



# JLX12864OLED-13016-PN 使用说明书

## 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	5
6	时序特性	5~6
7	指令功能及硬件接口与编程案例	6~页末

## 1. 概述

晶联讯电子专注于 OLED 屏及液晶模块的研发、制造。所生产 JLX12864OLED-13016-PN 型液晶模块由于使用方便、显示清晰、视角宽、清晰度高、无需背光广泛应用于各种人机交流面板。

JLX12864OLED-13016-PN 可以显示 128 列\*64 行点阵单色图片，或显示 16\*16 点阵的汉字 8 个\*4 行，或显示 8\*16 点阵的英文、数字、符号 16 个\*4 行。或显示 5\*8 点阵的英文、数字、符号 21 个\*8 行。

## 2. JLX12864OLED-13016-PN 图像型点阵液晶模块的特性

2.1 结构牢：焊接式 FPC。

2.2 IC 采用 SH1106, 功能强大，稳定性好

2.3 功耗低。

2.4 显示内容：

- 128\*64 点阵单色图片；

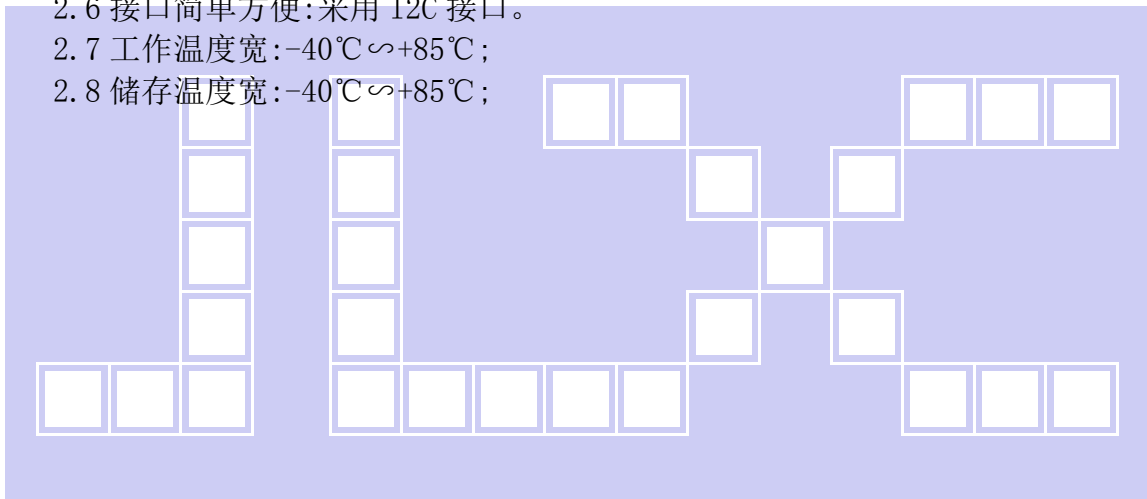
- 可选用 16\*16 点阵或其他点阵的图片来自编汉字，按照 16\*16 点阵汉字来计算可显示 8 字/行\*4 行。

