Manuals+ — User Manuals Simplified.

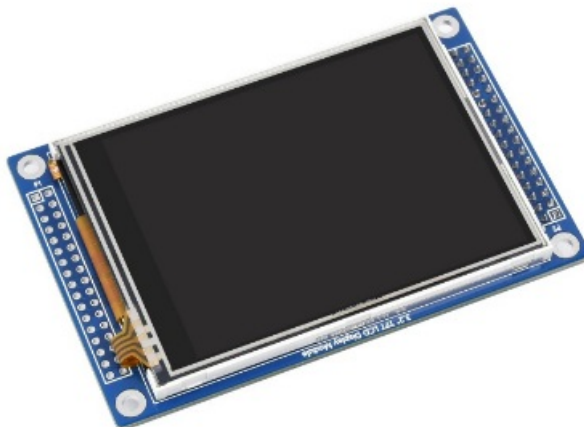# WAVESHARE XPT2046 3.2inch Touch LCD User Manual

Contents

WAVESHARE XPT2046 3.2inch Touch LCD

Specifications

- **Screen Size:** 3.2 inch
- **Resolution: 3**20×240 pixels

- **Touch Technology:** Resistive touchscreen
- **Touch Controller:** XPT2046
- **TFT LCD Driver Chip**: ILI9341
- **Color Depth:** 262144 colors
- **RAM:** 172820 bytes
- **Data Interface Modes:** i80-system MPU port, serial data transfer port (SPI), RGB 6-/16-/18-bit port

Overview

Hardware Resource

ILI9341

- ILI9341 is a 262144-color TFT LCD driver chip with a resolution of 240×320 (RGB) and 172820 (240 x 320x 18/8) bytes of RAM. Each pixel depth can reach 18 bits.
- ILI9341 has the following data interface modes:

1. i80-system MPU port (8-/9-/16-/18-bit bus width).
2. serial data transfer port (SPI).
3. RGB 6-/16-/18-bit port (DOTCLK, VSYNC, HSYNC, ENABLE, DB[17:0]).

The corresponding relationship between the 18-bit RGB assignment of ILI9341 and LCD GRAM in this screen is shown in the figure:
According to the above figure, ILI9341 is in the 16-bit mode and the available GRAM data are D17~D13 and D11~D1. D12 and D0 are not used. D12 and D0 of ILI9341 are not adapted in our LCD module. D17~D13 and D11~D1 of ILI9341 correspond to D15~D0 of MCU. In the 16-bit data of the MCU, the lowest 5 bits represent blue, the middle 6 bits are green, and the highest 5 bits are red. The larger the value, the darker the color.

8080 16-bit Interface Sequence Introduction

- For more details about the register, please refer to ILI9341.pdf.
- Here we only introduce the sequence requirements for reading and writing.

The 8080 interface is designed by Intel. It is a parallel, asynchronous, half-duplex communication protocol. It is used for external expansion of RAM and ROM and is also used for LCD interface later.

