Test CODE Example

1、Display with Arduino：

Now we will show how to use the Arduino (ATMega 168) to control the TFT LCD module.
To have a quicker refresh rate, we use the 16bit mode for LCD, it's two times faster than 8 bit mode.

Connect the pins to Arduino first:

DB0-DB16 to pin D0-D13, pin A0-A1 of Arduino
RESET to A2
CS to A3
WR to A4
RS to A5

All the data pins of Arduino is used ….So, we just can connect the RD to 3.3V ….We don't used it because we don't need to read back the data from the screen.
Connect the power pins: LCD-A to 3.3V, VCC to 5V , GND to GND
Note that the LCD is use the 3.3V voltage level , so if you want to connect the 5V Arduino pins to it , you need add a resister about 20K to reduce voltage. We use the 3.3V Arduino Pro which work in 8M , so we connect the pins directly — this is why we used 16 bit mode, 8 bit mode is too slow to refresh the screen.

Download the Demo code below into the controller ：

```
define LCD_RS 19
#define LCD_REST 16
#define LCD_WR 18
#define LCD_CS 17

void main_Write_COM(int DH)
{
  unsigned char i;
  int temp;
  digitalWrite(LCD_RS,LOW);
  digitalWrite(LCD_CS,LOW);
  for(i=0;i<16;i++)
  {
    temp=(DH&0x01);
    if(temp)
      digitalWrite(i,HIGH);
```

```
        else
            digitalWrite(i,LOW);
        DH=DH>>1;
    }
    digitalWrite(LCD_WR,LOW);
    digitalWrite(LCD_WR,HIGH);
    digitalWrite(LCD_CS,HIGH);
}

void main_Write_DATA(int DH)
{
    unsigned char i;
    int temp;
    digitalWrite(LCD_RS,HIGH);
    digitalWrite(LCD_CS,LOW);
    for(i=0;i<16;i++)
    {
        temp=(DH&0x01);
        if(temp)
            digitalWrite(i,HIGH);
        else
            digitalWrite(i,LOW);
        DH=DH>>1;
    }
    digitalWrite(LCD_WR,LOW);
    digitalWrite(LCD_WR,HIGH);
    digitalWrite(LCD_CS,HIGH);
}

void main_W_com_data(int com1,int dat1)
{
    main_Write_COM(com1);
    main_Write_DATA(dat1);
}

void address_set(unsigned int x1,unsigned int y1,unsigned int x2,unsigned int y2)
{
    main_W_com_data(0x0002,x1>>8);          // Column address start2
    main_W_com_data(0x0003,x1);         // Column address start1
    main_W_com_data(0x0004,x2>>8);          // Column address end2
    main_W_com_data(0x0005,x2);         // Column address end1
    main_W_com_data(0x0006,y1>>8);          // Row address start2
    main_W_com_data(0x0007,y1);         // Row address start1
    main_W_com_data(0x0008,y2>>8);          // Row address end2
```

