

# JLX128128G-560-BN 使用说明书

## 目 录

序号	内 容 标 题	页码
1	概述	2
2	字符型模块的特点	2
3	外形及接口引脚功能	3~4
4	基本原理	4~5
5	技术参数	5
6	时序特性	6~7
7	指令功能及硬件接口与编程案例	8~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 JLX128128G-560 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。

JLX128128G-560 可以显示不大于  $128 \times 128$  点阵单色或 4 灰度级的图片, 或显示 8 个  $\times 8$  行=64 个的  $16 \times 16$  点阵的汉字, 或显示 16 个  $\times 8$  行=128 个的  $8 \times 16$  点阵的英文、数字、符号。或显示 21 个  $\times 16$  行的  $5 \times 8$  点阵的英文、数字、符号。

## 2. JLX128128G-560 图像型点阵液晶模块的特性

1.1 结构牢: 背光带有挡墙, 焊接式 FPC。

1.2 IC 采用 ST7571, 功能强大, 稳定性好。

1.3 功耗低:  $1 - 100\text{mW}$  (不带背光  $1\text{mW} < 3.3\text{V}@0.3\text{mA}>$ , 带背光不大于  $100\text{mW} < 3.3\text{V}@30\text{mA}>$ );

1.4 显示内容:

- $128 \times 128$  点阵单色图片或 4 灰度级的图片,

- 或显示 8 个  $\times 8$  行=64 个的  $16 \times 16$  点阵的汉字, 按照  $12 \times 12$  点阵汉字来计算可显示 10 字/行  $\times 10$  行。

- 或显示 16 个  $\times 8$  行=128 个的  $8 \times 16$  点阵的英文、数字、符号。

- 或显示 21 个  $\times 16$  行的  $5 \times 8$  点阵的英文、数字、符号。;

- 可选用  $16 \times 16$  点阵或其他点阵的图片来自编汉字也可配合晶联讯字库 IC (JLX-GB2312) 来显示汉字。

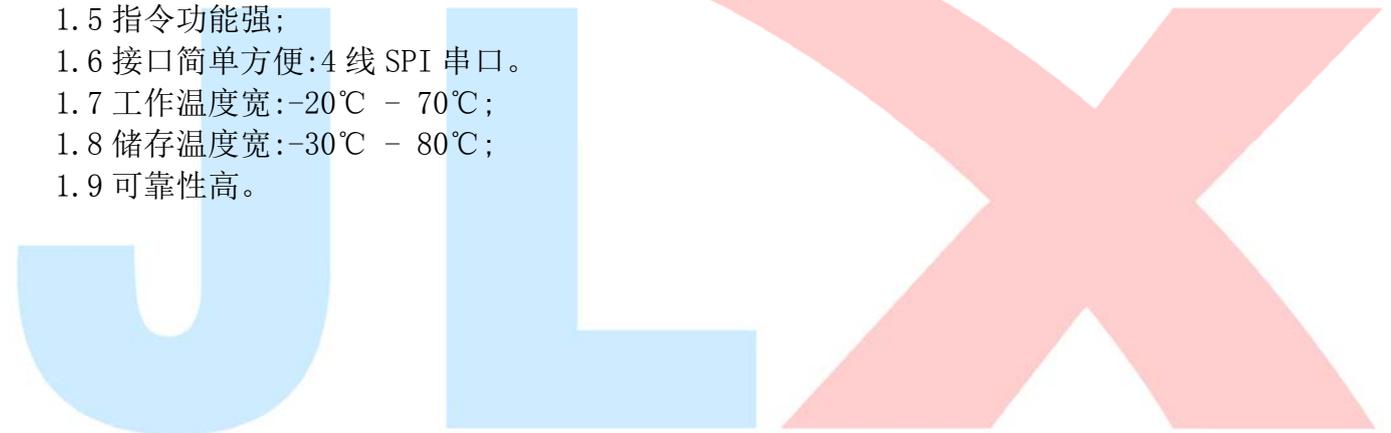
1.5 指令功能强;

1.6 接口简单方便: 4 线 SPI 串口。

1.7 工作温度宽:  $-20^\circ\text{C} - 70^\circ\text{C}$ ;

1.8 储存温度宽:  $-30^\circ\text{C} - 80^\circ\text{C}$ ;

