

JLX177-015-BN 使用说明书

(不带字库 IC)

目 录

序号	内 容 标 题	页 码
1	概述	2
2	特点	2
3	外形及接口引脚功能	3~4
4	基本原理	4
5	技术参数	4
6	时序特性	5~6
7	指令功能及硬件接口与编程案例	7~末页

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 JLX177-015-BN 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。

JLX177-015-BN 可以显示 160 列*128 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. JLX177-015-BN 彩色图像型点阵液晶模块的特性

2.1 结构轻、薄、带背光。

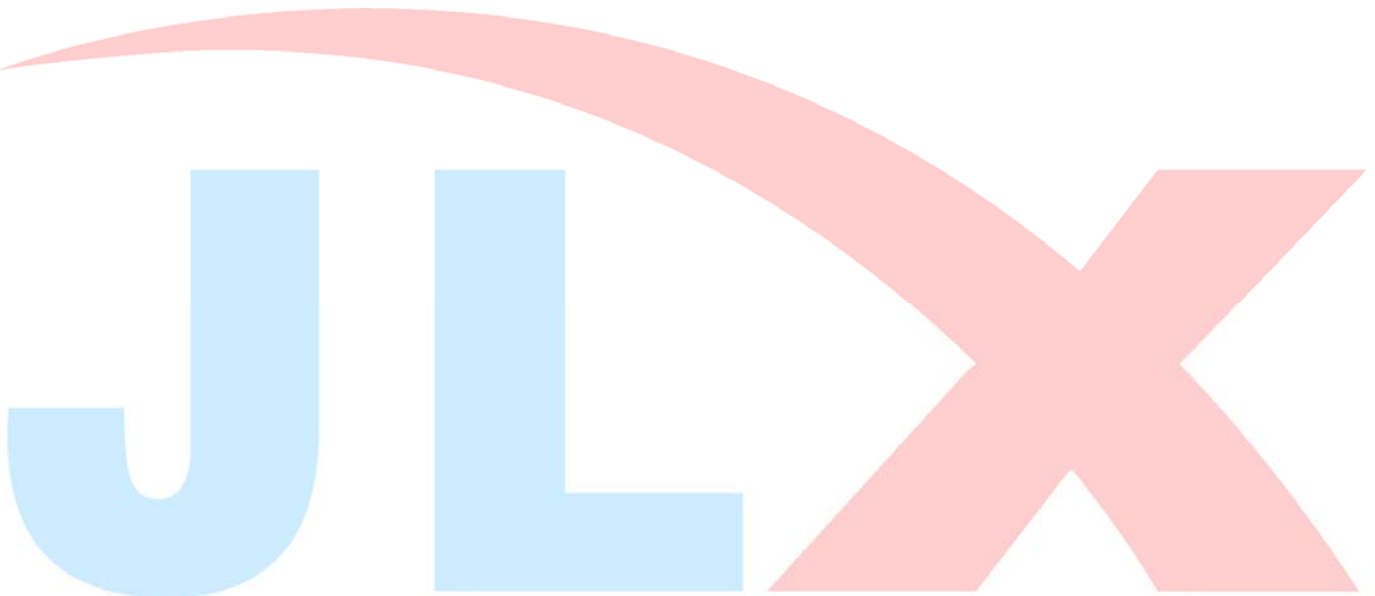
2.2 IC 采用 ST7735S, 功能强大, 稳定性好

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

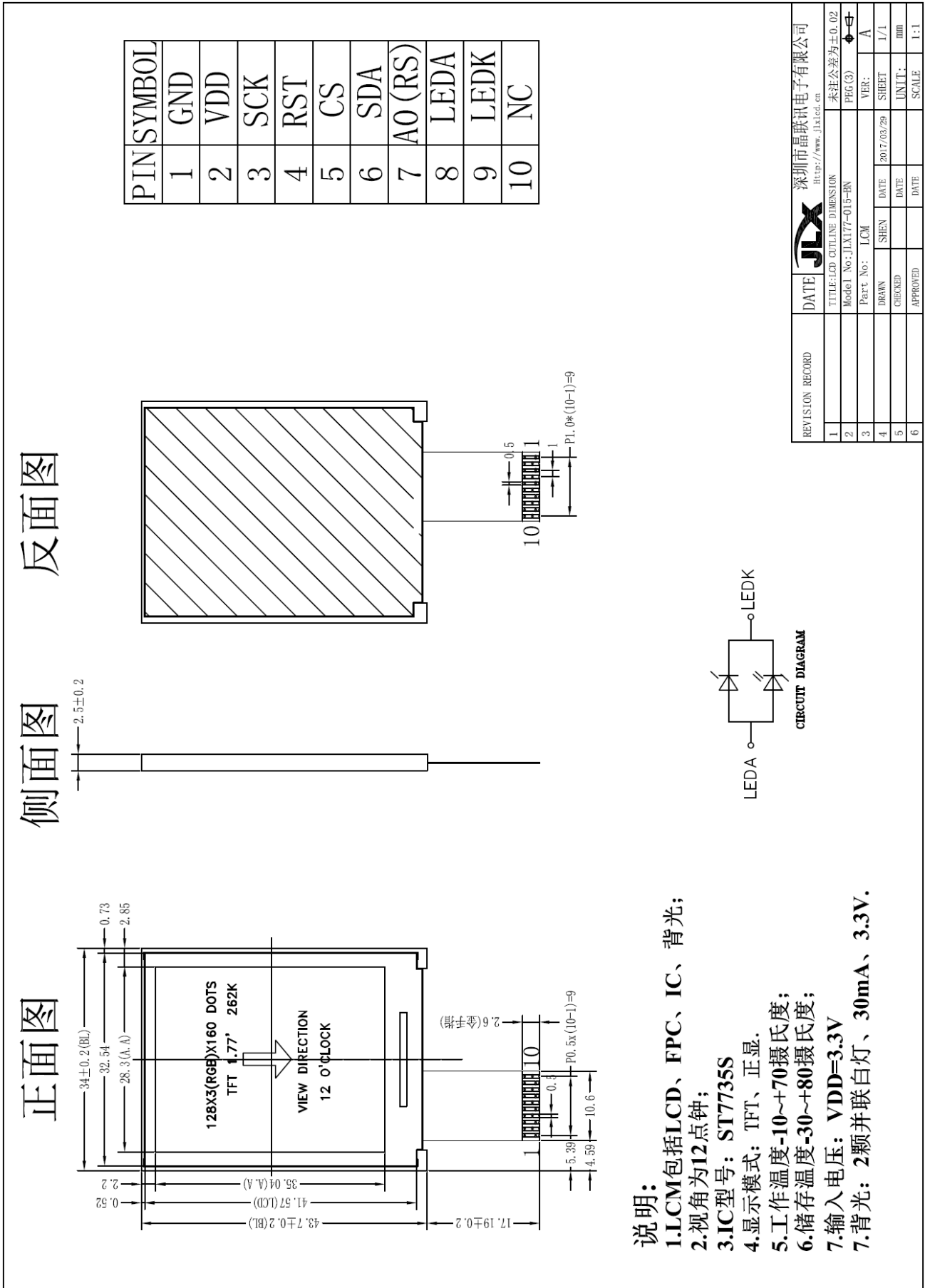


图 1. 外形尺寸

模块的接口引脚功能

引线号	符号	名称	功能
1	GND	接地	0V
2	VDD	电路电源	3.3V
3	SCK	I/O	串行时钟
4	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
5	CS	片选	低电平片选
6	SDA	I/O	串行数据
7	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")
8	LDEA	背光电源正极	背光电源正极, 接 3.0V
9	LEDK	背光电源负极	背光电源负极, 接地
10	NC	空脚	空脚

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 160×128 点阵, 160 个列信号与驱动 IC 相连, 128 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: 16~40mA (LED 灯数共 2 颗, 每颗灯是 8~20 mA)

工作电压: 3.0V。

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	3.3	V
工作温度		-20		+70	°C
储存温度		-30		+80	°C

