憶體區塊
ra8871m.canvasImageStartAddress(PATTERN1_RAM_START_ADDR);
ra8871m.canvasImageWidth(16);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(16,16);
ra8871m.putPicture_16bpp(0,0,16,16,pattern6);

//寫入圖片到指定的 pattern2 記憶體區塊
ra8871m.canvasImageStartAddress(PATTERN2_RAM_START_ADDR);
ra8871m.putPicture_16bpp(0,0,16,16,pattern11);

//寫入圖片到指定的 pattern3 記憶體區塊
ra8871m.canvasImageStartAddress(PATTERN3_RAM_START_ADDR);
ra8871m.putPicture_16bpp(0,0,16,16,bug1);

//set canvas and active window back
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);

ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
IGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
RA8871M_TEXT_CHROMA_KEY_ENABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,
RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"BTE pattern fill, fill 16x16 pattern to page1");

ra8871m.btePatternFill(1,PATTERN1_RAM_START_ADDR,16,0,0,PAGE1_START_ADDR,SC
REEN_WIDTH, 50,80,200,64);


ra8871m.btePatternFill(1,PATTERN2_RAM_START_ADDR,16,0,0,PAGE1_START_ADDR,SC
```

REEN_WIDTH, 50,160,200,64);

範例顯示截圖：



btePatternFillWithChromaKey()

描述：

使用樣版圖像並忽略透明色填滿指定畫布區塊。

函數原型：

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);

參數	說明
p8x8or16x16	選擇樣版尺寸 0 = 8*8, 1=16*16
s0_addr	樣版圖像來源(畫布)記憶體起始地址
s0_image_width	樣版圖像影像(畫布)記憶體寬度
s0_x	樣版來源影像 X 軸坐標
s0_y	樣版來源影像 Y 軸坐標
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體寬度
des_x	目的地影像 X 軸起始座標
des_y	目的地影像 Y 軸起始座標
width	要填滿的區塊寬

height	要填滿的區塊高
chromakey_color	透明色數據

註：

樣版圖像是預先寫入到使用者指定的記憶體位址。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

參考圖片：

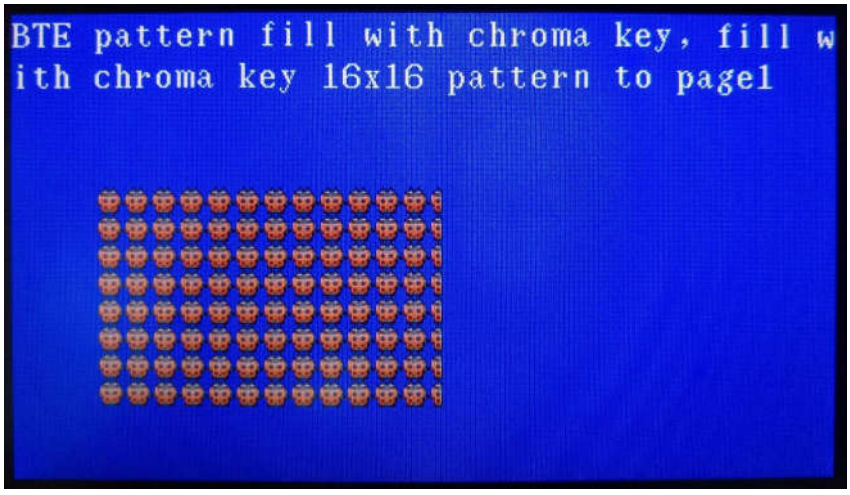
Bug1.bmp

以下範例使用者必須預先轉換圖片檔為 16bpp 格式(bug1.h)，並且包含到程式內。

範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);

ra8871m.putString(0,356,"BTE pattern fill with chroma key, fill with chroma key 16x16 pattern to page1");

ra8871m.btePatternFillWithChromaKey(1,PATTERN3_RAM_START_ADDR,16,0,0,PAGE1_START_ADDR,SCREEN_WIDTH, 50,100,200,128,0xe8e4);
```

範例顯示截圖：



第 8 章 DMA

RA8871M 提供了 DMA 功能，可以快速的讀取其外擴的 serial flash 內圖像資料，並寫入到指定的畫布區域。外擴的 serial flash 提供了讓使用者存放圖像數據的空間，彩色的圖像數據量是龐大的，低階的 MPU 內建的 ROM 通常小於 512Kbyte，只能存放少量圖型數據，低階 MPU 時脈通常低於 50MHz，如果寫入大量的數據就需要較久的時間，因此使用者可以選擇使用 DMA 功能，把圖像數據先燒錄到 serial flash，然後利用 DMA 功能執行快速的圖像存取。

函數	說明
setSerialFlash4BytesMode()	設定 serial flash 為 4Bytes 模式
dma_24bitAddressBlockMode()	DMA 讀取 24bit serial flash,區塊模式
dma_32bitAddressBlockMode()	DMA 讀取 32bit serial flash,區塊模式

註：

參考 RA8871M Arduino Wire Sketch.jpg 接線圖或附錄 Figure A-1。

Serial Flash 燒錄,參考 ArduinoDue_SpiFlashProgramWithSdCard 演示與說明。

第 8 章演示範例，使用者必須先對 Serial Flash 燒錄”**ArduinoDue_SpiFlashProgramWithSdCard**”演示專案的”**file2sdcard**”資料夾內”**All_Pic.bin**”文件。

圖像數據使用 Image_Tool_v1.1.0.1 圖像工具轉換。

setSerialFlash4BytesMode()

描述：

當使用 32bit address serial flash 時，必須先調用此函數，設定 serial flash 為 4Bytes mode。

函數原型：

void setSerialFlash4BytesMode(ru8 scs_select);

參數	說明
scs_select	選擇 serial IF0 或 serial IF1

註：

建議 serial IF0 連接到集通字庫，serial IF1 連接到 serial flash。

dma_24bitAddressBlockMode()