1.9 可靠性高。



### 3. 外形尺寸及接口引脚功能

#### 3.1 外形尺寸图

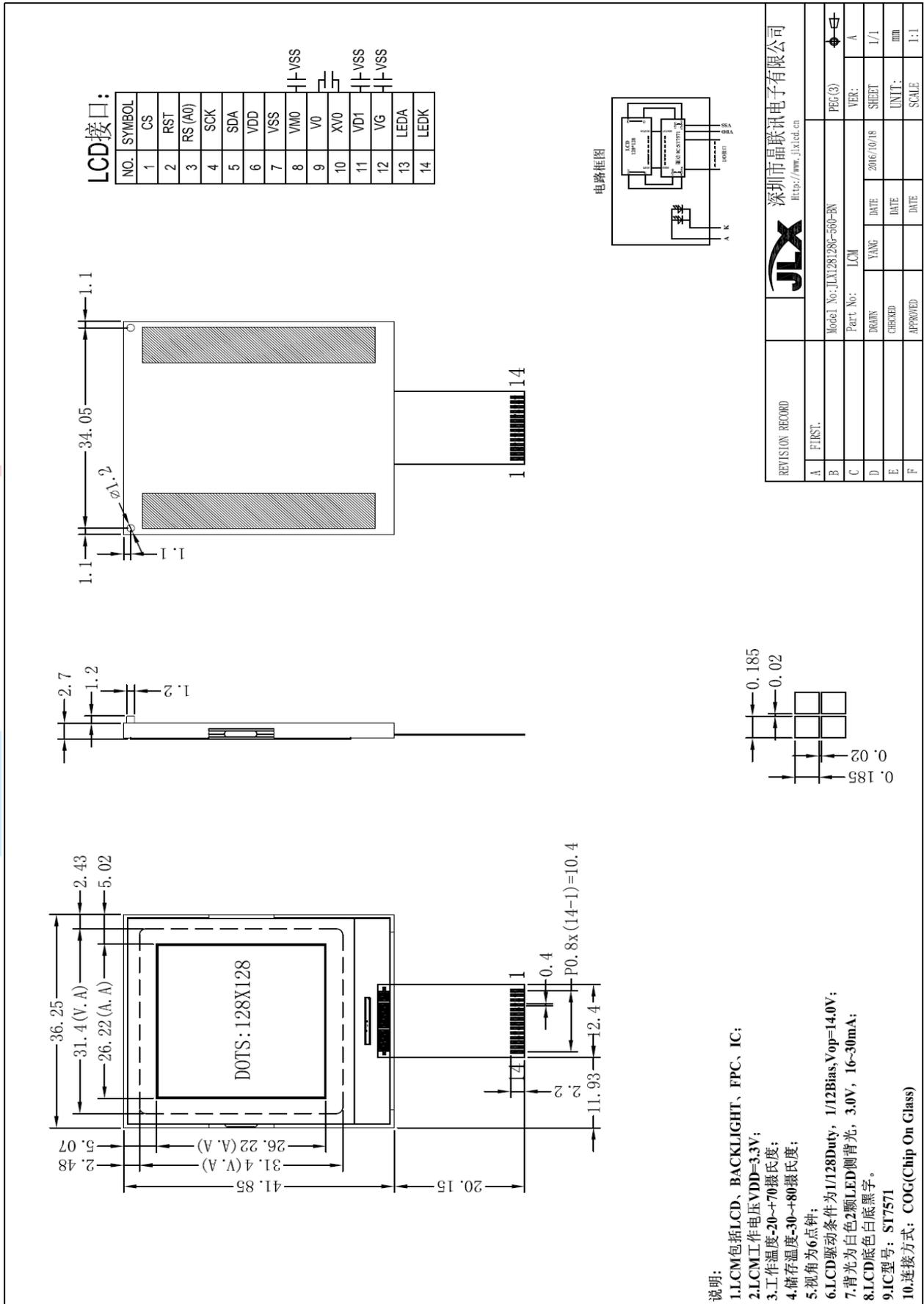


图 1. 外形尺寸

### 3.2 模块的接口引脚功能

#### 3.2.1 接口引脚功能

引线号	符号	名称	功能															
1	CS	片选	低电平片选															
2	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															
3	RS	寄存器选择信号	H: 数据寄存器 0: 指令寄存器															
4	SCK	I/O	串行时钟															
5	SDA	I/O	串行数据															
6	VDD	供电电源正极	供电电源正极															
7	VSS	接地	0V															
8	VM0	倍压电路	<table border="1"> <tr> <td>8</td> <td>VM0</td> <td>   VSS</td> </tr> <tr> <td>9</td> <td>V0</td> <td>  </td> </tr> <tr> <td>10</td> <td>XV0</td> <td>  </td> </tr> <tr> <td>11</td> <td>VD1</td> <td>   VSS</td> </tr> <tr> <td>12</td> <td>VG</td> <td>   VSS</td> </tr> </table> <p>电容为 1uf, 耐压 25V 以上</p>	8	VM0	VSS	9	V0		10	XV0		11	VD1	VSS	12	VG	VSS
8	VM0	VSS																
9	V0																	
10	XV0																	
11	VD1	VSS																
12	VG	VSS																
9	V0	倍压电路																
10	XV0	倍压电路																
11	VD1	倍压电路																
12	VG	倍压电路																
13	LEDA	背光电源正极	接 3.0V															
14	LEDK	背光电源负极	接地															

表 1

## 4. 基本原理

### 4.1 液晶屏 (LCD)

在 LCD 上排列着 128×128 点阵, 128 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)。

