



JLX12864OLED-S130 中文使用说明书

(焊接式 FPC)

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
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~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-S130 型 OLED 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。

JLX12864OLED-S130 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-S130 图像型点阵 OLED 模块的特性

2.1 结构牢：焊接式 FPC。

2.2 IC 采用 SSD1312, 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；

- 可選用 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。

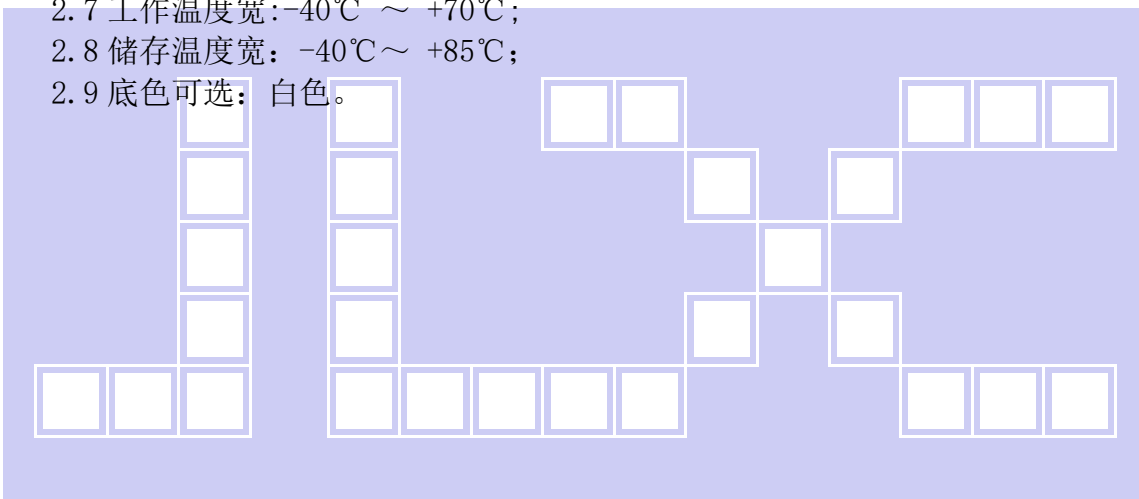
2.5 指令功能强:可组合成各种输入、显示、移位方式以满足不同的要求；

2.6 接口方式：I²C 接口。

2.7 工作温度宽:-40℃ ~ +70℃；

2.8 储存温度宽：-40℃~ +85℃；

2.9 底色可选：白色。



3. 外形尺寸及接口引脚功能

3.1 外形图

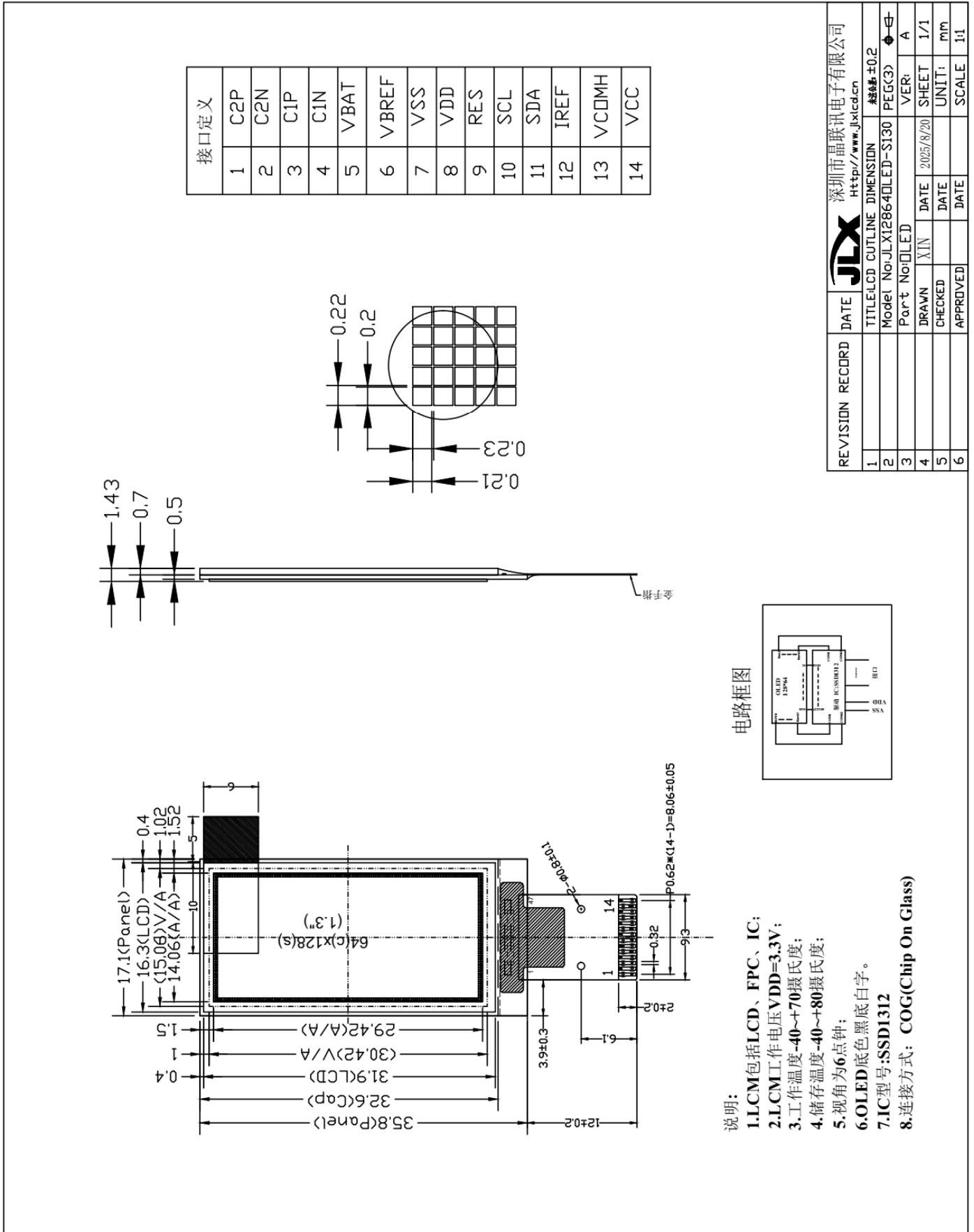


图 1. OLED 模块外形尺寸