描述：

由指定的 serial IF 從 24bit address serial flash 讀出圖像數據，並寫入到指定的當前畫布區塊。

函數原型：

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

參數	說明
scs_selct	RA8871M_SERIAL_FLASH_SELECT0 RA8871M_SERIAL_FLASH_SELECT1 選擇 serial IF0 或 serial IF1
clk_div	RA8871M_SPI_DIV2 RA8871M_SPI_DIV4 RA8871M_SPI_DIV6 RA8871M_SPI_DIV8 RA8871M_SPI_DIV10 選擇 SPI clock 除頻
x0	當前畫布上 X 軸坐標
y0	當前畫布上 Y 軸坐標
width	DMA 區塊寬度
height	DMA 區塊高度
picture_width	Serial flash 內圖像寬度
addr	Serial flash 內圖像的起始地址

範例：

DMA 功能可以執行讀取一整個圖像數據，然後寫入到指定的當前畫布區塊。

整張圖像數據讀寫範例：

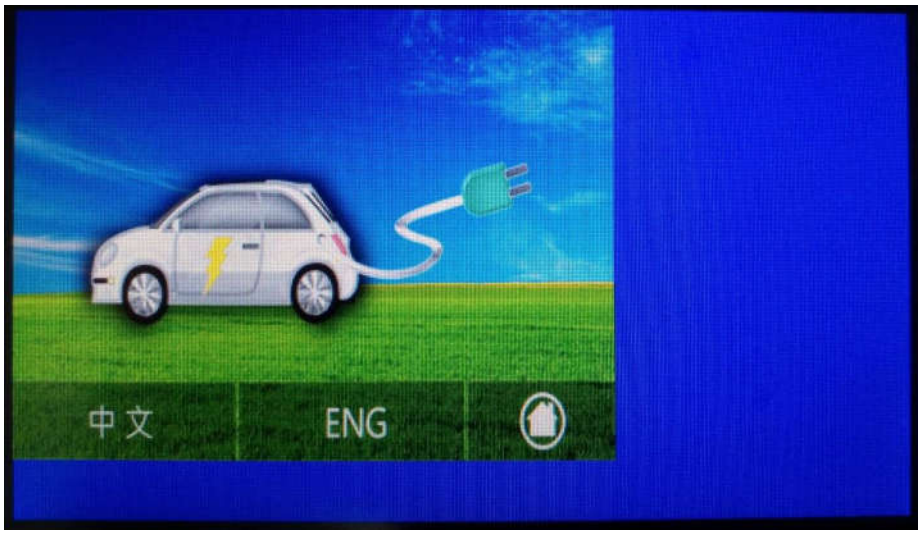
```
//設定當前畫布
//清除當前畫布(page1)的活動視窗為藍色
//DMA 讀取 Serial Flash 內圖像寫入當前畫布指定區塊

ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0,SCREEN_WIDTH-1,SCREEN_HEIGHT-1,COLOR65K_BLUE);
```

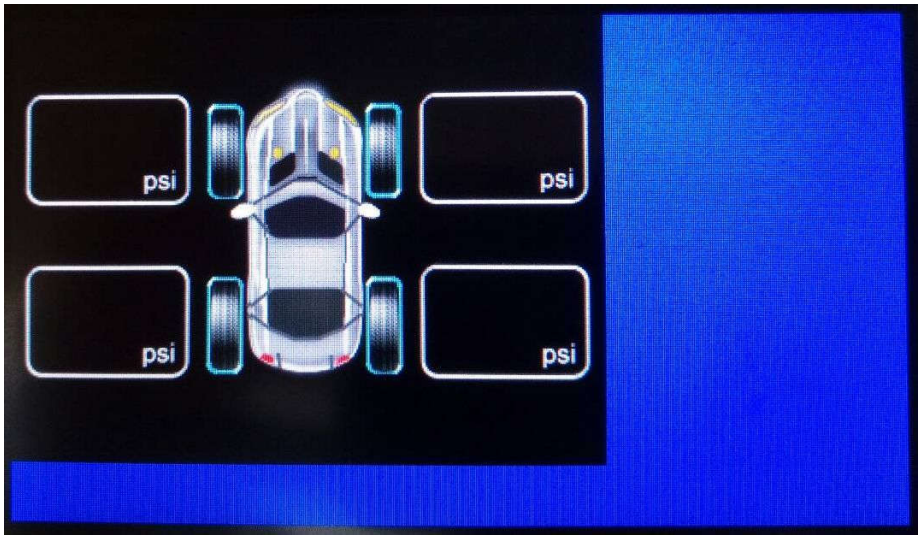
//DMA 讀取 serial flash 內圖像寫入當前畫布指定區塊
ra8871m.dma_24bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[carcharge320240].img_width,BINARY_INFO[carcharge320240].img_height,BINARY_INFO[carcharge320240].img_width,BINARY_INFO[carcharge320240].start_addr);

ra8871m.dma_24bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[refrigerator_320240].img_width,BINARY_INFO[refrigerator_320240].img_height,BINARY_INFO[refrigerator_320240].img_width,BINARY_INFO[refrigerator_320240].start_addr);

ra8871m.dma_24bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[tirepressure320240].img_width,BINARY_INFO[tirepressure320240].img_height,BINARY_INFO[tirepressure320240].img_width,BINARY_INFO[tirepressure320240].start_addr);

範例顯示截圖：





dma_32bitAddressBlockMode()

描述：

由指定的 serial IF 從 32bit address serial flash 讀出圖像數據，並寫入到指定的當前畫布區塊。

函數原型：

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

參數	說明
scs_selct	RA8871M_SERIAL_FLASH_SELECT0

	RA8871M_SERIAL_FLASH_SELECT1 選擇 serial IF0 或 serial IF1
clk_div	RA8871M_SPI_DIV2 RA8871M_SPI_DIV4 RA8871M_SPI_DIV6 RA8871M_SPI_DIV8 RA8871M_SPI_DIV10 選擇 SPI clock 除頻
x0	當前畫布上 X 軸坐標
y0	當前畫布上 Y 軸坐標
width	DMA 區塊寬度
height	DMA 區塊高度
picture_width	Serial flash 內圖像寬度
addr	Serial flash 內圖像的起始地址