### 4.2 工作电路图:

图 1 是 JLX128128G-560 图像点阵型模块的电路框图, 它由驱动 IC ST7571 及几个电阻电容组成。

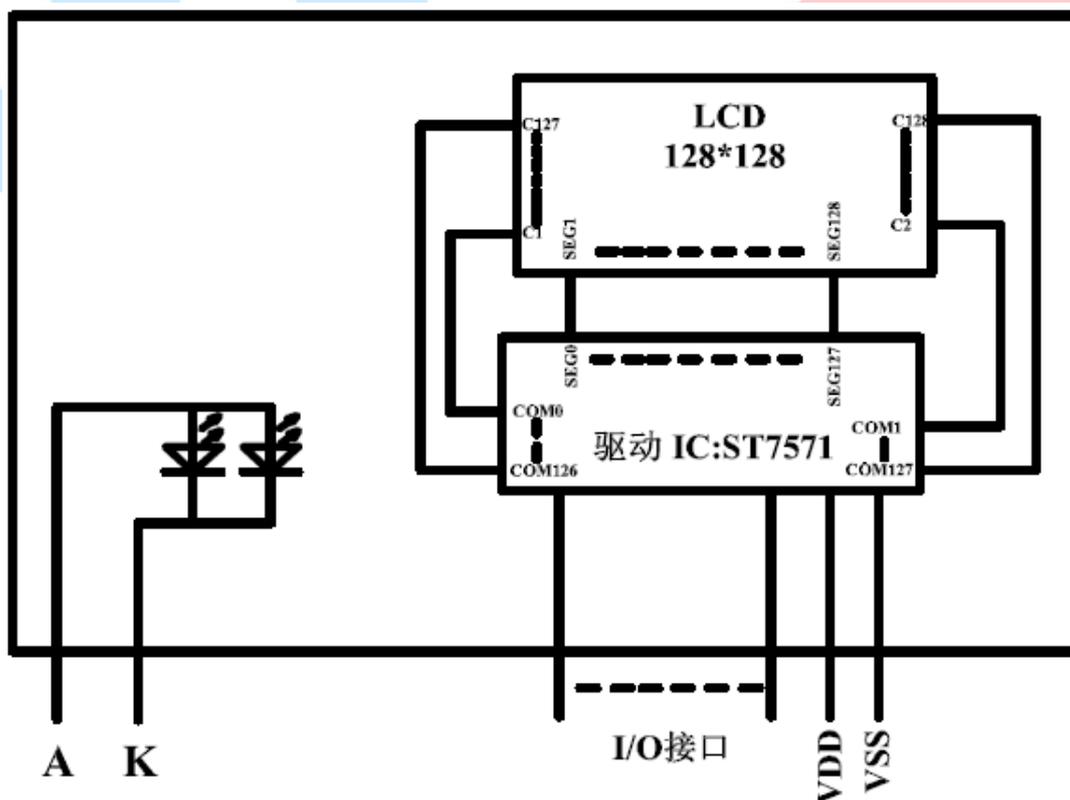


图 2: JLX128128G-560 图像点阵型液晶模块的电路框图

### 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:  $-20\sim+70^{\circ}\text{C}$ ;

存储温度:  $-30\sim+80^{\circ}\text{C}$ ;

背光板为白色。

正常工作电流为:  $16\sim 40\text{mA}$ ;

工作电压: 3.0V (**温馨提示: 3.3V 供电串一个 20 欧电阻, 5.0V 供电串一个 120 欧电阻**)

## 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 - VSS	-0.3		7.0	V
LCD 驱动电压	VDD - V0	VDD - 13.5		VDD + 0.3	V
静电电压		-	-	100	V
工作温度		-20		+70	$^{\circ}\text{C}$
储存温度		-30		+80	$^{\circ}\text{C}$

表 2: 最大极限参数

### 5.2 直流 (DC) 参数

#### 3.3V 供电

名称	符号	测试条件	标准值			单位
			MIN	TYPE	MAX	
工作电压	VIN	3.3V 供电	2.7	3.3	3.6	V
输入高电平	VIH	-	2.2		VDD	V
输入低电平	VIO	-	-0.3		0.6	V
输出高电平	VOH	IOH = 0.2mA	2.4		-	V
输出低电平	VOO	I00 = 1.2mA	-		0.4	V
模块工作电流	IDD	VDD = 3.3V	-		0.3	mA
背光工作电流	ILED	VLED=3.0V	16	30	40	mA

表 3: 直流 (DC) 参数

### 6. 读写时序特性 (AC 参数)

#### 6.1 4 线 SPI 串行接口写时序特性 (AC 参数)

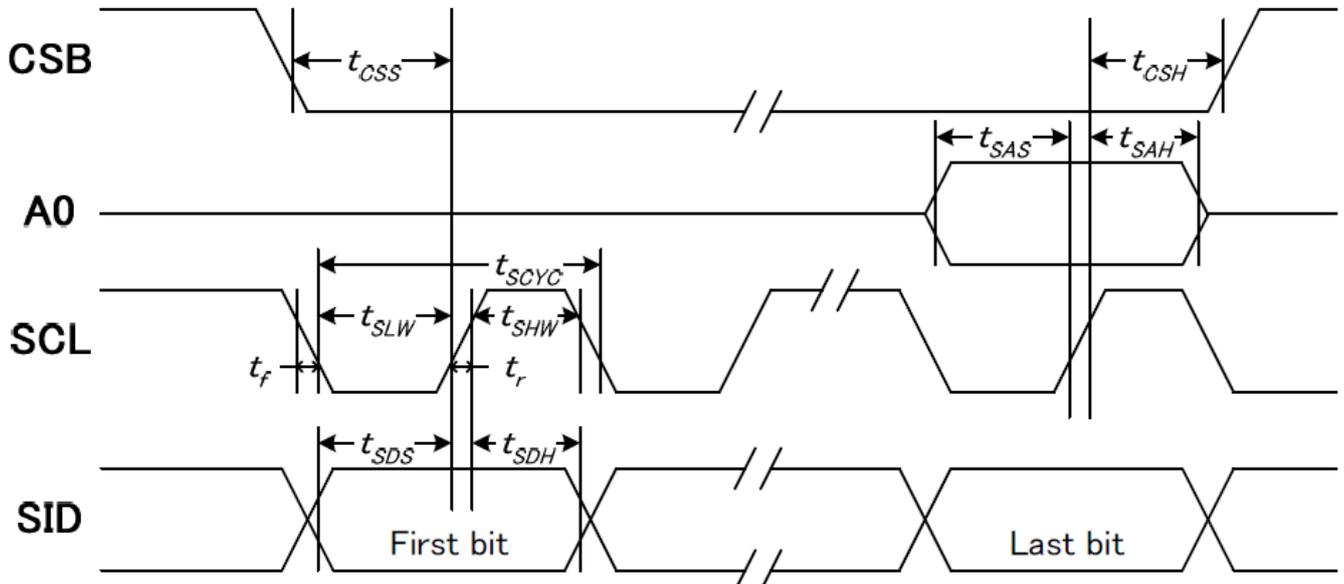


图 3. 从 CPU 写到 ST7571 (Writing Data from CPU to ST7571)

表 4. 写数据到 ST7571 的时序要求

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
4线 SPI 串口时钟周期 (4-line SPI Clock Period)	T <sub>SCYC</sub>	引脚: SCK	200		--	ns
保持SCK高电平脉宽 (SCLK "H" pulse width)	T <sub>SHW</sub>		80		--	ns
保持SCLK低电平脉宽 (SCLK "L" pulse width)	T <sub>SLW</sub>		80		--	ns
地址建立时间 (Address setup time)	T <sub>SAS</sub>	引脚: RS/CD	60		--	ns
地址保持时间 (Address hold time)	T <sub>SAH</sub>		30		--	ns
数据建立时间 (Data setup time)	T <sub>SDS</sub>	引脚: SDA	60		--	ns
数据保持时间 (Data hold time)	T <sub>SDH</sub>		30		--	ns
片选信号建立时间 (CS-SCL time)	T <sub>CSS</sub>	引脚: CS	40			ns
片选信号保持时间 (CS-SCL time)	T <sub>Csh</sub>		100			ns

