JLX64320LED-049 中文使用说明书

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 JLX6432OLED-049 型液晶模块由于使用方便、无需背光、视角宽、显示清晰、超薄，广泛应用于各种人机交流面板。

JLX6432OLED-049 可以显示 64 列*32 行点阵单色图片，或显示 16*16 点阵的汉字 4 个*2 行，或显示 8*16 点阵的英文、数字、符号 8 个*2 行。或显示 5*8 点阵的英文、数字、符号 10 个*4 行。

2. JLX6432OLED-049 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 SSD1306,功能强大，稳定性好
2.3 功耗低。
2.4 显示内容:
●64*32 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 4 字/行*2 行。按照 12*12 点阵汉字来计算可显示 5 字/行*2 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 接口方式：I^2C 接口。
2.7 工作温度宽:-20℃ ~ 70℃;
2.8 储存温度宽：-30℃-80℃;
2.9 底色可选：蓝色、白色。

3. 外形尺寸及接口引脚功能

3.1 外形图

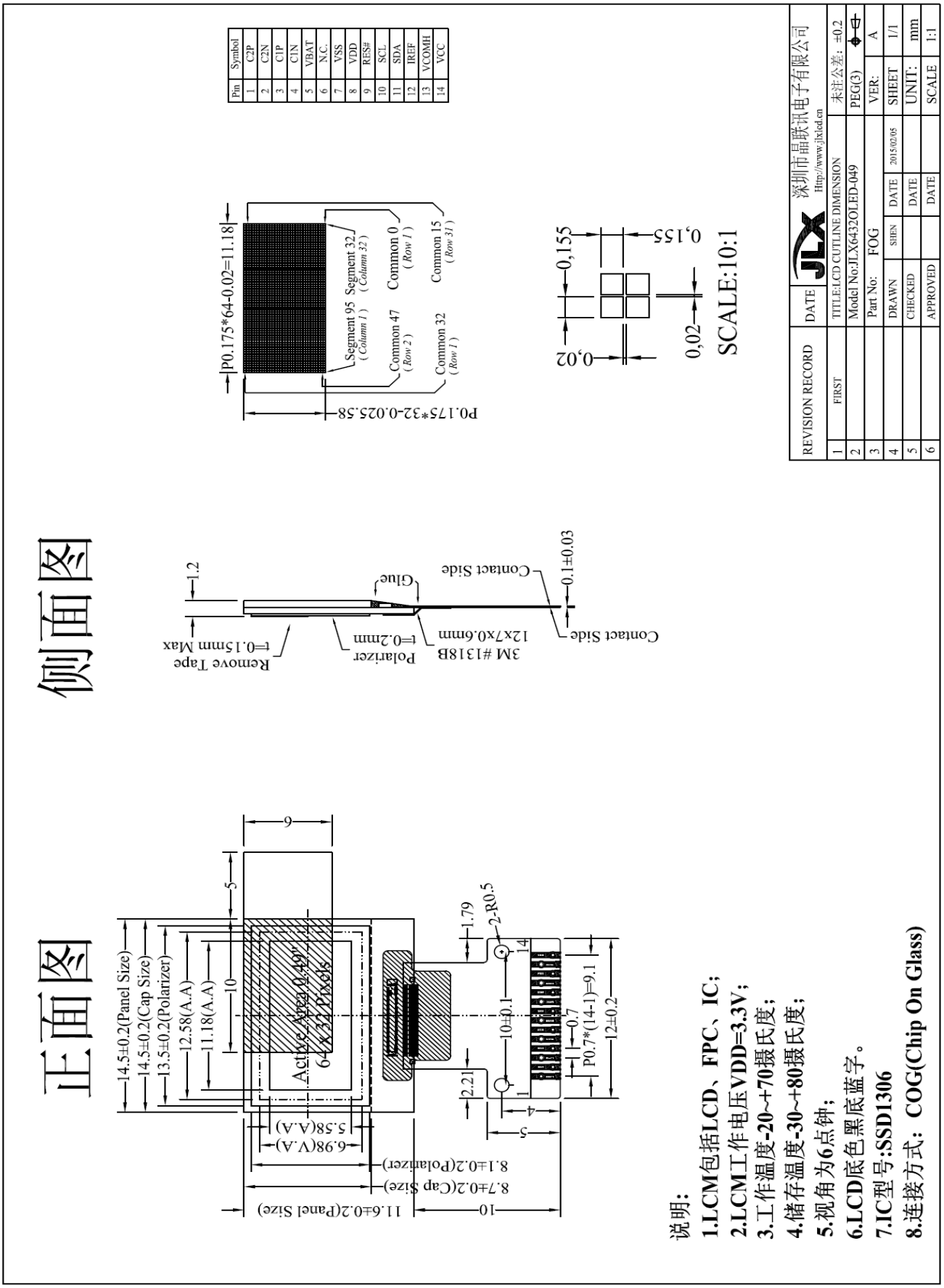


图 1. 液晶模块外形尺寸

模块的接口引脚功能

引 线 号	符 号	名 称	功 能
1	C2P	升压电容	C2P C2N
2	C2N	升压电容	
3	C1P	升压电容	C1P C1N
4	C1N	升压电容	
5	VBAT	VBAT	接 VDD