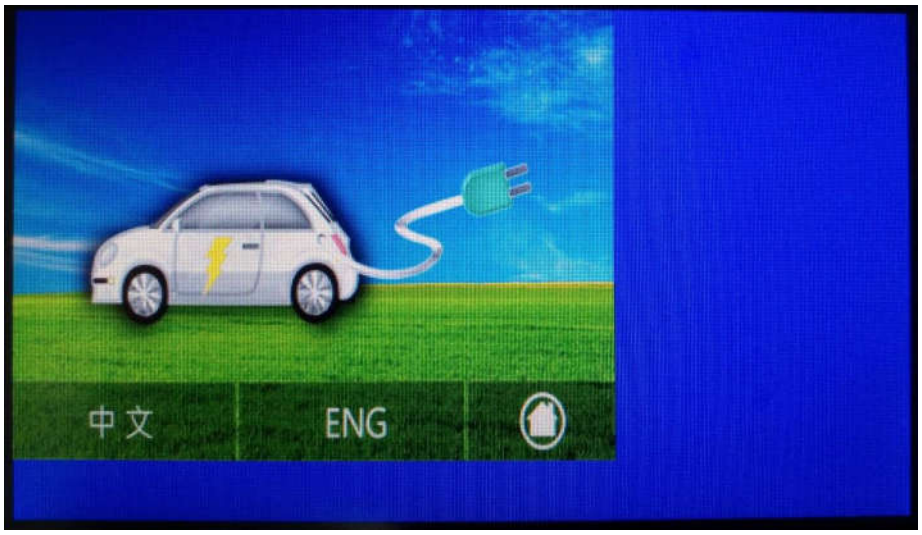
範例：

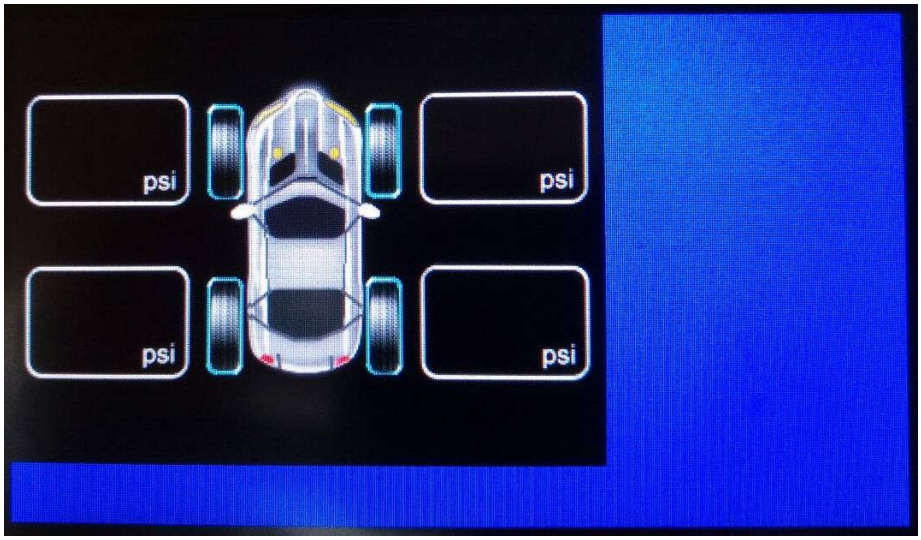
```
//DMA 32bit 位址演示
//當使用 32bit address serial flash 時，必須先設定 serial flash 為 4Bytes mode
//只需要在開電後設定一次
ra8871m.setSerialFlash4BytesMode(1);

//設定當前畫布
//清除當前畫布(page1)的活動視窗為藍色
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

//DMA 讀取 serial flash 內圖像寫入當前畫布指定區塊
ra8871m.dma_32bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[carcharge320240].img_width,BINARY_INFO[carcharge320240].img_height,BINARY_INFO[carcharge320240].img_width,BINARY_INFO[carcharge320240].start_addr);

ra8871m.dma_32bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[refrigerator_320240].img_width,BINARY_INFO[refrigerator_320240
```

].img_height,BINARY_INFO[refrigerator_320240].img_width,BINARY_INFO[refrigerator_320240].start_addr);

ra8871m.dma_32bitAddressBlockMode(RA8871M_SERIAL_FLASH_SELECT1,RA8871M_SPI_DIV2,0,0,BINARY_INFO[tirepressure320240].img_width,BINARY_INFO[tirepressure320240].img_height,BINARY_INFO[tirepressure320240].img_width,BINARY_INFO[tirepressure320240].start_addr);

範例顯示截圖：




psi
psi
psi
psi

第 9 章 PWM

函數	說明
pwm_Prescaler()	預分頻器設定
pwm_ClockMuxReg()	PWM 除頻器與 PWM 引腳的功能選擇
pwm_Configuration()	PWM 功能設定與開啟
pwm0_ClocksPerPeriod()	PWM0 每個周期時脈數量設定
pwm0_Duty()	PWM0 責任周期
pwm1_ClocksPerPeriod()	PWM1 每個周期時脈數量設定
pwm1_Duty()	PWM1 責任周期

註：參考 **RA8871M Arduino Wire Sketch.jpg** 接線圖或附錄 Figure A-1。

pwm_Prescaler()

描述：

預分頻器設定

函數原型：

void pwm_Prescaler(ru8 prescaler);

參數	說明
prescaler	RA8871M_PRESCALER

註：

PWM0 和 PWM1 的基頻 = Core_Freq(核心頻率) / (Prescaler + 1)

pwm_ClockMuxReg()