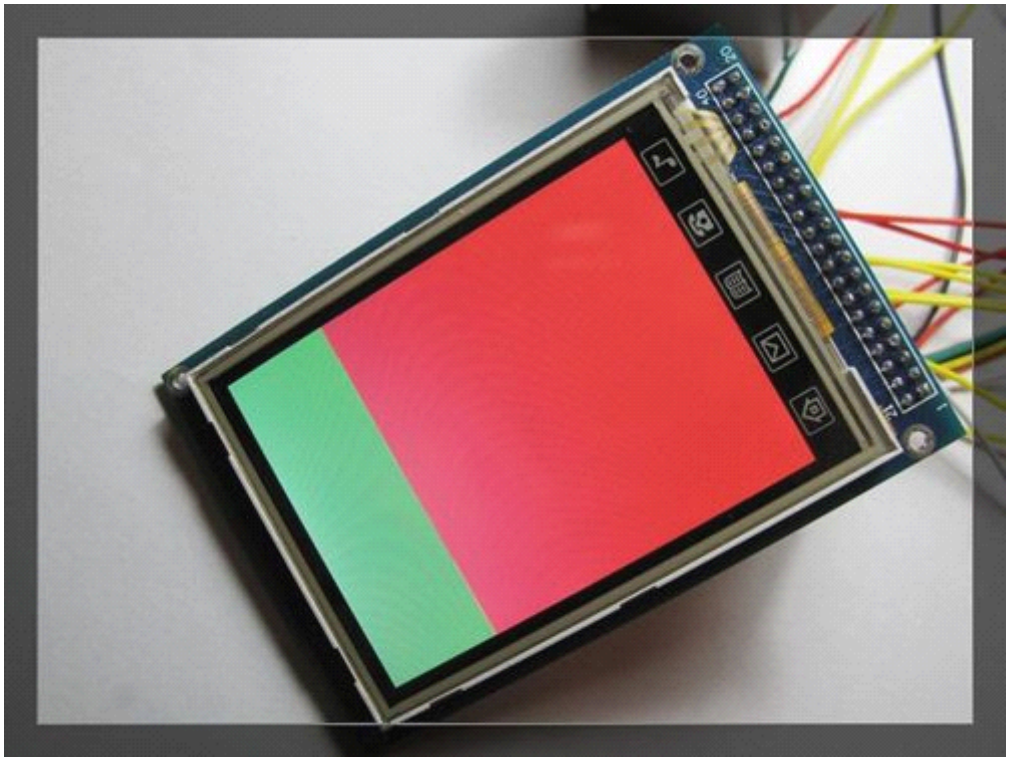
```
   main_W_com_data(0x0009,y2);        // Row address end1
   main_Write_COM(0x0022);

}

void main_init(void)
{

   digitalWrite(LCD_REST,HIGH);
   delay(5);
   digitalWrite(LCD_REST,LOW);
   delay(10);
   digitalWrite(LCD_REST,HIGH);
   delay(20);

   //    VENDOR
   main_W_com_data(0x0046,0x00A4);
   main_W_com_data(0x0047,0x0053);
   main_W_com_data(0x0048,0x0000);
   main_W_com_data(0x0049,0x0044);
   main_W_com_data(0x004a,0x0004);
   main_W_com_data(0x004b,0x0067);
   main_W_com_data(0x004c,0x0033);
   main_W_com_data(0x004d,0x0077);
   main_W_com_data(0x004e,0x0012);
   main_W_com_data(0x004f,0x004C);
   main_W_com_data(0x0050,0x0046);
   main_W_com_data(0x0051,0x0044);

   //240x320 window setting
   main_W_com_data(0x0002,0x0000); // Column address start2
   main_W_com_data(0x0003,0x0000); // Column address start1
   main_W_com_data(0x0004,0x0000); // Column address end2
   main_W_com_data(0x0005,0x00ef); // Column address end1
   main_W_com_data(0x0006,0x0000); // Row address start2
   main_W_com_data(0x0007,0x0000); // Row address start1
   main_W_com_data(0x0008,0x0001); // Row address end2
   main_W_com_data(0x0009,0x003f); // Row address end1

   // Display Setting
   main_W_com_data(0x0001,0x0006); // IDMON=0, INVON=1, NORON=1, PTLON=0
   main_W_com_data(0x0016,0x00C8);  //  MY=0,  MX=0,  MV=0,  ML=1,  BGR=0,  TEON=0
0048
   main_W_com_data(0x0023,0x0095); // N_DC=1001 0101
```

```
   main_W_com_data(0x0024,0x0095); // PI_DC=1001 0101
   main_W_com_data(0x0025,0x00FF); // I_DC=1111 1111

   main_W_com_data(0x0027,0x0002); // N_BP=0000 0010
   main_W_com_data(0x0028,0x0002); // N_FP=0000 0010
   main_W_com_data(0x0029,0x0002); // PI_BP=0000 0010
   main_W_com_data(0x002a,0x0002); // PI_FP=0000 0010
   main_W_com_data(0x002C,0x0002); // I_BP=0000 0010
   main_W_com_data(0x002d,0x0002); // I_FP=0000 0010

   main_W_com_data(0x003a,0x0001); // N_RTN=0000, N_NW=001        0001
   main_W_com_data(0x003b,0x0000); // P_RTN=0000, P_NW=001
   main_W_com_data(0x003c,0x00f0); // I_RTN=1111, I_NW=000
   main_W_com_data(0x003d,0x0000); // DIV=00
   delay(1);
   main_W_com_data(0x0035,0x0038); // EQS=38h
   main_W_com_data(0x0036,0x0078); // EQP=78h
   main_W_com_data(0x003E,0x0038); // SON=38h
   main_W_com_data(0x0040,0x000F); // GDON=0Fh
   main_W_com_data(0x0041,0x00F0); // GDOFF

   // Power Supply Setting
   main_W_com_data(0x0019,0x0049); // CADJ=0100, CUADJ=100, OSD_EN=1 ,60Hz
   main_W_com_data(0x0093,0x000F); // RADJ=1111, 100%
   delay(1);
   main_W_com_data(0x0020,0x0040); // BT=0100
   main_W_com_data(0x001D,0x0007); // VC1=111      0007
   main_W_com_data(0x001E,0x0000); // VC3=000
   main_W_com_data(0x001F,0x0004); // VRH=0011

   //VCOM SETTING
   main_W_com_data(0x0044,0x004D); // VCM=101 0000    4D
   main_W_com_data(0x0045,0x000E); // VDV=1 0001      0011
   delay(1);
   main_W_com_data(0x001C,0x0004); // AP=100
   delay(2);

   main_W_com_data(0x001B,0x0018); // GASENB=0, PON=0, DK=1, XDK=0, VLCD_TRI=0,
STB=0
   delay(1);
   main_W_com_data(0x001B,0x0010); // GASENB=0, PON=1, DK=0, XDK=0, VLCD_TRI=0,
STB=0
   delay(1);
   main_W_com_data(0x0043,0x0080); //set VCOMG=1
```

```
   delay(2);

   // Display ON Setting
   main_W_com_data(0x0090,0x007F); // SAP=0111 1111
   main_W_com_data(0x0026,0x0004); //GON=0, DTE=0, D=01
   delay(1);
   main_W_com_data(0x0026,0x0024); //GON=1, DTE=0, D=01
   main_W_com_data(0x0026,0x002C); //GON=1, DTE=0, D=11
   delay(1);
   main_W_com_data(0x0026,0x003C); //GON=1, DTE=1, D=11

   // INTERNAL REGISTER SETTING
   main_W_com_data(0x0057,0x0002); // TEST_Mode=1: into TEST mode
   main_W_com_data(0x0095,0x0001); // SET DISPLAY CLOCK AND PUMPING CLOCK TO
SYNCHRONIZE
   main_W_com_data(0x0057,0x0000); // TEST_Mode=0: exit TEST mode
   //main_W_com_data(0x0021,0x0000);
   main_Write_COM(0x0022);

}

void Pant(unsigned int color)
{
   int i,j;
   address_set(0,0,239,319);

   for(i=0;i<320;i++)
   {
      for (j=0;j<240;j++)
      {
         main_Write_DATA(color);
      }

   }
}

void setup()
{
   unsigned char p;
   for(p=0;p<20;p++)
   {
      pinMode(p,OUTPUT);
   }
   main_init();
```

```
}

void loop()
{
   Pant(0xf800); //Red
   delay(1000);
   Pant(0X07E0); //Green
   delay(1000);
   Pant(0x001f); //Blue
   delay(1000);
}
```