模块的接口引脚功能

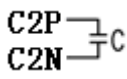
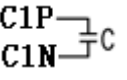
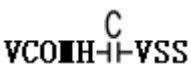
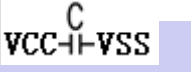
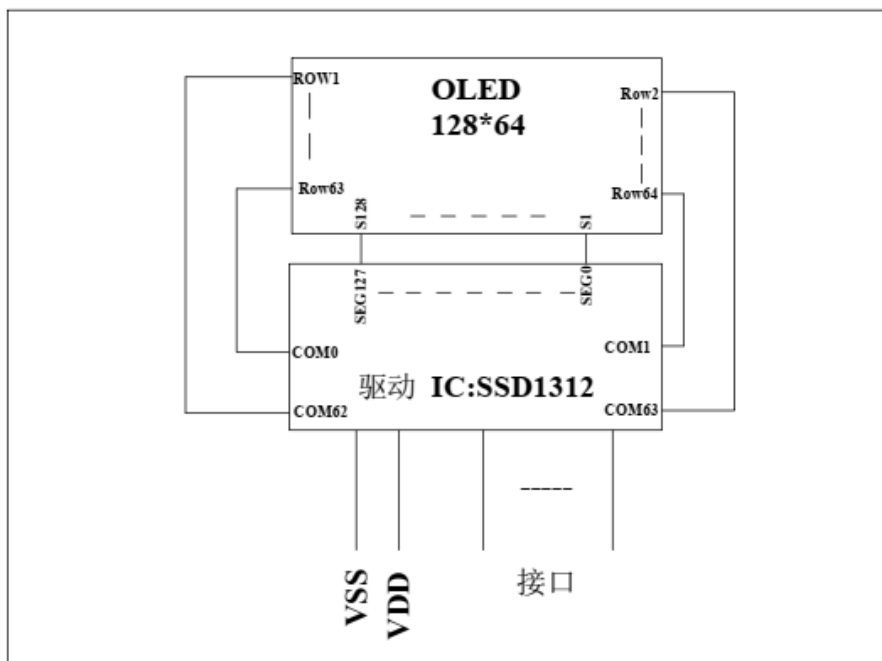
引线号	符号	名称	功能
1	C2P	升压电容	
2	C2N	升压电容	
3	C1P	升压电容	
4	C1N	升压电容	
5	VBAT	VBAT	接 VDD
6	VBREF	NC	空脚
7	VSS	接地	0V
8	VDD	电源电路	3.3V
9	RES#	复位	低电平复位, 复位完成后, 回到高电平, OLED 模块开始工作
10	SCL	I/O	串行时钟
11	SDA	I/O	串行数据
12	IREF	IREF	串 390K 电阻到 VSS
13	VCOMH	VCOMH	
14	VCC	面板电源	

表 1: 模块的接口引脚功能

4. 基本原理
4.1 OLED 屏 (LCD)

在 LCD 上排列着 128×64 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


5. 技术参数

5.1 最大极限参数 (超过极限参数则会损坏 OLED 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	3.5	V
OLED 驱动电压	VCC	7.0	—	16	V
静电电压		—	—	100	V
工作温度		-40		+70	°C
储存温度		-40		+80	°C