描述：

PWM 除頻器與 PWM 引腳的功能選擇

函數原型：

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

參數	說明
pwm1_clk_div	PWM1 基頻除頻設定

	RA8871M_PWM_TIMER_DIV1 RA8871M_PWM_TIMER_DIV2 RA8871M_PWM_TIMER_DIV4 RA8871M_PWM_TIMER_DIV8
pwm0_clk_div	PWM0 基頻除頻設定 RA8871M_PWM_TIMER_DIV1 RA8871M_PWM_TIMER_DIV2 RA8871M_PWM_TIMER_DIV4 RA8871M_PWM_TIMER_DIV8
xpwm1_ctrl	PWM1 引腳功能選擇 RA8871M_XPWM1_OUTPUT_ERROR_FLAG RA8871M_XPWM1_OUTPUT_PWM_TIMER1 RA8871M_XPWM1_OUTPUT_OSC_CLK
xpwm0_ctr	PWM0 引腳功能選擇 RA8871M_XPWM0_GPIO_C7 RA8871M_XPWM0_OUTPUT_PWM_TIMER0 RA8871M_XPWM0_OUTPUT_CORE_CLK

pwm_Configuration()

描述：

PWM 功能設定與開啟

函數原型：

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);

參數	說明
pwm1_inverter	PWM1 輸出反向 RA8871M_PWM_TIMER1_INVERTER_OFF RA8871M_PWM_TIMER1_INVERTER_ON
pwm1_auto_reload	PWM1 單次輸出或重複輸出 RA8871M_PWM_TIMER1_ONE_SHOT RA8871M_PWM_TIMER1_AUTO_RELOAD
pwm1_start	PWM1 停止或開啟 RA8871M_PWM_TIMER1_STOP RA8871M_PWM_TIMER1_START

pwm0_dead_zone	PWM0 死區功能選擇不啟用或啟用 RA8871M_PWM_TIMER0_DEAD_ZONE_DISABLE RA8871M_PWM_TIMER0_DEAD_ZONE_ENABLE
pwm0_inverter	PWM0 輸出反向 RA8871M_PWM_TIMER0_INVERTER_OFF RA8871M_PWM_TIMER0_INVERTER_ON
pwm0_auto_reload	PWM0 單次輸出或重複輸出 RA8871M_PWM_TIMER0_ONE_SHOT RA8871M_PWM_TIMER0_AUTO_RELOAD
pwm0_start	PWM0 停止或開啟 RA8871M_PWM_TIMER0_STOP RA8871M_PWM_TIMER0_START

pwm0_ClocksPerPeriod()
pwm1_ClocksPerPeriod()
描述：
PWM0 每個周期的時脈數量設定.
PWM1 每個周期的時脈數量設定.

函數原型：
void pwm0_ClocksPerPeriod(ru16 clocks_per_period);
void pwm1_ClocksPerPeriod(ru16 clocks_per_period);

參數	說明
clocks_per_period	每個周期時脈數量(1~65535)

註：
此設定也可以說是 PWM 分辨率的設定，例如設定值為 1000，那麼責任周期(Duty cycle)可以調整的範圍就是 0~1000。

pwm0_Duty()
pwm1_Duty()

描述：
PWM0 責任周期設定.
PWM1 責任周期設定.

函數原型：

void pwm0_Duty(ru16 duty);

void pwm1_Duty(ru16 duty);

參數	說明
duty	責任周期設定值

註：

責任周期 duty 範圍為 clocks_per_period 設定值決定。

範例：

```
//PWM 演示，請用示波器測量頻率
ra8871m.pwm_Prescaler(RA8871M_PRESCALER); //if core_freq = 100MHz, pwm base clock
                                          //= 100/(3+1) = 25MHz
ra8871m.pwm_ClockMuxReg(RA8871M_PWM_TIMER_DIV4,  RA8871M_PWM_TIMER_DIV4,
  RA8871M_XPWM1_OUTPUT_PWM_TIMER1,RA8871M_XPWM0_OUTPUT_PWM_TIMER0
  );
//pwm timer clock = 25/4 = 6.25MHz

ra8871m.pwm0_ClocksPerPeriod(1024); // pwm0 = 6.25MHz/1024 = 6.1KHz
ra8871m.pwm0_Duty(10);//pwm0 set 10/1024 duty

ra8871m.pwm1_ClocksPerPeriod(256);    // pwm1 = 7.5MHz/256 = 24.4KHz
ra8871m.pwm1_Duty(5); //pwm1 set 5/256 duty

ra8871m.pwm_Configuration(RA8871M_PWM_TIMER1_INVERTER_ON,RA8871M_PWM_TI
  MER1_AUTO_RELOAD,RA8871M_PWM_TIMER1_START,RA8871M_PWM_TIMER0_DEA
  D_ZONE_DISABLE ,RA8871M_PWM_TIMER0_INVERTER_ON,RA8871M_PWM_TIMER0_
  AUTO_RELOAD,RA8871M_PWM_TIMER0_START);
```

第 10 章 Arduino SD

使用 Arduino due 連接 SD 卡，使用者可以利用 SD 卡做為圖片數據的儲存器，把已經轉換過的圖像數據(.bin)透過 PC 存放到 SD，然後使用 Arduino Due 讀取 SD 卡內的圖像數據填入到 RA8871M 的記憶體。

函數	說明
sdCardShowPicture16bpp()	讀取 SD 卡內指定檔名的圖像數據並寫入至當前畫布指定位置
sdCardShowPicture16bppBteMpuWriteWithROP()	讀取 SD 卡內指定檔名的圖像數據並透過 BTE 寫與邏輯運算寫入至目的地畫布指定位置
sdCardShowPicture16bppBteMpuWriteWithChromaKey()	讀取 SD 卡內指定檔名的圖像數據透過 BTE 透明色忽略寫入至目的地畫布指定位置
sdCardShowPicture16bppBteMpuWriteColorExpansion()	讀取 SD 卡內指定檔名的 1bpp 圖像數據透過 BTE MPU 寫入與顏色擴展至目的地畫布指定位置
sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey()	讀取 SD 卡內指定檔名的 1bpp 圖像數據透過 BTE 執行顏色擴展與透明色忽略寫入至目的地畫布指定位置