VDD = 3.0V ± 5%, Ta = 25°C

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

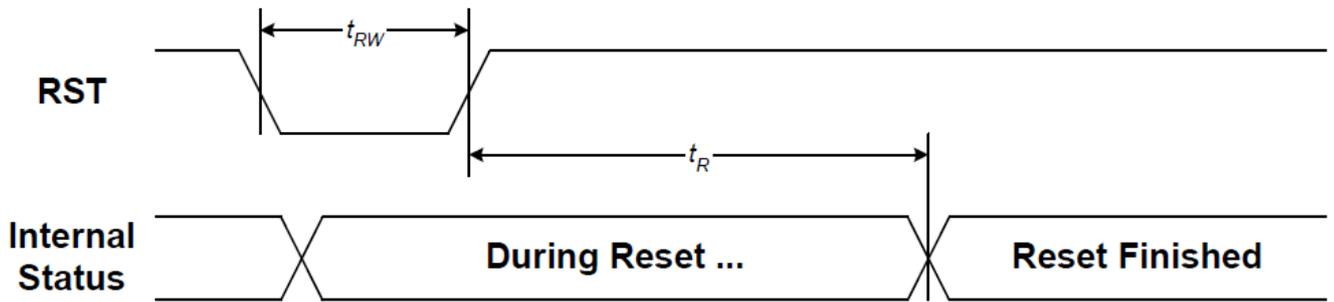
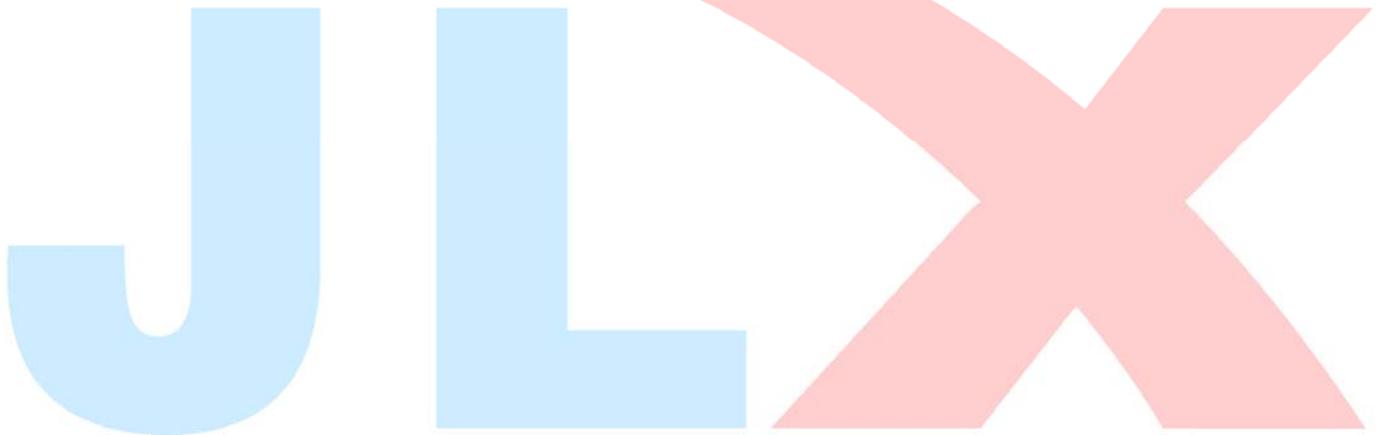


图 4: 电源启动后复位的时序

表 5: 电源启动后复位的时序要求

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
复位时间	$T_R$		120	--	--	us
复位保持低电平的时间	$T_{RW}$		2.0	--	--	ms



7. 指令功能:

7.1 指令表

表 6

指令名称		指令码										说明
		RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
(1) 设置模式 (Set Mode)		0	0	0	0	1	1	1	0	0	0	2 字节的指令 FR [ 3 : 0 ]: 设置帧频 BE [1: 0 ]: 设置增压效率 <b>模式: 0X38, FR 、 BE 0Xb8</b>
		0	0	FR3	FR2	FR1	FR0	BE1	BE0	--	0	
(2) 写显示数据 (Write Display Data)		1	0	写数据							将数据写入 DDRAM	
(3) 设置图标 (Set Icon)		0	0	1	0	1	0	0	0	1	ION	ION = 0: 禁止图标功能 ION = 1: 启用图标功能 并设置页地址= 16
(4) 设置页地址 (Set Page Address)		0	0	1	0	1	1	<b>显示页地址, 共 4 位</b>				设置页地址。每 8 行为一个页, 128 行分为 16 个页, 可设置值为: 0XB0~0XBF 分别对应第一页到第 16 页,
(5) 列地址高 4 位设置 (Set Column Address (MSB))		0	0	0	0	0	1	<b>列地址的高 4 位</b>				高 4 位与低 4 位共同组成列地址, 指定 128 列中的其中一列。比如液晶模块的第 100 列地址十六进制为 0x64, 那么此指令由 2 个字节来表达: 0x16, 0x04
	列地址低 4 位设置 (Set Column Address (LSB))	0	0	0	0	0	0	<b>列地址的低 4 位</b>				
(6) 显示开/关 (Display ON/OFF)		0	0	1	0	1	0	1	1	1	0 1	显示开/关: <b>0XAE: 关, 0XAF: 开</b>
(7) 设置起始行 (Set Display Start Line)		0	0	0	1	0	0	0	0	--	--	设置显示存储器的显示初始行, 可设置值为 <b>0X40~0XBF</b> , 分别代表第 <b>0~63</b> 行, 针对该液晶屏一般设置为 <b>0x40</b>
		0	0	--	<b>显示初始行地址, 共 7 位</b>							
(8) 设置 COM0 (Set COM0)		0	0	0	1	0	0	0	1	--	--	2 字节的指令。 指定 COM 引脚为 COM0 <b>0X44</b>
		0	0	--	C6	C5	C4	C3	C2	C1	C0	
(9) 设置显示 Duty (Set Display Duty)		0	0	0	1	0	0	1	0	--	--	2 字节的指令。 显示设置 Duty
		0	0	L7	L6	L5	L4	L3	L2	L1	L0	
(10) Set N-line Inversion		0	0	0	1	0	0	1	0	--	--	2-byte instruction. Set N-line inversion counter
		0	0	--	--	--	N4	N3	N2	N1	N0	
(11) Release N-line Inversion		0	0	1	1	1	0	0	1	0	0	Exit N-line inversion mode
(12) 正显/反显 (Reverse Display)		0	0	1	0	1	0	0	1	1	0 1	显示正显/反显: <b>0xA6: 常规: 正显 0xA7: 反显</b>
(13) 显示全部点阵 (Entire Display ON)		0	0	1	0	1	0	0	1	0	0 1	显示全部点阵: <b>0xA4: 常规 0xA5: 显示全部点阵</b>