6	NC	NC	空脚
7	VSS	接地	0V
8	VDD	电源电路	5V，或 3.3V 可选
9	RES#	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
10	SCL	I/O	串行时钟
11	SDA	I/O	串行数据
12	IREF	IREF	接电阻调整模块亮度（即限电流）
13	VCOMH	VCOMH	VCOMH C VSS
14	VCC	面板电源	VCC C VSS

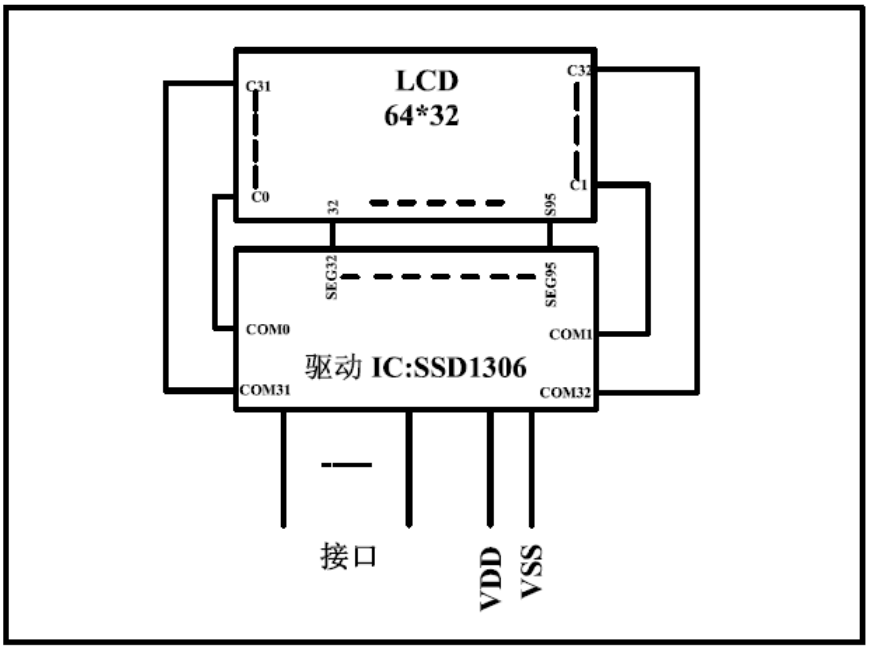
表 1：模块的接口引脚功能

4. 基本原理

4.1 液晶屏（LCD）

在 LCD 上排列着 64×32 点阵,64 个列信号与驱动 IC 相连，32 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



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	℃
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
			MIN	TYPE	MAX	
工作电压（当 3.3V 供电时）	VDD		2.4	3.3	3.6	V
工作电压（当 5.0V 供电时）			4.8	5.0	5.2	V
输入高电平	V_{IHC}		0.8xVDD	—	VDD	V
输入低电平	V_{ILC}		VSS	—	0.2xVDD	V
输出高电平	V_{OHC}	IOH = 0.2mA	0.8xVDD	—	VDD	V
输出低电平	V_{OHC}	IOO = 1.2mA	VSS	—	0.2xVDD	V
模块工作电流	I_{DD}	VDD = 3.3V	—		0.3	mA