- Four control cables
    - **RD:** Write enable (write information to register)
    - **WR:** read enable (read information from the register)
    - **DC(RS):** data/command (1 for reading and writing data, 0 for reading and writing commands)
    - **CS:** chip select
- The sequence diagram for writing commands or data is as follows:
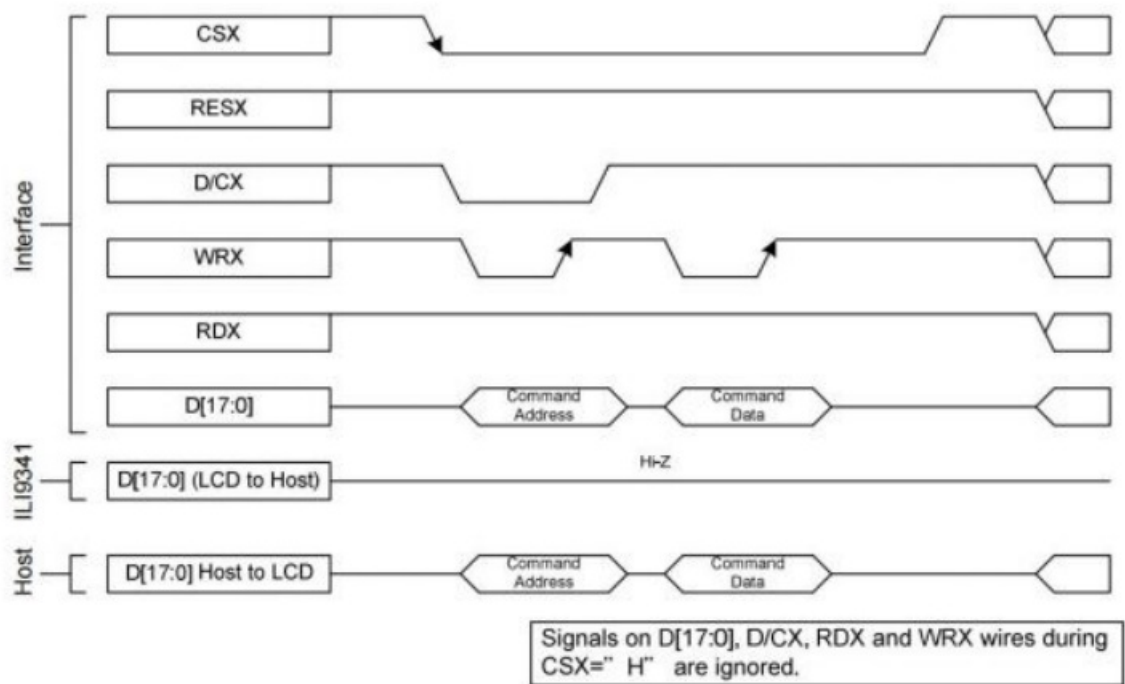
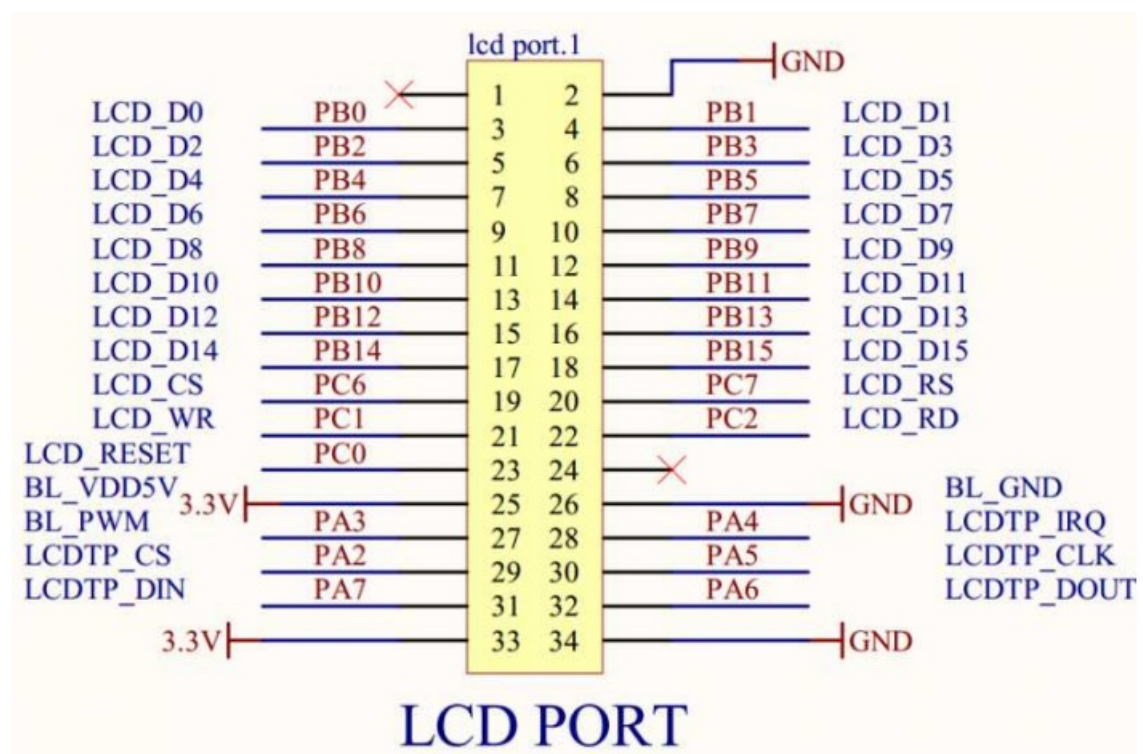- The sequence diagram for reading information is as follows:
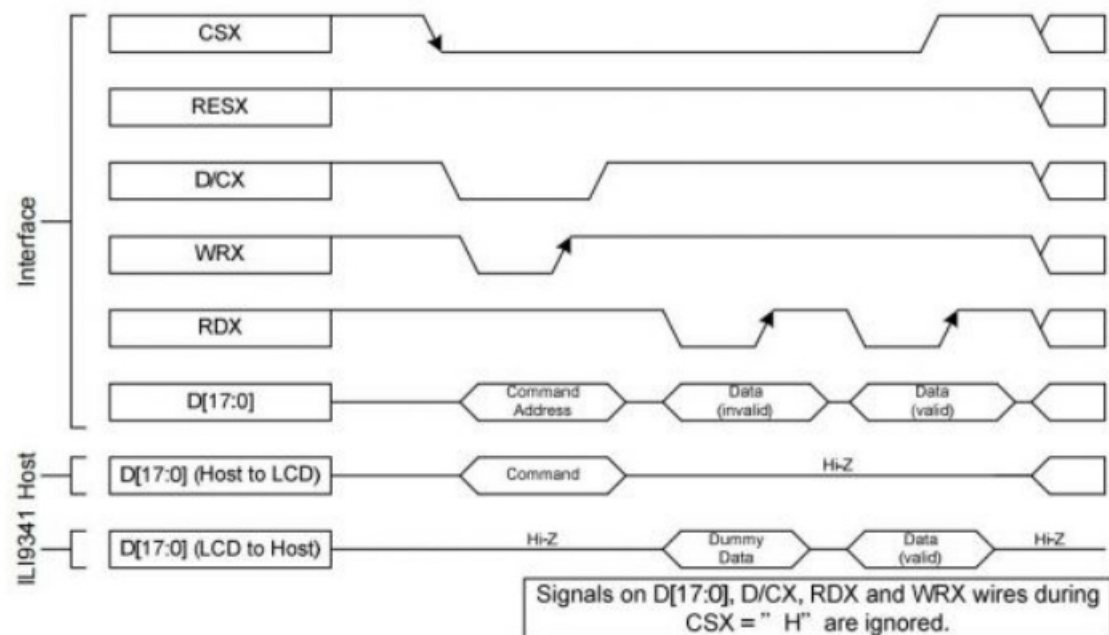
XPT2046