(14)电源控制 (Power Control)	0	0	0	0	1	0	1	电压操作模式选择, 共 3 位			选择内部电压供应操作模式: D2、D1、D0 位分别对应内部升压是否打开(1 为打开, 0 为不打开), 电压调整电路是否打开(1 为打开, 0 为不打开), 电压跟随器是否打开(1 为打开, 0 为不打开)。  通常是 <b>0x2C,0x2E,0x2F</b> 三条指令按顺序紧接着写, 表示依次打开内部升压、电压调整电路、电压跟随器。也可以单写 <b>0x2F</b> , 一次性打开三部分电路。
(15)选择内部电阻比例 (Select Regulator Register)	0	0	0	0	1	0	0	内部电压值电阻设置			选择内部电阻比例 (Rb/Ra): 可以理解为粗调对比度值。可设置范围为: <b>0x20~0x27</b> , 数值越大对比度越浓, 越小越淡
(16) Set Contrast	内部设置液晶电压模式	0	0	1	0	0	0	0	0	1	设置内部电阻微调, 可以理解为微调对比度值, 此两个指令需紧接着使用。上面一条指令 <b>0x81</b> 是不改的, 下面一条指令可设置范围为: <b>0x00~0x3F</b> , 数值越大对比度越浓, 越小越淡
	设置的电压值	0	0	--	--	6 位电压值数据, 0~63 共 64 级					
(17) LCD 偏压比设置(Select LCD bias)	0	0	0	1	0	1	0	B2	B1	B0	设置偏压比: 此液晶: <b>0X54 1/9bias</b>
(18)设定行扫描方向(Set COM Scan Direction)	0	0	1	1	0	0	1	--	--	--	行扫描顺序选择: <b>0XC0</b> : 普通扫描顺序: 从上到下 <b>0XC8</b> : 反转扫描顺序: 从下到上
(19)设定列扫描方向(Set SEG Scan Direction)	0	0	1	0	1	0	0	0	0	1	列扫描顺序选择: <b>0xA0</b> : 常规: 列地址从左到右, <b>0xA1</b> : 反转: 列地址从右到左
(20)开振荡器电路(Oscillator ON)	0	0	1	0	1	0	1	0	1	1	<b>0x8B</b> : 开启内部震荡电路
(21)设置睡眠模式 (Release Power-Save Mode)	0	0	1	0	1	0	1	0	0	1	<b>0XA8</b> : 正常模式 <b>0XA9</b> : 睡眠模式
(22)退出省电模式 (Release Power-Save Mode)	0	0	1	1	1	0	0	0	0	1	<b>0XE1</b> : 退出睡眠模式
(23)RESET	0	0	1	1	1	0	0	0	1	0	<b>0XE2</b> : 软件复位
(24)设置显示数据长度 (Set Display Data Length)	-	-	1	1	1	0	1	0	0	0	2 字节指令, 设定数据仅用在 3-SPI, 此液晶为 4-SPI, 不用此功能
	-	-	DL7	DL6	DL5	DL4	DL3	DL2	DL1	DL0	
(25) 扩展指令1 (Extension Command Set1)	0	0	1	1	1	1	1	1	0	1	<b>0XFD</b>

(26) 扩展指令2 (Extension Command Set2)	0	0	1	1	0	1	0	0	0	1	0XD1
(27) 扩展指令3 (Extension Command Set3)	0	0	0	1	1	1	1	0	1	1	0X7B
扩展指令 1											
(1)增加Vop偏移 (Increase Vop offset)	0	0	0	1	0	1	0	0	0	1	0X51
(2)降低Vop偏移 (Decrease Vop offset)	0	0	0	1	0	1	0	0	1	0	0X52
(3)返回正常模 式 (Return normal mode)	0	0	0	0	0	0	0	0	0	0	0X00
扩展指令 2											
(1)禁用自动读 (Disable autoread)	0	0	1	0	1	0	1	0	1	0	0XAA
(2) 进入EEPROM 模式 (Enter EEPROM mode)	0	0	0	0	0	1	0	0	1	1	0X13
(3)启用阅读模 式 (Enable read mode)	0	0	0	0	1	0	0	0	0	0	0X20
(4) 设置读取脉冲 (Set read pulse)	0	0	0	1	1	1	0	0	0	1	0X71
(5) 退出EEPROM 模式 (Exit EEPROM mode)	0	0	1	0	0	0	0	0	1	1	0X83
(6) 启用擦除模式 (Enable erase mode)	0	0	0	1	0	0	1	0	1	0	0X4A
(7) 设置擦除脉冲 (Set erase pulse)	0	0	0	1	0	1	0	1	0	1	0X55
(8) 启用写入模式 (Enable write mode)	0	0	0	0	1	1	0	1	0	1	0X35
(9) 设置写入脉冲 (Set write pulse)	0	0	0	1	1	0	1	0	1	0	0X6A
(10) 返回正常模 式 (Return normal mode)	0	0	0	0	0	0	0	0	0	0	0X00
扩展指令 3											
(1)设置显示模 式 (Set Color Mode)	0	0	0	0	0	1	0	0	0	1	显示模式选择 0 黑白模式: 0X11, 四灰阶模式: 0X10

