JLX096-023-PN 使用说明书

(不带字库 IC，3.3V 供电)

目 录

1．概述

晶联讯电子专注于液晶屏及液晶模块的研发、制造。所生产 JLX096-023-PN 型液晶模块由于使用方便、显示清晰，广泛应用于各种人机交流面板。

JLX096-023-PN 可以显示 160 列*80 行点阵彩色图片。

本产品可选择带中文字库 IC 与不带中文字库 IC 两种。

2．JLX096-023-PN 彩色图像型点阵液晶模块的特性

2.1 结构轻、薄、带背光、带 PCB。

2.2 IC 采用 ST7735S,功能强大，稳定性好

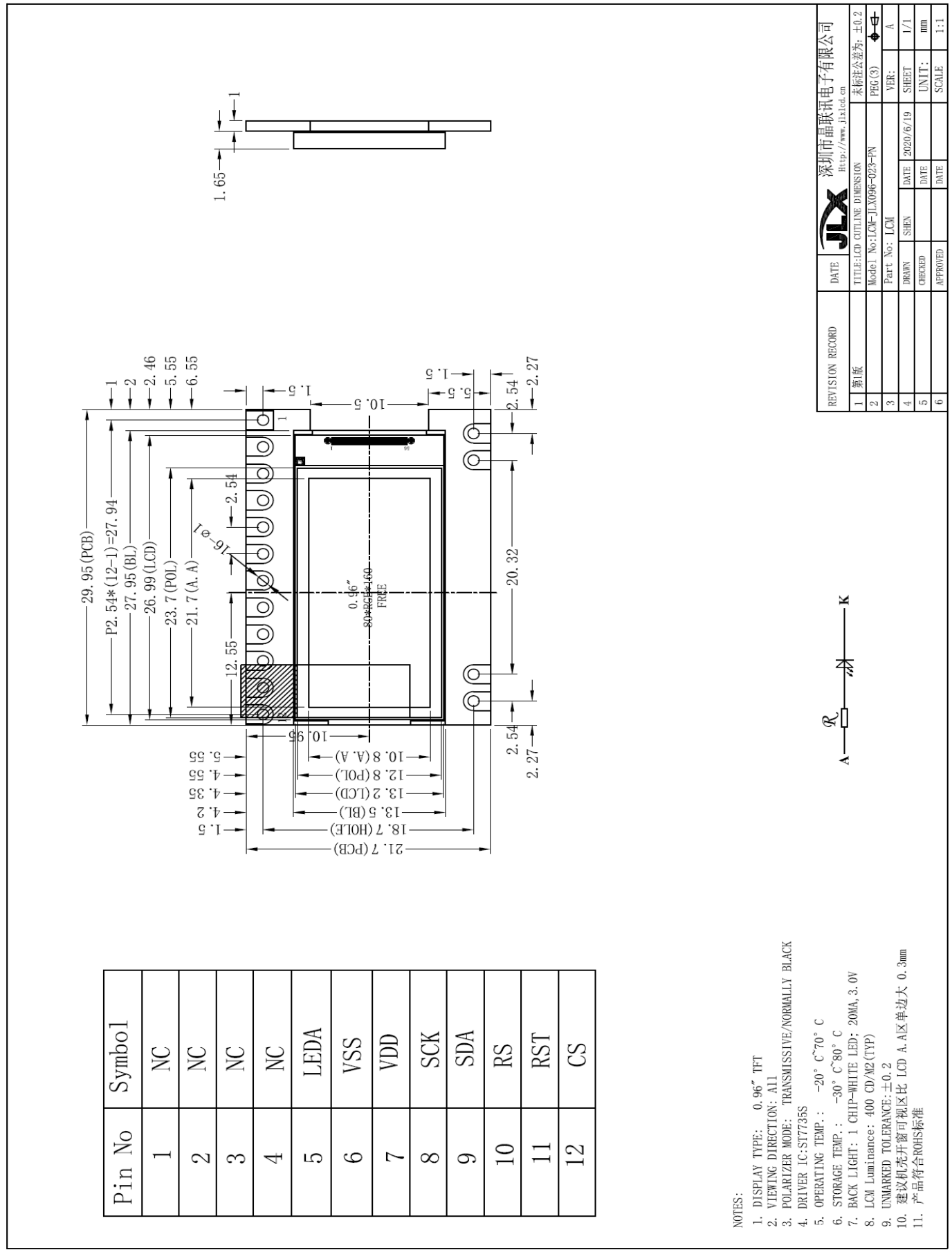
2.3 指令功能强：例如可以用指令控制显示内容顺时针旋转 90、逆时针旋转 90°或倒立竖放。

2.4 接口简单方便:采用 4_SPI 串口。

2.5 工作温度宽:-20℃ - 70℃;

2.6 储存温度宽:-30℃ - 80℃;

3. 外形尺寸及接口引脚功能



NOTES:
1. DISPLAY TYPE: 0.96" TFT
2. VIEWING DIRECTION: All
3. POLARIZER MODE: TRANSMISSIVE/NORMALLY BLACK
4. DRIVER IC:ST7735S
5. OPERATING TEMP.: -20°C~70°C
6. STORAGE TEMP.: -30°C~80°C
7. BACK LIGHT: 1 CHIP-WHITE LED; 20MA, 3.0V
8. LCM Luminance: 400 CD/M2(TYP)
9. UNMARKED TOLERANCE:±0.2
10. 建议机壳开窗可视区比 LCD A.A区单边大 0.3mm
11. 产品符合ROHS标准