- The XPT2046 is a 4-wire resistive touchscreen controller with a successive approximation A/D converter with 12-bit resolution and a conversion rate of 125KHZ.
- The XPT2046 supports low-voltage I/O interfaces from 1.5V to 5.25V.
- The XPT2046 can detect the touched position on the screen by performing two A/D conversions, in addition to measuring the pressure applied on the touch screen. The built-in 2.5V reference voltage can be used as an auxiliary input, temperature measurement, and battery monitoring. The voltage range of battery monitoring can be from 0V to 5V.
- The XPT2046 has a temperature sensor integrated on-chip. Under typical operating conditions of 2.7V, with the reference voltage off, the power consumption can be less than 0.75mW. The XPT2046 is available in tiny packages: TSSOP-16, QFN-16, and VFBGA-48. The working temperature range is -40°C +85°C. Fully compatible with ADS7846, TSC2046, and AK4182A.
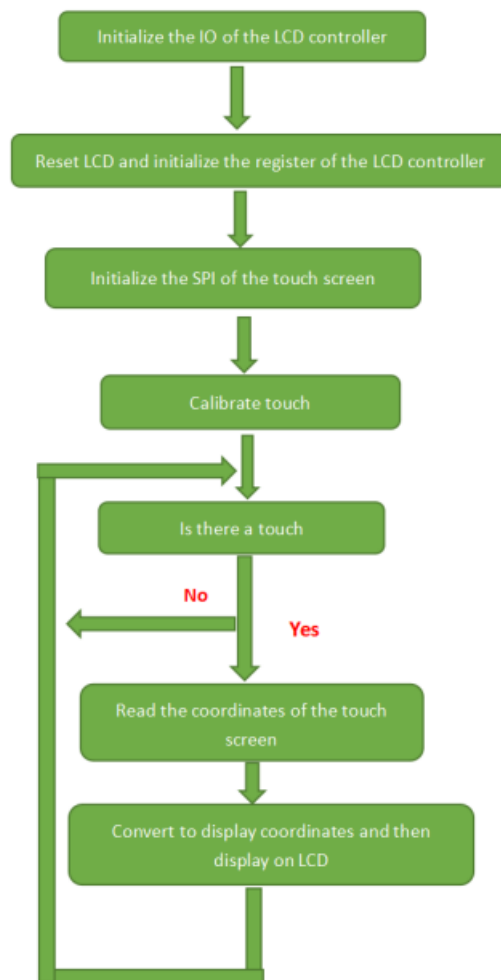
Hardware Description

