

# JLX9664G-959-BN 使用说明书

## (焊接式 FPC)

### 目 录

序号	内 容 标 题	页 码
1	概述	2
2	图像型点阵液晶模块的特性	2
3	外形尺寸及接口引脚功能	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	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 JLX9664G-959-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。

JLX9664G-959-BN 可以显示 96 列\*64 行点阵单色图片，或显示 16\*16 点阵的汉字 6 个\*4 行，或显示 8\*16 点阵的英文、数字、符号 12 个\*4 行。或显示 5\*8 点阵的英文、数字、符号 16 个\*8 行。

## 2. JLX9664G-959-BN 图像型点阵液晶模块的特性

2.1 结构牢：背光带有挡墙，焊接式 FPC。

2.2 IC 采用 ST7567A, 功能强大，稳定性好

2.3 功耗低：不带背光 1mW (3.3V\*0.3mA)，带背光不大于 140mW (3.3V\*30mA)；

2.4 显示内容：

(1) 96\*64 点阵单色图片，或其它小于 96\*64 点阵的单色图片；

(2) 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 6 字\*4 行；

(3) 按照 8\*16 点阵汉字来计算可显示 12 字\*4 行；

(4) 按照 5\*8 点阵汉字来计算可显示 16 字\*8 行；

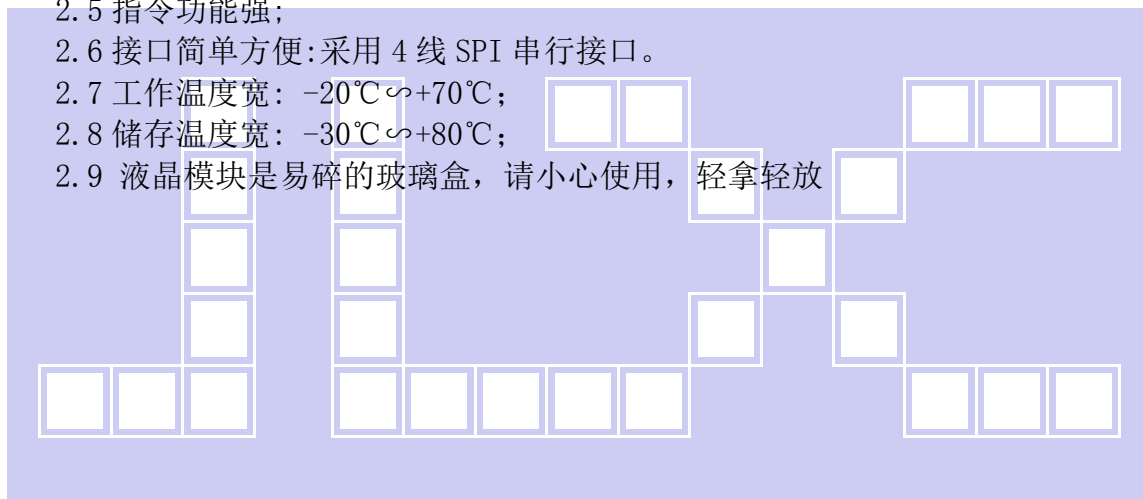
2.5 指令功能强；

2.6 接口简单方便:采用 4 线 SPI 串行接口。

2.7 工作温度宽：-20℃~+70℃；

2.8 储存温度宽：-30℃~+80℃；

2.9 液晶模块是易碎的玻璃盒，请小心使用，轻拿轻放



## 3. 外形尺寸及接口引脚功能

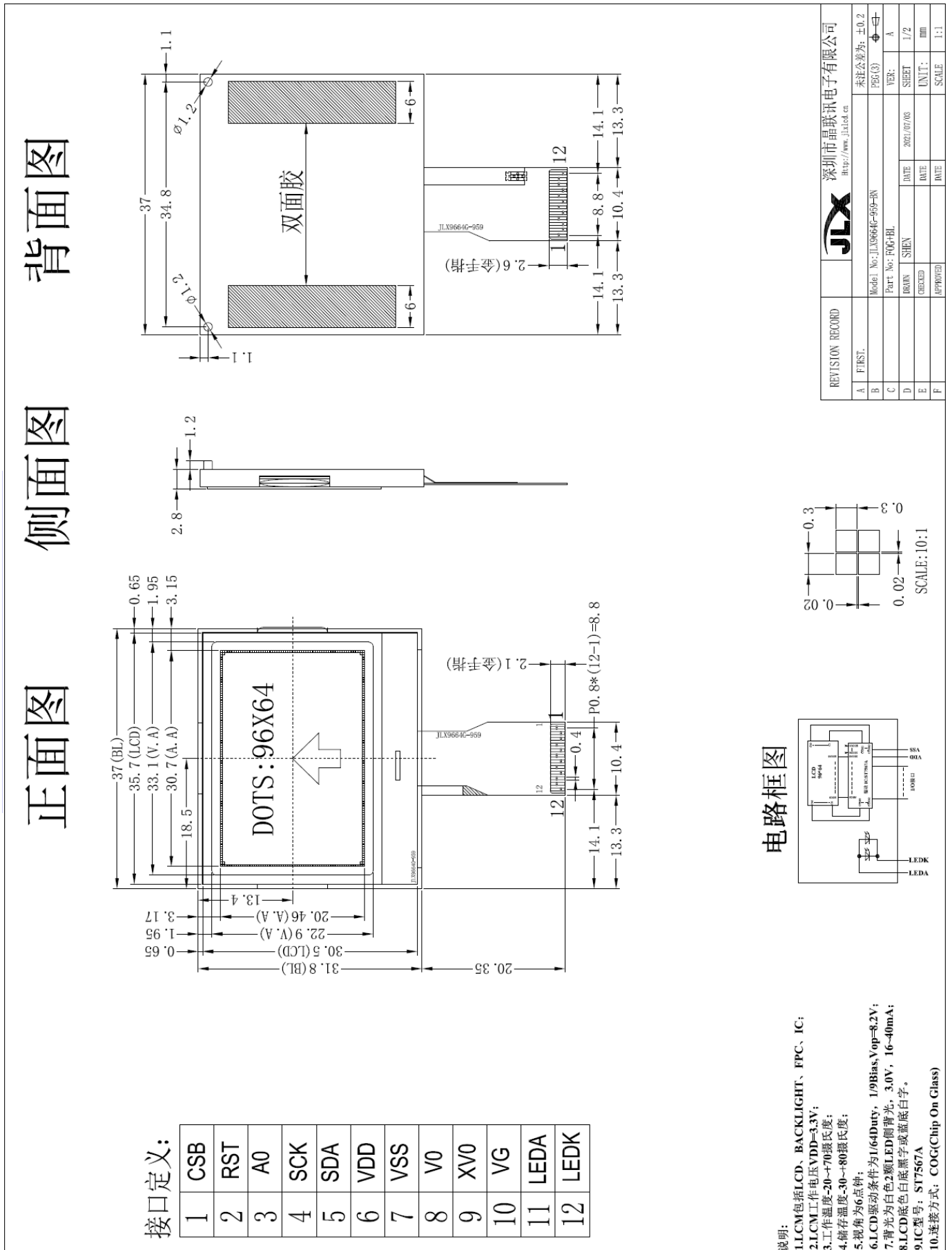


图 1. 外形尺寸

## 模块的接口引脚功能

引线号	符号	名称	功能
1	CS (CSB)	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	A0 (RS)	寄存器选择信号	H: 数据寄存器 0: 指令寄存器
4	SCK	串行时钟	串行时钟
5	SDA	串行数据	串行数据
6	VDD	供电电源正极	供电电源正极, 3.3V
7	VSS	接地	0V
8	VO	偏置电压	
9	XVO	偏置电压	
10	VG	LCD 倍压输出	
11	LEDA	背光电源正极	背光电源正极, 电压 3.0V (接 3.3V 串 20 欧以上阻值的电阻)
12	LEDK	背光电源负极	背光电源负极

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  $96 \times 64$  点阵, 96 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)。