表 3：直流（DC）参数

6. 读写时序特性

6.1 I²C 接口：

从 CPU 写到 SSD1306（Writing Data from CPU to SSD1306）

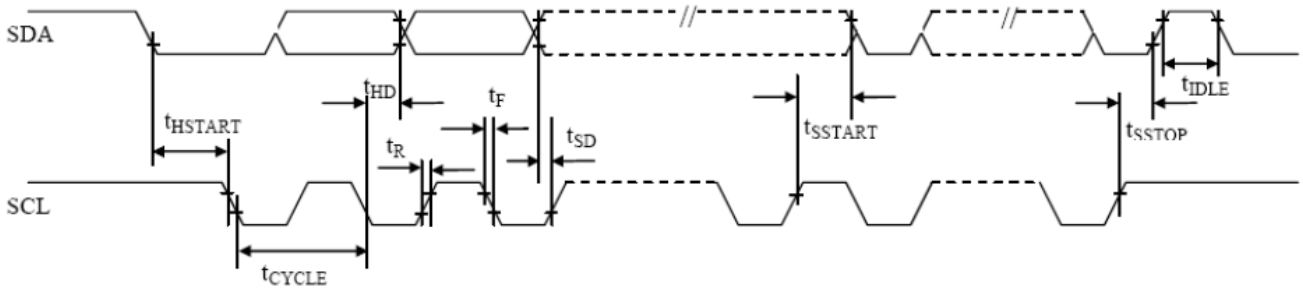


图 4. 从 CPU 写到 SSD1306（Writing Data from CPU to SSD1306）

6.2 I²C 接口：时序要求（AC 参数）：

写数据到 SSD1306 的时序要求： 表 4.

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）	Tscyc	引脚：SCK	2.5	—	—	ns
保持SCK高电平脉宽（SCK “H” pulse width）	Tshw	引脚：SCK	0.6	—	—	ns

保持SCK低电平脉宽 （SCK “L” pulse width）	T_{SLW}	引脚：SCK	0.6	—	—	ns
数据建立时间 (Data setup time)	T_{sds}	引脚：SDA	100	—	—	ns
数据保持时间 (Data hold time)	T_{SDH}	引脚：SDA	300	—	—	ns

*（VDD =1.65V～3.3V, Ta = 25℃）

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

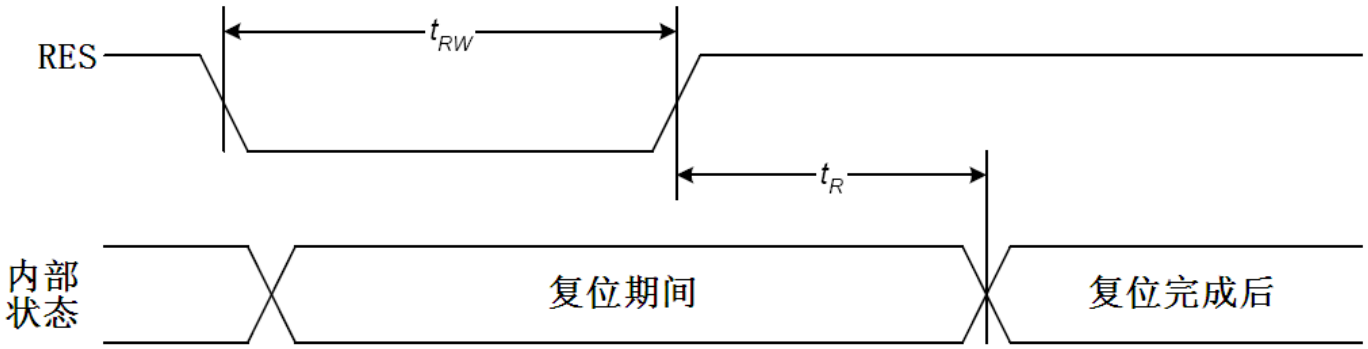


图 7：**电源启动后复位的时序**

表 6：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		—	—	1.0	us
复位保持低电平的时间	t_{RW}	引脚：RES	3.0	—	—	us