Pin No.	Symbol	Description	Function
1	5V	5V power supply	When powered from 5V supply, pin 1 & Pin 2 as power input, pin 33 & Pin 34 provide 3.3V output.
2	GND	Ground	GND
3	D0	Data cable	D0-D15
4	D1		
5	D2		
6	D3		
7	D4		
8	D5		
9	D6		
10	D7		
11	D8		
12	D9		
13	D10		
14	D11		
15	D12		
16	D13		
17	D14		
18	D15		
19	CS	LCD chip selection signal	Low active
20	RS	Command/Data Register Selection	RS = 0: Command Register RS = 1: Data Register
21	WR	Write	WR = 0, RD = 1
22	RD	Read	WR = 1, RD = 0
23	RESET	Reset the controller chip	Low active
24	NC	Not connect	Not connect
25	BLVCC	5V or 3.3V	Backlight VCC
26	BLGND	Ground	Backlight GND
27	BLCNT	Backlight brightness adjustment	Control the backlight brightness via PWM
28	TP_IRQ	Touch panel interrupt	Low level while the touch panel detects pressing
29	TP_CS	Touch panel chip select	Low active
30	TP_SCK	Touch panel SPI clock	Connects to SPI SCK
31	TP_SI	Touch panel data input	Connects to SPI MOSI
32	TP_SO	Touch panel SPI data output	Connects to SPI MISO
33	3.3V	3.3V power supply	When powered from 3.3V supply, pin 33 & pin 34 as power input, pin 1 & pin 2 keep NC.
34	GND	Ground	

Sample Demo

This manual uses the development board of the main control chip STM32F103RCT6 to illustrate the basic usage of this LCD. Users can also use other similar development boards for development. 3.2inch 320×240 Touch LCD (D) and STM32F103RCT6 connection interface diagram:

LCD PORT

Procedure flow chart

- Initialize the IO of the LCD controller
- Reset LCD and initialize the register of the LCD controller
- Initialize the SPI of the touch screen
- Calibrate touch
- Is there a touch
  - No
  - Yes
- Read the coordinates of the touch screen
- Convert to display coordinates and then display on LCD