表 2：最大极限参数

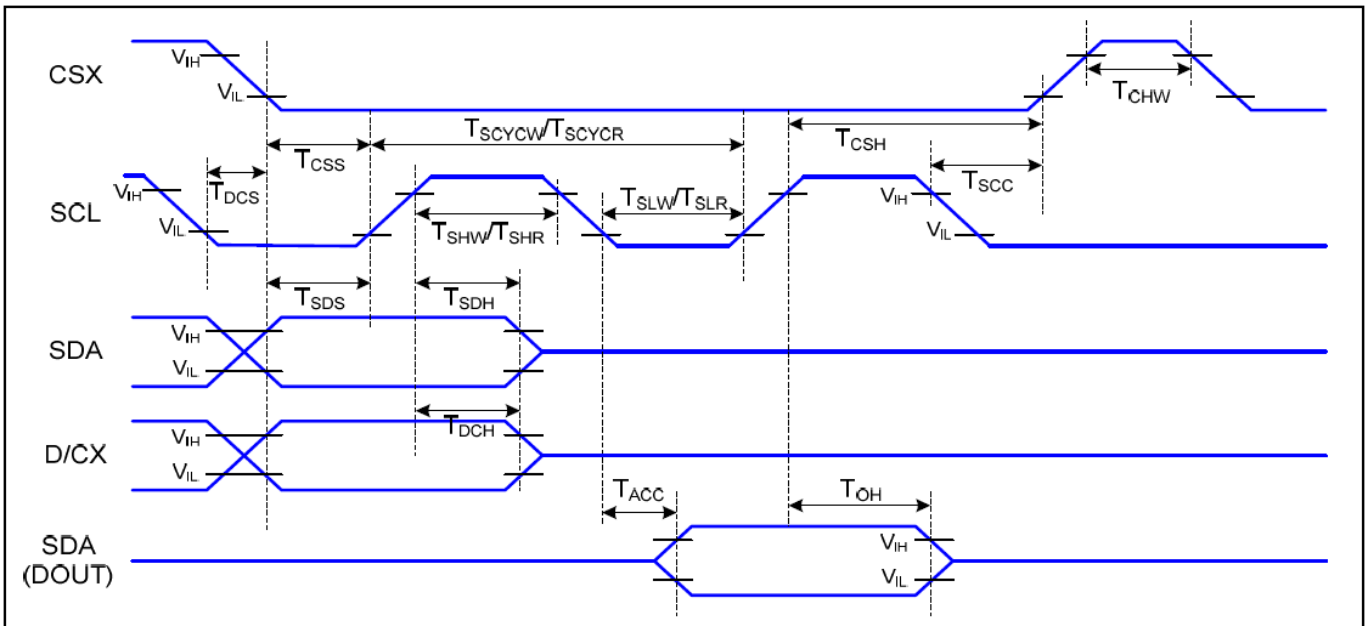
5.2 直流 (DC) 参数

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	3.3	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

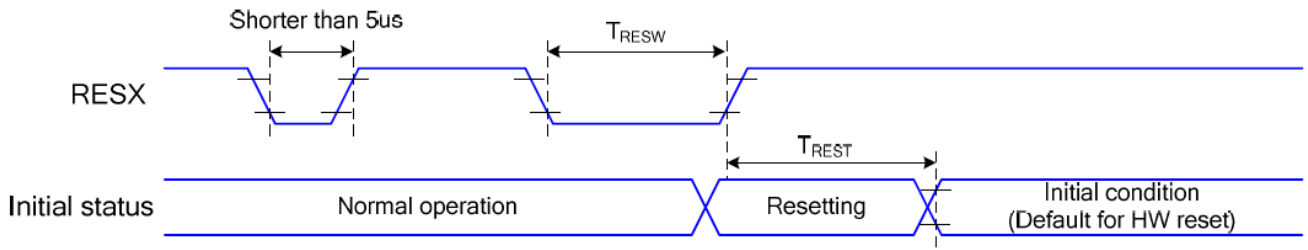
详见 IC 资料 “ST7735S”，请找相关客服人员索要。



$T_a=25\text{ }^\circ\text{C}$, $V_{DDI}=1.65\sim 3.7\text{V}$, $V_{DD}=2.3\sim 4.8\text{V}$

Signal	Symbol	Parameter	MIN	MAX	Unit	Description
CSX	T_{CSS}	Chip Select Setup Time (Write)	TBD		ns	
	T_{CSH}	Chip Select Hold Time (Write)	TBD		ns	
	T_{CSS}	Chip Select Setup Time (Read)	TBD		ns	
	T_{SCC}	Chip Select Hold Time (Read)	TBD		ns	
	T_{CHW}	Chip Select "H" Pulse Width	TBD		ns	
SCL	T_{SCYCW}	Serial Clock Cycle (Write)	TBD		ns	-Write Command & Data Ram
	T_{SHW}	SCL "H" Pulse Width (Write)	TBD		ns	
	T_{SLW}	SCL "L" Pulse Width (Write)	TBD		ns	
	T_{SCYCR}	Serial Clock Cycle (Read)	TBD		ns	-Read Command & Data Ram
	T_{SHR}	SCL "H" Pulse Width (Read)	TBD		ns	
	T_{SLR}	SCL "L" Pulse Width (Read)	TBD		ns	
D/CX	T_{DCS}	D/CX Setup Time	TBD		ns	
	T_{DCH}	D/CX Hold Time	TBD		ns	
SDA (DIN) (DOUT)	T_{SDS}	Data Setup Time	TBD		ns	For Maximum $CL=30\text{pF}$ For Minimum $CL=8\text{pF}$
	T_{SDH}	Data Hold Time	TBD		ns	
	T_{ACC}	Access Time	TBD	TBD	ns	
	T_{OH}	Output Disable Time	TBD	TBD	ns	