表 2: 最大极限参数

5.2 直流 (DC) 参数

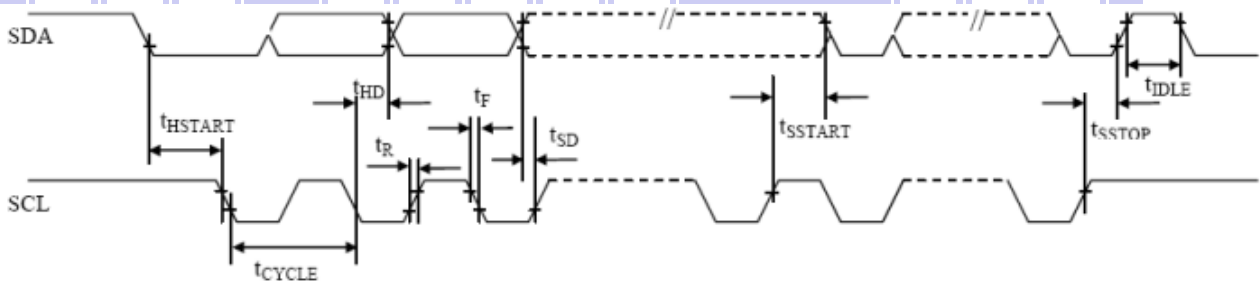
名称	符号	测试条件	标准值			单位
			MIN	TYPE	MAX	
工作电压 (当 3.3V 供电时)	VDD		2.4	3.3	3.5	V
输入高电平	V _{IHC}		0.8xVDD	—	VDD	V
输入低电平	V _{ILC}		VSS	—	0.2xVDD	V
输出高电平	V _{OHC}	I _{OH} = 0.2mA	0.9xVDD	—	VDD	V
输出低电平	V _{OHC}	I _{OO} = 1.2mA	VSS	—	0.2xVDD	V
模块工作电流	I _{DD}	VDD = 3.3V	—		0.3	mA

表 3: 直流 (DC) 参数

6. 读写时序特性

6.1 IIC 接口:

从 CPU 写到 SSD1312 (Writing Data from CPU to SSD1312)



从 CPU 写到 SSD1312 (Writing Data from CPU to SSD1312)

6.2 IIC 接口: 时序要求 (AC 参数):

写数据到 SSD1312 的时序要求:

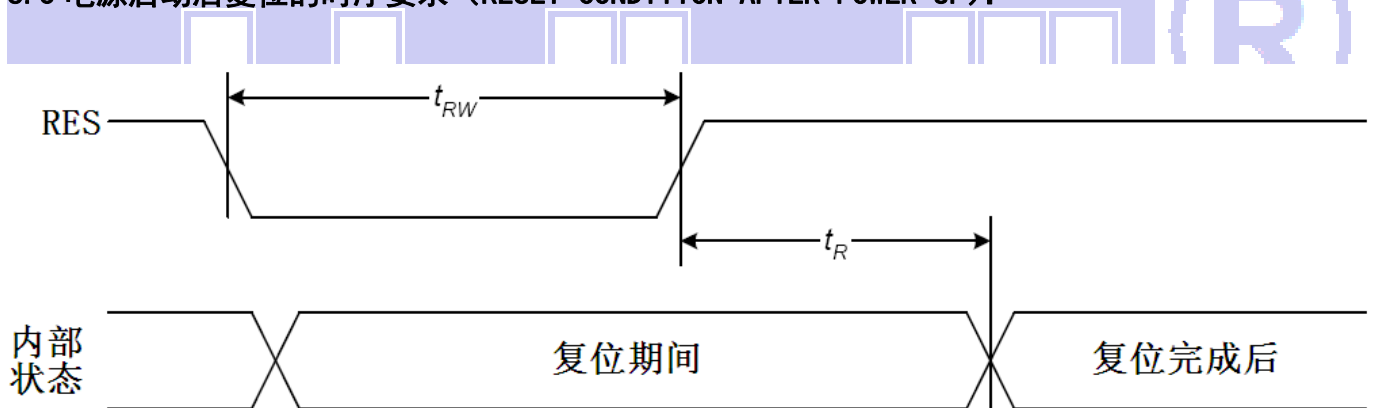
表 4.

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
时钟周期 (Clock Cycle Time)	t _{cycle}	引脚: SCK	2.5	—	—	ns
启动条件设置时间 (Start Condition Hold Time)	t _{SSSTART}	引脚: SCK	0.6	—	—	ns
停止条件设置时间 (Stop Condition Setup Time)	t _{SSSTOP}	引脚: SCK	0.6	—	—	ns

地址建立时间 (Address setup time)	t_{HSTART}	引脚: SCK	0.6	—	—	ns
地址保持时间 (Address hold time)	t_{HSTART}	引脚: SCK	0.6	—	—	ns
数据建立时间 (Data setup time)	t_{HD}	引脚: SDA	0	—	—	ns
数据保持时间 (Data hold time)	t_{HD}	引脚: SDA	300	—	—	ns
上升时间 (Rise Time for Data and Clock Pin)	t_R	引脚: SCK	—	—	300	ns
下降时间 (Fall Time for Data and Clock Pin)	t_F	引脚: SCK	—	—	300	ns
启动前时间 (Idle Time before a New Transmission can Start)	t_{IDLE}		1.3	—	—	ns

* (VDD = 1.65V ~ 3.3V, Ta = 25°C)