Source code analysis:

```c
/*The following macro definition is for the rotation angle of the image and the setti
ng of the control cable*/<br />
//#define DISP_ORIENTATION                                        0
//#define DISP_ORIENTATION                                        90
//#define DISP_ORIENTATION                                        180
#define DISP_ORIENTATION                                          270

#define Set_Cs       GPIOC->BSRR = GPIO_Pin_6  //CS=1
#define Clr_Cs       GPIOC->BRR = GPIO_Pin_6           //CS=0

#define Set_Rs       GPIOC->BSRR = GPIO_Pin_7  //RS=1
#define Clr_Rs       GPIOC->BRR = GPIO_Pin_7           //RS=0

#define Set_nWr      GPIOC->BSRR = GPIO_Pin_1  //WR=1
#define Clr_nWr      GPIOC->BRR = GPIO_Pin_1           //WR=0

#define Set_nRd      GPIOC->BSRR = GPIO_Pin_2  //RD=1
#define Clr_nRd      GPIOC->BRR = GPIO_Pin_2           //RD=0
/* Write command function */
__inline void LCD_WriteIndex(uint16_t index)
{
Clr_Rs;             //RS=0
Set_nRd;            //RD=0
LCD_Delay(0);       //Delay
GPIOB->ODR = index; /*Write command */
LCD_Delay(0);       //Delay
Clr_nWr;            //WR=0
Set_nWr;            //WR=1
}
/*  */
__inline void ILI9341_LCD_WriteData(uint16_t data)
{
        Clr_Cs;                               //CS=1
        Set_Rs;                               //RS=1
        LCD_Delay(0);                         //delay
        GPIOB->ODR = data;            /* GPIO_Write(GPIOB,data); */
        LCD_Delay(0);                         //delay
        Clr_nWr;                              //WR=0
        Set_nWr;                              //WR=1
```

```c
        Set_Cs;                                        //CS=1
}
/* Read data function */
__inline uint16_t LCD_ReadData(void)
{
uint16_t value;
Set_Rs;
Set_nWr;
Clr_nRd;
GPIOB->CRH = 0x44444444;    //Set PB0-PB15 as input
GPIOB->CRL = 0x44444444;
value = GPIOB->IDR;          //Read data
    GPIOB->CRH = 0x33333333;    //Set PB0-PB15 as output
    GPIOB->CRL = 0x33333333;
    Set_nRd;
    return value;
}
/*******************************************************************************

Write data at the specified address, LCD_Reg is the address, and LCD_RegValue is the
written value.

*******************************************************************************/
__inline void LCD_WriteReg(uint16_t LCD_Reg,uint16_t LCD_RegValue)
{
Clr_Cs;
LCD_WriteIndex(LCD_Reg);         //Write command; that is, the address to write data i
nto;
LCD_WriteData(LCD_RegValue);    //data writing;
Set_Cs;
}


/*******************************************************************************

Read data from the specified address, LCD_Reg is the address, and the function return
s the read value.
<pre>
*******************************************************************************/
```

```c
 *************************************************************************/
__inline uint16_t LCD_ReadReg(uint16_t LCD_Reg)
{
uint16_t LCD_RAM;
Clr_Cs;
LCD_WriteIndex(LCD_Reg);      //Write command; that is, the address of the data to be
read;
LCD_RAM = LCD_ReadData();  //data readout;
Set_Cs;
return LCD_RAM;
}
//The above are the basic functions for reading and writing; if you want to operate I
O analog with the FSMC of the STM32, you can refer to LCD + TouchPanel (8080 FSMC) de
mo.
/***************************************************************************
Initialization of LCD registers, the following initialization values of the registers
are provided by the original LCD manufacturer and can be displayed normally according
to the following configuration, please refer to the chip manual for registers.
***************************************************************************/
void LCD_Initializtion(void)
{
uint16_t DeviceCode;
LCD_Configuration();                   //Pin initialization
GPIO_PinRemapConfig(GPIO_Remap_SWJ_JTAGDisable , ENABLE);//Change the mapping of the
specified pin
GPIO_ResetBits(GPIOC, GPIO_Pin_0);    /* LCD reset*/
delay_ms(100);
GPIO_SetBits(GPIOC, GPIO_Pin_0);
GPIO_SetBits(GPIOA, GPIO_Pin_3);      /*Enable backlight */
DeviceCode = LCD_ReadReg(0x0000);     /* Read ID */
if(DeviceCode == 0 || DeviceCode == 0xffff)
{
    ILI9341_LCD_WriteReg(0XD3);
    Clr_Cs;
    DeviceCode = LCD_ReadData();
    DeviceCode = LCD_ReadData();
    DeviceCode = LCD_ReadData();
    DeviceCode <<= 8;
    DeviceCode |= LCD_ReadData();
```

```c
    Set_Cs;
}
if(DeviceCode == 0x9341)
{
        LCD_Code = ILI9341;
        ILI9341_LCD_WriteReg(0x3A);
        ILI9341_LCD_WriteData(0x55);

        ILI9341_LCD_WriteReg(0xB5);
        ILI9341_LCD_WriteData(0X04);
        ILI9341_LCD_WriteData(0X04);
        ILI9341_LCD_WriteData(0X0A);
        ILI9341_LCD_WriteData(0x14);

        ILI9341_LCD_WriteReg(0x35);
        ILI9341_LCD_WriteData(0x00);

        ILI9341_LCD_WriteReg(0xCF);
        ILI9341_LCD_WriteData(0x00);
        ILI9341_LCD_WriteData(0xEA);
        ILI9341_LCD_WriteData(0XF0);

        ILI9341_LCD_WriteReg(0xED);
        ILI9341_LCD_WriteData(0x64);
        ILI9341_LCD_WriteData(0x03);
        ILI9341_LCD_WriteData(0X12);