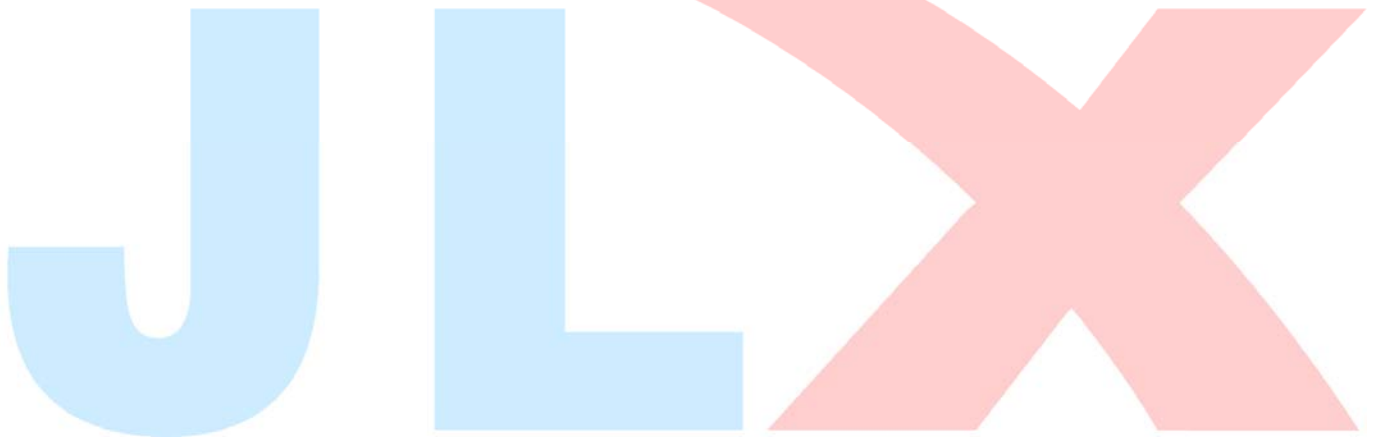
6.1 电源启动后复位的时序要求 (RESET CONDITION AFTER POWER UP):



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: $0 \leq XS \leq X$
		1	↑	1	-	XS7	XS6	XS5	XS4	XS3	XS2	XS1	XS0		
		1	↑	1	-	XE15	XE14	XE13	XE12	XE11	XE10	XE9	XE8		X Address End: $S \leq XE \leq X$
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: $0 \leq YS \leq Y$
		1	↑	1	-	YS7	YS6	YS5	YS4	YS3	YS2	YS1	YS0		
		1	↑	1	-	YE15	YE14	YE13	YE12	YE11	YE10	YE9	YE8		Y Address End: $S \leq YE \leq Y$
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
		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	-	-	-	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	
		1	↑	1	-	PSL15	PSL14	PSL13	PSL12	PSL11	PSL10	PSL9	PSL8		Partial Start Address (0,1,2, ..P)	
		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)
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	-	-	-	-	-	-	-	-	-		SSA=0,1,2,...,161	
		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

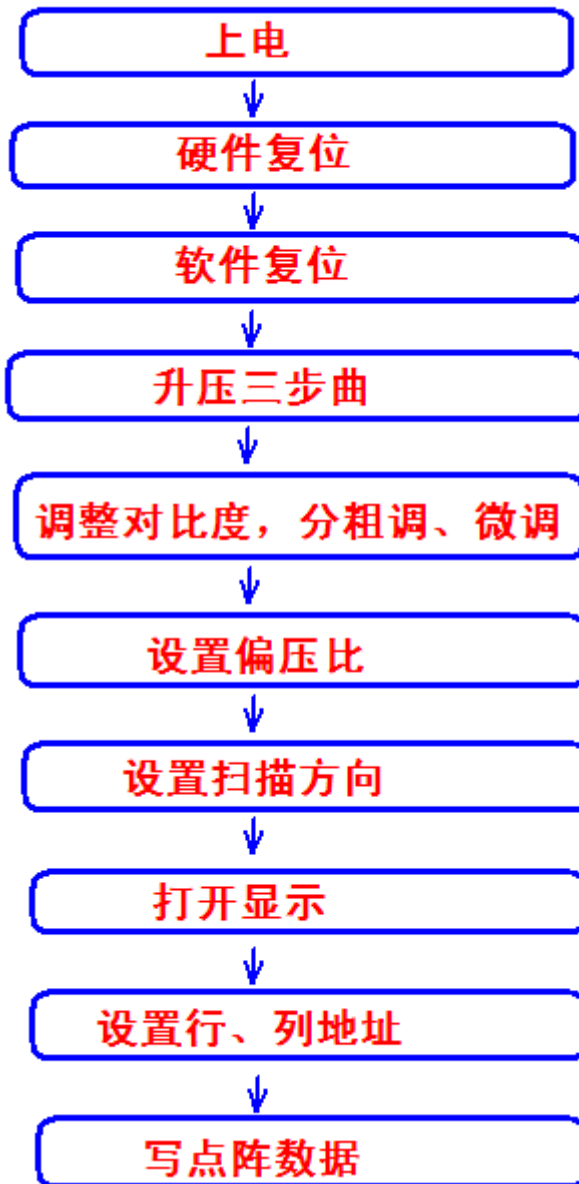
点亮液晶模块的步骤

硬件准备:
开发板 (或专门设计的主板)、单片机、电源、连接线、仿真器或程序下载器 (又名烧录器)