Pin No	Symbol
1	NC
2	NC
3	NC
4	NC
5	LEDA
6	VSS
7	VDD
8	SCK
9	SDA
10	RS
11	RST
12	CS

图 1. 外形尺寸

模块的接口引脚功能

引 线 号	符 号	名 称	功 能
1	NC	空脚	空脚
2	NC	空脚	空脚
3	NC	空脚	空脚
4	NC	空脚	空脚
5	LDEA	背光电源正极	背光电源正极，（同 VDD 电压 3.3V）。
6	VSS	接地	0V
7	VDD	电路电源	3.3V
8	SCK	I/O	串行时钟
9	SDA	I/O	串行数据
10	A0（RS）	寄存器选择信号	H:数据寄存器 0:指令寄存器（IC 资料上所写为”A0”）
11	RST	复位	低电平复位，复位完成后，回到高电平，液晶模块开始工作
12	CS	片选	低电平片选

表 1：模块的接口引脚功能

4. 基本原理

4.1 液晶屏（LCD）

在 LCD 上排列着 160×80 点阵,160 个列信号与驱动 IC 相连，80 个行信号也与驱动 IC 相连，IC 邦定在 LCD 玻璃上（这种加工工艺叫 COG）.

4.3 背光参数

该型号液晶模块带 LED 背光源。它的性能参数如下:
工作温度:-20∽+70° C;
存储温度:-30∽+80° C;
背光板是白色。
正常工作电流为：8∽20mA（LED 灯数共 1 颗）
工作电压：3.0V（由于 PCB 上面加了限流电阻，所以电压同 VDD 电压 3.3V 即可）。

5. 技术参数

5.1 最大极限参数（超过极限参数则会损坏液晶模块）

名称	符号	标准值			单位
		最小	典型	最大	
电路电源	VDD	-0.3	3.3	4.6	V
工作温度		-20		+70	℃
储存温度		-30		+80	℃

表 2：最大极限参数

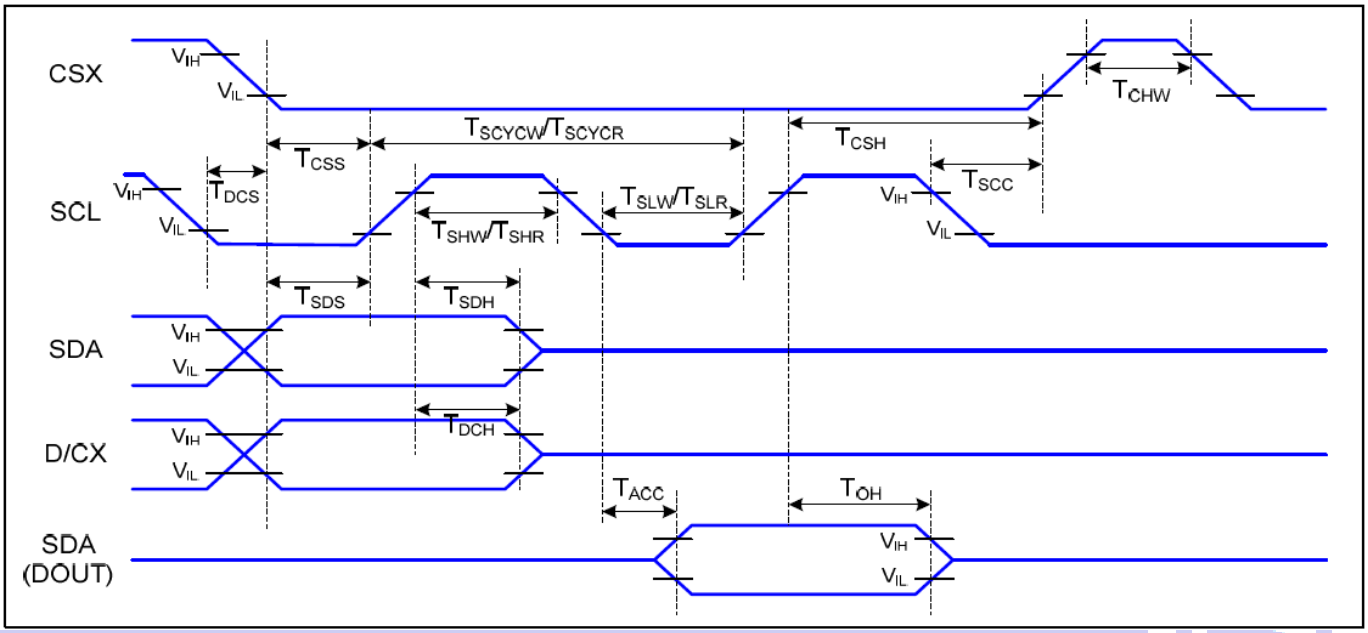
5.2 直流（DC）参数

名 称	符 号	测 试 条 件	标 准 值			单位
			最小	典型值	最大	
工作电压	VDD		2.8	3.0	4.6	V
背光工作电压	VLED		2.9	3.0	3.1	V
背光工作电流	ILED	VLED=3.0V， 共 2 颗 LED 灯并联	16	30	40	mA