2.5 指令功能强:可组合成各种输入、显示、移位方式以满足不同的要求；

2.6 接口简单方便:采用 I2C 接口。

2.7 工作温度宽:-40℃~+85℃；

2.8 储存温度宽:-40℃~+85℃；



### 3. 外形尺寸及接口引脚功能

#### 3.1 外形图

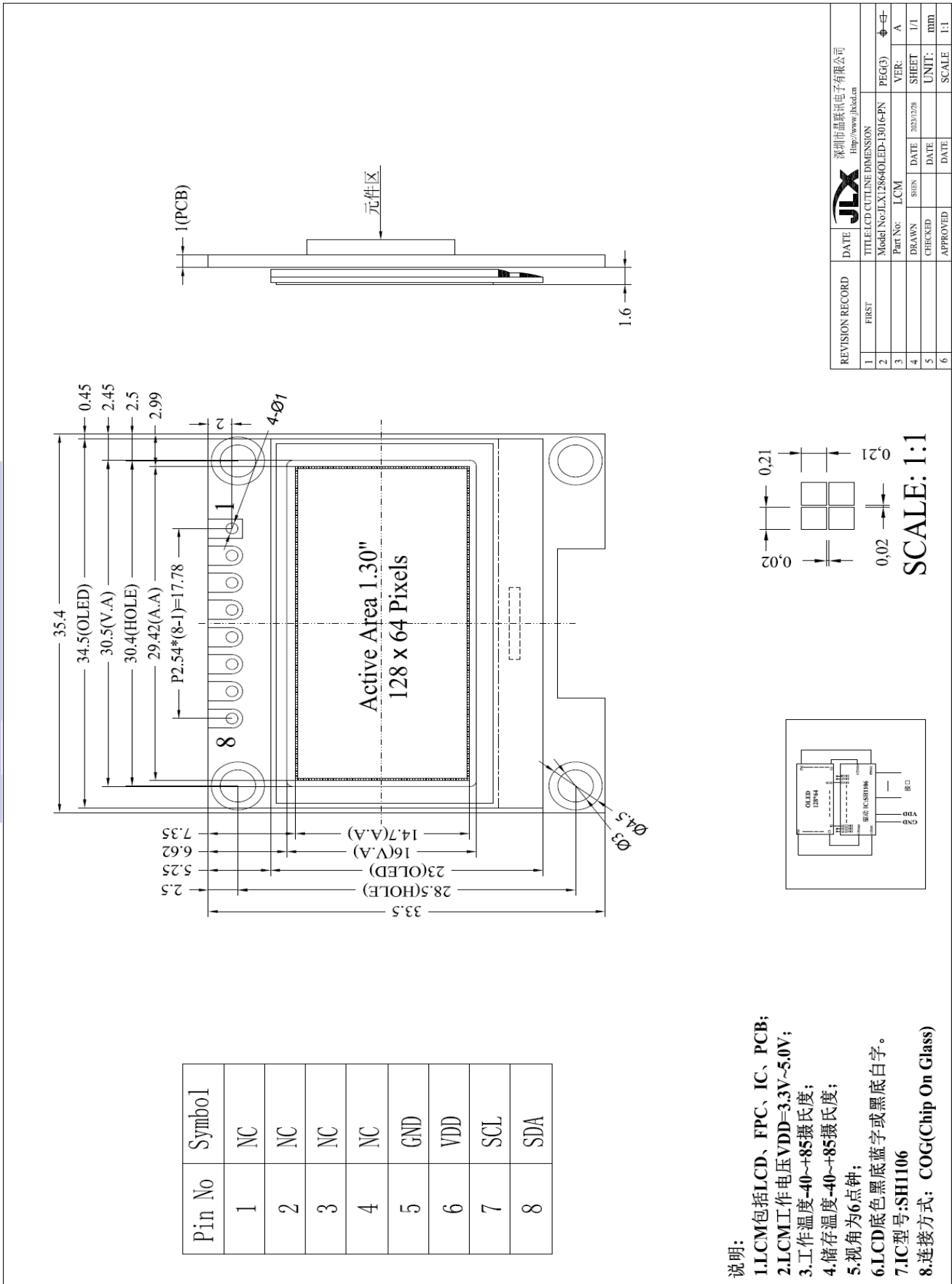


图 1. 液晶模块外形尺寸

## 模块的接口引脚功能

引 线 号	符 号	名 称	功 能
1	NC	NC	
2	NC	NC	
3	NC	NC	
4	NC	NC	
5	VSS	供电电源负极	0V
6	VDD	供电电源正极	2.7V—5.0V
7	SCL	I/O	串行时钟
8	SDA	I/O	串行数据

表 1：模块的接口引脚功能

## 4. 基本原理

### 4.1 OLED 屏（OLED）

在 LCD 上排列着  $128 \times 64$  点阵, 128 个列信号与驱动 IC 相连, 64 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）。