註：

這些副程式是額外提供的，並不包含在 Ra8871m_Lite.cpp 內，如需要使用，參考 Ra8871m_Lite_Arduino_SD.ino，複製需要使用的函數到自己的專案內。

Arduino and SD card and RA8871M 接線圖，參考 RA8871MArduinoDueSD Wire Sketch.jpg 或附錄 Figure A-2。

圖像數據使用 Image_Tool_v1.1.0.10 圖像工具轉換。

sdCardShowPicture16bpp()

描述：

讀取 SD 卡內指定檔名內圖像數據，並寫入至當前畫布指定位置。

函數原型：

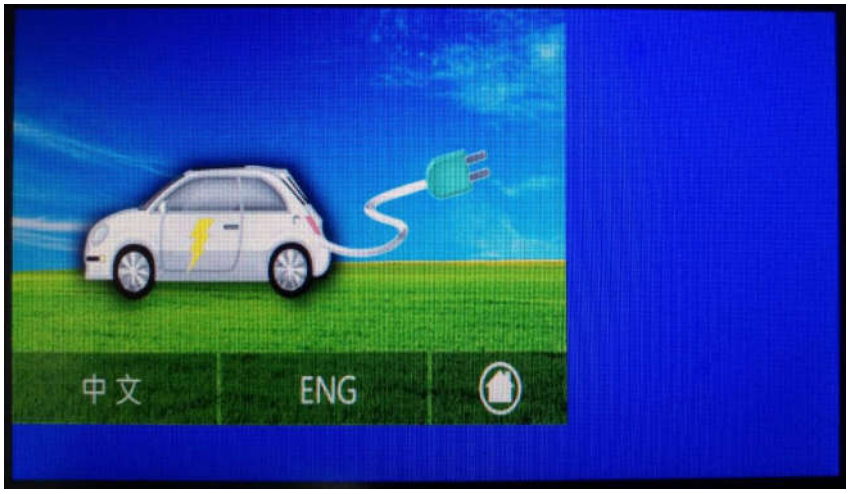
```
void sdCardShowPicture16bpp(unsigned short x, unsigned short y, unsigned short width,
  unsigned short height, char *filename);
```

參數	說明
x	X 軸起始座標
y	Y 軸起始座標
width	圖像寬度
height	圖像高度
*filename	圖像檔名

範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

sdCardShowPicture16bpp(0,0,320,240,"carc.bin");
```

範例顯示截圖：



sdCardShowPicture16bppBteMpuWriteWithROP()

描述：

讀取 SD 卡內指定檔名的圖像數據，並透過 BTE 寫與邏輯運算寫入至目的地畫布指定位置。

函數原型：

```
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);

參數	說明
s1_addr	Source1 記憶體起始地址
s1_image_width	Source1 記憶體影像寬度
s1_x	Source1 X 軸座標
s1_y	Source1 Y 軸座標
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	圖像寬度
height	圖像高度
rop_code	邏輯運算選擇碼 RA8871M_BTE_ROP_CODE_0 (Blackness) RA8871M_BTE_ROP_CODE_1 ~S0・~S1 or ~ (S0+S1) RA8871M_BTE_ROP_CODE_2 ~S0・S1 RA8871M_BTE_ROP_CODE_3 ~S0 RA8871M_BTE_ROP_CODE_4 S0・~S1 RA8871M_BTE_ROP_CODE_5 ~S1 RA8871M_BTE_ROP_CODE_6 S0^S1 RA8871M_BTE_ROP_CODE_7 ~S0+~S1 or ~ (S0・S1) RA8871M_BTE_ROP_CODE_8 S0・S1 RA8871M_BTE_ROP_CODE_9 ~ (S0^S1) RA8871M_BTE_ROP_CODE_10

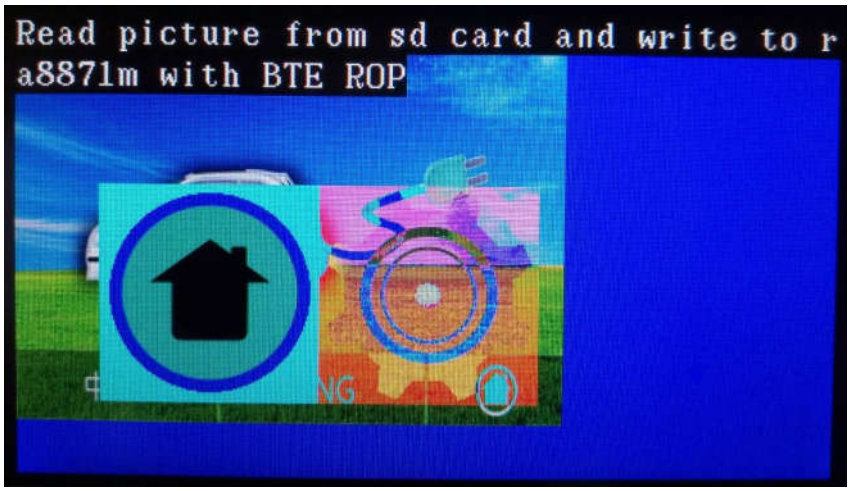
	S1 RA8871M_BTE_ROP_CODE_11 ~S0+S1 RA8871M_BTE_ROP_CODE_12 S0 RA8871M_BTE_ROP_CODE_13 S0+~S1 RA8871M_BTE_ROP_CODE_14 S0+S1 RA8871M_BTE_ROP_CODE_15 (Whiteness)
*filename	圖像檔名