正确地接线
根据说明书正确地与开发板连接, 连接的线包括: 液晶模块电源线、背光电源线、IO端口 (接口)
IO端口包括: 并口时: CS、RESET、RW、E、RS、D0--D7, 串口时: CS、SCLK、SDA、RESET、RS

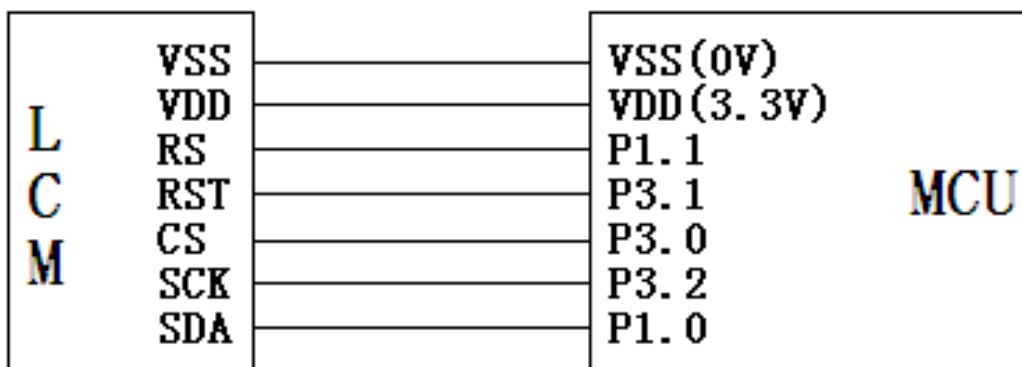
编写软件
背光给合适的直流电可以点亮, 但液晶屏里面没有程序, 只给电不能让液晶屏显示 (我们通常说“点亮”), 程序须另外编写, 并烧录 (下载) 到单片机里液晶模块才能工作。