## 4.2 工作电路框图:

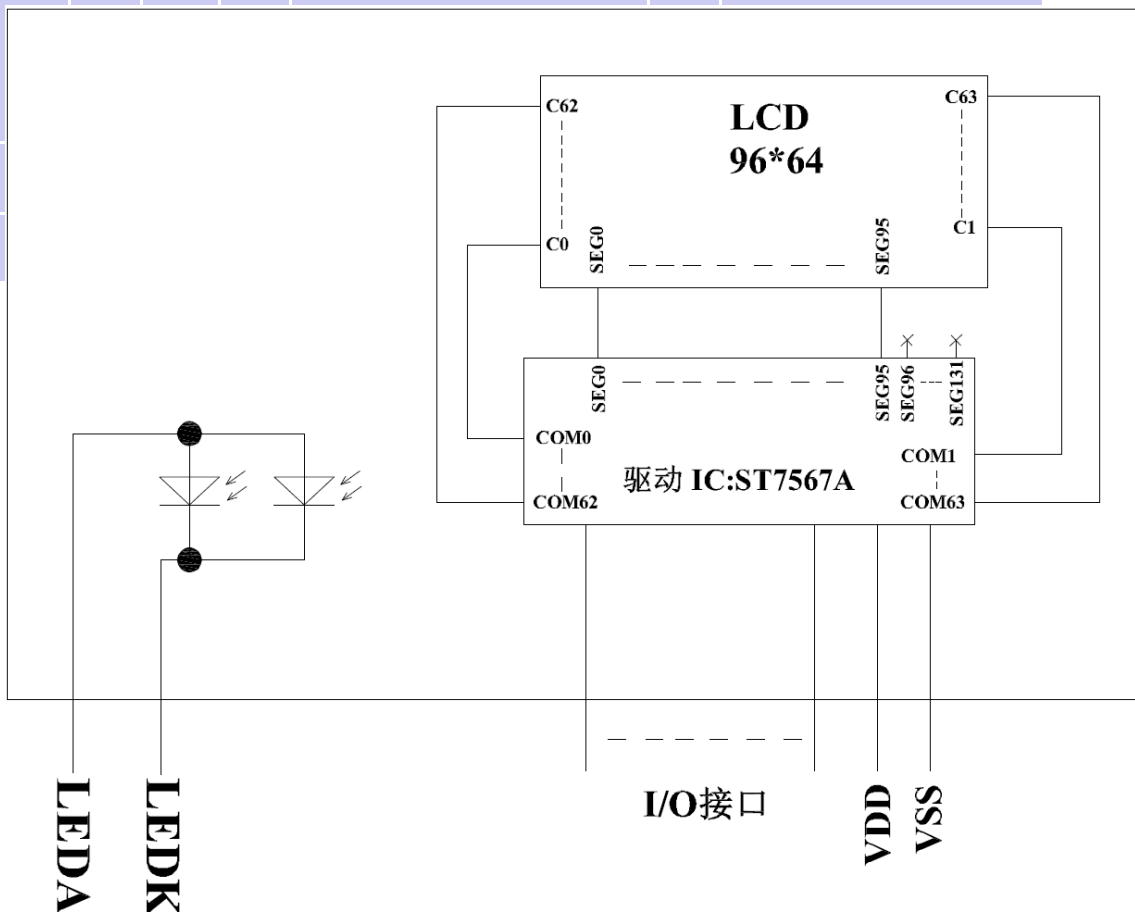


图 2: JLX9664G-959-BN 图像点阵型液晶模块的电路框图

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

存储温度:  $-30^{\circ}\text{C} \sim +80^{\circ}\text{C}$ ;

背光白色;

正常工作电流为:  $16 \sim 40\text{mA}$  (LED 灯数共 3 颗);

工作电压:  $3.0\text{V}$ ; (串 20 欧电阻接 3.3V)

## 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	
电源	VDD - VSS	-0.3		3.6	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