表 3：直流（DC）参数

6. 读写时序特性

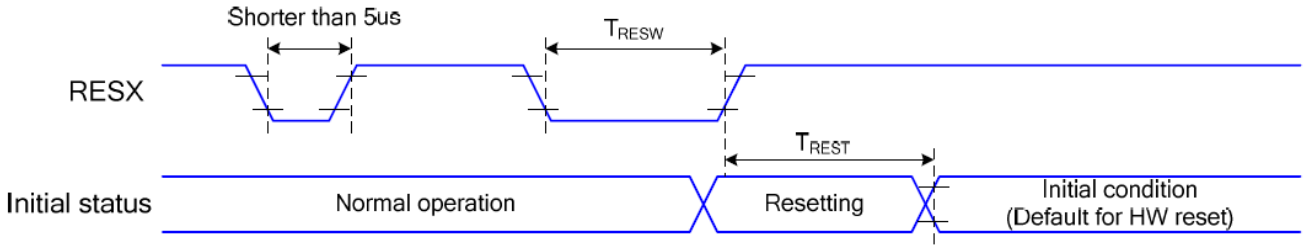
详见 IC 资料“ST7735S”，请找相关客服人员索要。



Ta=25 ℃, VDDI=1.65~3.7V, VDD=2.3~4.8V

Signal	Symbol	Parameter	MIN	MAX	Unit	Description
CSX	TCSS	Chip Select Setup Time (Write)	TBD		ns	
	TCSH	Chip Select Hold Time (Write)	TBD		ns	
	TCSS	Chip Select Setup Time (Read)	TBD		ns	
	TSCC	Chip Select Hold Time (Read)	TBD		ns	
	TCHW	Chip Select “H” Pulse Width	TBD		ns	
SCL	TSCYCW	Serial Clock Cycle (Write)	TBD		ns	-Write Command & Data Ram
	TSHW	SCL “H” Pulse Width (Write)	TBD		ns	
	TSLW	SCL “L” Pulse Width (Write)	TBD		ns	
	TSCYCR	Serial Clock Cycle (Read)	TBD		ns	-Read Command & Data Ram
	TSHR	SCL “H” Pulse Width (Read)	TBD		ns	
	TSLR	SCL “L” Pulse Width (Read)	TBD		ns	
D/CX	TDCS	D/CX Setup Time	TBD		ns	
	TDCH	D/CX Hold Time	TBD		ns	
SDA (DIN) (DOUT)	TSDS	Data Setup Time	TBD		ns	For Maximum CL=30pF For Minimum CL=8pF
	TSDH	Data Hold Time	TBD		ns	
	TACC	Access Time	TBD	TBD	ns	
	TOH	Output Disable Time	TBD	TBD	ns	

6.1 电源启动后复位的时序要求（RESET CONDITION AFTER POWER UP）：



图为**电源启动后复位的时序**

电源启动后复位的时序要求

Related Pins	Symbol	Parameter	MIN	MAX	Unit
RESX	tRESW	Reset Pulse Duration	10	-	us
	tREST	Reset Cancel	-	5	ms
				120	ms

7. 指令功能：

7.1 指令表

指 令 表

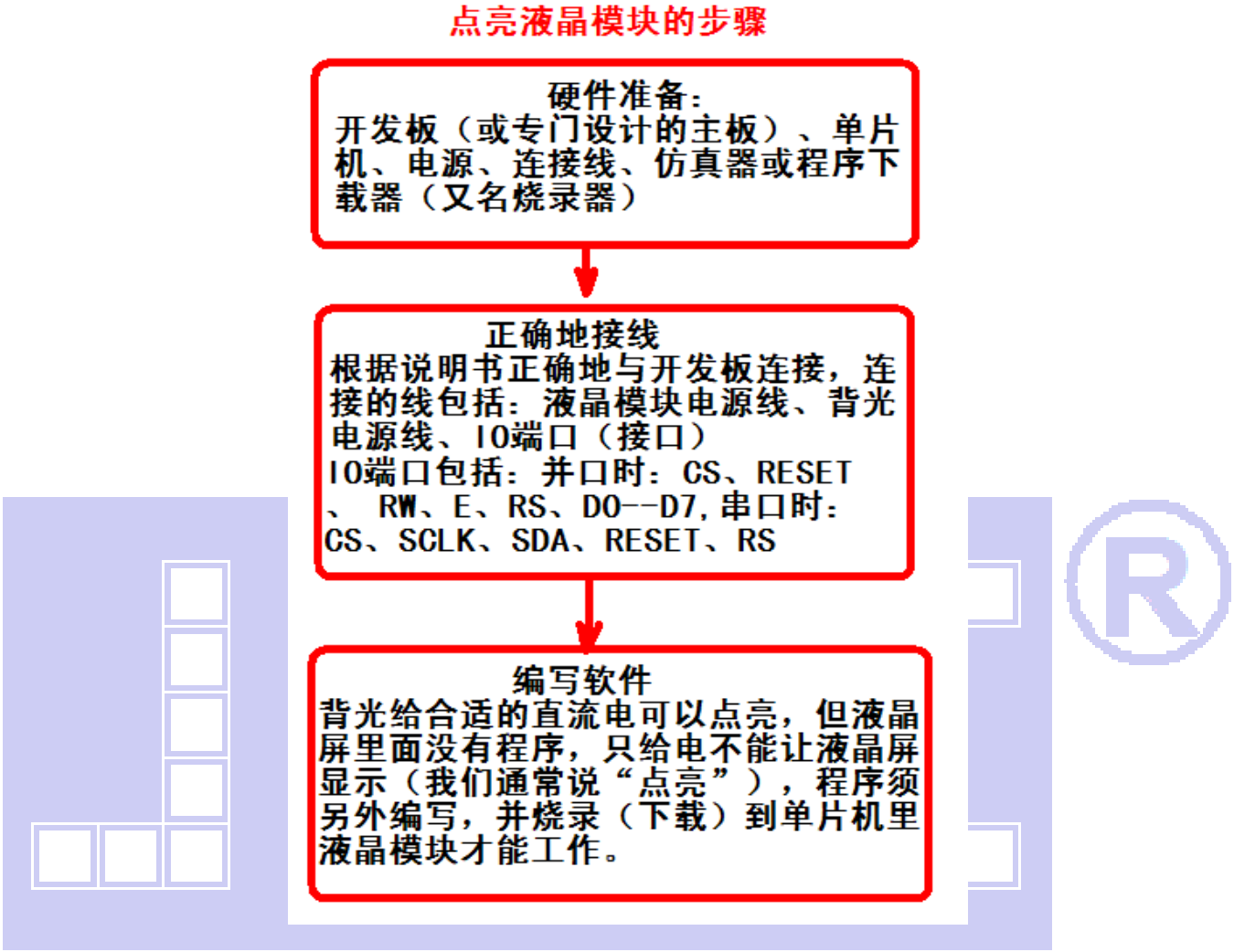
Instruction	Refer	D/CX	WRX	RDX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	Hex	Function
NOP	0	0	↑	1	-	0	0	0	0	0	0	0	0	(00h)	No Operation
SWRESET	0	0	↑	1	-	0	0	0	0	0	0	0	1	(01h)	Software Reset
RDDID	0	0	↑	1	-	0	0	0	0	0	1	0	0	(04h)	Read Display ID
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10		ID1 Read
		1	1	↑	-	1	ID26	ID25	ID24	ID23	ID22	ID21	ID20		ID2 Read
		1	1	↑	-	ID37	ID36	ID35	ID34	ID33	ID32	ID31	ID30		ID3 Read
RDDST	0	0	↑	1	-	0	0	0	0	1	0	0	1	(09h)	Read Display Status
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	BSTON	MY	MX	MV	ML	RGB	MH	ST24		-
		1	1	↑	-	ST23	IFPF2	IFPF1	IFPF0	IDMON	PTLON	SLOUT	NORON		-
		1	1	↑	-	VSSON	ST14	INVON	ST12	ST11	DISON	TEON	GCS2		-
		1	1	↑	-	GCS1	GCS0	TEM	ST4	ST3	ST2	ST1	ST0		-
RDDPM	0	0	↑	1	-	0	0	0	0	1	0	1	0	(0Ah)	Read Display Power Mode
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	BSTON	IDMON	PTLON	SLPOUT	NORON	DISON	-	-		-
RDD MADCTL	0	0	↑	1	-	0	0	0	0	1	0	1	1	(0Bh)	Read Display MADCTL
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	MY	MX	MV	ML	RGB	MH	-	-		-
RDD COLMOD	0	0	↑	1	-	0	0	0	0	1	1	0	0	(0Ch)	Read Display Pixel Format
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	0	0	0	0	-	IFPF2	IFPF1	IFPF0		-
RDDIM	0	0	↑	1	-	0	0	0	0	1	1	0	1	(0Dh)	Read Display Image Mode
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	VSSON	D6	INVON	-	-	GCS2	GCS1	GCS0		-
RDDSM	0	0	↑	1	-	0	0	0	0	1	1	1	0	(0Eh)	Read Display Signal Mode
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	TEON	TEM	-	-	-	-	-	-		-
RDDSDR	0	0	↑	1	-	0	0	0	0	1	1	1	1	(0Fh)	Read Display Self-diagnostic
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	RELD	FUND	ATTD	BRD	-	-	-	-		-