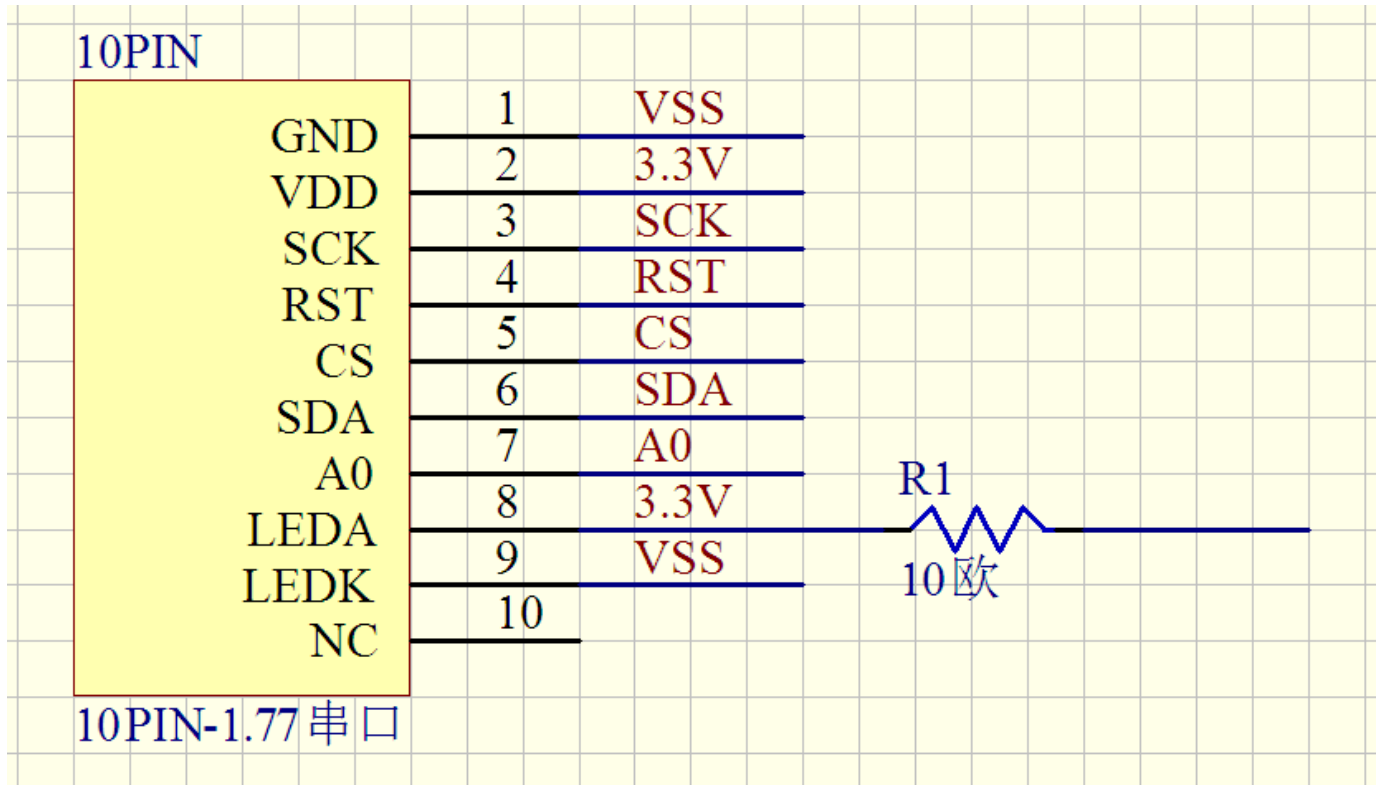
点亮液晶模块的编程步骤



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:JLX177-015-BN 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 <128160_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
//=====
```

```
#define uchar unsigned char
#define uint unsigned int
#define ulong unsigned long
```

```
//定义彩屏旋转方向
#define normal 0xc8
```

```

#define CW90 0x68
#define CCW90 0xa8
#define CW180 0xc8

#define red 0xf800 //定义红色
#define blue 0x001f //定义蓝色
#define green 0x07e0 //定义绿色
#define deep_green 0x0600 //定义深绿色
#define white 0xffff //定义白色
#define black 0x0000 //定义黑色
#define orange 0xfc08 //定义橙色
#define yellow 0xffe0 //定义黄色
#define pink 0xf3f3 //定义粉红色
#define purple 0xald6 //定义紫色
#define brown 0x8200 //定义棕色
#define gray 0xc618 //定义灰色

uchar code jing[]={ //横向取模
/*-- 文字： 晶 --*/
/*-- 宋体 23； 此字体下对应的点阵为：宽 x 高=31x31 --*/
/*-- 宽度不是 8 的倍数，现调整为：宽度 x 高度=32x31 --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x00,0x00,0xF0,0x07,0x80,
0x00,0xFF,0xFF,0x80,0x00,0xF0,0x07,0x00,0x00,0xF0,0x07,0x00,0x00,0xF0,0x07,0x00,
0x00,0xFF,0xFF,0x00,0x00,0xF0,0x07,0x00,0x00,0xF0,0x07,0x00,0x00,0xF0,0x07,0x00,
0x00,0xFF,0xFF,0x00,0x00,0xF0,0x07,0x00,0x00,0xC0,0x00,0x00,0x18,0x0C,0x40,0x30,
0x1F,0xFE,0x7F,0xF8,0x1C,0x1E,0x70,0x78,0x1C,0x1C,0x70,0x70,0x1C,0x1C,0x70,0x70,
0x1C,0x1C,0x70,0x70,0x1C,0x1C,0x70,0x70,0x1F,0xFC,0x7F,0xF0,0x1C,0x1C,0x70,0x70,
0x1C,0x1C,0x70,0x70,0x1C,0x1C,0x70,0x70,0x1C,0x1C,0x70,0x70,0x1F,0xFC,0x7F,0xF0,
0x1C,0x1C,0x70,0x70,0x18,0x00,0x60,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
};

uchar code lian[]={ //横向取模
/*-- 文字： 联 --*/
/*-- 宋体 23； 此字体下对应的点阵为：宽 x 高=31x31 --*/
/*-- 宽度不是 8 的倍数，现调整为：宽度 x 高度=32x31 --*/
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x81,0xC0,0x00,0x0D,0xC1,0xF0,
0x00,0x1E,0xE1,0xC0,0x7F,0xFB,0xF3,0x80,0x1C,0x70,0x73,0x80,0x1C,0x70,0x77,0x00,
0x1C,0x70,0x06,0x70,0x1C,0x70,0x0E,0xF8,0x1C,0x7F,0xFF,0xD8,0x1F,0xF0,0x1C,0x00,
0x1C,0x70,0x1C,0x00,0x1C,0x70,0x1C,0x00,0x1C,0x70,0x1C,0x18,0x1C,0x70,0x1C,0x3C,
0x1C,0x7F,0xFF,0xFE,0x1F,0xF0,0x1E,0x00,0x1C,0x70,0x1E,0x00,0x1C,0x70,0x3E,0x00,
0x1C,0x70,0x3F,0x00,0x1C,0x7E,0x3B,0x00,0x1D,0xF0,0x3B,0x80,0x1F,0xF0,0x73,0x80,
0x7C,0x70,0x71,0xC0,0x30,0x70,0xE1,0xE0,0x00,0x71,0xC0,0xF0,0x00,0x73,0x80,0x7C,
0x00,0x77,0x00,0x3C,0x00,0x7C,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
};