7. 指令功能：

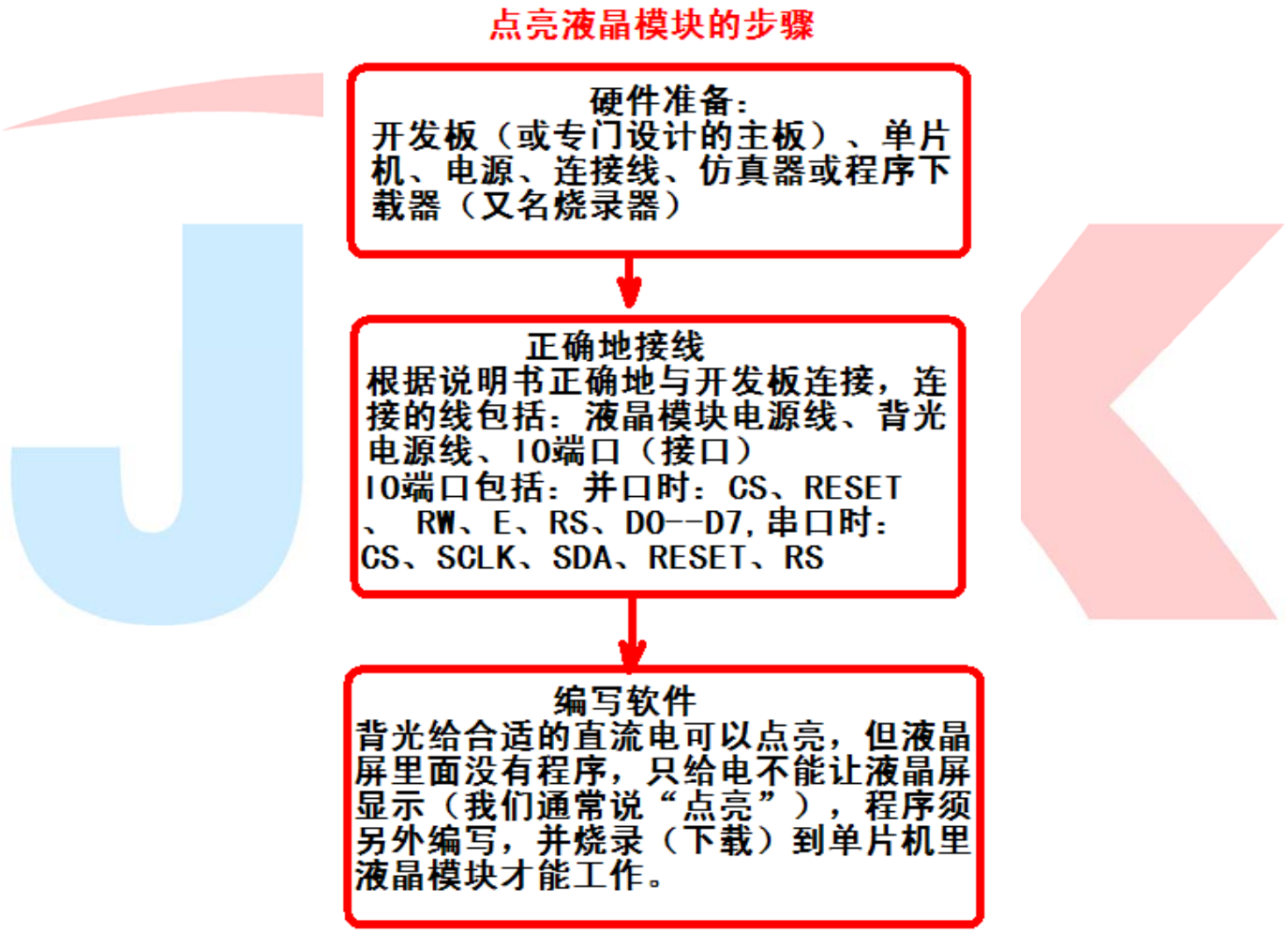
7.1 指令表

指令名称		指令码									说明
		RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
(1)显示开/关（display on/off）		0	1	0	1	0	1	1	1	0 1	显示开/关: 0XAE:关，0XAF：开
(2)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，针对该液晶屏一般设置为 0x40
(3)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：0XB0～0XB8 分别对应第一页到第九页，第九页是一个单独的一行图标，本液晶屏没有这一行图标，所以设置值为 0XB0～0XB7 分别对应第一页～第八页。
(4)	列地址高4位设置	0	0	0	0	1	列地址的高 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		0	0	0	0	列地址的低 4 位				
(5) 读状态 (Status read)		0	状态				0	0	0	0	并口时：读驱动IC的当前状态,串口时不能用此指令。
(6)写显示数据到液晶屏 (Display data write)		1	8 位显示数据								从 CPU 写数据到液晶屏，每一位对应一个点阵，1 个字节对应 8 个竖置的点阵
(7)读液晶屏的显示数据 (Display data read)		1	8 位显示数据								并口时：读已经显示到液晶屏上的点阵数据。串口时不能用此指令。
(8) 显示列地址增减 (ADC select)			1	0	1	0	0	0	0	0 1	显示列地址增减： 0xA0：反转：列地址从右到左， 0xA1：常规：列地址从左到右
(9)显示正显/反显 (Display normal/reverse)		0	1	0	1	0	0	1	1	0 1	显示正显/反显: 0xA6：常规：正显 0xA7：反显
(10)显示全部点阵 (Display all points)		0	1	0	1	0	0	1	0	0 1	显示全部点阵: 0xA4：常规 0xA5：显示全部点阵
(11) 行扫描顺序选择 (Common output mode select)			1	1	0	0	0 1	0	0	0	行扫描顺序选择： 0XC0:普通扫描顺序：从上到下 0XC8:反转扫描顺序：从下到上
(12)OLED 振荡频率设置 (Oscillator Frequency)		0	1	1	0	1	0	1	0	1	设置振荡频率：范围：0000-1111， 参考指令：0Xd5 0X80
(13) 电源控制 (Power control set)		0	1	0	0	0	1	1	0	1	设置升压：0X8d 0X14
(14)	内部设置液晶电压模式	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

	设置的电压值		0	0	6 位电压值数据，0～63 共 64 级						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
(15)静态图标显示：开/关		0	1	0	1	0	1	1	1	0 1	静态图标的开关设置： 0xAE: 关, 0xAF: 开。 此指令在进入及退出睡眠模式时起作用
(16) 省电模式（Power save）											省电模式，此非一条指令，是由“(10)显示全部点阵”、 (19)静态图标显示：开/关等指令合成一个“省电功能”。详细看 IC 规格书 “POWER SAVE”部分
(17)空指令（ NOP）		0	1	1	1	0	0	0	1	1	空操作

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



7.5 程序举例:

液晶模块与 MPU(以 8051 系列单片机为例)接口图如下:

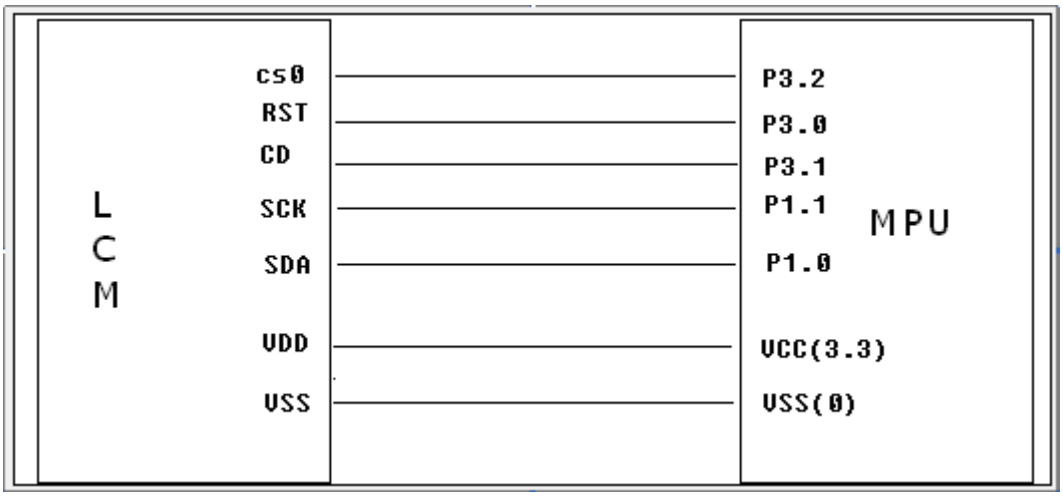
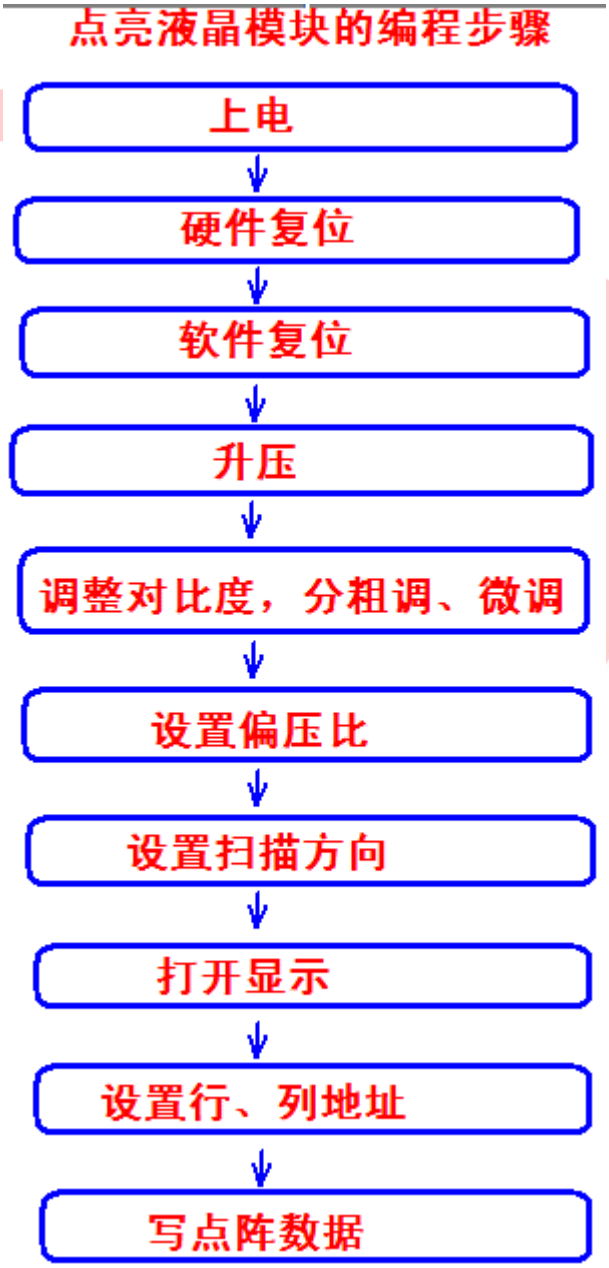