Instructi	Refer	D/CX	WRX	RDX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	Hex	Function
SLPIN	0	0	↑	1	-	0	0	0	1	0	0	0	0	(10h)	Sleep In & Booster Off
SLPOUT	0	0	↑	1	-	0	0	0	1	0	0	0	1	(11h)	Sleep Out & Booster On
PTLON	0	0	↑	1	-	0	0	0	1	0	0	1	0	(12h)	Partial Mode On
NORON	0	0	↑	1	-	0	0	0	1	0	0	1	1	(13h)	Partial Off (Normal)
INVOFF	0	0	↑	1	-	0	0	1	0	0	0	0	0	(20h)	Display Inversion Off (Normal)
INVON	0	0	↑	1	-	0	0	1	0	0	0	0	1	(21h)	Display Inversion On
GAMSET	0	0	↑	1	-	0	0	1	0	0	1	1	0	(26h)	Gamma Curve Select
		1	↑	1	-	-	-	-	-	GC3	GC2	GC1	GC0		-
DISPOFF	0	0	↑	1	-	0	0	1	0	1	0	0	0	(28h)	Display Off
DISPON	0	0	↑	1	-	0	0	1	0	1	0	0	1	(29h)	Display On
CASET	0	0	↑	1	-	0	0	1	0	1	0	1	0	(2Ah)	Column Address Set
		1	↑	1	-	XS15	XS14	XS13	XS12	XS11	XS10	XS9	XS8		X Address Start: 0≦XS≦X
		1	↑	1	-	XS7	XS6	XS5	XS4	XS3	XS2	XS1	XS0		
		1	↑	1	-	XE15	XE14	XE13	XE12	XE11	XE10	XE9	XE8		X Address End: S≦XE≦X
		1	↑	1	-	XE7	XE6	XE5	XE4	XE3	XE2	XE1	XE0		
RASET	0	0	↑	1	-	0	0	1	0	1	0	1	1	(2Bh)	Row Address Set
		1	↑	1	-	YS15	YS14	YS13	YS12	YS11	YS10	YS9	YS8		Y Address Start: 0≦YS≦Y
		1	↑	1	-	YS7	YS6	YS5	YS4	YS3	YS2	YS1	YS0		
		1	↑	1	-	YE15	YE14	YE13	YE12	YE11	YE10	YE9	YE8		Y Address End:S≦YE≦Y
		1	↑	1	-	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0		
RAMWR	0	0	↑	1	-	0	0	1	0	1	1	0	0	(2Ch	Memory Write
		1	↑	1	-	D7	D6	D5	D4	D3	D2	D1	D0		Write Data
RGBSET	0	0	↑	1	-	0	0	1	0	1	1	0	1	(2Dh	LUT for 4k,65k,262k Color display
		1	↑	1	-	-	-	R005	R004	R003	R002	R001	R000		Red Tone 0
		1	↑	1	-	-	-	:	:	:	:	:	:		:
		1	↑	1	-	-	-	Ra5	Ra4	Ra3	Ra2	Ra1	Ra0		Red Tone “a”
		1	↑	1	-	-	-	G005	G004	G003	G002	G001	G000		Green Tone 0
		1	↑	1	-	-	-	:	:	:	:	:	:		:
		1	↑	1	-	-	-	Gb5	Gb4	Gb3	Gb2	Gb1	Gb0		Green Tone “b”
		1	↑	1	-	-	-	B005	B004	B003	B002	B001	B000		Blue Tone 0
		1	↑	1	-	-	-	:	:	:	:	:	:		:
		1	↑	1	-	-	-	Bc5	Bc4	Bc3	Bc2	Bc1	Bc0		Blue Tone “c”
RAMRD	0	0	↑	1	-	0	0	1	0	1	1	1	0	(2Eh)	Memory Read
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	D7	D6	D5	D4	D3	D2	D1	D0		Read Data