OK, reset your board, you can find the screen will display in red, green and blue.



2、Demo code for TFT LCD

Here is a demo code that use the STC 51 MCU to drive the module. The controller communicate with the screen in 16bit parallel mode , and use 4 more pins for timing and mode control.
The code display a image of the image[3200] in the module .

First, initial the TFT screen:

```
void LCD_Init(void)
{

    LCD_REST=1;
    delayms(5);
    LCD_REST=0;
    delayms(5);
    LCD_REST=1;
    delayms(5);

    LCD_CS =0;    //enable cs
    //************* Start Initial Sequence **********//
    LCD_Write_COM(0x00,0xE5); LCD_Write_DATA(0x78,0xF0); // set SRAM internal timing
    LCD_Write_COM(0x00,0x01); LCD_Write_DATA(0x01,0x00); // set SS and SM bit
    LCD_Write_COM(0x00,0x02); LCD_Write_DATA(0x07,0x00); // set 1 line inversion
    LCD_Write_COM(0x00,0x03); LCD_Write_DATA(0x10,0x30); // set GRAM write direction and BGR=1.
    LCD_Write_COM(0x00,0x04); LCD_Write_DATA(0x00,0x00); // Resize register
    LCD_Write_COM(0x00,0x08); LCD_Write_DATA(0x02,0x07); // set the back porch and front porch
    LCD_Write_COM(0x00,0x09); LCD_Write_DATA(0x00,0x00); // set non-display area refresh cycle ISC[3:0]
    LCD_Write_COM(0x00,0x0A); LCD_Write_DATA(0x00,0x00); // FMARK function
    LCD_Write_COM(0x00,0x0C); LCD_Write_DATA(0x00,0x00); // RGB interface setting
    LCD_Write_COM(0x00,0x0D); LCD_Write_DATA(0x00,0x00); // Frame marker Position
    LCD_Write_COM(0x00,0x0F); LCD_Write_DATA(0x00,0x00); // RGB interface polarity
    //*************Power On sequence***************//
    LCD_Write_COM(0x00,0x10); LCD_Write_DATA(0x00,0x00); // SAP, BT[3:0], AP, DSTB, SLP, STB
    LCD_Write_COM(0x00,0x11); LCD_Write_DATA(0x00,0x07); // DC1[2:0], DC0[2:0], VC[2:0]
    LCD_Write_COM(0x00,0x12); LCD_Write_DATA(0x00,0x00); // VREG1OUT voltage
    LCD_Write_COM(0x00,0x13); LCD_Write_DATA(0x00,0x00); // VDV[4:0] for VCOM amplitude
    LCD_Write_COM(0x00,0x07); LCD_Write_DATA(0x00,0x01);
    delayms(50); // Dis-charge capacitor power voltage
    LCD_Write_COM(0x00,0x10); LCD_Write_DATA(0x10,0x90); //SAP, BT[3:0], AP, DSTB, SLP, STB
    LCD_Write_COM(0x00,0x11); LCD_Write_DATA(0x02,0x27); // DC1[2:0], DC0[2:0], VC[2:0]
    delayms(50); // Delay 50ms
    LCD_Write_COM(0x00,0x12); LCD_Write_DATA(0x00,0x1F); // Internal reference voltage= Vci;
```

```
    delayms(50); // Delay 50ms
    LCD_Write_COM(0x00,0x13); LCD_Write_DATA(0x15,0x00); //Set VDV[4:0] for VCOM amplitude
    LCD_Write_COM(0x00,0x29); LCD_Write_DATA(0x00,0x27);
    LCD_Write_COM(0x00,0x2B); LCD_Write_DATA(0x00,0x0D); // Set Frame Rate      000C
    delayms(50); // Delay 50ms
    LCD_Write_COM(0x00,0x20);  LCD_Write_DATA(0x00,0x00);  // GRAM horizontal Address
    LCD_Write_COM(0x00,0x21); LCD_Write_DATA(0x00,0x00); // GRAM Vertical Address
    // ----------- Adjust the Gamma Curve ----------//
    LCD_Write_COM(0x00,0x30); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x31); LCD_Write_DATA(0x07,0x07);
    LCD_Write_COM(0x00,0x32); LCD_Write_DATA(0x03,0x07);
    LCD_Write_COM(0x00,0x35); LCD_Write_DATA(0x02,0x00);
    LCD_Write_COM(0x00,0x36); LCD_Write_DATA(0x00,0x08);
    LCD_Write_COM(0x00,0x37); LCD_Write_DATA(0x00,0x04);
    LCD_Write_COM(0x00,0x38); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x39); LCD_Write_DATA(0x07,0x07);
    LCD_Write_COM(0x00,0x3C); LCD_Write_DATA(0x00,0x02);
    LCD_Write_COM(0x00,0x3D); LCD_Write_DATA(0x1D,0x04);

    //------------------ Set GRAM area ---------------//
    LCD_Write_COM(0x00,0x50); LCD_Write_DATA(0x00,0x00); // Horizontal GRAM Start Address
    LCD_Write_COM(0x00,0x51); LCD_Write_DATA(0x00,0xEF); // Horizontal GRAM End Address
    LCD_Write_COM(0x00,0x52); LCD_Write_DATA(0x00,0x00); // Vertical GRAM Start Address
    LCD_Write_COM(0x00,0x53); LCD_Write_DATA(0x01,0x3F); // Vertical GRAM Start Address
    LCD_Write_COM(0x00,0x60); LCD_Write_DATA(0xA7,0x00); // Gate Scan Line
    LCD_Write_COM(0x00,0x61); LCD_Write_DATA(0x00,0x01); // NDL,VLE, REV
    LCD_Write_COM(0x00,0x6A); LCD_Write_DATA(0x00,0x00); // set scrolling line
    //-------------- Partial Display Control ---------//
    LCD_Write_COM(0x00,0x80); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x81); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x82); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x83); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x84); LCD_Write_DATA(0x00,0x00);
    LCD_Write_COM(0x00,0x85); LCD_Write_DATA(0x00,0x00);
    //-------------- Panel Control --------------------//
    LCD_Write_COM(0x00,0x90); LCD_Write_DATA(0x00,0x10);
    LCD_Write_COM(0x00,0x92); LCD_Write_DATA(0x06,0x00);
    LCD_Write_COM(0x00,0x07); LCD_Write_DATA(0x01,0x33); // 262K color and display
```

```
ON
    LCD_CS =1; //disable    cs

}
```

Second, build the function to calculate the address:

```
void Address_set(unsigned int x1,unsigned int y1,unsigned int x2,unsigned int y2)
{
    LCD_Write_COM(0x00,0x20);LCD_Write_DATA(x1&gt;&gt;8,x1);
    LCD_Write_COM(0x00,0x21);LCD_Write_DATA(y1&gt;&gt;8,y1);
    LCD_Write_COM(0x00,0x50);LCD_Write_DATA(x1&gt;&gt;8,x1);
    LCD_Write_COM(0x00,0x52);LCD_Write_DATA(y1&gt;&gt;8,y1);
    LCD_Write_COM(0x00,0x51);LCD_Write_DATA(x2&gt;&gt;8,x2);
    LCD_Write_COM(0x00,0x53);LCD_Write_DATA(y2&gt;&gt;8,y2);
    LCD_Write_COM(0x00,0x22);
}
```