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



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		—	—	1.0	us
复位保持低电平的时间	t_{RW}	引脚: RES	5.0	—	—	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)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)设定寻址方式(双指令) (Set Memory Addressing Mode)	0	0	0	1	0	0	0	0	0	0x20: 设定寻址方式																			
	0	0	0	0	0	0	0	A1	A0	<table border="1"> <thead> <tr> <th>A1</th> <th>A0</th> <th>寻址方式</th> <th>指令</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>水平</td> <td>0x00</td> </tr> <tr> <td>0</td> <td>1</td> <td>垂直</td> <td>0x01</td> </tr> <tr> <td>1</td> <td>0</td> <td>页</td> <td>0x02</td> </tr> <tr> <td>1</td> <td>1</td> <td colspan="2">无效指令</td> </tr> </tbody> </table>	A1	A0	寻址方式	指令	0	0	水平	0x00	0	1	垂直	0x01	1	0	页	0x02	1	1	无效指令
A1	A0	寻址方式	指令																										
0	0	水平	0x00																										
0	1	垂直	0x01																										
1	0	页	0x02																										
1	1	无效指令																											
(4)设置列地址起始-结束(Set Column Address)	0	0	0	1	0	0	0	0	1	0x21: 设置列地址																			
	0	0	A6	A5	A4	A3	A2	A1	A0	设置起始列地址;范围0~127																			
	0	0	B6	B5	B4	B3	B2	B1	B0	设置结束列地址;范围0~127																			
(5)设置页地址起始-结束(Set Page Address)	0	0	0	1	0	0	0	1	0	0x22: 设置页地址																			
	0	0	0	0	0	0	A2	A1	A0	设置起始页地址;范围0~7																			
	0	0	0	0	0	0	B2	B1	B0	设置结束页地址;范围0~7																			
(6)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																					
(7)	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																										
(8)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: 列地址从右到左, 1 0xA1: 列地址从左到右																			
(9)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																			
(10)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																			
(11)设置显示行数	0	1	0	1	0	1	0	0	0	0xA8: 设置显示行数																			



(Set Multiplex Ration)	0	*	*	共 6 位, 0~63 共 64 级						设置范围: 00~3f 针对本型号为 0x3f, 64 行																				
(12) 显示开/关 (display on/off)	0	1	0	1	0	1	1	1	0	显示开/关: 0xAE :关, 0xAF : 开																				
(13)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。																				
(14) 行扫描顺序选择 (Common output mode select)	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 0xC8 :反转扫描顺序: 从上到下																				
(15) 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																				
(16) OLED 显示时钟/ 振荡频率设 (Oscillator Frequency(Double Bytes Command))	0	1	1	0	1	0	1	0	1	0xD5 :振荡频率设置																				
	0	A7	A6	A5	A4	A3	A2	A1	A0	D3—D0 : 显示时钟分频 D7—D4 : 显示频率 详情请看 IC 资料第 40 页																				
(17)设置预充电周 (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	A7	A6	A5	A4	A3	A2	A1	A0	设置预充电时间: 0xF1 :默认值, 范围: 00—ff																				
(18)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	A5	A4	0	0	1	0	设置COM配置模式 0x02 :上下映射 0x32 :左右映射 0x12 :正常显示																				
(19)设置VCOM (Set VCOMH Deselect Level)	0	1	1	0	1	1	0	1	1	0xDB :设置VCOM																				
	0	0	A6	A5	A4	0	0	0	0	<table border="1"> <thead> <tr> <th>A6</th> <th>A5</th> <th>A4</th> <th>VCOMH</th> <th>指令</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0.65xVcc</td> <td>0x00</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0.77xVcc</td> <td>0x20</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>0.83xVcc</td> <td>0x30</td> </tr> </tbody> </table>	A6	A5	A4	VCOMH	指令	0	0	0	0.65xVcc	0x00	0	1	0	0.77xVcc	0x20	0	1	1	0.83xVcc	0x30
A6	A5	A4	VCOMH	指令																										
0	0	0	0.65xVcc	0x00																										
0	1	0	0.77xVcc	0x20																										
0	1	1	0.83xVcc	0x30																										
(20)空指令 (NOP)	0	1	1	1	0	0	0	1	1	0xE3 :空操作																				
(21) 读状态 (Status read)	0	0	A6	0	0	0	0	0	0	暂不可用																				
(22) 设置内部/外部升 压 (Charge Pump Setting)	0	1	0	0	0	1	1	0	1	0x8d : 设置内部/外部升压																				
	0	0	0	1	0	0	A2	0	0	0x10 : 使用外部升压 0x14 : 使用内部升压																				
(23)连续水平滚动设置 (Continuous Horizontal Scroll)	0	0	0	1	0	0	1	1	X0	0x26 : 水平向右滚动 0x27 : 水平向左滚动																				
	0	0	0	0	0	0	0	0	0	0x00 : 虚拟指令																				