註：
BTE 與 MPU 寫入數據相關的功能 S0(Source0) = MPU 寫入數據。
S1(Source1)設定可以與 Des(destination)相同。

範例：

```
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);

ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
 IGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
 RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X
 1,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"Read picture from sd card and write to ra8871m with BTE ROP");

sdCardShowPicture16bppBteMpuWriteWithROP(PAGE1_START_ADDR, SCREEN_WIDTH,
 50,100,PAGE1_START_ADDR,SCREEN_WIDTH,50,100,128,128,RA8871M_BTE_ROP_CO
 DE_3,"home.bin");
sdCardShowPicture16bppBteMpuWriteWithROP(PAGE1_START_ADDR, SCREEN_WIDTH,
 50+128,100,PAGE1_START_ADDR,SCREEN_WIDTH,50+128,100,128,128,RA8871M_BTE
 _ROP_CODE_6,"appli.bin");
```

範例顯示截圖：



sdCardShowPicture16bppBteMpuWriteWithChromaKey()

描述：

讀取 SD 卡內指定檔名的圖像數據，透過 BTE 透明色忽略寫入至目的地畫布指定位置。

函數原型：

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);

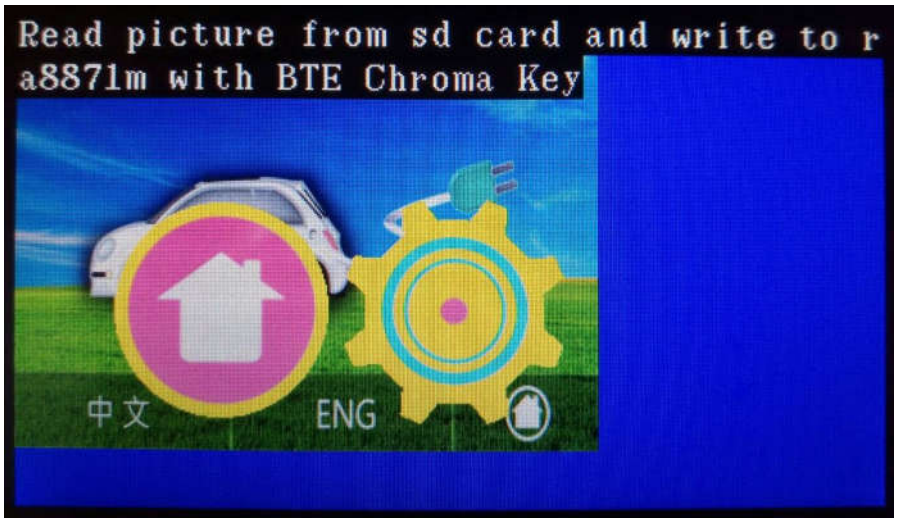
參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	圖像寬度
height	圖像高度
chromakey_color	透明色數據
* filename	圖像檔名

範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
```

```
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

sdCardShowPicture16bpp(0,0,320,240,"carc.bin");

ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
  IGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
  RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X
  1,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"Read picture from sd card and write to ra8871m with BTE Chroma
  Key");

sdCardShowPicture16bppBteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WI
  DTH,50,100,128,128,0xf800,"home.bin");
sdCardShowPicture16bppBteMpuWriteWithChromaKey(PAGE1_START_ADDR,SCREEN_WI
  DTH,50+128,100,128,128,0xf800,"appli.bin");
```

範例顯示截圖：



sdCardShowPicture16bppBteMpuWriteColorExpansion()

描述：

讀取 SD 卡內指定檔名的 1bpp 圖像數據，透過 BTE MPU 寫入與顏色擴展至目的地畫布指定位置。

函數原型：

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);

參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標
des_y	目的地 Y 軸座標
width	圖像寬度
height	圖像高度
foreground_color	前景色
background_color	背景色
* filename	圖像檔名

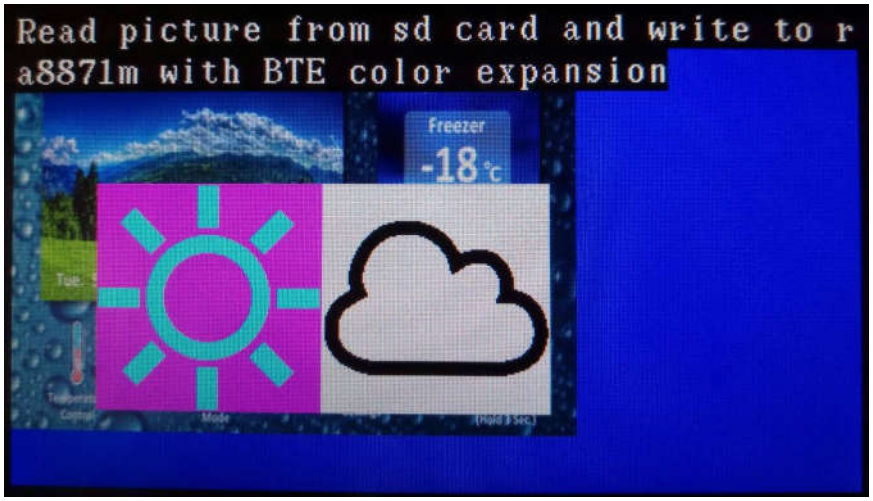
範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

sdCardShowPicture16bpp(0,0,320,240,"refri.bin");

ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HEIGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X1,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"Read picture from sd card and write to ra8871m with BTE color
```

expansion");

sdCardShowPicture16bppBteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50,100, 128, 128,COLOR65K_CYAN,COLOR65K_MAGENTA,"sun.bin");
sdCardShowPicture16bppBteMpuWriteColorExpansion(PAGE1_START_ADDR,SCREEN_WIDTH,50+128, 100, 128, 128,COLOR65K_BLACK,COLOR65K_WHITE,"cloud.bin");