名称	符号	测试条件	标准值			单位
			MIN	TYPE	MAX	
工作电压	VDD		2.7	3.3	3.6	V
背光工作电压	VLED		2.6	3.0	3.1	V
输入高电平	VIH	-	$0.8\text{VDD}$	-	VDD	V
输入低电平	VIO	-	0	-	$0.2\text{VDD}$	V
输出高电平	VOH	$\text{IOH} = 0.2\text{mA}$	$0.8\text{VDD}$		VDD	V
输出低电平	VOO	$\text{IOO} = 1.2\text{mA}$	0	-	$0.2\text{VDD}$	V
模块工作电流	IDD	VDD = $3.0\text{V}$	-	0.3	1.0	mA
背光工作电流	ILED	VLED= $3.0\text{V}$ (共 3 颗 LED 灯并联)	16	30	40	mA

表 3: 直流 (DC) 参数

## 6. 读写时序特性

### 6.1 串行接口:

从 CPU 写到 ST7567A (Writing Data from CPU to ST7567A)

System Bus Timing for 4-Line Serial Interface

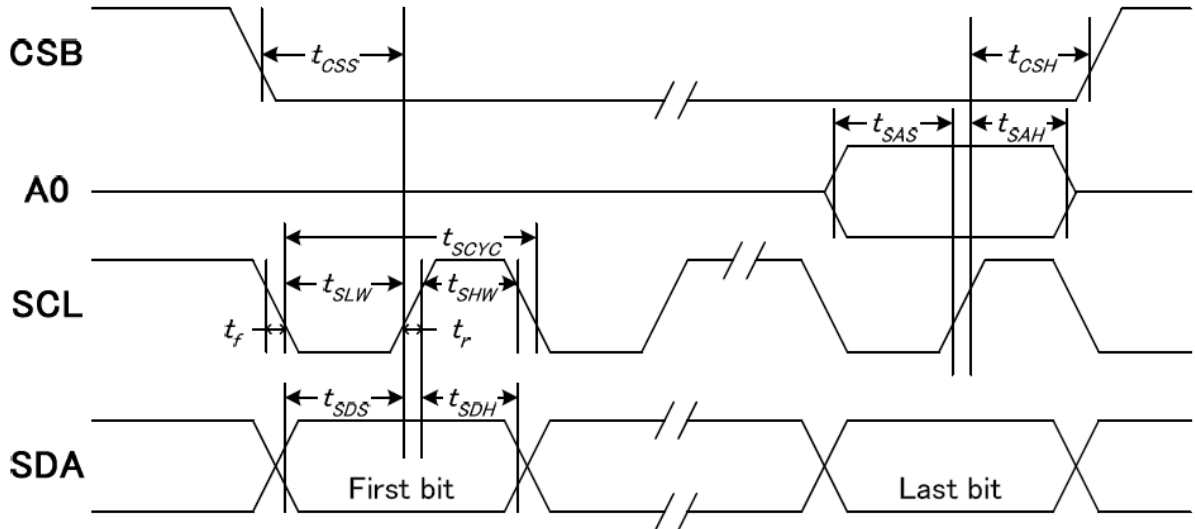


图 3. 从 CPU 写到 ST7567A (Writing Data from CPU to ST7567A)

### 6.2 串行接口: 时序要求 (AC 参数):

写数据到 ST7567A 的时序要求:

4 表

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)	$T_{scyc}$	引脚: SCK	50	—	—	ns
保持SCK高电平脉宽 (SCK "H" pulse width)	$T_{shw}$	引脚: SCK	25	—	—	ns
保持SCK低电平脉宽 (SCK "L" pulse width)	$T_{slw}$	引脚: SCK	25	—	—	ns
地址建立时间 (Address setup time)	$T_{sas}$	引脚: RS	20	—	—	ns
地址保持时间 (Address hold time)	$T_{sah}$	引脚: RS	10	—	—	ns
数据建立时间 (Data setup time)	$T_{sds}$	引脚: SI	20	—	—	ns
数据保持时间 (Data hold time)	$T_{sdh}$	引脚: SI	10	—	—	ns
片选信号建立时间 (CS-SCL time)	$T_{css}$	引脚: CS	20	—	—	ns
片选信号保持时间 (CS-SCL time)	$T_{csh}$	引脚: CS	40	—	—	ns