uchar code xun[]={ //横向取模

```

```
/*- 文字: 讯 -*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=31x31 -*/
/*- 宽度不是 8 的倍数, 现调整为: 宽度 x 高度=32x31 -*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0x00, 0x00, 0x80,
0x07, 0x80, 0x01, 0xC0, 0x03, 0xFF, 0xFF, 0xE0, 0x03, 0xC0, 0x01, 0xC0, 0x01, 0x81, 0xE1, 0xC0,
0x00, 0x01, 0xE1, 0xC0, 0x00, 0x01, 0xC1, 0xC0, 0x00, 0x01, 0xC1, 0xC0, 0x00, 0x01, 0xC1, 0xC0,
0x03, 0xC1, 0xC1, 0xC0, 0x7F, 0xE1, 0xCD, 0xC0, 0x03, 0x81, 0xDF, 0xC0, 0x03, 0xBF, 0xFB, 0xC0,
0x03, 0x81, 0xC1, 0xC0, 0x03, 0x81, 0xC1, 0xC0, 0x03, 0x81, 0xC1, 0xC0, 0x03, 0x81, 0xC1, 0xC0,
0x03, 0x81, 0xC1, 0xCC, 0x03, 0x9D, 0xC1, 0xCC, 0x03, 0xB9, 0xC1, 0xEC, 0x03, 0xF1, 0xC0, 0xFC,
0x03, 0xE1, 0xC0, 0xFC, 0x03, 0xE1, 0xC0, 0xFC, 0x03, 0xC1, 0xC0, 0x7C, 0x01, 0x81, 0xC0, 0x3C,
0x00, 0x01, 0xC0, 0x1C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

uchar code jing1[]={//横向取模
/*- 文字: 晶 -*/
/*- 宋体 12; 此字体下对应的点阵为: 宽 x 高=16x16 -*/
0x00, 0x00, 0x0F, 0xF0, 0x08, 0x10, 0x0F, 0xF0, 0x08, 0x10, 0x0F, 0xF0, 0x08, 0x10, 0x00, 0x00,
0x7E, 0x7E, 0x42, 0x42, 0x7E, 0x7E, 0x42, 0x42, 0x42, 0x42, 0x7E, 0x7E, 0x42, 0x42, 0x00, 0x00,
};

uchar code lian1[]={//横向取模
/*- 文字: 联 -*/
/*- 宋体 12; 此字体下对应的点阵为: 宽 x 高=16x16 -*/
0x01, 0x08, 0xFE, 0x8C, 0x44, 0x48, 0x44, 0x50, 0x7F, 0xFE, 0x44, 0x20, 0x44, 0x20, 0x7C, 0x20,
0x47, 0xFE, 0x44, 0x20, 0x4E, 0x20, 0xF4, 0x20, 0x44, 0x50, 0x04, 0x48, 0x04, 0x86, 0x05, 0x04};

uchar code xun1[]={//横向取模
/*- 文字: 讯 -*/
/*- 宋体 12; 此字体下对应的点阵为: 宽 x 高=16x16 -*/
0x40, 0x00, 0x27, 0xF8, 0x31, 0x08, 0x21, 0x08, 0x01, 0x08, 0xF1, 0x08, 0x17, 0xE8, 0x11, 0x08,
0x11, 0x08, 0x11, 0x08, 0x11, 0x08, 0x11, 0x0A, 0x15, 0x0A, 0x19, 0x0A, 0x11, 0x04, 0x00, 0x00};

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

```
/*写数据到 LCD 模块*/  
void transfer_data(int 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=data1<<=1;  
    }  
}
```

//连写 2 个字节（即 16 位）数据到 LCD 模块

```
void transfer_data_16(uint data2)  
{  
    transfer_data(data2>>8);  
    transfer_data(data2);  
}
```