## 电路框图

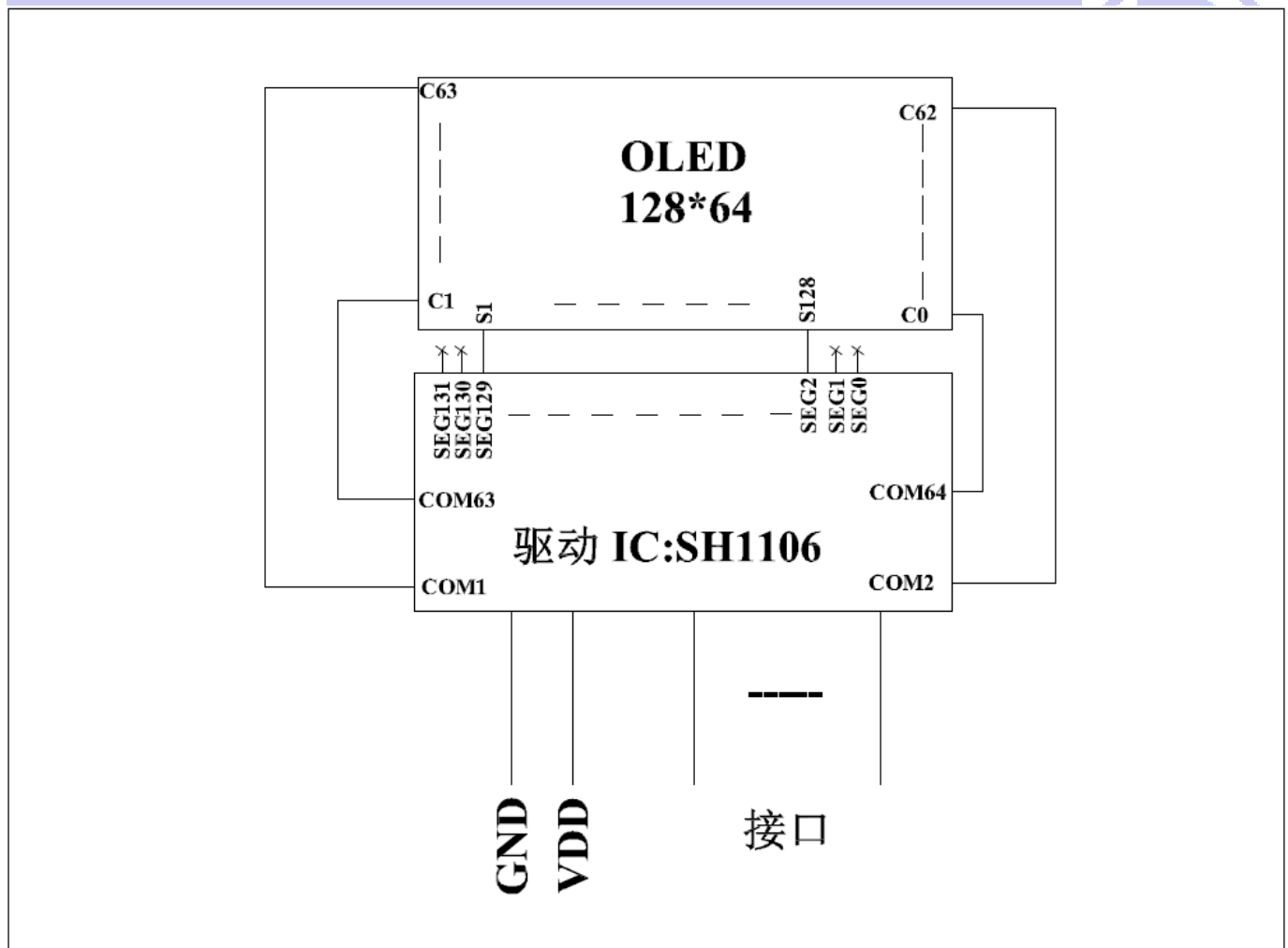


图 2. JLX12864OLED-13016-PN

## 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	3.3	5.0	V
OLED 驱动电压	VCC	7.0	—	16	V
静电电压		—	—	100	V
工作温度		-40		+85	°C
储存温度		-40		+85	°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
			MIN	TYPE	MAX	
工作电压	VDD		2.4	3.3	5.2	V
输入高电平	V <sub>IHC</sub>		0.8xVDD	—	VDD	V
输入低电平	V <sub>ILC</sub>		VSS	—	0.2xVDD	V
输出高电平	V <sub>OHC</sub>	I <sub>OH</sub> = 0.2mA	0.8xVDD	—	VDD	V
输出低电平	V <sub>OHC</sub>	I <sub>OO</sub> = 1.2mA	VSS	—	0.2xVDD	V
模块工作电流	I <sub>DD</sub>	VDD = 3.3V	—		150	mA

表 3: 直流 (DC) 参数

## 6. 读写时序特性

### 6.1 I2C 接口:

从 CPU 写到 SH1106 (Writing Data from CPU to SH1106)