Third, build the function to send data/command into data bus:

```
void LCD_Write_COM(char VH,char VL)     //snet command
{
    LCD_RS=0;
    LCD_Writ_Bus(VH,VL);
}

void LCD_Write_DATA(char VH,char VL)//sent data
{
    LCD_RS=1;
    LCD_Writ_Bus(VH,VL);
}

void LCD_Writ_Bus(char VH,char VL)     //Write Data
{
    LCD_DataPortH=VH;
    LCD_DataPortL=VL;
    LCD_WR=0;
    LCD_WR=1;
}
```

Fourth, build a clean screen function:

```
void Pant(char VH,char VL)
{
```

```
    int i,j;
    LCD_CS =0;
    Address_set(0,0,240,320);
    for(i=0;i&lt;320;i++)
     {
      for (j=0;j&lt;240;j++)
        {
            LCD_Write_DATA(VH,VL);
        }

      }
     LCD_CS =1;
}
```

Last, the mail loop , display the image:

```
main()
{
    int i,j,k;
    LCD_Init();             //TFT initial
    Pant(0xff,0xff);// clean screen
    LCD_CS =0;
    for(k=0;k&lt;8;k++)      //display
    {
        for(j=0;j&lt;6;j++)
        {
            Address_set(40*j,40*k,40*j+39,40*k+39);
            for(i=0;i&lt;1600;i++)
              {
              LCD_Write_DATA(image[i*2+1],image[i*2]);
               }
         }
    }
    LCD_CS =1;
    while(1)
    {
    LCD_Init();
    Pant(0xff,0xff);
    LCD_CS =0;
    for(k=0;k&lt;8;k++)
    {
        for(j=0;j&lt;6;j++)
        {
            Address_set(40*j,40*k,40*j+39,40*k+39);
```

```
                for(i=0;i&lt;1600;i++)
                  {
                  LCD_Write_DATA(image[i*2+1],image[i*2]);
                   }
            }
      }
      LCD_CS =1;
      delayms(1000);
      }
}
```

3、8Bit Mode Demo

We have showed how to use the Arduino control the module for refreshing the screen , but the demo is in 16 bit mode , we received some E-mail require the 8 Bit mode demo, so we now publish the new demo which using the Arduino to control the module display a picture in 8 bit mode.

The connection is:

DB9 – DB16 to D0 – D7 of Arduino ; DB0 – DB8 to GND
RS to D8 ; WR to D9 ; RD to 3.3V
CS to D10 ; RESET to D11 ; LDEA to 3.3V
VCC to 5V; GND to GND