//LCD 初始化

```
void LCD_initial()  
{  
    delay(50);    //50ms  
    RST=0;       //低电平：复位  
    delay(10);   //10ms
```

```
RST=1;          //高电平：复位结束
delay(10);      //10ms
//开始初始化：
transfer_command(0x11); //退出睡眠

transfer_command(0xb1); //帧速率控制：在正常模式下（全色）
transfer_data(0x05);
transfer_data(0x3a);
transfer_data(0x3a);

transfer_command(0xb2); //帧速率控制（空闲模式/8色）
transfer_data(0x05);
transfer_data(0x3a);
transfer_data(0x3a);

transfer_command(0xb3); //帧速率控制（部分模式/全色）
transfer_data(0x05);
transfer_data(0x3a);
transfer_data(0x3a);
transfer_data(0x05);
transfer_data(0x3a);
transfer_data(0x3a);

transfer_command(0xb4); //显示反转控制
transfer_data(0x03);

transfer_command(0xc0); //电源控制 1
transfer_data(0x62);
transfer_data(0x02);
transfer_data(0x04);

transfer_command(0xc1); //电源控制 2
transfer_data(0xc0);

transfer_command(0xc2); //电源控制 3
transfer_data(0x0D);
transfer_data(0x00);

transfer_command(0xc3); //电源控制 4
transfer_data(0x8d);
transfer_data(0x6a);

transfer_command(0xc4); //电源控制 5
transfer_data(0x8d);
transfer_data(0xee);
```



```
transfer_command(0xc5); //VCOM 控制 1
transfer_data(0x12);
```

```
transfer_command(0xE0);
transfer_data(0x03);
transfer_data(0x1B);
transfer_data(0x12);
transfer_data(0x11);
transfer_data(0x3F);
transfer_data(0x3A);
transfer_data(0x32);
transfer_data(0x34);
transfer_data(0x2F);
transfer_data(0x2B);
transfer_data(0x30);
transfer_data(0x3A);
transfer_data(0x00);
transfer_data(0x01);
transfer_data(0x02);
transfer_data(0x05);
```

```
transfer_command(0xE1);
transfer_data(0x03);
transfer_data(0x1B);
transfer_data(0x12);
transfer_data(0x11);
transfer_data(0x32);
transfer_data(0x2F);
transfer_data(0x2A);
transfer_data(0x2F);
transfer_data(0x2E);
transfer_data(0x2C);
transfer_data(0x35);
transfer_data(0x3F);
transfer_data(0x00);
transfer_data(0x00);
transfer_data(0x01);
transfer_data(0x05);
```

```
transfer_command(0x3a); //界面像素格式
transfer_data(0x05);
```

```
transfer_command(0x36); //行扫描顺序，列扫描顺序，横放/竖放
transfer_data(normal); //MX=1（行地址顺序：从左到右），MY=1（列地址顺序：从上到下），MV=0（竖放），ML=0（纵向刷新：从上到下），RGB=1（依次为 RGB），MH=0（横向刷新顺序：从左到右）
//定义：“normal”就是“0xc8”——正常竖放；
```

```
//定义：“CW180”就是“0x08”---在正常竖放基础上转 180 度竖放；  
//定义：“CCW90”就是“0xa8”---在竖放基础上逆时针转 90 度横放；  
//定义：“CW90”就是“0x68”---在竖放基础上顺转 90 度横放；
```

```
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)
```

```
void lcd_address(int XS,int YS,int x_total,int y_total)  
{  
    int XE,YE;  
    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 i, j;
lcd_address(XS, YS, x_total, y_total);
for(i=0; i<128; i++)
{
    for(j=0; j<160; j++)
    {
        transfer_data_16(color);
    }
}

//显示一幅全屏彩图
void display_image(uchar *dp)
{
    uchar i, j;
    lcd_address(0, 0, 128, 160);
    for(i=0; i<160; i++)
    {
        for(j=0; j<128; 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<128; 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<126; 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(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(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++;
        }
    }
}
```

//显示 8x16 点阵的字符串

```
void disp_string_8x16(int x,int y,char *text,int font_color,int back_color)
{
    int i=0,j,k;
```

```

while(text[i]>0x00)
{
    if((text[i]>=0x20)&&(text[i]<=0x7e))
    {
        j=text[i]-0x20;
        lcd_address(x, y, 8, 16);
        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++;
}
}

```

//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(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)
    {
        display_color(0, 0, 128, 160, blue);
        disp_16x16(8+16*2, 8, jing1, red, white);
        disp_16x16(8+16*3, 8, lian1, red, white);
        disp_16x16(8+16*4, 8, xun1, red, white);
        disp_32x32(16+32*0, 32, jing, red, white);
        disp_32x32(16+32*1, 32, lian, red, white);
        disp_32x32(16+32*2, 32, xun, red, white);
        disp_string_8x16(0, 70, "  JLX177-015  ", yellow, blue); //在 (x, y) 位置开始, 显示 ASCII 字符串
        disp_string_8x16(0, 88, "  1.77 TFT  ", yellow, blue); //在 (x, y) 位置开始, 显示 ASCII 字符串
        disp_string_5x8( 0, 109, "  JLX177-015, 1.77TFT", yellow, blue); //在 (x, y) 位置开始, 显示 ASCII 字符串
        disp_string_5x8( 0, 120, "  www.jlxlcd.cn", yellow, blue); //在 (x, y) 位置开始, 显示 ASCII 字符串
        disp_string_5x8( 0, 133, "  tel:0755-29784961", yellow, blue); //在 (x, y) 位置开始, 显示 ASCII 字符串
        waitkey();
        display_color(0, 0, 128, 160, red);
        waitkey();
        display_color(0, 0, 128, 160, green);
        waitkey();
        display_color(0, 0, 128, 160, blue);
        waitkey();
        display_black();
        waitkey();
        display_image(pic1);           //显示单幅彩图, 编码是通过专用取模工具获取的, 取模工具请找销售人员索要。
        waitkey();
    }
}

```