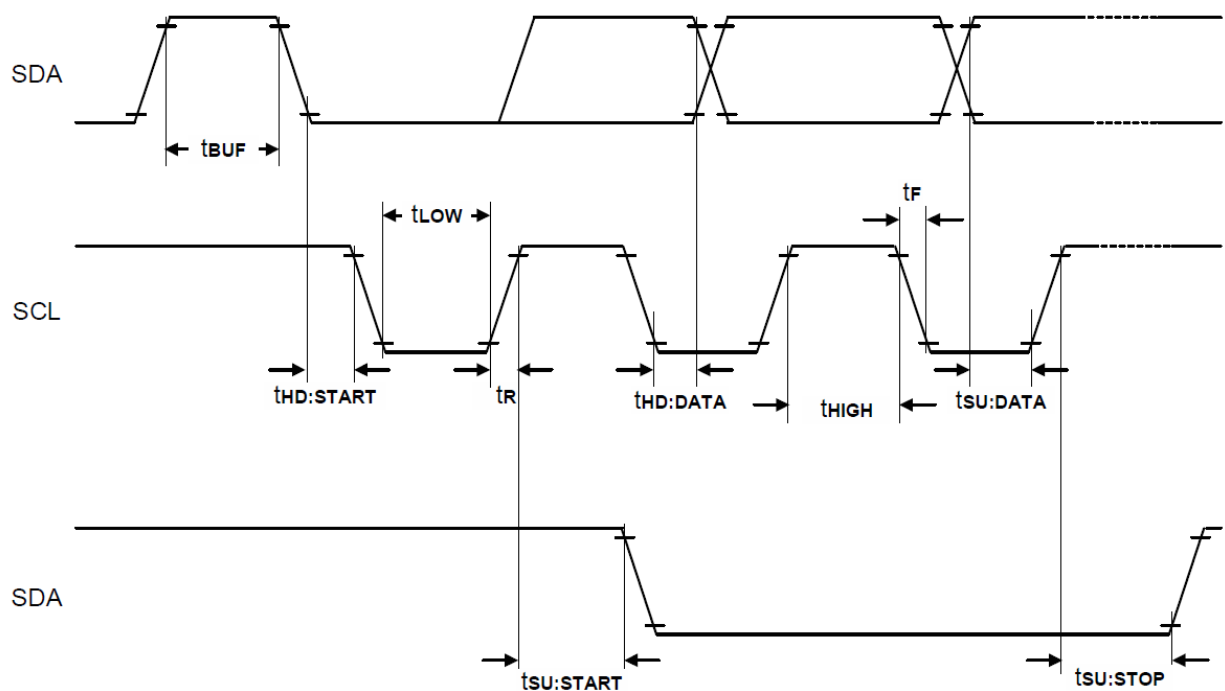


图 3. 从 CPU 写到 SH1106 (Writing Data from CPU to SH1106)

6.2 时序要求 (AC 参数):  
写数据到 SH1106 的时序要求:

表 4.

Symbol	Parameter	Min.	Typ.	Max.	Unit	Condition
f <sub>SCL</sub>	SCL clock frequency	DC	-	400	kHz	
T <sub>LOW</sub>	SCL clock Low pulse width	1.3	-	-	uS	
T <sub>HIGH</sub>	SCL clock H pulse width	0.6	-	-	uS	
T <sub>SU:DATA</sub>	data setup time	100	-	-	nS	
T <sub>HD:DATA</sub>	data hold time	0	-	0.9	uS	
T <sub>R</sub>	SCL , SDA rise time	20+0.1Cb	-	300	nS	
T <sub>F</sub>	SCL , SDA fall time	20+0.1Cb	-	300	nS	
C <sub>b</sub>	Capacity load on each bus line	-	-	400	pF	
T <sub>SU:START</sub>	Setup time for re-START	0.6	-	-	uS	
T <sub>HD:START</sub>	START Hold time	0.6	-	-	uS	
T <sub>SU:STOP</sub>	Setup time for STOP	0.6	-	-	uS	
T <sub>BUF</sub>	Bus free times between STOP and START condition	1.3	-	-	uS	

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

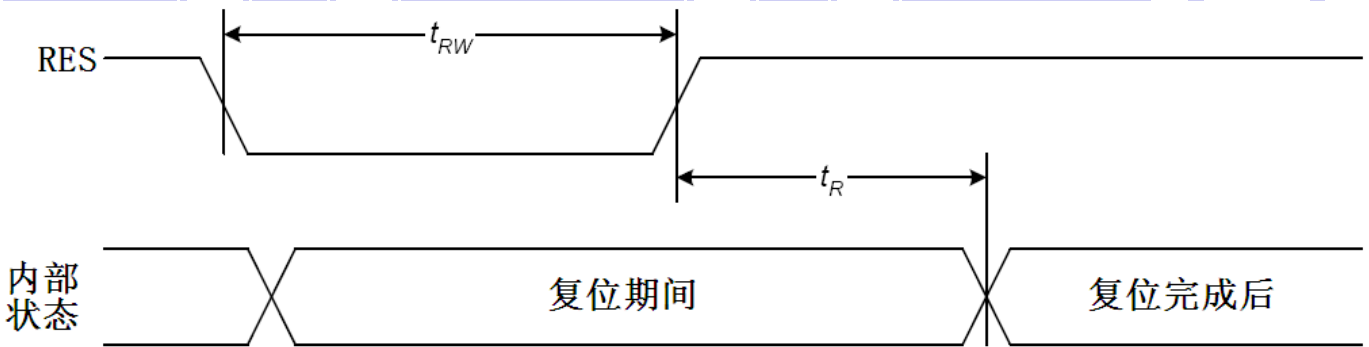


图 4: 电源启动后复位的时序

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 <sub>R</sub>		200	—	—	ms
复位保持低电平的时间	t <sub>RW</sub>	引脚: RES	200	—	—	ms