Download the code below into Arduino :

```
#include <avr/pgmspace.h>

#define LCD_RS 8
#define LCD_WR 9
#define LCD_CS 10
#define LCD_REST 11

extern unsigned char image[3200];

void LCD_Writ_Bus(char VH,char VL)
{
   unsigned char i,temp,data;
   data=VH;
   for(i=0;i<8;i++)
   {
```

```
    temp=(data&0x01);
    if(temp)
      digitalWrite(i,HIGH);
    else
      digitalWrite(i,LOW);
    data=data>>1;
  }
  digitalWrite(LCD_WR,LOW);
  digitalWrite(LCD_WR,HIGH);
  data=VL;
  for(i=0;i<8;i++)
  {
    temp=(data&0x01);
    if(temp)
      digitalWrite(i,HIGH);
    else
      digitalWrite(i,LOW);
    data=data>>1;
  }
  digitalWrite(LCD_WR,LOW);
  digitalWrite(LCD_WR,HIGH);
}


void LCD_Write_COM(char VH,char VL)    //send order
{
  digitalWrite(LCD_RS,LOW);
  LCD_Writ_Bus(VH,VL);
}


void LCD_Write_DATA(char VH,char VL)       //send data
{
  digitalWrite(LCD_RS,HIGH);
  LCD_Writ_Bus(VH,VL);
}

void Address_set(unsigned int x1,unsigned int y1,unsigned int x2,unsigned int y2)
{
  LCD_Write_COM(0x00,0x20);
  LCD_Write_DATA(x1>>8,x1);
  LCD_Write_COM(0x00,0x21);
  LCD_Write_DATA(y1>>8,y1);
  LCD_Write_COM(0x00,0x50);
```

```
  LCD_Write_DATA(x1>>8,x1);
  LCD_Write_COM(0x00,0x52);
  LCD_Write_DATA(y1>>8,y1);
  LCD_Write_COM(0x00,0x51);
  LCD_Write_DATA(x2>>8,x2);
  LCD_Write_COM(0x00,0x53);
  LCD_Write_DATA(y2>>8,y2);
  LCD_Write_COM(0x00,0x22);
}

void LCD_Init(void)
{

  digitalWrite(LCD_REST,HIGH);
  delay(5);
  digitalWrite(LCD_REST,LOW);
  delay(5);
  digitalWrite(LCD_REST,HIGH);
  delay(5);

  digitalWrite(LCD_CS,LOW);
  //************ Start Initial Sequence **********//
  LCD_Write_COM(0x00,0xE5);
  LCD_Write_DATA(0x78,0xF0); // set SRAM internal timing
  LCD_Write_COM(0x00,0x01);
  LCD_Write_DATA(0x01,0x00); // set SS and SM bit
  LCD_Write_COM(0x00,0x02);
  LCD_Write_DATA(0x07,0x00); // set 1 line inversion
  LCD_Write_COM(0x00,0x03);
  LCD_Write_DATA(0x10,0x30); // set GRAM write direction and BGR=1.
  LCD_Write_COM(0x00,0x04);
  LCD_Write_DATA(0x00,0x00); // Resize register
  LCD_Write_COM(0x00,0x08);
  LCD_Write_DATA(0x02,0x07); // set the back porch and front porch
  LCD_Write_COM(0x00,0x09);
  LCD_Write_DATA(0x00,0x00); // set non-display area refresh cycle ISC[3:0]
  LCD_Write_COM(0x00,0x0A);
  LCD_Write_DATA(0x00,0x00); // FMARK function
  LCD_Write_COM(0x00,0x0C);
  LCD_Write_DATA(0x00,0x00); // RGB interface setting
  LCD_Write_COM(0x00,0x0D);
  LCD_Write_DATA(0x00,0x00); // Frame marker Position
  LCD_Write_COM(0x00,0x0F);
  LCD_Write_DATA(0x00,0x00); // RGB interface polarity
```

```
//*************Power On sequence ***************//
LCD_Write_COM(0x00,0x10);
LCD_Write_DATA(0x00,0x00); // SAP, BT[3:0], AP, DSTB, SLP, STB
LCD_Write_COM(0x00,0x11);
LCD_Write_DATA(0x00,0x07); // DC1[2:0], DC0[2:0], VC[2:0]
LCD_Write_COM(0x00,0x12);
LCD_Write_DATA(0x00,0x00); // VREG1OUT voltage
LCD_Write_COM(0x00,0x13);
LCD_Write_DATA(0x00,0x00); // VDV[4:0] for VCOM amplitude
LCD_Write_COM(0x00,0x07);
LCD_Write_DATA(0x00,0x01);
delay(50); // Dis-charge capacitor power voltage
LCD_Write_COM(0x00,0x10);
LCD_Write_DATA(0x10,0x90); // 1490//SAP, BT[3:0], AP, DSTB, SLP, STB
LCD_Write_COM(0x00,0x11);
LCD_Write_DATA(0x02,0x27); // DC1[2:0], DC0[2:0], VC[2:0]
delay(50); // Delay 50ms
LCD_Write_COM(0x00,0x12);