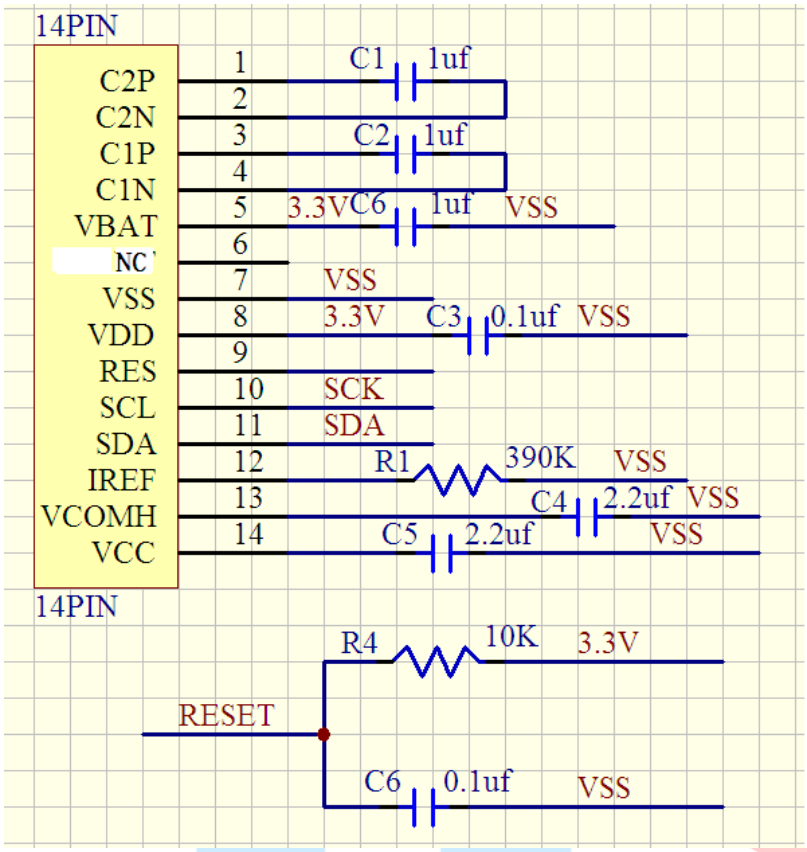
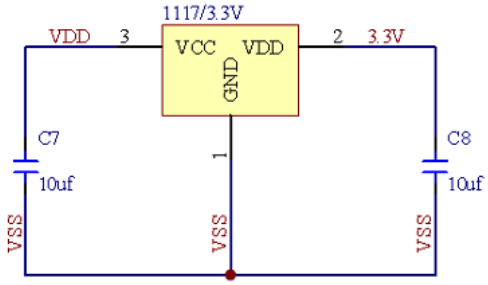