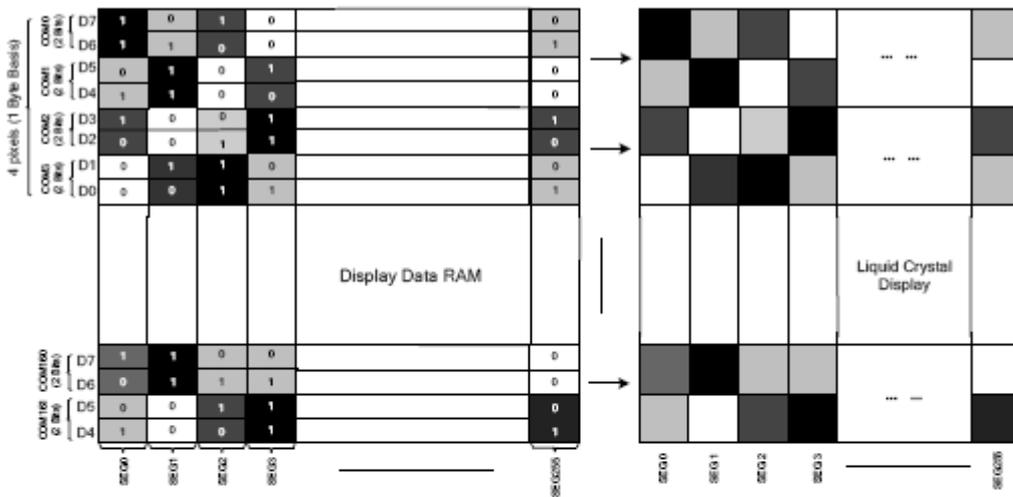
(2)返回正常模式	0	0	0	0	0	0	0	0	0	0	0X00
-----------	---	---	---	---	---	---	---	---	---	---	------

请详细参考 IC 资料”ST7571.PDF

### 7.3 点阵与 DD RAM 地址的对应关系

请留意页的定义: PAGE, 与平时所讲的“页”并不是一个意思, 在此表示 8 个行就是一个“页”, 一个 128\*128 点阵的屏分为 16 个“页”, 从第 0“页”到第 15“页”。

DB7—DB0 的排列方向: 数据是从下向上排列的。最低位 D0 是在最上面, 最高位 D7 是在最下面。每一位 (bit) 数据对应一个点阵, 通常“1”代表点亮该点阵, “0”代表关掉该点阵. 如下图所示:



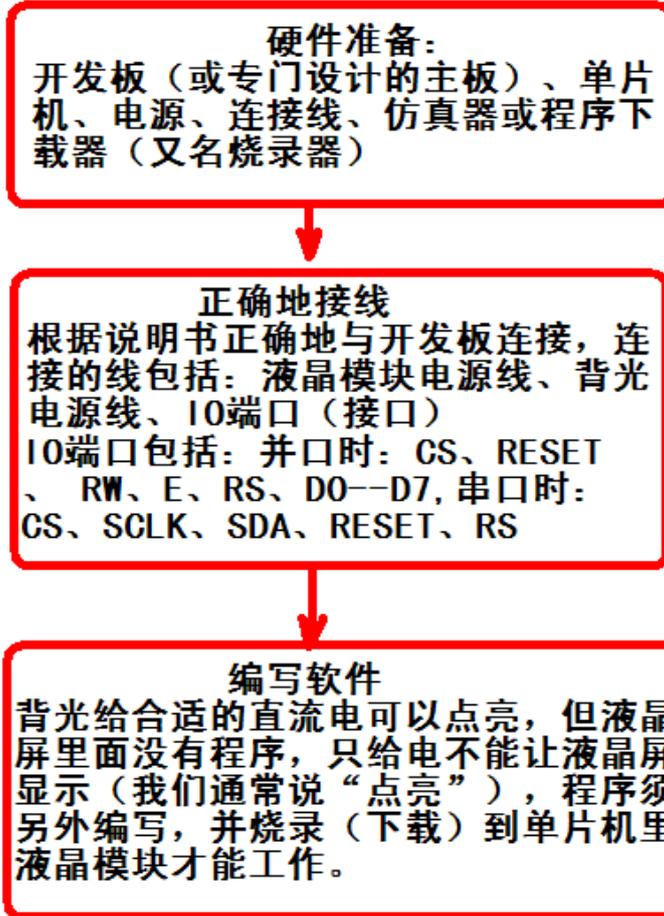
2 Bits Data N=0~3		DDRAM		LCD
D2N+1	D2N			
1	1	1	1	■
0	0	0	0	□
1	0	1	0	■
0	1	0	1	■

Figure 21 DDRAM Mapping (4-Level Gray Scale Mode)

### 7.4 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



#### 7.5 程序举例:

##### 7.5.1 并行接口

液晶模块与 MPU (以 8051 系列单片机为例) 接口图如下:

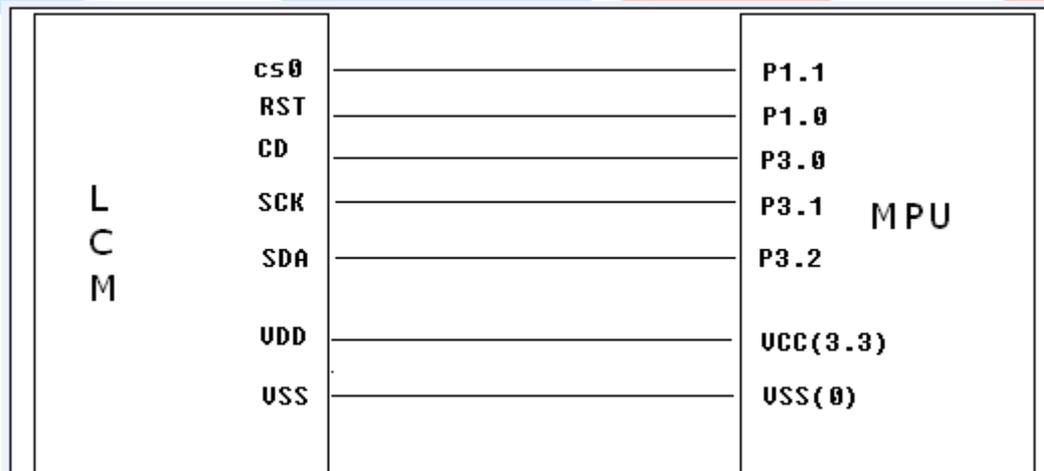
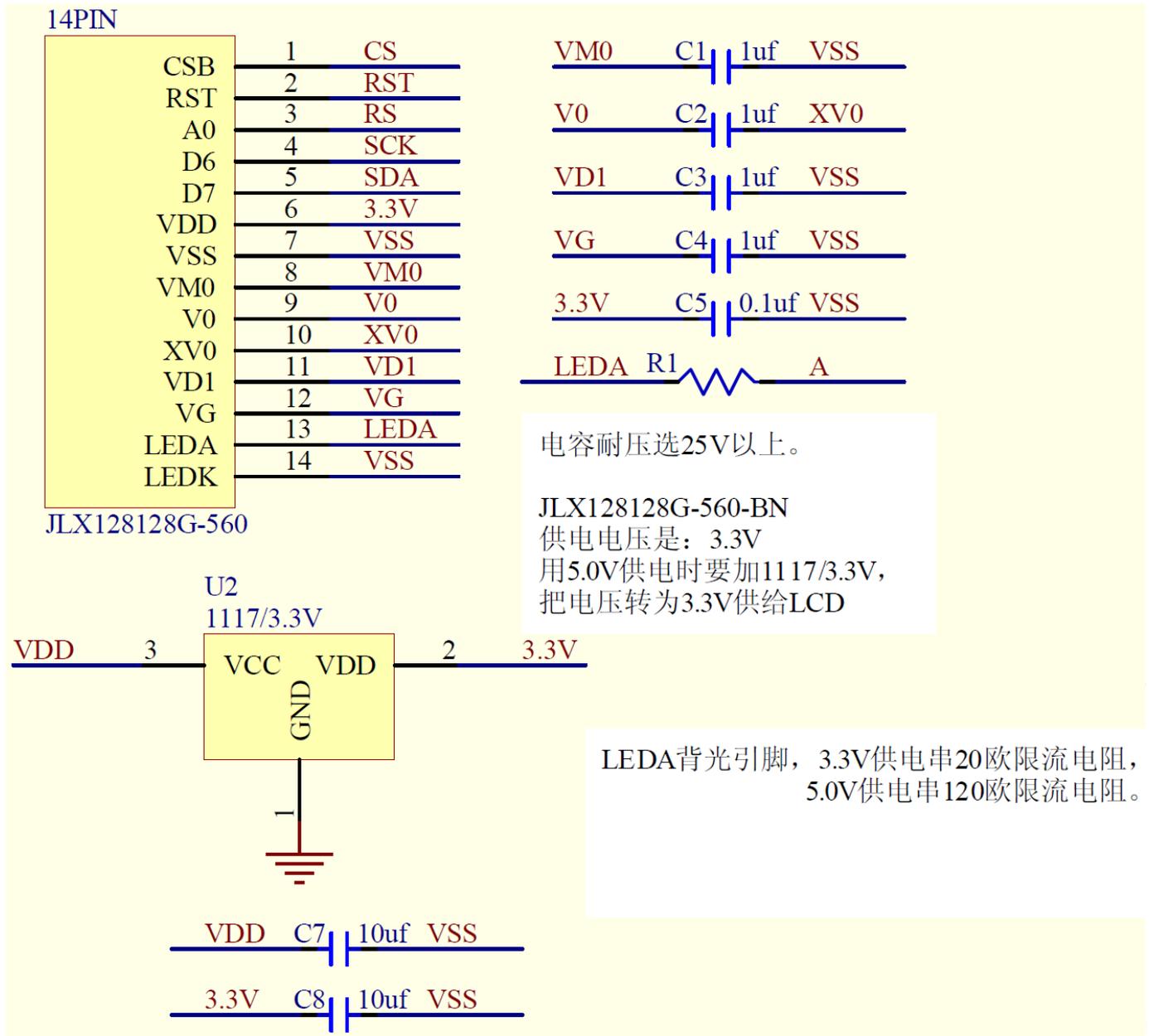


图 5. 串行接口



### 串行接口

7.5.2 以下是串行接口例程序

```
#include <reg51.h>
#include <intrins.h>
#include <Ctype.h>
#include <chinese_code.h>

sbit rs=P3^0; /*接口定义:LCD 的 rs*/
sbit sck=P3^1; /*接口定义:LCD 的 sck*/
sbit sda=P3^2; /*接口定义:LCD 的 sda*/
sbit reset=P1^0; /*接口定义:LCD 的 reset*/
sbit cs0=P1^1; /*接口定义:LCD 的 cs0*/

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

void delay_us(int i);

uchar code bmp1[];
uchar code bmp2[];
uchar code bmp3[];

/*写指令到 LCD 模块*/
void transfer_command(int data1)
{
    char i;
    cs0=0;
    rs=0;
    for(i=0;i<8;i++)
    {
        sck=0;
        if(data1&0x80) sda=1;
        else sda=0;
        sck=1;
        delay_us(1);
        data1<<=1;
    }
    cs0=1;
}

/*写数据到 LCD 模块*/
void transfer_data(int data1)
{
    char i;
    cs0=0;
    rs=1;
    for(i=0;i<8;i++)
    {
        sck=0;
        if(data1&0x80) sda=1;
        else sda=0;
        sck=1;
        data1<<=1;
    }
    cs0=1;
}