## 7. 指令功能:

## 7.1 指令表

指令名称		指 令 码									说 明																							
		RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0																								
(1)列地址低4位设置		0	0	0	0	0	列地址的低 4 位				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 OLED 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																							
(2)列地址高4位设置			0	0	0	1	列地址的高 4 位																											
(3)设定升压峰值 (Set Pump voltage value)		0	0	0	1	1	0	0	A1	A0	<table><tr><th>A1</th><th>A0</th><th>升压峰值</th><th>指令</th></tr><tr><td>0</td><td>0</td><td>6.4V</td><td>0x30</td></tr><tr><td>0</td><td>1</td><td>7.4V</td><td>0x31</td></tr><tr><td>1</td><td>0</td><td>8.0V</td><td>0x32</td></tr><tr><td>1</td><td>1</td><td>9.0V</td><td>0x33</td></tr></table>				A1	A0	升压峰值	指令	0	0	6.4V	0x30	0	1	7.4V	0x31	1	0	8.0V	0x32	1	1	9.0V	0x33
A1	A0	升压峰值	指令																															
0	0	6.4V	0x30																															
0	1	7.4V	0x31																															
1	0	8.0V	0x32																															
1	1	9.0V	0x33																															
(4)显示初始行设置 (Display start line set)		0	0	1	显示初始行地址, 共 6 位				设置显示存储器的显示初始行,可设置值为 0X40~0X7F,分别代表第 0~63 行, 针对该 OLED 屏一般设置为 0x40																									
(5)	设置对比度	0	1	0	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 0x81 是不改的, 下面一条指令可设置范围为: 0x00~0xFF,数值越大对比度越浓, 越小越淡																							
	设置对比度值	0	8 位电压值数据, 0~255 共 256 级																															
(6) 显示列地址增减 (ADC select)		0	1	0	1	0	0	0	0	ADC	显示列地址增减: 0 0xA0: 常规: 列地址从右到左, 1 0xA1: 反转: 列地址从左到右																							
(7) 设置常规/打开全部点阵 (Set Entire Display OFF/ON)		0	1	0	1	0	0	1	0	D	设置常规显示/打开全部点阵 0 0xA4: 常规显示, 写什么内容显示什么 1 0xA5: 全部点阵点亮,之前的显示会被覆盖																							
(8)显示正显/反显 (Display normal/reverse)		0	1	0	1	0	0	1	1	D	显示正显/反显: 0 0xA6: 常规: 正显 1 0xA7: 反显																							
(9)设置显示行数 (Set Multiplex Ration)		0	1	0	1	0	1	0	0	0	0xA8: 设置显示行数																							
		0	*	*	共 6 位, 0~63 共 64 级				设置范围: 00~3f 针对本型号为 0x3f, 64 行																									
(10) 设置内部/外部升压 Set DC-DC OFF/ON: (Double Bytes Command)		0	1	0	1	0	1	1	0	1	0xAD: 设置内部/外部升压																							
		0	1	0	0	0	1	0	1	D	0x8A: 使用外部升压																							
										0	0x8B: 使用内部升压																							
(11) 显示开/关 (display on/off)		0	1	0	1	0	1	1	1	0	显示开/关: 0xAE: 关, 0xAF: 开																							
(12)页地址设置 (Page address set)		0	1	0	1	1	显示页地址, 共 4 位				设置页地址。每8行为一个页, 64行分为8个页, 可设置值为: 0XB0~0XB7 分别对应第一页到第八页。																							
(13) 行扫描顺序选择 (Common output mode)		0	1	1	0	0	D	0	0	0	行扫描顺序选择: 0XC0: 普通扫描顺序: 从下到上																							



select)						1				0XC8:反转扫描顺序: 从上到下
(14) 设置显示行偏移 (Set Display Offset: (Double Bytes Command))	0	1	1	0	1	0	0	1	1	0XD3:设置显示行偏移
	0	*	*	共 6 位, 0~63 共 64 级						0X00:默认, 范围: 00—3f
(15) OLED 振荡频率设置 (Oscillator Frequency(Double Bytes Command))	0	1	1	0	1	0	1	0	1	0Xd5:振荡频率设置
	0	共 8 位, 0~255 共 256 级								0X80:默认值, 范围: 00—ff
(16)设置预充电周期 (Set Dis-charge/Pre-charge Period: (Double Bytes Command))	0	1	1	0	1	1	0	0	1	预充电周期模式设置: 0XD9:
	0	共 8 位, 0~255 共 256 级								设置预充电时间: 0X1f:默认值, 范围: 00—ff
(17)设置COM硬件配置 (Set Common pads hardware configuration: (Double Bytes Command))	0	1	1	0	1	1	0	1	0	设置COM硬件配置 0XDA:
	0	0	0	0	D 0 1	0	0	1	0	设置COM配置模式 0X02: 0X12:本款型号配置
(18)设置VCOM (Set VCOM Deselect Level: (Double Bytes Command))	0	1	1	0	1	1	0	1	1	设置VCOM 0XDB:
	0	共 8 位, 0~255 共 256 级								0X40: 本款型号配置 范围: 00—ff
(19)读-改-写 (Read-Modify-Write)	0	1	1	1	0	0	0	0	0	0XE0: “读-改-写” 开始。 列地址的增加: 写入时: 列地址+1 读出时: 列地址不加 详情请参考IC资料第28页
(20)退出上述“读-改- 写”( End)	0	1	1	1	0	1	1	1	0	0XEE :上述“读-改-写”指令结束 详情请参考IC资料第28页
(21)空指令 ( NOP)	0	1	1	1	0	0	0	1	1	0XE3:空操作
(22) 写显示 (Write Display Data)	1	8 位显示数据								从CPU写数据到OLED屏, 每一位对应一个 点阵, 1个字节对应8个竖置的点阵
(23) 读状态 (Status read)	0	BUSY	ON/OFF	*	*	*	0	0	0	暂不可用
(24) 读OLED屏的显示 数据 (Read Display Data)	1	8 位显示数据								并口时: 读已经显示到OLED屏上的点阵数 据。