Instruction	Refer	D/CX	WRX	RDX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	Hex	Function
PTLAR	0	0	↑	1	-	0	0	1	1	0	0	0	0	(30h)	Partial Start/End Address Set
		1	↑	1	-	PSL15	PSL14	PSL13	PSL12	PSL11	PSL10	PSL9	PSL8		Partial Start Address (0,1,2, ..P)
		1	↑	1	-	PSL7	PSL6	PSL5	PSL4	PSL3	PSL2	PSL1	PSL0		
		1	↑	1	-	PEL15	PEL14	PEL13	PEL12	PEL11	PEL10	PEL9	PEL8		Partial End Address (0,1,2, .., P)
		1	↑	1	-	PEL7	PEL6	PEL5	PEL4	PEL3	PEL2	PEL1	PEL0		
SCRLAR	10.1.26	0	↑	1	-	0	0	1	1	0	0	1	1	(33h)	Scroll area set
		1	↑	1	-	-	-	-	-	-	-	-	-		Top fixed area (0,1, 2, .., 161)
		1	↑	1	-	TFA7	TFA6	TFA5	TFA4	TFA3	TFA2	TFA1	TFA0		
		1	↑	1	-	-	-	-	-	-	-	-	-		Vertical scroll area (0,1, 2, .., 161)
		1	↑	1	-	VSA7	VSA6	VSA5	VSA4	VSA3	VSA2	VSA1	VSA0		
		1	↑	1	-	-	-	-	-	-	-	-	-		Bottom fixed area (0,1, 2, .., 161)
		1	↑	1	-	BFA7	BFA6	BFA5	BFA4	BFA3	BFA2	BFA1	BFA0		
TEOFF	10.1.27	0	↑	1	-	0	0	1	1	0	1	0	0	(34h)	Tearing effect line off
TEON	08	0	↑	1	-	0	0	1	1	0	1	0	1	(35h)	Tearing Effect Mode Set & on
		1	↑	1	-	-	-	-	-	-	-	-	TEM		Mode1: TEM="0" Mode2: TEM="1"
MADCTL	09	0	↑	1	-	0	0	1	1	0	1	1	0	(36h)	Memory Data Access Control
		1	↑	1	-	MY	MX	MV	ML	RGB	MH	-	-		-
VSCSAD	10.1.30	0	↑	1	-	0	0	1	1	0	1	1	1	(37h)	Scroll Start Address of RAM
		1	1	1	-	-	-	-	-	-	-	-			SSA=0,1,2,…,161
		1	1	1	-	SSA7	SSA6	SSA5	SSA4	SSA3	SSA2	SSA1	SSA0		
IDMOFF	031	0	↑	1	-	0	0	1	1	1	0	0	0	(38h)	Idle Mode Off
IDMON	02	0	↑	1	-	0	0	1	1	1	0	0	1	(39h)	Idle Mode On
COLMOD	03	0	↑	1	-	0	0	1	1	1	0	1	0	(3Ah)	Interface Pixel Format
		1	↑	1	-	-	-	-	-	-	IFPF2	IFPF1	IFPF0		Interface Format
RDID1	04	0	↑	1	-	1	1	0	1	1	0	1	0	(DAh)	Read ID1
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10		Read Parameter
RDID2	05	0	↑	1	-	1	1	0	1	1	0	1	1	(DBh)	Read ID2
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	1	ID26	ID25	ID24	ID23	ID22	ID21	ID20		Read Parameter
RDID3	06	0	↑	1	-	1	1	0	1	1	1	0	0	(DCh)	Read ID3
		1	1	↑	-	-	-	-	-	-	-	-	-		Dummy Read
		1	1	↑	-	ID37	ID36	ID35	ID34	ID33	ID32	ID31	ID30		Read Parameter

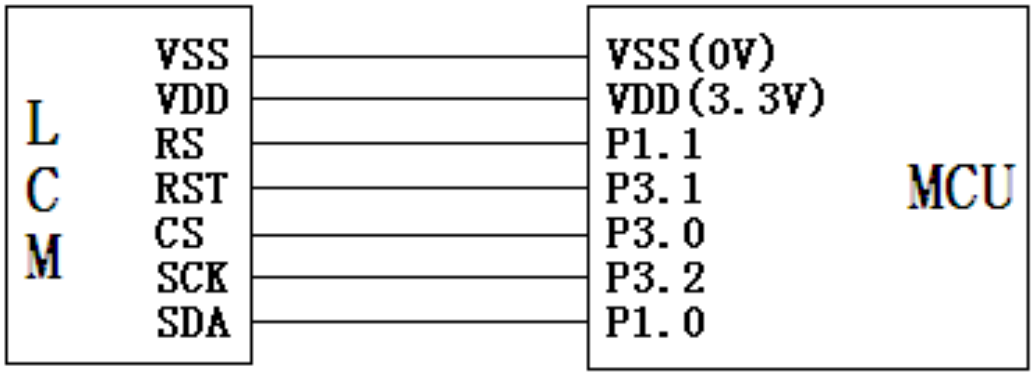
7.2 初始化方法

用户所编的显示程序,开始必须进行初始化,否则模块无法正常显示,过程请参考程序



7.3 程序举例:

液晶模块与 MPU(以 8051 系列单片机为例)接口图如下:并行接口



详细例程请找销售索要

```
//通过指令可以正常竖屏、顺时针旋转 90 度、逆时针旋转 90 度、倒转 180 度竖屏
//本程序针对:JLX096-023 TFT 彩屏
// IC：ST7735S
//版权所有：深圳市晶联讯电子有限公司，网址：www.jlxlcd.cn

#include <reg52.h>
#include <stdio.h>
#include <16080_22.h>
#include <ASCII_TABLE_5X8_8X16_horizontal.h>

//液晶屏 IC 所需要的信号线的接口定义
sbit CS=P3^0;
sbit RST=P3^1;
sbit RS=P1^1;
sbit SCK=P3^2;
sbit SDA=P1^0;
sbit key=P2^0;          //P2.0 口与 GND 之间接一个按键

void delay(long int i)
{
    long int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<110;k++);
}

//等待按键
void waitkey()
{
repeat:
    if(key==1) goto repeat;
    else delay(1000);
}

/*写指令到 LCD 模块*/
void transfer_command(int data1)
{
    char i;
    CS=0;
    RS=0;
    for(i=0;i<8;i++)
    {
        SCK=0;
        if(data1&0x80) SDA=1;
        else SDA=0;
```

```
        SCK=1;
        data1<<=1;
    }
}

/*写数据到 LCD 模块*/
void transfer_data(uchar data1)
{
    char i;
    CS=0;
    RS=1;
    for(i=0;i<8;i++)
    {
        SCK=0;
        if(data1&0x80) SDA=1;
        else SDA=0;
        SCK=1;
        data1<<=1;
    }
}

//连写 2 个字节（即 16 位）数据到 LCD 模块
void transfer_data_16(int data2)
{
    transfer_data(data2>>8);
    transfer_data(data2);
}

//LCD 初始化
void LCD_initial()
{
    delay(50);
    RST=0;                  //低电平：复位
    delay(1);
    RST=1;                  //高电平：复位结束
    delay(10);
    //开始初始化：
    transfer_command(0x11);
    delay(10);
//  transfer_command(0x21); //Display Inversion On

    transfer_command(0xb1);
    transfer_data(0x05);
    transfer_data(0x3A);
    transfer_data(0x3A);
```

```
transfer_command(0xb2);
transfer_data(0x05);
transfer_data(0x3A);
transfer_data(0x3A);

transfer_command(0xb3);
transfer_data(0x05);
transfer_data(0x3A);
transfer_data(0x3A);
transfer_data(0x05);
transfer_data(0x3A);
transfer_data(0x3A);

transfer_command(0xb4);
transfer_data(0x03);

transfer_command(0xc0);
transfer_data(0x62);
transfer_data(0x02);
 transfer_data(0x04);

transfer_command(0xc1);
transfer_data(0xc0);

transfer_command(0xc2);
transfer_data(0x0D);
transfer_data(0x00);

transfer_command(0xc3);
transfer_data(0x8D);
transfer_data(0x6a);

transfer_command(0xc4);
transfer_data(0x8D);
transfer_data(0xee);

transfer_command(0xc5);
transfer_data(0x0e);

transfer_command(0xe0);
transfer_data(0x10);
transfer_data(0x0E);
transfer_data(0x02);
transfer_data(0x03);
transfer_data(0x0E);
transfer_data(0x07);
```

```
 transfer_data(0x02);
 transfer_data(0x07);
transfer_data(0x0A);
transfer_data(0x12);
transfer_data(0x27);
transfer_data(0x37);
transfer_data(0x00);
transfer_data(0x0D);
transfer_data(0x0E);
transfer_data(0x10);

transfer_command(0xe1);
transfer_data(0x10);
transfer_data(0x0E);
transfer_data(0x03);
transfer_data(0x03);
transfer_data(0x0F);
transfer_data(0x06);
transfer_data(0x02);
transfer_data(0x08);
transfer_data(0x0A);
transfer_data(0x13);
transfer_data(0x26);
transfer_data(0x36);
transfer_data(0x00);
transfer_data(0x0D);
transfer_data(0x0E);
transfer_data(0x10);

 transfer_command(0x3a);
transfer_data(0x05);

 transfer_command(0x36);   //行扫描顺序，列扫描顺序，横放/竖放 ,RGB/BGR
transfer_data(0x08);

transfer_command(0x2a);    //定义 X 地址的开始及结束位置
transfer_data(0x00);
transfer_data(0x1A);
transfer_data(0x00);
transfer_data(0x69);

transfer_command(0x2b);     //定义 Y 地址的开始及结束位置
transfer_data(0x00);
transfer_data(0x01);
 transfer_data(0x00);
transfer_data(0xA0);
```

```
    transfer_command(0x29);      //开显示
}

//定义窗口坐标：开始坐标（XS,YS)以及窗口大小（x_total,y_total)
//垂直或竖向显示的地址函数
void lcd_address_v(int XS,int YS,int x_total,int y_total)
{
     int XE,YE;
     XS+=23;
     YS-=1;
    XS=XS;
     YS=YS;
     XE=XS+x_total-1;
     YE=YS+y_total-1;
     transfer_command(0x2a);     // 设置 X 开始及结束的地址
     transfer_data_16(XS);  // X 开始地址(16 位)
     transfer_data_16(XE);  // X 结束地址(16 位)
     transfer_command(0x2b);     // 设置 Y 开始及结束的地址
     transfer_data_16(YS);  // Y 开始地址(16 位)
     transfer_data_16(YE);  // Y 结束地址(16 位
     transfer_command(0x2c);    // 写数据开始
}

//定义窗口坐标：开始坐标（XS,YS)以及窗口大小（x_total,y_total)
//水平或横向显示的地址函数
void lcd_address_h(int XS,int YS,int x_total,int y_total)
{
     int XE,YE;
     XS-=1;
     YS+=23;
    XS=XS;
     YS=YS;
     XE=XS+x_total-1;
     YE=YS+y_total-1;
     transfer_command(0x2a);     // 设置 X 开始及结束的地址
     transfer_data_16(XS);  // X 开始地址(16 位)
     transfer_data_16(XE);  // X 结束地址(16 位)
     transfer_command(0x2b);     // 设置 Y 开始及结束的地址