Setup)	0	0	0	0	0	0	B2	B1	B0	设置起始滚动页地址, 范围: 00--07																																													
	0	0	0	0	0	0	C2	C1	C0	<table border="1"> <thead> <tr> <th>C2</th><th>C1</th><th>C0</th><th>间隔</th><th>指令</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td><td>5帧</td><td>0x00</td></tr> <tr> <td>0</td><td>0</td><td>1</td><td>64帧</td><td>0x01</td></tr> <tr> <td>0</td><td>1</td><td>0</td><td>128帧</td><td>0x02</td></tr> <tr> <td>0</td><td>1</td><td>1</td><td>256帧</td><td>0x03</td></tr> <tr> <td>1</td><td>0</td><td>0</td><td>3帧</td><td>0x04</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>4帧</td><td>0x05</td></tr> <tr> <td>1</td><td>1</td><td>0</td><td>25帧</td><td>0x06</td></tr> <tr> <td>1</td><td>1</td><td>1</td><td>2帧</td><td>0x07</td></tr> </tbody> </table>	C2	C1	C0	间隔	指令	0	0	0	5帧	0x00	0	0	1	64帧	0x01	0	1	0	128帧	0x02	0	1	1	256帧	0x03	1	0	0	3帧	0x04	1	0	1	4帧	0x05	1	1	0	25帧	0x06	1	1	1	2帧	0x07
	C2	C1	C0	间隔	指令																																																		
	0	0	0	5帧	0x00																																																		
	0	0	1	64帧	0x01																																																		
	0	1	0	128帧	0x02																																																		
	0	1	1	256帧	0x03																																																		
	1	0	0	3帧	0x04																																																		
	1	0	1	4帧	0x05																																																		
	1	1	0	25帧	0x06																																																		
1	1	1	2帧	0x07																																																			
0	0	0	0	0	0	D2	D1	D0	设置结束滚动页地址, 范围: 00--07																																														
0	0	0	0	0	0	0	0	0	0x00: 虚拟指令																																														
0	1	1	1	1	1	1	1	1	0xFF: 虚拟指令																																														
(24)连续垂直和水平滚动设置 (Continuous Vertical and Horizontal Scroll Setup)	0	0	0	1	0	1	0	X1	X0	0x29: 垂直向右水平滚动 0x2A: 垂直向左水平滚动																																													
	0	0	0	0	0	0	0	0	0	0x00: 虚拟指令																																													
	0	0	0	0	0	0	B2	B1	B0	设置起始滚动页地址, 范围: 00--07																																													
	0	0	0	0	0	0	C2	C1	C0	<table border="1"> <thead> <tr> <th>C2</th><th>C1</th><th>C0</th><th>间隔</th><th>指令</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td><td>5帧</td><td>0x00</td></tr> <tr> <td>0</td><td>0</td><td>1</td><td>64帧</td><td>0x01</td></tr> <tr> <td>0</td><td>1</td><td>0</td><td>128帧</td><td>0x02</td></tr> <tr> <td>0</td><td>1</td><td>1</td><td>256帧</td><td>0x03</td></tr> <tr> <td>1</td><td>0</td><td>0</td><td>3帧</td><td>0x04</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>4帧</td><td>0x05</td></tr> <tr> <td>1</td><td>1</td><td>0</td><td>25帧</td><td>0x06</td></tr> <tr> <td>1</td><td>1</td><td>1</td><td>2帧</td><td>0x07</td></tr> </tbody> </table>	C2	C1	C0	间隔	指令	0	0	0	5帧	0x00	0	0	1	64帧	0x01	0	1	0	128帧	0x02	0	1	1	256帧	0x03	1	0	0	3帧	0x04	1	0	1	4帧	0x05	1	1	0	25帧	0x06	1	1	1	2帧	0x07
	C2	C1	C0	间隔	指令																																																		
	0	0	0	5帧	0x00																																																		
	0	0	1	64帧	0x01																																																		
	0	1	0	128帧	0x02																																																		
	0	1	1	256帧	0x03																																																		
	1	0	0	3帧	0x04																																																		
1	0	1	4帧	0x05																																																			
1	1	0	25帧	0x06																																																			
1	1	1	2帧	0x07																																																			
0	0	0	0	0	0	D2	D1	D0	设置结束滚动页地址, 范围: 00--07																																														
0	0	0	E5	E4	E3	E2	E1	E0	设置垂直滚动行偏移, 范围: 01--3f																																														
(25)停止滚动 (Deactivate scroll)	0	0	0	1	0	1	1	1	0	0x2E: 停止滚动 配合: 0x26、0x27、0x29、0x2A使用																																													
(26)开始滚动 (Activate scroll)	0	0	0	1	0	1	1	1	1	0x2F: 开始滚动																																													
(27)设置垂直滚动区域 (Set Vertical Scroll Area)	0	1	0	1	0	0	0	1	1	0xA3: 设置垂直滚动区域																																													
	0	0	0	A5	A4	A3	A2	A1	A0	滚动起始行, 范围: 00--63																																													
	0	0	0	B6	B5	B4	B3	B2	B1	B0	滚动结束行, 范围: 01--64																																												