LCD_Write_DATA(0x00,0x1F); //001C// Internal reference voltage= Vci;
delay(50); // Delay 50ms
LCD_Write_COM(0x00,0x13);
LCD_Write_DATA(0x15,0x00);  //0x1000//1400        Set VDV[4:0] for VCOM amplitude
1A00
LCD_Write_COM(0x00,0x29);
LCD_Write_DATA(0x00,0x27);  //0x0012  //001a     Set VCM[5:0] for VCOMH     //0x0025
0034
LCD_Write_COM(0x00,0x2B);
LCD_Write_DATA(0x00,0x0D); // Set Frame Rate      000C
delay(50); // Delay 50ms
LCD_Write_COM(0x00,0x20);
LCD_Write_DATA(0x00,0x00); // GRAM horizontal Address
LCD_Write_COM(0x00,0x21);
LCD_Write_DATA(0x00,0x00); // GRAM Vertical Address
// ----------- Adjust the Gamma Curve ----------//
LCD_Write_COM(0x00,0x30);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x31);
LCD_Write_DATA(0x07,0x07);
LCD_Write_COM(0x00,0x32);
LCD_Write_DATA(0x03,0x07);
LCD_Write_COM(0x00,0x35);
LCD_Write_DATA(0x02,0x00);
LCD_Write_COM(0x00,0x36);
LCD_Write_DATA(0x00,0x08);//0207
```

```
LCD_Write_COM(0x00,0x37);
LCD_Write_DATA(0x00,0x04);//0306
LCD_Write_COM(0x00,0x38);
LCD_Write_DATA(0x00,0x00);//0102
LCD_Write_COM(0x00,0x39);
LCD_Write_DATA(0x07,0x07);//0707
LCD_Write_COM(0x00,0x3C);
LCD_Write_DATA(0x00,0x02);//0702
LCD_Write_COM(0x00,0x3D);
LCD_Write_DATA(0x1D,0x04);//1604

   //------------------ Set GRAM area ---------------//
LCD_Write_COM(0x00,0x50);
LCD_Write_DATA(0x00,0x00); // Horizontal GRAM Start Address
LCD_Write_COM(0x00,0x51);
LCD_Write_DATA(0x00,0xEF); // Horizontal GRAM End Address
LCD_Write_COM(0x00,0x52);
LCD_Write_DATA(0x00,0x00); // Vertical GRAM Start Address
LCD_Write_COM(0x00,0x53);
LCD_Write_DATA(0x01,0x3F); // Vertical GRAM Start Address
LCD_Write_COM(0x00,0x60);
LCD_Write_DATA(0xA7,0x00); // Gate Scan Line
LCD_Write_COM(0x00,0x61);
LCD_Write_DATA(0x00,0x01); // NDL,VLE, REV
LCD_Write_COM(0x00,0x6A);
LCD_Write_DATA(0x00,0x00); // set scrolling line
//-------------- Partial Display Control ---------//
LCD_Write_COM(0x00,0x80);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x81);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x82);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x83);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x84);
LCD_Write_DATA(0x00,0x00);
LCD_Write_COM(0x00,0x85);
LCD_Write_DATA(0x00,0x00);
//-------------- Panel Control --------------------//
LCD_Write_COM(0x00,0x90);
LCD_Write_DATA(0x00,0x10);
LCD_Write_COM(0x00,0x92);
LCD_Write_DATA(0x06,0x00);
```

```
  LCD_Write_COM(0x00,0x07);
  LCD_Write_DATA(0x01,0x33); // 262K color and display ON
  digitalWrite(LCD_CS,HIGH);

}

void Pant(char VH,char VL)
{
  int i,j;
  digitalWrite(LCD_CS,LOW);
  Address_set(0,0,240,320);
  for(i=0;i<320;i++)
  {
    for (j=0;j<240;j++)
    {
      LCD_Write_DATA(VH,VL);
    }

  }
  digitalWrite(LCD_CS,HIGH);
}

void setup()
{
  unsigned char p;
  char hi,lo;
  int i,j,k;
  for(p=0;p<20;p++)
  {
    pinMode(p,OUTPUT);
  }

  LCD_Init();
  // Pant(0xff,0xff);

  digitalWrite(LCD_CS,LOW);
  for(k=0;k<8;k++)
  {
    for(j=0;j<6;j++)
    {
      Address_set(40*j,40*k,40*j+39,40*k+39);
      for(i=0;i<1600;i++)
      {
        hi=pgm_read_byte(&image[i*2+1]);
```

```
            lo=pgm_read_byte(&image[i*2]);
            LCD_Write_DATA(hi,lo);

          }
        }
    }
    digitalWrite(LCD_CS,HIGH);
}

void loop()
{

}
```