     transfer_data_16(YS);  // Y 开始地址(16 位)
     transfer_data_16(YE);  // Y 结束地址(16 位
     transfer_command(0x2c);    // 写数据开始
}

//将单色的 8 位的数据（代表 8 个像素点）转换成彩色的数据传输给液晶屏
void mono_transfer_data(int mono_data,int font_color,int back_color)
```

```
{
    int i;
    for(i=0;i<8;i++)
    {
        if(mono_data&0x80)
        {
            transfer_data_16(font_color);    //当数据是 1 时，显示字体颜色
        }
        else
        {
            transfer_data_16(back_color);    //当数据是 0 时，显示底色
        }
        mono_data<<=1;
    }
}

//显示单一色彩
void display_color(int XS,int YS,int x_total,int y_total,int color,int Vertical)
{
    int i,j;
    if(Vertical==1)
    {
        lcd_address_v(XS,YS,x_total,y_total);
    }
    else
        lcd_address_h(XS,YS,x_total,y_total);

    for(i=0;i<160;i++)
    {
        for(j=0;j<80;j++)
        {
            transfer_data_16(color);
        }
    }
}

//显示一幅全屏彩图
void display_image(uchar *dp)
{
    uchar i,j;
    lcd_address_v(1,1,80,160);
    for(i=0;i<160;i++)
    {
        for(j=0;j<80;j++)
        {
            transfer_data(*dp);              //传一个像素的图片数据的高位
```

```
            dp++;
            transfer_data(*dp);                 //传一个像素的图片数据的低位
            dp++;
        }
    }
}

void display_black(void)
{
    int i,j,k;
    transfer_command(0x2c);     // 写数据开始
    for(i=0;i<80;i++)
    {
        transfer_data_16(0xffff);
    }
    for(i=0;i<158;i++)
    {
        for(k=0;k<1;k++)
        {
            transfer_data_16(0xffff);
        }
        for(j=0;j<78;j++)
        {
            transfer_data_16(0x0000);
        }
        for(k=0;k<1;k++)
        {
            transfer_data_16(0xffff);
        }
    }
    for(i=0;i<160;i++)
    {
        transfer_data_16(0xffff);
    }
}

//显示 32x32 点阵的汉字，或相当于 16x16 点阵的图像。温馨提示，数据指针*dp 是字符型数据（char *dp）
void disp_32x32(int x,int y,char *dp,int font_color,int back_color) //int x X 轴坐标，int y,Y 轴坐标
{
    int i,j;
    lcd_address_v(x,y,32,32);
    for(i=0;i<32;i++)
    {
        for(j=0;j<4;j++)
        {
            mono_transfer_data(*dp,font_color,back_color);
```

```
            dp++;
        }
    }
}

//显示 16x16 点阵的汉字，或相当于 16x16 点阵的图像。温馨提示，数据指针*dp 是字符型数据（char *dp）
void disp_16x16(int x,int y,char *dp,int font_color,int back_color) //int x X 轴坐标，int y,Y 轴坐标
{
    int i,j;
    lcd_address_v(x,y,16,16);
    for(i=0;i<2;i++)
    {
        for(j=0;j<16;j++)
        {
            mono_transfer_data(*dp,font_color,back_color);
            dp++;
        }
    }
}

void display_chinese_16x16(int x,int y,uchar *text,int font_color,int back_color,int Vertical)
{
    uchar i,j,k;
    uint address;
    j = 0;
    while(text[j] != '\0')                          //'\0' 字符串结束标志
    {
        i = 0;
        address = 1;
        while(Chinese_horizontal_text_16x16[i] > 0x7e)  // >0x7f 即说明不是 ASCII 码字符
        {
            if(Chinese_horizontal_text_16x16[i] == text[j])
            {
                if(Chinese_horizontal_text_16x16[i + 1] == text[j + 1])
                {
                    address = i * 16;
                    break;
                }
            }
            i += 2;
        }

        if(address != 1)// 显示汉字
        {   if(Vertical ==1)
            {
                lcd_address_v(x,y,16,16);
```

```
            }
            else
                lcd_address_h(x,y,16,16);
            for(i=0;i<2;i++)
            {
                for(k = 0; k <16; k++)
                {
                mono_transfer_data(Chinese_horizontal_code_16x16[address],font_color,back_color);
                address++;
                }
            }
            j+=2;
        }
        else              //显示空白字符
        {
            if(Vertical ==1)
            {
                lcd_address_v(x,y,16,16);
            }
            else
                lcd_address_h(x,y,16,16);
            for(i = 0; i <2; i++)
            {
            for(k = 0; k < 16; k++)
            {
            mono_transfer_data(0x00,font_color,back_color);
            }
            }
            j+=2;
        }
        x=x+16;
    }
}

//显示 8x16 点阵的字符串
void disp_char_8x16(int x,int y,char *text,int font_color,int back_color,int Vertical)
{
    int i=0,j,k;

    while(text[i]>0x00)
    {
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;

            if(Vertical ==1)
```

```
            {
                lcd_address_v(x,y,8,16); //y-2 是为了字母与汉字显示对齐.
            }