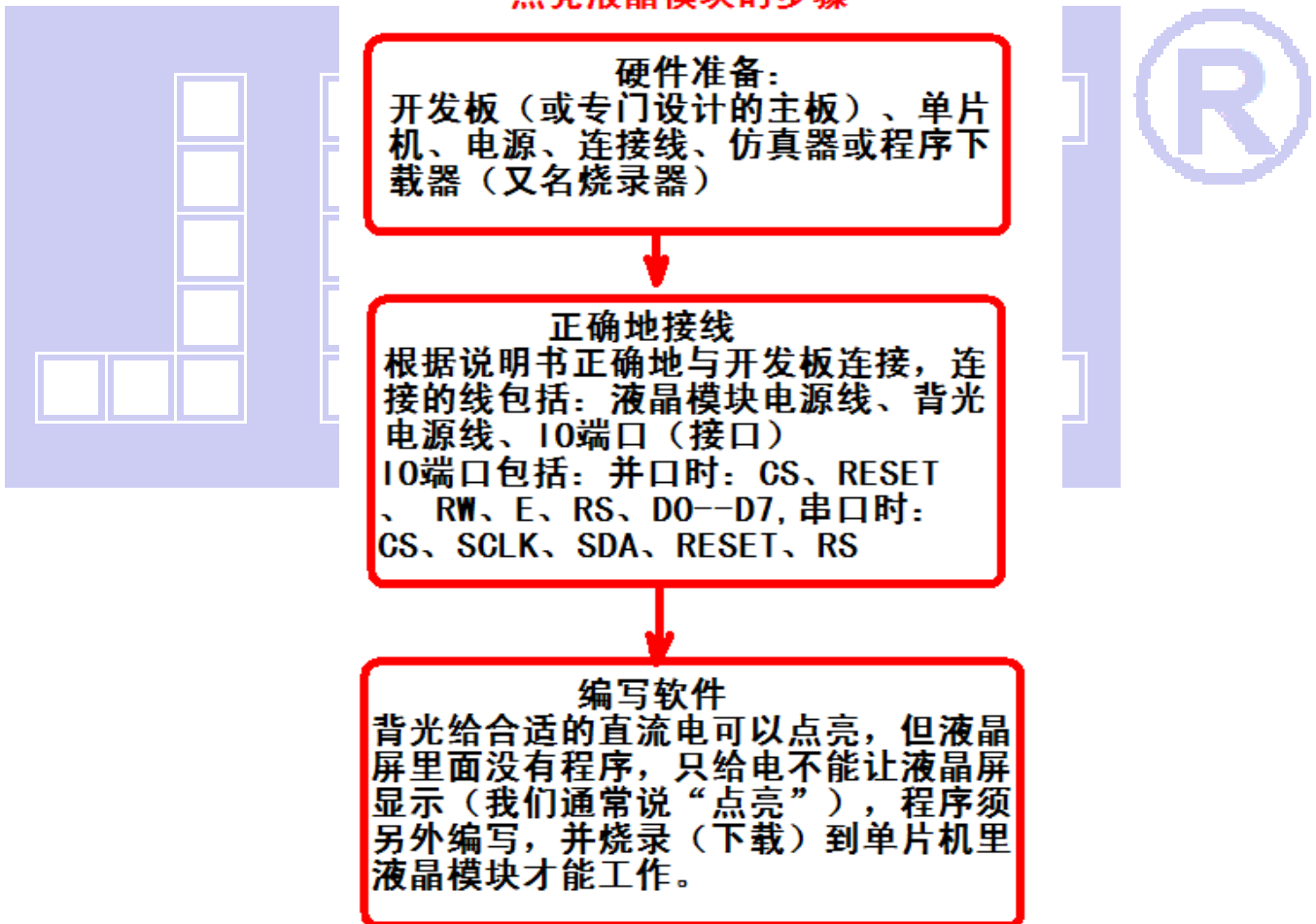
(28)设置淡出和闪烁 (Set Fade Out and Blinking)	0	0	0	1	0	0	0	1	1	0x23 : 设置淡出和闪烁
	0	0	0	A5	A4	A3	A2	A1	A0	A[5:4]=00b禁用淡出/闪烁模式 A[5:4]=10b启用淡出模式 A[5:4]=11b启用闪烁模式 A[3:0]=0000—FFFF,8—128帧, 闪烁或淡出间隔
(29) 设置放大 (Set Zoom In)	0	1	1	0	1	0	1	1	0	0xD6 : 设置放大
	0	0	0	0	0	0	0	0	A0	0x00 : 禁用 0x01 : 启用

请详细参考 IC 资料”SSD1312.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



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:

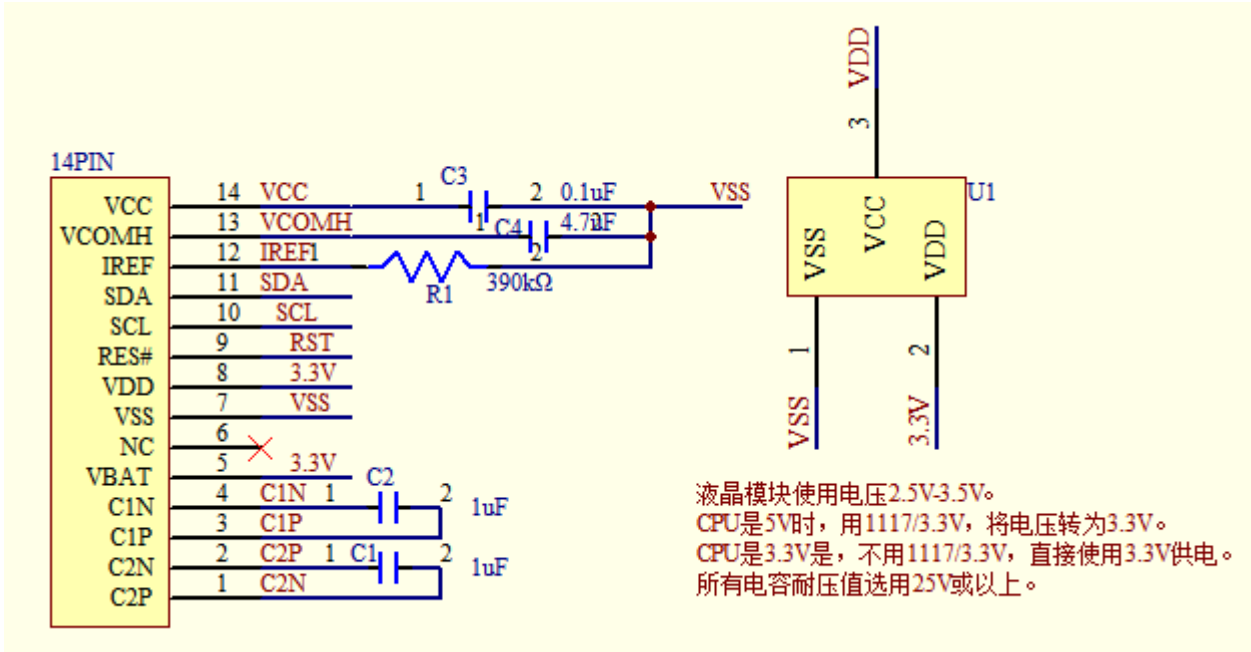


IIC 行接口

7.3.1 程序:

点亮液晶模块的编程步骤





```
// OLED 演示程序
// OLED 模块型号: JLX12864OLED-S130, IIC 行接口!
// 驱动 IC 是:SSD1312
// 资料(源程序、驱动手册、使用说明书等) 销售统一发
#include <STC15F2K60S2.H>
//=====IIC 接口=====
sbit lcd_scl =P3^1; //接口定义:lcd_sclk 就是 LCD 的 SCLK //SCLK 接到“D0”脚
sbit lcd_sda =P3^2; //接口定义:lcd_sda 就是 LCD 的 SDA //SDIN 接到“D1”脚
sbit lcd_reset=P3^0; //接口定义:lcd_reset 就是 LCD 的 RESET
//sbit IM1=P3^2;
//sbit lcd_dc=P1^0;
//sbit lcd_cs1=P3^1;
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
```



```

    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()
{
    lcd_reset=0;        //低电平复位
    delay(500);
    lcd_reset=1;        //复位完毕
    delay(200);

    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(0xa6);
    
```

```

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(0x16); //微调对比度的值, 可设置范围 0x00~0xff
                        // (0x16) 62cd/m2

transfer_command(0xd9); //Set Pre-Charge Period
transfer_command(0x42);

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)+0x10); //设置列地址的高 4 位
    transfer_command(column&0x0f);          //设置列地址的低 4 位
}
    
```

```
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<64; j++)
        {
            transfer_data(data1);
            transfer_data(data2);
        }
    }
}
```

```
//测试外框是否缺划（少行、少列）
```

```
void test_box()
```

```
{
    int i;
    //上横线:
    for(i=1; i<129; i++)
    {
        lcd_address(1, i);
        transfer_data(0x01);
    }
    //下横线:
    for(i=1; i<129; i++)
    {
        lcd_address(8, i);
        transfer_data(0x80);
    }
    //左竖线:
    for(i=1; i<9; i++)
```



```

    {
        lcd_address(i, 1);
        transfer_data(0xff);
    }
//右竖线:
    for(i=1;i<9;i++)
    {
        lcd_address(i, 128);
        transfer_data(0xff);
    }
}

```

```

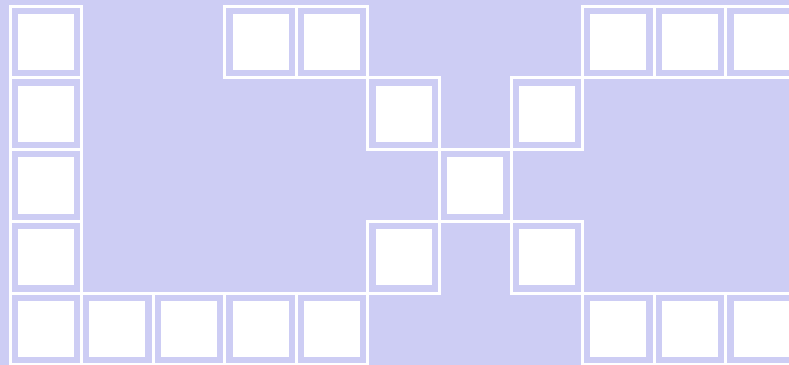
//测试
void test()

```

```

{
    test_box();
    waitkey();
    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();
}