Reset the Arduino board , you will see the screen full with the picture that in image[3200] .

4、Touch Screen Handwrite Demo

Today we show a demo that use the Arduino to control the module 4 display, and used the touch screen to achieve handwrite function.
The LCD connection is the same as that in "3、8Bit Mode Demo", and the touch screen connection is :

DCLK to D14(A0) pin of Arduino
CS to D15(A1) pin of Arduino
DIN to D16(A2) pin of Arduino
DOUT to D18(A4) pin of Arduino
IRQ to D19(A5) pin of Arduino

The display code is the same as we have released , now we just give a brief introduction of SPI and touch IC control.

SPI Start:

```
void spistar()
{
    digitalWrite(CS,HIGH);
    digitalWrite(DCLK,HIGH);
    digitalWrite(DIN,HIGH);
    digitalWrite(DCLK,HIGH);
}
```

SPI Write Data function:

```
void WriteCharTo7843(unsigned char num)
{
   unsigned char count=0;
   unsigned char temp;
   unsigned nop;
   temp=num;
   digitalWrite(DCLK,LOW);
   for(count=0;count&amp;lt;8;count++)
   {
      if(temp&amp;amp;0x80)
         digitalWrite(DIN,HIGH);
      else
         digitalWrite(DIN,LOW);

      temp=temp&amp;lt;&amp;lt;1;

      digitalWrite(DCLK,LOW);
      nop++;
      nop++;
      digitalWrite(DCLK,HIGH);
      nop++;
      nop++;
   }
}
```

SPI Read Data function:

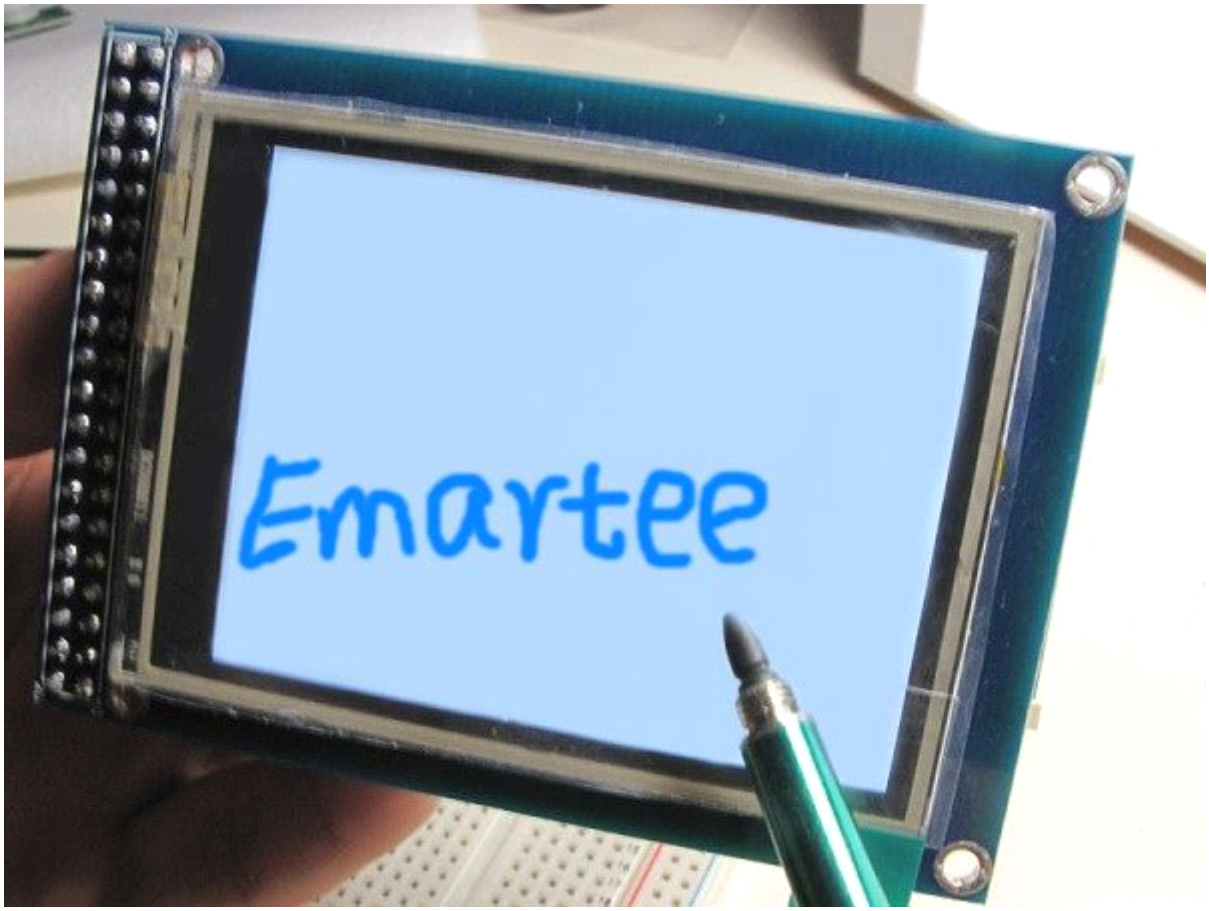
```
unsigned int ReadFromCharFrom7843()
{
   unsigned nop;
   unsigned char count=0;
   unsigned int Num=0;
   for(count=0;count&amp;lt;12;count++)
   {
      Num&amp;lt;&amp;lt;=1;
      digitalWrite(DCLK,HIGH);//DCLK=1; _nop_();_nop_();_nop_();
      nop++;
      digitalWrite(DCLK,LOW);//DCLK=0; _nop_();_nop_();_nop_();
      nop++;
      if(digitalRead(DOUT)) Num++;
   }
```

```
    return(Num);
}
```

Get coordinates:

```
void AD7843(void)
{
    digitalWrite(CS,LOW);
    WriteCharTo7843(0x90);
    digitalWrite(DCLK,HIGH);
    digitalWrite(DCLK,LOW);
    TP_Y=ReadFromCharFrom7843();
    WriteCharTo7843(0xD0);
    digitalWrite(DCLK,HIGH);
    digitalWrite(DCLK,LOW);
    TP_X=ReadFromCharFrom7843();
    digitalWrite(CS,HIGH);
}
```

Download the code into Arduino , and you can see the handwrite effects as the previous image shown :



Note: the touch and LCD is 3V3, so if you want to use the Arduino 5V pin to connect it , reduction voltage and limiting current part is necessary .