请详细参考 IC 资料”SH1106.PDF”。



## 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、背光源电源线、IO 端口 (接口)  
IO 端口包括: 并口时: CS、RESET、RW、E、RS、DO--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。

## 7.3 程序举例:

OLED 模块与 MPU (以 8051 系列单片机为例) 接口图如下:

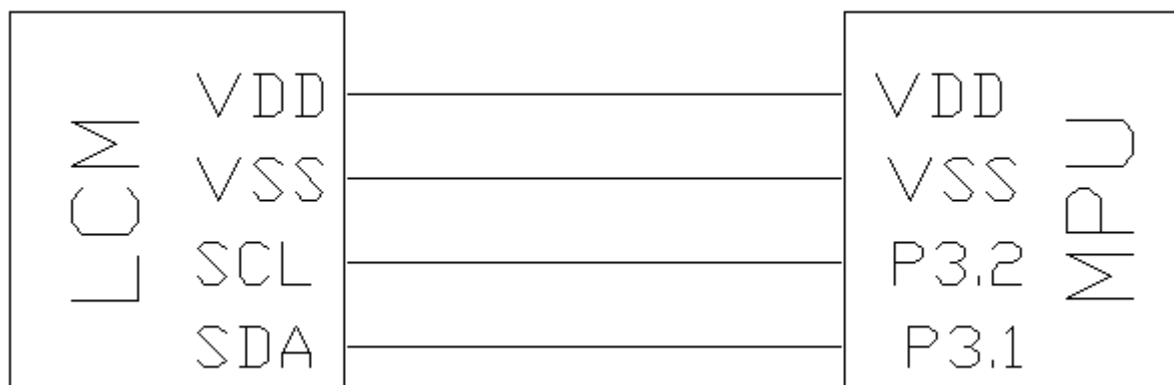


图 5.I2C 接口

## 7.3.1 程序:

## 点亮液晶模块的编程步骤



```
// 液晶演示程序
//      液晶模块型号: JLX12864OLED-13016-PN-IIC, IIC 接口!
//      驱动 IC 是:SSD1306

#include <reg52.h>

//=====
sbit lcd_scl =P3^2;    //接口定义:lcd_sclk 就是 LCD 的 SCLK    //SCLK 接到 "D0" 脚
sbit lcd_sda =P3^1;    //接口定义:lcd_sda 就是 LCD 的 SDA    //SDIN 接到 "D1" 脚

sbit key=P2^0;         //定义一个按键: P2.0 口与 GND 之间接一个按键
```

```
#define uchar unsigned char
```

```
#define uint unsigned int
```

```
#define ulong unsigned long
```

```
#include <ASCII_CODE_8X16_5X8_VERTICAL.H>
```

```
#include <Chinese_And_Graphic.H>
```

```
//延时
```

```
void delay(int i)
```

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

```
//短延时
```

```
void delay_us(int i)
```

```
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<1;k++);
}
```

```
//等待按键: P2.0 口与 GND 之间接一个按键
```

```
void waitkey()
```

```
{
repeat:  if(key==1) goto repeat;
        else delay(3000);
}
```

```
void start_flag()
```

```
{
    lcd_scl=1;
    delay_us(1);
    lcd_sda=1;
    delay_us(1);
    lcd_sda=0;
    delay_us(1);
    lcd_scl=0;
    delay_us(1);
}
```

```
void stop_flag()
```

```
{
    lcd_scl=0;
    delay_us(1);
    lcd_sda=0;
    delay_us(1);
    lcd_sda=1;
}
```

```

delay_us(1);
lcd_scl=1;
delay_us(1);
}

```

//传 8 位指令或数据到 OLED 显示模块

```
void transfer(uchar data1)
```

```

{
    unsigned char j;
    for(j=0;j<8;j++)
    {
        lcd_scl=0;
        if(data1&0x80)    lcd_sda=1;
        else
            lcd_sda=0;
        lcd_scl=1;
        lcd_scl=0;
        data1<<=1;

```

```
//    delay_us(1);
```

```

    }
    lcd_sda=0;
    lcd_scl=0;
    lcd_scl=1;
}

```

//写指令到 OLED 显示模块

```
void transfer_command(uchar com)
```

```

{
    start_flag();
    transfer(0x78);
    transfer(0x00);
    transfer(com);
    stop_flag();
}

```

//写数据到 OLED 显示模块

```
void transfer_data(uchar dat)
```

```

{
    start_flag();
    transfer(0x78);
    transfer(0x40);
    transfer(dat);
    stop_flag();
}