        ILI9341_LCD_WriteData(0X81);

        ILI9341_LCD_WriteReg(0xE8);
        ILI9341_LCD_WriteData(0x85);
        ILI9341_LCD_WriteData(0x10);
        ILI9341_LCD_WriteData(0x78);

        ILI9341_LCD_WriteReg(0xCB);
        ILI9341_LCD_WriteData(0x39);
        ILI9341_LCD_WriteData(0x2C);
        ILI9341_LCD_WriteData(0x00);
        ILI9341_LCD_WriteData(0x33);
        ILI9341_LCD_WriteData(0x06);
```

```c
ILI9341_LCD_WriteReg(0xF7);
ILI9341_LCD_WriteData(0x20);

ILI9341_LCD_WriteReg(0xEA);
ILI9341_LCD_WriteData(0x00);
ILI9341_LCD_WriteData(0x00);

ILI9341_LCD_WriteReg(0xC0);    //Power control
ILI9341_LCD_WriteData(0x21);   //VRH[5:0]

ILI9341_LCD_WriteReg(0xC1);    //Power control
ILI9341_LCD_WriteData(0x10);   //SAP[2:0];BT[3:0]

ILI9341_LCD_WriteReg(0xC5);    //VCM control
ILI9341_LCD_WriteData(0x4F);     //3F
ILI9341_LCD_WriteData(0x38);     //3C

ILI9341_LCD_WriteReg(0xC7);    //VCM contro12
ILI9341_LCD_WriteData(0XB7);

ILI9341_LCD_WriteReg(0x36);    // Memory Access Control
if (DISP_ORIENTATION == 0)
        ILI9341_LCD_WriteData(0x08 | 0x00);
else if (DISP_ORIENTATION == 90)
        ILI9341_LCD_WriteData(0x08 | 0xa0);
else if(DISP_ORIENTATION == 180)
        ILI9341_LCD_WriteData(0x08 | 0xc0);
else
        ILI9341_LCD_WriteData(0x08 | 0x60);


ILI9341_LCD_WriteReg(0xB1);
ILI9341_LCD_WriteData(0x00);
ILI9341_LCD_WriteData(0x13);

ILI9341_LCD_WriteReg(0xB6);    // Display Function Control
ILI9341_LCD_WriteData(0x0A);
ILI9341_LCD_WriteData(0xA2);
```

```c
ILI9341_LCD_WriteReg(0xF2);     // 3Gamma Function Disable
ILI9341_LCD_WriteData(0x02);

ILI9341_LCD_WriteReg(0x26);     //Gamma curve selected
ILI9341_LCD_WriteData(0x01);

ILI9341_LCD_WriteReg(0xE0);     //Set Gamma
ILI9341_LCD_WriteData(0x0F);
ILI9341_LCD_WriteData(0x27);
ILI9341_LCD_WriteData(0x24);
ILI9341_LCD_WriteData(0x0C);
ILI9341_LCD_WriteData(0x10);
ILI9341_LCD_WriteData(0x08);
ILI9341_LCD_WriteData(0x55);
ILI9341_LCD_WriteData(0X87);
ILI9341_LCD_WriteData(0x45);
ILI9341_LCD_WriteData(0x08);
ILI9341_LCD_WriteData(0x14);
ILI9341_LCD_WriteData(0x07);
ILI9341_LCD_WriteData(0x13);
ILI9341_LCD_WriteData(0x08);
ILI9341_LCD_WriteData(0x00);

ILI9341_LCD_WriteReg(0XE1);     //Set Gamma
ILI9341_LCD_WriteData(0x00);
ILI9341_LCD_WriteData(0x0F);
ILI9341_LCD_WriteData(0x12);
ILI9341_LCD_WriteData(0x05);
ILI9341_LCD_WriteData(0x11);
ILI9341_LCD_WriteData(0x06);
ILI9341_LCD_WriteData(0x25);
ILI9341_LCD_WriteData(0x34);
ILI9341_LCD_WriteData(0x37);
ILI9341_LCD_WriteData(0x01);
ILI9341_LCD_WriteData(0x08);
ILI9341_LCD_WriteData(0x07);
ILI9341_LCD_WriteData(0x2B);
ILI9341_LCD_WriteData(0x34);
```

```c
        ILI9341_LCD_WriteData(0x0F);

        ILI9341_LCD_WriteReg(0x11); //Exit Sleep
        delay_ms(120);
        ILI9341_LCD_WriteReg(0x29); //Display on
}
delay_ms(50);
}
/*******************************************************************************
Set the positions X, and Y of the display window;
*******************************************************************************/
void LCD_Address_Set(u16 x1,u16 y1,u16 x2,u16 y2)
{
        ILI9341_LCD_WriteReg(0x2a);//Column address setting
        ILI9341_LCD_WriteData(x1>>8);
        ILI9341_LCD_WriteData(x1&0xff);
        ILI9341_LCD_WriteData(x2>>8);
        ILI9341_LCD_WriteData(x2&0xff);

        ILI9341_LCD_WriteReg(0x2b);//Line address setting
        ILI9341_LCD_WriteData(y1>>8);
        ILI9341_LCD_WriteData(y1&0xff);
        ILI9341_LCD_WriteData(y2>>8);
        ILI9341_LCD_WriteData(y2&0xff);

        ILI9341_LCD_WriteReg(0x2c);//Write in memory
}

static void LCD_SetCursor( uint16_t Xpos, uint16_t Ypos )
{
        uint16_t temp;
        #if (DISP_ORIENTATION == 0)
        #elif (DISP_ORIENTATION == 90)
        temp = Xpos;
        Xpos =Ypos;
        Ypos = MAX_X - 1 - temp;
        #elif (DISP_ORIENTATION == 180)
        Xpos = MAX_X - 1 - Xpos;
        Ypos = MAX_Y - 1 - Ypos;
```

```
        #elif (DISP_ORIENTATION == 270)
        temp = Ypos;
        Ypos = Xpos;
        Xpos = MAX_Y - 1 - temp;
        #endif

        ILI9341_LCD_WriteReg(0x2a);     //Column address setting
        ILI9341_LCD_WriteData(Xpos>>8);
        ILI9341_LCD_WriteData(Xpos&0xff);

        ILI9341_LCD_WriteReg(0x2b);     //Line address setting
        ILI9341_LCD_WriteData(Ypos>>8);
        ILI9341_LCD_WriteData(Ypos&0xff);
}
/*************************************************************************
Clearing screen function to make the whole screen display a certain color,
*************************************************************************/
void LCD_Clear(uint16_t Color)
{
uint32_t index=0;

LCD_Address_Set(0,0,MAX_X-1,MAX_Y-1);//Set the display range
for( index = 0; index < MAX_X * MAX_Y; index++ )
{
    ILI9341_LCD_WriteData(Color);
}


}

int main(void)
{
//Delay and initialization system
  LCD_Initializtion();        //LCD initialization
//LCD initialization
  LCD_Clear(Red);         //Clear the screen to red
//You can write a function to calibrate the screen.
  /* Infinite loop */
  {
//You can write a function to display the coordinates of the touch point on the LCD.
  }
}
```