图 8.串行接口

7.5.1 程序：

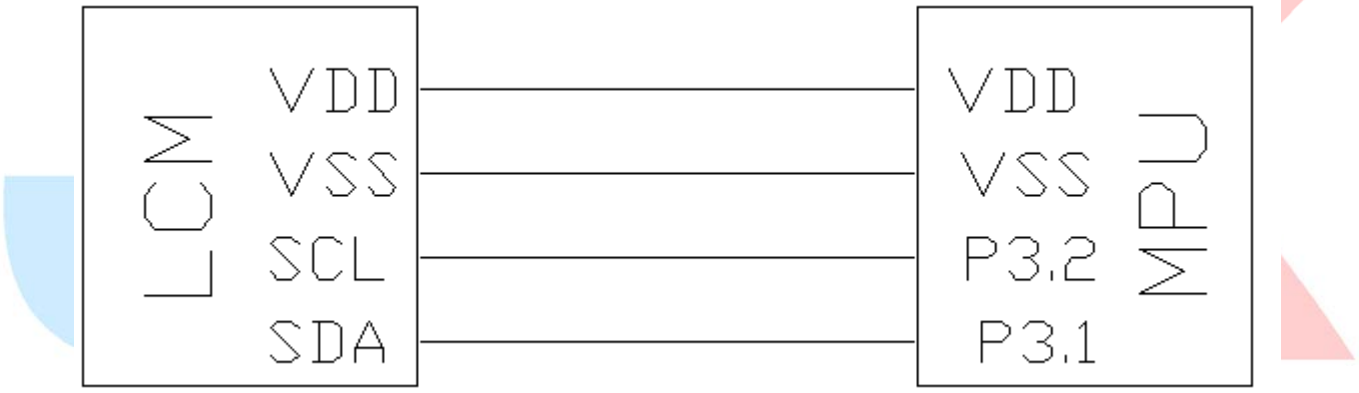






液晶模块使用电压是2.5V-3.5V
CPU是5V时,用1117/3.3V，将电压转成3.3V
CPU是3.3V时，不用1117/3.3V，直接接3.3V

所有电容的耐压值选25V或以上




```
//    液晶演示程序
//    液晶模块型号：JLX6448OLED-049，IIC 接口！
//    驱动 IC 是:SSD1306
//    版权所有：晶联讯电子：网址  http://www.jlxlcd.cn;

#include <reg52.H>
#include <intrins.h>
#include <string.h>
#include <stdio.h>

//=========================================================================
sbit SCL =P3^2;         //接口定义:lcd_sclk 就是 LCD 的 SCLK    //SCLK 接到 “D0” 脚
sbit 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>
void start();
void stop();
void waitkey();
//延时
void delay(int i)
{
      int j,k;
      for(j=0;j<i;j++)
            for(k=0;k<110;k++);
}
void write_w(unsigned char dat)
{
     unsigned char m,da;
     unsigned char j;
     da=dat;
     for(j=0;j<8;j++)
{
     m=da;
     SCL=0;
     m=m&0x80;
     if(m==0x80)
{
     SDA=1;
}
     else
{
     SDA=0;
}
     da=da<<1;
     SCL=1;
}
     SCL=0;
     SCL=1;
}

void transfer_command(unsigned char ins)
{
     start();
     write_w(0x78);
     write_w(0x00);
     write_w(ins);
     stop();
```

```
}
void transfer_data(unsigned char dat)
{
     start();
     write_w(0x78);
     write_w(0x40);
     write_w(dat);
     stop();
}
void start()
{
     SCL=1;
     SDA=1;
     SDA=0;
     SCL=0;
}
void stop()
{
     SCL=0;
     SDA=0;
     SDA=1;
     SCL=1;
}
//OLED 显示模块初始化
void initial_lcd()
{
      transfer_command(0xae);     //关显示
      transfer_command(0xd5);     //晶振频率
      transfer_command(0x80);
      transfer_command(0xa8);     //duty 设置
      transfer_command(0x3f);     //duty=1/64
      transfer_command(0xd3);     //显示偏移
      transfer_command(0x00);
      transfer_command(0x40);     //起始行
      transfer_command(0x8d);     //升压允许
      transfer_command(0x14);
      transfer_command(0x20);     //page address mode
      transfer_command(0x02);
      transfer_command(0xc8); //行扫描顺序：从上到下
      transfer_command(0xa1); //列扫描顺序：从左到右
      transfer_command(0xda); //sequential configuration
      transfer_command(0x12);
      transfer_command(0x81); //微调对比度,本指令的 0x81 不要改动，改下面的值
      transfer_command(0xcf); //微调对比度的值，可设置范围 0x00～0xff
      transfer_command(0xd9); //Set Pre-Charge Period
      transfer_command(0xf1);
      transfer_command(0xdb); //Set VCOMH Deselect Level
```

```
	transfer_command(0x40);
	transfer_command(0xaf); //开显示
}
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。所以在这里减去 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)+0x12);  //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);
		}
	}
}
//full display test
void full_display(uchar data1,uchar data2)
{
	int i,j;
	for(i=0;i<8;i++)
	{