```



```

//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;
}

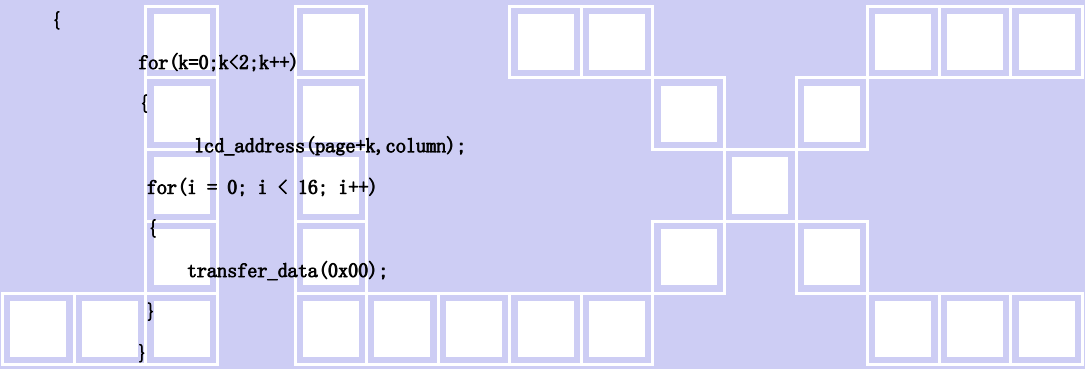
if(address != 1)// 显示汉字
{
    for(k=0;k<2;k++)
    {
        lcd_address(page+k, column);
        for(i = 0; i < 16; i++)
        {
            transfer_data(Chinese_code_16x16[address]);
            address++;
        }
    }
    j += 2;
}

```

```

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)
        {

```

```

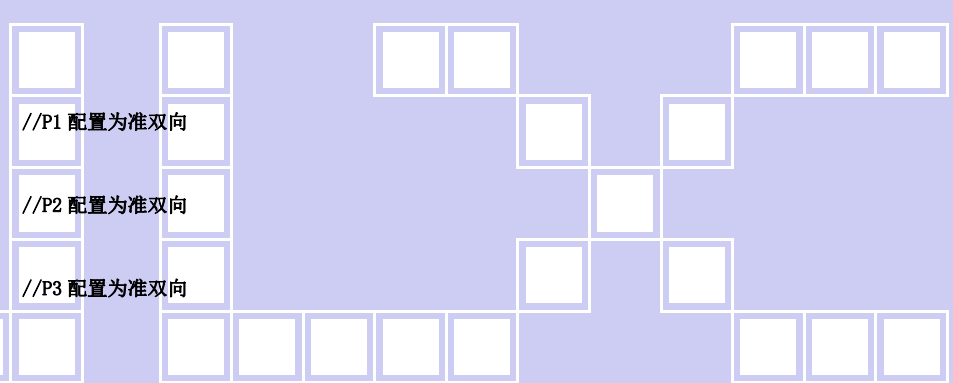
temp[0] = text[i];
temp[1] = text[i + 1];
temp[2] = '\0';          //汉字为两个字节
display_string_16x16(page, column, temp); //显示汉字
column += 16;
i += 2;
}
else
{
temp[0] = text[i];
temp[1] = '\0';          //字母占一个字节
display_string_8x16(page, column, temp); //显示字母
column += 8;
i++;
}
}
}


```

```

void main(void)
{
P1M1=0x00;
P1M0=0x00; //P1 配置为准双向
P2M1=0x00;
P2M0=0x00; //P2 配置为准双向
P3M1=0x00;
P3M0=0x00; //P3 配置为准双向
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, 字符串指针)
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");

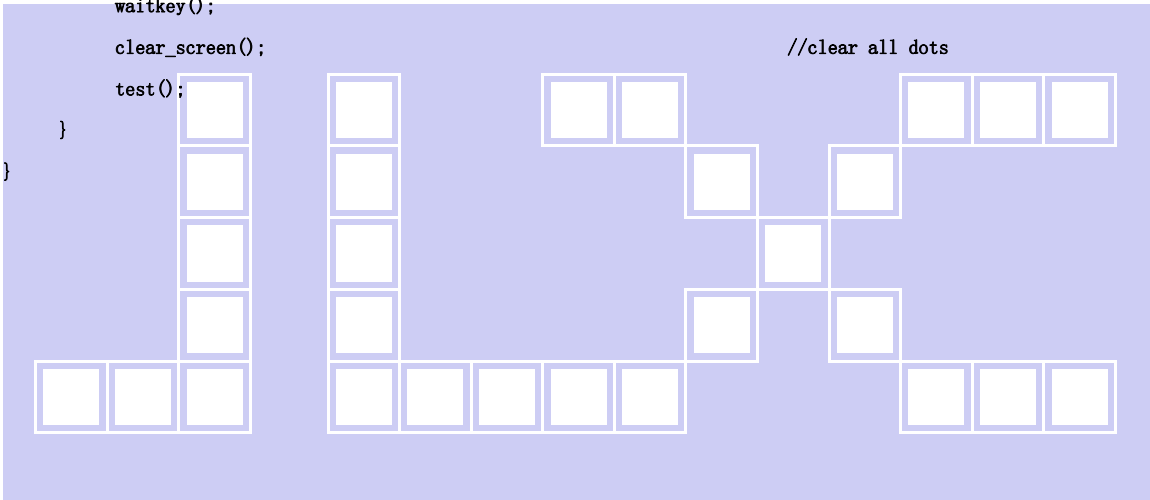
```

```

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(); //clear all dots
display_128x64(bmp1);
waitkey();
clear_screen(); //clear all dots
display_128x64(bmp2);
waitkey();
clear_screen(); //clear all dots
test();
}
}

```



-END-