Resources

Documentation

- Schematic diagram
- Dimensions

Demo

- Sample demo

Datasheet

- ILI9325
- XPT2046

Support

If you require technical support, please go to the page and open a ticket.

Frequently Asked Questions (FAQ)

- **Q: What is the resolution of the LCD screen?**
- A: The resolution of the LCD screen is 320×240 pixels.
- **Q: What is the color depth of the LCD screen?**
- A: The LCD screen has a color depth of 262144 colors.
- **Q: What are the available data interface modes for the ILI9341 TFT LCD driver chip?**
- A: The available data interface modes are the i80-system MPU port, serial data transfer port (SPI), and RGB 6-/16-/18-bit port.
- **Q: What is the resolution of the XPT2046 touchscreen controller?**
- A: The XPT2046 touchscreen controller does not have a resolution. It detects the touched position on the screen by performing two A/D conversions.

Documents / Resources

WAVESHARE XPT2046 3.2inch Touch LCD [pdf] User Manual
ILI9341, XPT2046, XPT2046 3.2inch Touch LCD, 3.2inch Touch LCD, Touch LCD, LCD

References

- 🌿 **Log in - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 Touch LCD (D).jpg - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 Touch LCD Spec 03.jpg - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 Touch LCD Spec 04.jpg - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 Touch LCD Spec01.jpg - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 TouchHardware007.png - Waveshare Wiki**
- 🌿 **File:3.2inch 320x240 TouchHardware01.png - Waveshare Wiki**
- 🌿 **Waveshare Wiki**
- **User Manual**