```

//OLED 显示模块初始化

```
void initial_lcd()
```

```

{
    delay(2000);

    transfer_command(0xAE);    /*display off*/ //该模式停止 OLED 显示系统的每一次操作，并且可以将电流消耗降低到接近静态电流
                                //保存睡眠模式开始前提供的显示数据和操作模式

    transfer_command(0x02);    /*set lower column address*/

    transfer_command(0x10);    /*set higher column address*/

    transfer_command(0x40);    /*set display start line*/ //设置起始行 0x40

    transfer_command(0xB0);    /*set page address*/

    transfer_command(0x81);    /*contrast control*/
    transfer_command(0xff);    /*128*/

    transfer_command(0xA1);    /*set segment remap*/ //设置列扫描方向，0xA1 列地址从左到右

    transfer_command(0xA6);    /*normal / reverse*/ //正常显示/A7 反显

    transfer_command(0xA8);    /*multiplex ratio*/
    transfer_command(0x3F);    /*duty = 1/64*/ //0x3F 设置显示行数

    transfer_command(0xad);    /*set charge pump enable*/
    transfer_command(0x8b);    /* 0x8B 内供 VCC */
    transfer_command(0x33);    /*0x30—0x33 set VPP 9V */ //设定升压峰值

    transfer_command(0xC8);    /*Com scan direction*/ //从上到下/0xC0: 从下到上

    transfer_command(0xD3);    /*set display offset*/ //设置显示行偏移
    transfer_command(0x00);    /* 0x20 */ //范围: 00—3F

    transfer_command(0xD5);    /*set osc division*/
    transfer_command(0x80);

    transfer_command(0xD9);    /*set pre-charge period*/
    transfer_command(0x1f);    /*0x22*/

    transfer_command(0xDA);    /*set COM pins*/
    transfer_command(0x12);    //0x12

    transfer_command(0xdb);    /*set vcomh*/
    transfer_command(0x40);    //0x40 范围: 00—ff

    transfer_command(0xAF);    /*display ON*/
}

```

```

void lcd_address(uchar page, uchar column)

```

```

{

```

```

column=column+1; //我们平常所说的第 1 列，在 LCD 驱动 IC 里是第 0 列。所以在这里+1.
page=page-1;
transfer_command(0xb0+page); //设置页地址。每页是 8 行。一个画面的 64 行被分成 8 个页。我们平常所说的第 1 页，在
LCD 驱动 IC 里是第 0 页，所以在这里减去 1
transfer_command(((column>>4)&0x0f)+0x10); //设置列地址的高 4 位
transfer_command(column&0x0f); //设置列地址的低 4 位
}
//全屏清屏
void clear_screen()
{
    unsigned char i, j;
    for(j=0; j<8; j++)
    {
        lcd_address(1+j, 1);
        for(i=0; i<128; i++)
        {
            transfer_data(0x00);
        }
    }
}
//显示 128x64 点阵图像
void display_128x64(uchar *dp)
{
    uint i, j;
    for(j=0; j<8; j++)
    {
        lcd_address(j+1, 1);
        for (i=0; i<128; i++)
        {
            transfer_data(*dp); //写数据到 LCD, 每写完一个 8 位的数据后列地址自动加 1
            dp++;
        }
    }
}
//显示 32x32 点阵图像、汉字、生僻字或 32x32 点阵的其他图标
void display_graphic_32x32(uchar page, uchar column, uchar *dp)
{
    uchar i, j;
    for(j=0; j<4; j++)
    {
        lcd_address(page+j, column);
        for (i=0; i<32; i++)
        {
            transfer_data(*dp); //写数据到 LCD, 每写完一个 8 位的数据后列地址自动加 1
            dp++;
        }
    }
}

```

```

}

//显示 16x16 点阵图像、汉字、生僻字或 16x16 点阵的其他图标
void display_graphic_16x16(uchar page, uchar column, uchar *dp)
{
    uchar i, j;
    for(j=0; j<2; j++)
    {
        lcd_address(page+j, column);
        for (i=0; i<16; i++)
        {
            transfer_data(*dp);          //写数据到 LCD, 每写完一个 8 位的数据后列地址自动加 1
            dp++;
        }
    }
}

//显示 8x16 点阵图像、ASCII, 或 8x16 点阵的自造字符、其他图标
void display_graphic_8x16(uchar page, uchar column, uchar *dp)
{
    uchar i, j;
    for(j=0; j<2; j++)
    {
        lcd_address(page+j, column);
        for (i=0; i<8; i++)
        {
            transfer_data(*dp);          //写数据到 LCD, 每写完一个 8 位的数据后列地址自动加 1
            dp++;
        }
    }
}

//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(uint page, uint column, uchar *text)
{
    uint i=0, j, k, n;
    if(column>123)
    {
        column=1;
        page+=2;
    }
    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]); //写数据到LCD, 每写完1字节的数据后列地址自动加1
        }
    }
    i++;
    column+=8;
}
else
    i++;
}
}

//显示 5x8 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_5x8(uint page, uint column, uchar reverse, uchar *text)
{
    uint i=0, j, k, disp_data;
    while(text[i]>0x00)
    {
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;
            lcd_address(page, column);
            for(k=0; k<5; k++)
            {
                if(reverse==1)
                {
                    disp_data=~ascii_table_5x8[j][k];
                }
                else
                {
                    disp_data=ascii_table_5x8[j][k];
                }
                transfer_data(disp_data); //写数据到LCD, 每写完1字节的数据后列地址自动加1
            }
            if(reverse==1) transfer_data(0xff); //写入一列空白列, 使得 5x8 的字符与字符之间有一列间隔, 更美观
            else transfer_data(0x00); //写入一列空白列, 使得 5x8 的字符与字符之间有一列间隔, 更美观
            i++;
            column+=6;
            if(column>123)
            {
                column=1;
                page++;
            }
        }
        else
            i++;
    }
}