```

```
void delay_us(int i)
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<10;k++);
}
```

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

//等待一个按键

```
void waitkey()
{
    repeat:
        if (key==1) goto repeat;
        else;
            delay(1000);
}
```

```
void initial_lcd()
{
```

```
    reset=0;
    delay(100);
    reset=1;
    delay(100);
```

```
    transfer_command(0x2c);
    delay(200);
    transfer_command(0x2e);
    delay(200);
    transfer_command(0x2f); //升压三步曲
    delay(10);
```

```
    transfer_command(0xae); //关显示
    transfer_command(0x38); //模式设置
    transfer_command(0xb8); // FR=1011 => 85Hz
        // BE[1:0]=1,0 => booster efficiency Level-3
```

```
    transfer_command(0xc8); //行扫描顺序
    transfer_command(0xa0); //列扫描顺序
```

```
    transfer_command(0x54); //0x54 1/9 bias
    transfer_command(0xf3); //释放偏压省电模式
```

```
transfer_command(0x04);
transfer_command(0x93); //集 FRC 和 PWM 模式 (4frc 和 15pwm)

transfer_command(0x44); //设置初始 COM0 Set initial COM0 register
transfer_command(0x00);
transfer_command(0x40); //设置起始行
transfer_command(0x00);
transfer_command(0xab); //OSC. ON
transfer_command(0x67); //DC-DC step up , 8 times boosting circuit

transfer_command(0x7b); //选用扩展指令 3
transfer_command(0x11); //黑白模式
// transfer_command(0x10); //四灰阶模式
transfer_command(0x00);

transfer_command(0x24); //0x24 粗调对比度, 可设置范围 0x20~0x27
transfer_command(0x81); //微调对比度
transfer_command(0x33); //36 微调对比度的值, 可设置范围 0x00~0x3f

transfer_command(0xaf); //开显示
}

void lcd_address(uchar page, uchar column)
{
    cs0=0;
    column=column-1;
    page=page-1;
    transfer_command(0xb0+page);
    transfer_command(((column>>4)&0x0f)+0x10);
    transfer_command(column&0x0f);
}

void clear_screen()
{
    uchar i, j;
    for(j=0; j<16; j++)
    {
        lcd_address(j+1, 1);
        for(i=0; i<128; i++)
        {
            transfer_data(0x00);
        }
    }
}

//显示 8x16 的点阵的字符串, 括号里的参数分别为 (页, 列, 字符串指针)
```

```
void display_string_8x16(uchar page,uchar column,uchar *text)
{
    uint i=0, j, k, n;

    while(text[i]>0x00)
    {
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;
            for(n=0;n<2;n++)
            {
                lcd_address(page+n, column);
                for(k=0;k<8;k++)
                {
                    transfer_data(ascii_table_8x16[j][k+8*n]);
                }
            }
            i++;
            column+=8;

            if(column>127)
            {
                column=0;
                page+=2;
            }
        }
        else
            i++;
    }
}
```

```
void display_16x16(uchar page,uchar column,uchar *dp)
{
    int i, j;
    for(j=0;j<2;j++)
    {
        lcd_address(page+j, column+1);
        for(i=0;i<16;i++)
        {
            transfer_data(*dp);
            dp++;
        }
    }
}
```

```
void display_32x32(uchar page,uchar column,uchar *dp)
```

```
{
    int i, j;
    for(j=0; j<4; j++)
    {
        lcd_address(page+j, column+1);
        for(i=0; i<31; i++)
        {
            transfer_data(*dp);
            dp++;
        }
    }
}
```

```
void display_graphic(uchar *dp)
```

```
{
    int i, j;
    for(j=0; j<16; j++)
    {
        lcd_address(j+1, 1);
        for(i=0; i<128; i++)
        {
            transfer_data(*dp);
            dp++;
        }
    }
}
```

```
//=====display a picture of 128*64 dots=====
```

```
void full_display(uchar data_left, uchar data_right)
```

```
{
    int i, j;
    for(i=0; i<16; i++)
    {
        lcd_address(i+1, 1);
        for(j=0; j<64; j++)
        {
            transfer_data(data_left);
            transfer_data(data_right);
        }
    }
}
```

```
void main(void)
```

```
{
    initial_lcd();
    while(1)
```

```
{
    clear_screen();
    display_graphic(bmp2);
    waitkey();
    clear_screen();
    display_32x32(1, 32*0, shen32);
    display_32x32(1, 32*1, zhen32);
    display_32x32(1, 32*2, shi32);
    display_32x32(1, 32*3, jing32);
    display_32x32(5, 32*0, lian32);
    display_32x32(5, 32*1, xun32);
    display_32x32(5, 32*2, dian32);
    display_32x32(5, 32*3, zi32);
    display_16x16(9, 16*0, shen);
    display_16x16(9, 16*1, zhen);
    display_16x16(9, 16*2, shi);
    display_16x16(9, 16*3, jing);
    display_16x16(9, 16*4, lian);
    display_16x16(9, 16*5, xun);
    display_16x16(9, 16*6, dian);
    display_16x16(9, 16*7, zi);
    display_string_8x16(11, 1, " JLX128128G-560");
    display_16x16(13, 16*1, dian1);
    display_16x16(13, 16*2, zhen1);
    display_string_8x16(13, 8*6-1, ":128*128");
    display_16x16(15, 16*0, shi1);
    display_16x16(15, 16*1, qu);
    display_string_8x16(15, 8*4-1, ":31.4X31.4mm");
    waitkey();
    full_display(0xff, 0xff);
    waitkey();
    full_display(0xff, 0x00);
    waitkey();
    full_display(0x00, 0xff);
    waitkey();
}
```