            else
                lcd_address_h(x,y,8,16); //y-2 是为了字母与汉字显示对齐.

            for(k=0;k<16;k++)
            {
                mono_transfer_data(ascii_table_8x16[j*16+k],font_color,back_color);    //        这        个
"ascii_table_8x16[]"这个数组在"ASCII_TABLE_5X8_8X16_horizontal.h"里
            }
            x+=8;
            i++;
        }
        else
        i++;
    }
}

void LCD_ShowString(int x,int y,uchar *text,int font_color,int back_color,int Vertical)
{
    uchar temp[3];
    uchar i=0;

    while(text[i] !='\0')
    {
        if(x>160){x=1;y+=16;}
        if(y>80)break;
        if(text[i]>0x7e)
        {
            temp[0]=text[i];
            temp[1]=text[i+1];
            temp[2]='\0';
            display_chinese_16x16(x,y,temp,font_color,back_color,Vertical);
            x +=16;
            i +=2;
        }
        else
        {
            temp[0]=text[i];
            temp[1]='\0';
            disp_char_8x16(x,y,temp,font_color,back_color,Vertical);
            x +=8;
            i++;
        }
```

```
    }
}

//将单色的 8 位的数据的高 5 位（代表 5 个像素点）转换成彩色的数据传输给液晶屏
void mono_data_out_5x8(char mono_data,int font_color,int back_color)
{
    int i;
    for(i=0;i<6;i++)                    //显示 6 列，因为 5x8 点阵的字中间最好是隔 1 列，美观一点
    {
        if(mono_data&0x80)
        {
            transfer_data_16(font_color);    //当数据是 1 时，显示字体颜色
        }
        else
        {
            transfer_data_16(back_color);    //当数据是 0 时，显示底色
        }

        mono_data<<=1;
    }
}

//显示 5x8 点阵的字符串
void disp_string_5x8(int x,int y,uchar *text,int font_color,int back_color)
{
    uint i=0,j,k;
    while(text[i]>0x00)
    {
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;
            lcd_address_v(x,y,6,8);
            for(k=0;k<8;k++)
            {
                mono_data_out_5x8(ascii_table_5x8[j*8+k],font_color,back_color); //这个"ascii_table_5x8[]"这个
数组在"ASCII_TABLE_5X8_8X16_horizontal.h"
            }
            x+=6;
            i++;
        }
        else
        i++;
    }
}

//主程序
```

```
void main(void)
{
    LCD_initial();                  //初始化
    while(1)
    {
//===============竖屏显示=======================//
         transfer_command(0x36);   //行扫描顺序，列扫描顺序，横放/竖放 ,RGB/BGR
        transfer_data(0xc8);
         display_color(1,1,80,160,white,1);
         display_color(1,1,80,26,red,1);
         disp_16x16(10,5,jing1,yellow,red);
         disp_16x16(36,5,lian1,yellow,red);
         disp_16x16(62,5,xun1, yellow,red);
         display_color(1,27,80,38,green,1);
         disp_32x32(24,30,jing,blue,green);
         display_color(1,65,80,38,blue,1);
         disp_32x32(24,68,lian,white,blue);
         display_color(1,103,80,38,yellow,1);
         disp_32x32(24,106,xun,red,yellow);
         display_color(1,141,80,20,brown,1);
         disp_char_8x16(1,143,"JLX096-023",green ,brown,1);     //在（x，y）位置开始，显示 ASCII 字符串
         waitkey();

//===============横屏显示=======================//
         transfer_command(0x36);   //行扫描顺序，列扫描顺序，横放/竖放 ,RGB/BGR
        transfer_data(0xa8);

         display_color(1,1,160,80,white,0);
         display_color(1,1,160,20,red,0);
         display_chinese_16x16(10,2,"晶联讯电子有限公司",yellow,red,0);
         display_color(1,21,160,20,green,0);
         disp_char_8x16(40,23,"JLX096-023",black,green,0);
         display_color(1,41,160,20,blue,0);
         display_chinese_16x16(16,43,"视区",white,blue,0);
         disp_char_8x16(48,43,":10.8x21.7mm",white,blue,0);
         display_color(1,61,160,20,yellow,0);
         display_chinese_16x16(16,63,"点阵",red,yellow,0);
         disp_char_8x16(48,63,":80(RGB)x160",red,yellow,0);
         waitkey();

         display_color(1,1,160,80,blue,0);
         LCD_ShowString(1,1,"JLX096-023 是 0.96 寸 IPS 显示模组,拥有 80*160 分辨率采用 4 线制 SPI 通信方式,IPS 全视角超宽可视范围.",white,blue,0);
         waitkey();

//===============竖屏显示=======================//
```

```
        transfer_command(0x36);   //行扫描顺序，列扫描顺序，横放/竖放 ,RGB/BGR
       transfer_data(0xc8);
        display_image(pic1);       //显示单幅彩图，编码是通过专用取模工具获取的，取模工具请找客服人员索取。
        waitkey();
        display_color(1,1,80,160,red,1);
        waitkey();
        display_color(1,1,80,160,green,1);
        waitkey();
        display_color(1,1,80,160,blue,1);
        waitkey();
        display_color(1,1,80,160,white,1);
        waitkey();
        display_black();
        waitkey();

    }
}
```