```



//写入一组 16x16 点阵的汉字字符串（字符串表格中需含有此字）

//括号里的参数：（页，列，汉字字符串）

```
void display_string_16x16(uchar page,uchar column,uchar *text)
```

```
{
    uchar i,j,k;
    uint address;
    j = 0;
    while(text[j] != '\0')
    {
        i = 0;
        address = 1;
        while(Chinese_text_16x16[i] > 0x7e)    // >0x7f 即说明不是 ASCII 码字符
        {
            if(Chinese_text_16x16[i] == text[j])
            {
                if(Chinese_text_16x16[i + 1] == text[j + 1])
                {
                    address = i * 16;
                    break;
                }
            }
            i += 2;
        }
        if(column > 113)
        {
            column = 0;
            page += 2;
        }

        if(address != 1)// 显示汉字
        {
            for(k=0;k<2;k++)
            {
                lcd_address(page+k, column);
                for(i = 0; i < 16; i++)
                {
                    transfer_data(Chinese_code_16x16[address]);
                    address++;
                }
            }
            j += 2;
        }
        else    //显示空白字符
        {
            for(k=0;k<2;k++)
            {
                lcd_address(page+k, column);
                for(i = 0; i < 16; i++)
```



```

        {
            transfer_data(0x00);
        }
    }

    j++;
}

column+=16;
}
}

```

//显示 16x16 点阵的汉字或者 ASCII 码 8x16 点阵的字符混合字符串

//括号里的参数: (页, 列, 字符串)

```
void disp_string_8x16_16x16(uchar page, uchar column, uchar *text)
```

```

{
    uchar temp[3];
    uchar i = 0;

    while(text[i] != '\0')
    {
        if(text[i] > 0x7e)
        {
            temp[0] = text[i];
            temp[1] = text[i + 1];
            temp[2] = '\0'; //汉字为两个字节
            display_string_16x16(page, column, temp); //显示汉字
            column += 16;
            i += 2;
        }
        else
        {
            temp[0] = text[i];
            temp[1] = '\0'; //字母占一个字节
            display_string_8x16(page, column, temp); //显示字母
            column += 8;
            i++;
        }
    }
}

```

```
void main(void)
```

```

{
    while(1)
    {
        initial_lcd(); //初始化
        clear_screen(); //清屏

        //演示 32x32 点阵的汉字, 16x16 点阵的汉字, 8x16 点阵的字符, 5x8 点阵的字符
        display_string_5x8(1, 1, 0, "{(5x8dot ASCII char)}"); //显示字符串, 括号里的参数分别为 (PAGE, 列, 字符串指针)
        display_string_5x8(2, 1, 0, "{[(<~!@#$$%^&*_+=?>)]}");
    }
}

```

```

disp_string_8x16_16x16(3,1,"标准 16x16dot 汉字");           //显示 16x16 点阵汉字串或 8x16 点阵的字符串, 括号里的参数分别为 (页, 列,
字符串指针)

display_graphic_32x32 (5,1+32*0, jing1);                      //显示单个 32x32 点阵的汉字, 括号里的参数分别为 (PAGE, 列, 字
符指针)

display_graphic_32x32 (5,1+32*1, lian1);
display_graphic_32x32 (5,1+32*2, xun1);
disp_string_8x16_16x16(5,1+32*3,"JLX:");
disp_string_8x16_16x16(7,1+32*3,"OLED");
waitkey();

//演示显示一页纯英文的 5x8 点阵的菜单界面

clear_screen();                                               //clear all dots
display_string_5x8(1,1,1,"012345678901234567890");
display_string_5x8(1,1,1,"      MENU      ");//显示 5x8 点阵的字符串, 括号里的参数分别为 (页, 列, 是否反显, 数据指针)
display_string_5x8(3,1,0,"Select>>>>");
display_string_5x8(3,64,1,"1. Graphic ");
display_string_5x8(4,64,0,"2. Chinese ");
display_string_5x8(5,64,0,"3. Movie  ");
display_string_5x8(6,64,0,"4. Contrast");
display_string_5x8(7,64,0,"5. Mirror  ");
display_string_5x8(8,1,1,"PRE USER DEL NEW");
display_string_5x8(8,19,0," ");
display_string_5x8(8,65,0," ");
display_string_5x8(8,97,0," ");
waitkey();
clear_screen();
display_128x64(bmp1);
waitkey();
clear_screen();
display_128x64(bmp2);
waitkey();
}
}

```

**-END-**