範例顯示截圖：



sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey()

描述：

讀取 SD 卡內指定檔名的 1bpp 圖像數據，透過 BTE 執行顏色擴展與透明色忽略寫入至目的地畫布指定位置。

函數原型：

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);

參數	說明
des_addr	目的地(畫布)記憶體起始地址
des_image_width	目的地(畫布)記憶體影像寬度
des_x	目的地 X 軸座標

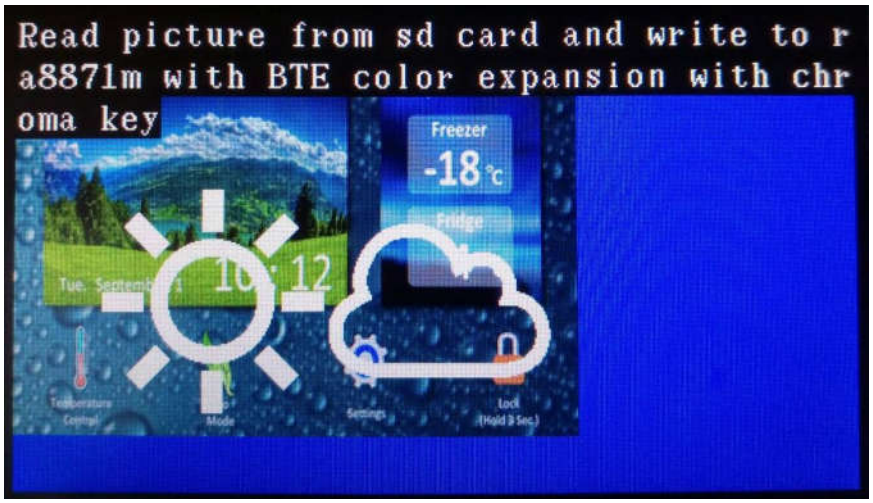
des_y	目的地 Y 軸座標
width	圖像寬度
height	圖像高度
foreground_color	前景色
background_color	背景色
* filename	圖像檔名

foreground_color 和 background_color 必須設定為不同顏色數據。

範例：

```
ra8871m.canvasImageStartAddress(PAGE1_START_ADDR);
ra8871m.canvasImageWidth(SCREEN_WIDTH);
ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.drawSquareFill(0, 0, SCREEN_WIDTH-1, SCREEN_HEIGHT-1, COLOR65K_BLUE);

sdCardShowPicture16bpp(0,0,320,240,"refri.bin");

ra8871m.activeWindowXY(0,0);
ra8871m.activeWindowWH(SCREEN_WIDTH,SCREEN_HEIGHT);
ra8871m.textColor(COLOR65K_WHITE,COLOR65K_BLACK);
ra8871m.setTextParameter1(RA8871M_SELECT_INTERNAL_CGROM,RA8871M_CHAR_HE
 IGHT_24,RA8871M_SELECT_8859_1);//cch
ra8871m.setTextParameter2(RA8871M_TEXT_FULL_ALIGN_ENABLE,
 RA8871M_TEXT_CHROMA_KEY_DISABLE,RA8871M_TEXT_WIDTH_ENLARGEMENT_X
 1,RA8871M_TEXT_HEIGHT_ENLARGEMENT_X1);
ra8871m.putString(0,0,"Read picture from sd card and write to ra8871m with BTE color
 expansion with chroma key");

sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADD
 R,SCREEN_WIDTH, 50, 100, 128, 128,COLOR65K_WHITE,COLOR65K_BLACK,"sun.bin");
sdCardShowPicture16bppBteMpuWriteColorExpansionWithChromaKey(PAGE1_START_ADD
 R,SCREEN_WIDTH, 50+128, 100, 128,
 128,COLOR65K_WHITE,COLOR65K_BLACK,"cloud.bin");
```

範例顯示截圖：


Read picture from sd card and write to r
a8871m with BTE color expansion with chr
oma key
Freezer
-18℃
Fridge
Mode
Settings