		lcd_address(i+1,1);
		for(j=0;j<32;j++)
		{
			transfer_data(data1);
			transfer_data(data2);
		}
	}
}
//测试外框是否缺划（少行、少列）
void test_box()
{
	int i,j;
//第 1 页：
	lcd_address(1,1);
	transfer_data(0xff);
	for(i=1;i<31;i++)
```

```
    {
        transfer_data(0x01);
    }
    transfer_data(0xff);
//第 2 页：
    lcd_address(2,1);
    transfer_data(0xff);
    for(i=1;i<31;i++)
    {
        transfer_data(0x80);
    }
    transfer_data(0xff);

//第 3 页：
    lcd_address(3,1);
    transfer_data(0xff);
    for(i=1;i<31;i++)
    {
        transfer_data(0x01);
    }
    transfer_data(0xff);
//第 4 页～第 7 页：
    for(j=4;j<=7;j++)
    {
        lcd_address(2,1);
        transfer_data(0xff);
        for(i=1;i<31;i++)
        {
            transfer_data(0x00);
        }
        transfer_data(0xff);
    }
//第 8 页：
    lcd_address(4,1);
    transfer_data(0xff);
    for(i=1;i<31;i++)
    {
        transfer_data(0x80);
    }
    transfer_data(0xff);
}
//测试
void test()
{
    full_display(0xff,0xff);
    waitkey();
    full_display(0x55,0x55);
```

```
    waitkey();
    full_display(0xaa,0xaa);
    waitkey();
    full_display(0xff,0x00);
    waitkey();
    full_display(0x00,0xff);
    waitkey();
    full_display(0x55,0xaa);
    waitkey();
    full_display(0xaa,0x55);
    waitkey();
    test_box();
    waitkey();
}
//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 64x32 点阵图像
void display_64x32(uchar page,uchar column,uchar *dp)
{
    uint i,j;
    for(j=0;j<4;j++)
    {
        //lcd_address(j+1,1);
      lcd_address(page+j,column);
    for (i=0;i<64;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 128x16 点阵图像
void display_128x16(uchar page,uchar column,uchar *dp)
{
    uint i,j;
```

```
    for(j=0;j<2;j++)
    {
        lcd_address(page+j,column);
        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 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++;
    }
}
//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
       display_64x32(5,1,bmp6432_1);
     waitkey();
       clear_screen();
       display_64x32(5,1,bmp6432_2);
       waitkey();
       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_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）
        waitkey();
     clear_screen();
       display_graphic_32x32 (5,1+32*1,lian1);
          waitkey();
       clear_screen();
       display_graphic_32x32 (5,1+32*0,xun1);
          waitkey();
       clear_screen();
       disp_string_8x16_16x16(5,1," 晶联讯");
       disp_string_8x16_16x16(7,1+32*0,"JLX:");
          disp_string_8x16_16x16(7,1+32*1,"OLED");
          waitkey();
          test();
     }
}
//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(2000);
}
```