附錄 A

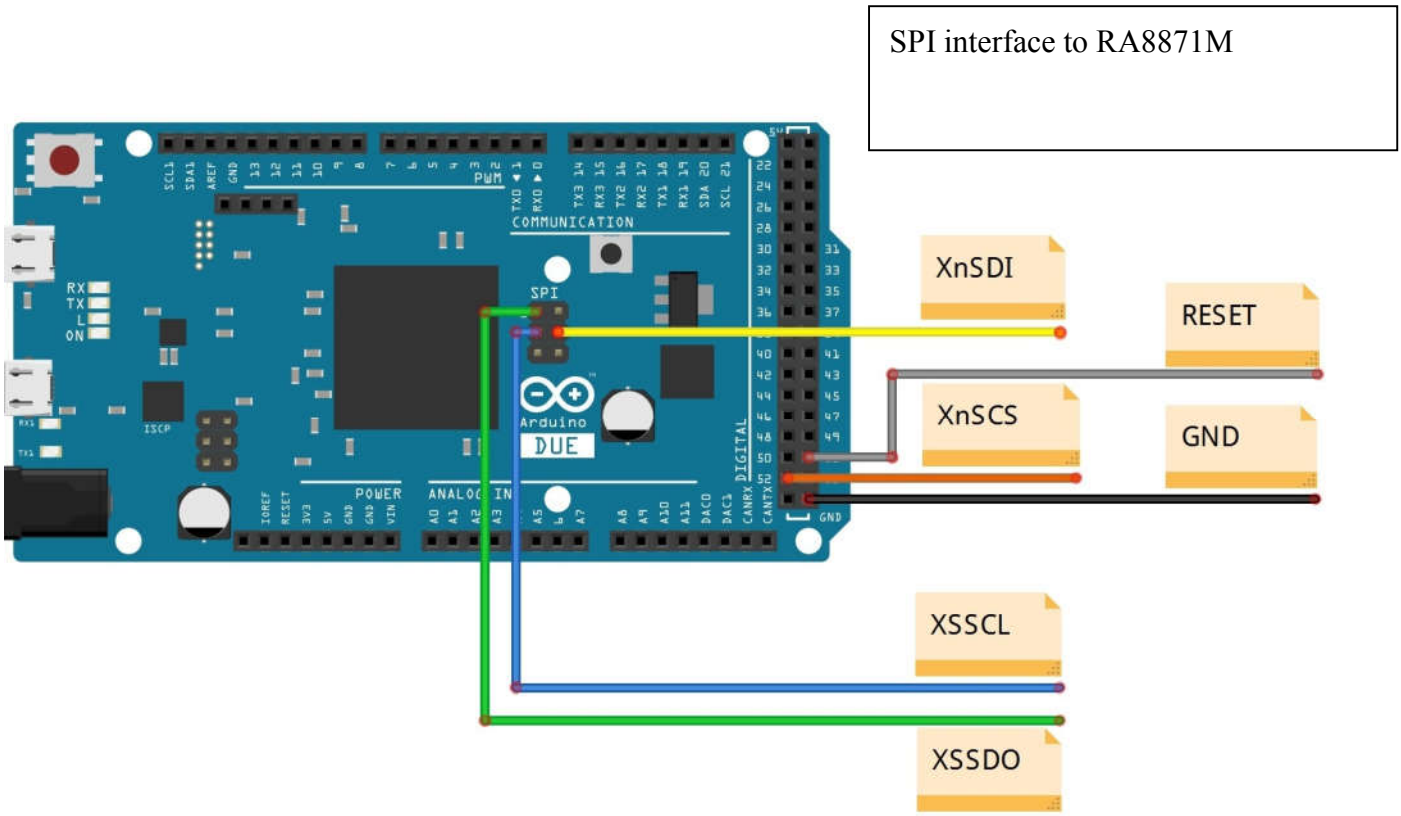


Figure A-1

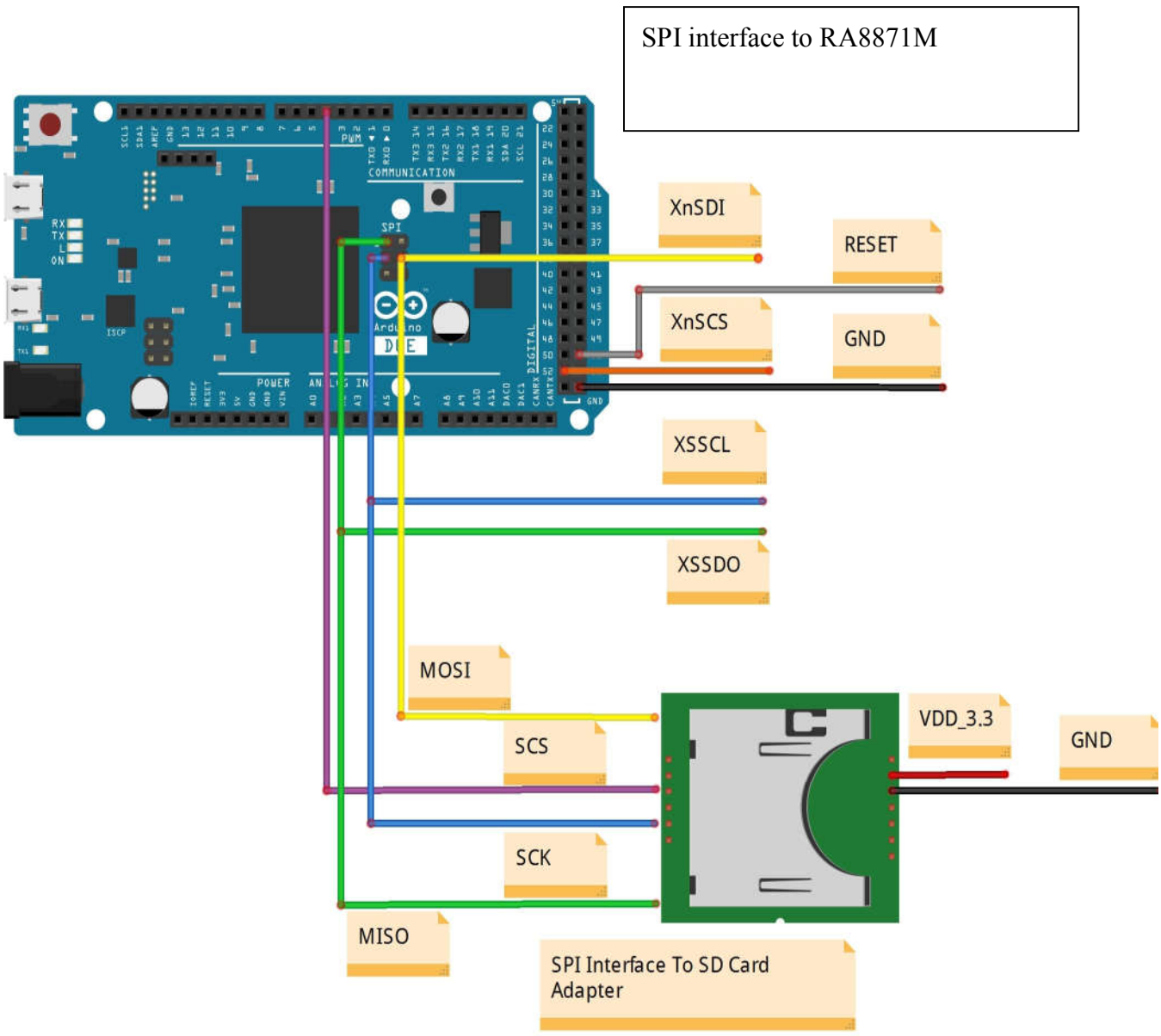


Figure A-2

全文完