VDD = 3.3V, Ta = 25°C

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

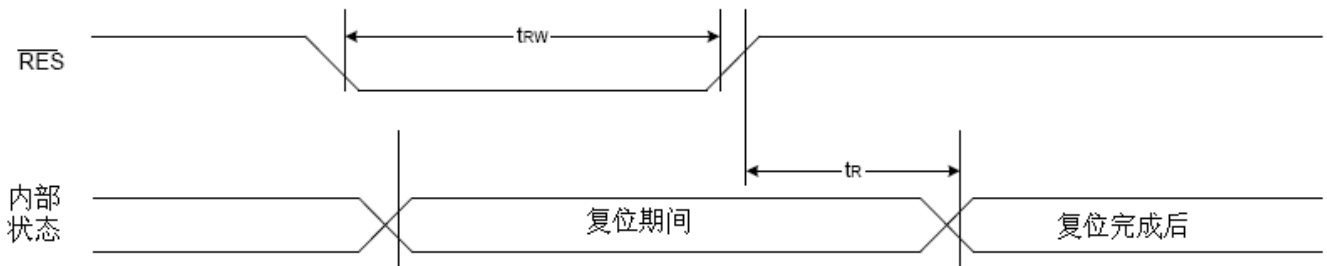


图 4: 电源启动后复位的时序

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	tr		10	—	200	ms
复位保持低电平的时间	trw	引脚: RES	200	—	—	ms

## 7. 指令功能:

### 7.1 指令表

表 6.

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	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	
		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。本液晶模块使用串行接口, 不具备此功能。	
(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。

										<b>本液晶模块使用串行接口, 不具备此功能。</b>
(8) 显示列地址增减 (ADC select)		1	0	1	0	0	0	0	0	显示列地址增减: <b>0xA0</b> : 常规: 列地址从左到右, <b>0xA1</b> : 反转: 列地址从右到左
(9) 显示正显/反显 (Display normal/reverse)	0	1	0	1	0	0	1	1	0	显示正显/反显: <b>0xA6</b> : 常规: 正显 <b>0xA7</b> : 反显
(10) 显示全部点阵 (Display all points)	0	1	0	1	0	0	1	0	0	显示全部点阵: <b>0xA4</b> : 常规 <b>0xA5</b> : 显示全部点阵
(11) LCD 偏压比设置 (LCD bias set)	0	1	0	1	0	0	0	1	0	设置偏压比: <b>0xA2</b> : BIAS=1/9 (常用) <b>0xA3</b> : BIAS=1/7
(12) 读-改-写 (Read-modify-write)	0	1	1	1	0	0	0	0	0	<b>0XE0</b> : “读-改-写” 开始。 <b>本液晶模块使用串行接口, 不具备此功能。</b> 详情请参考IC资料
(13) 退出上述“读-改-写”指令(End)	0	1	1	1	0	1	1	1	0	<b>0xEE</b> : 上述“读-改-写”指令结束 <b>本液晶模块使用串行接口, 不具备此功能。</b> 详情请参考 IC 资料
(14) 软件复位 (Reset)	0	1	1	1	0	0	0	1	0	<b>0XE2</b> : 软件复位。
(15) 行扫描顺序选择 (Common output mode select)		1	1	0	0	0	0	0	0	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
(16) 电源控制 (Power control set)		0	0	1	0	1				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。
(17) 选择内部电阻比例	0	0	0	1	0	0				<b>内部电压值电阻设置</b> 选择内部电阻比例 (Rb/Ra): 可以理解为 <b>粗调</b> 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
(18)	内部设置液晶电压模式	0	1	0	0	0	0	0	1	设置内部电阻微调, 可以理解为 <b>微调</b> 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	<b>6 位电压值数据, 0~63 共 64 级</b>					
(19) 静态图标显示: 开/关	0	1	0	1	0	1	1	0	0	静态图标的开关设置: <b>0xAC</b> : 关, <b>0xAD</b> : 开。 此指令在进入及退出睡眠模式时起作用
(20) 升压倍数选择	0	1	1	1	1	1	0	0	0	选择升压倍数:



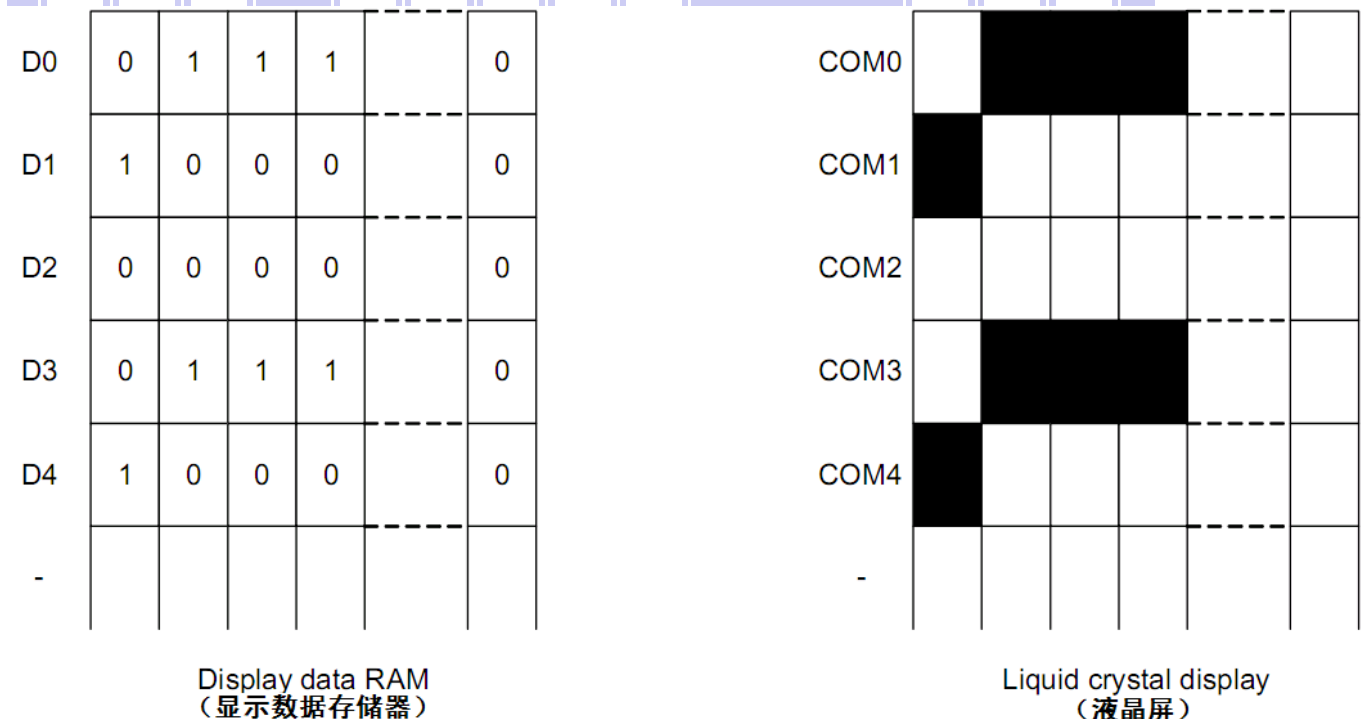
(Booster ratio set)		0	0	0	0	0	0	2 位数设置 升压倍数	00: 2 倍, 3 倍, 4 倍 01: 5 倍 11: 6 倍。本模块外部已设置升压倍数为 4 倍, 不必使用此指令	
(21) 省电模式 (Power save)									省电模式, 此非一条指令, 是由“(10)显示全部点阵”、“(19)静态图标显示: 开/关等指令合成一个“省电功能”。详细看 IC 规格书 “POWER SAVE”部分	
(22)空指令 (NOP)	0	1	1	1	0	0	0	1	1	空操作
(23) 测试 (Test)	0	1	1	1	1	*	*	*	*	内部测试用, 千万别用!

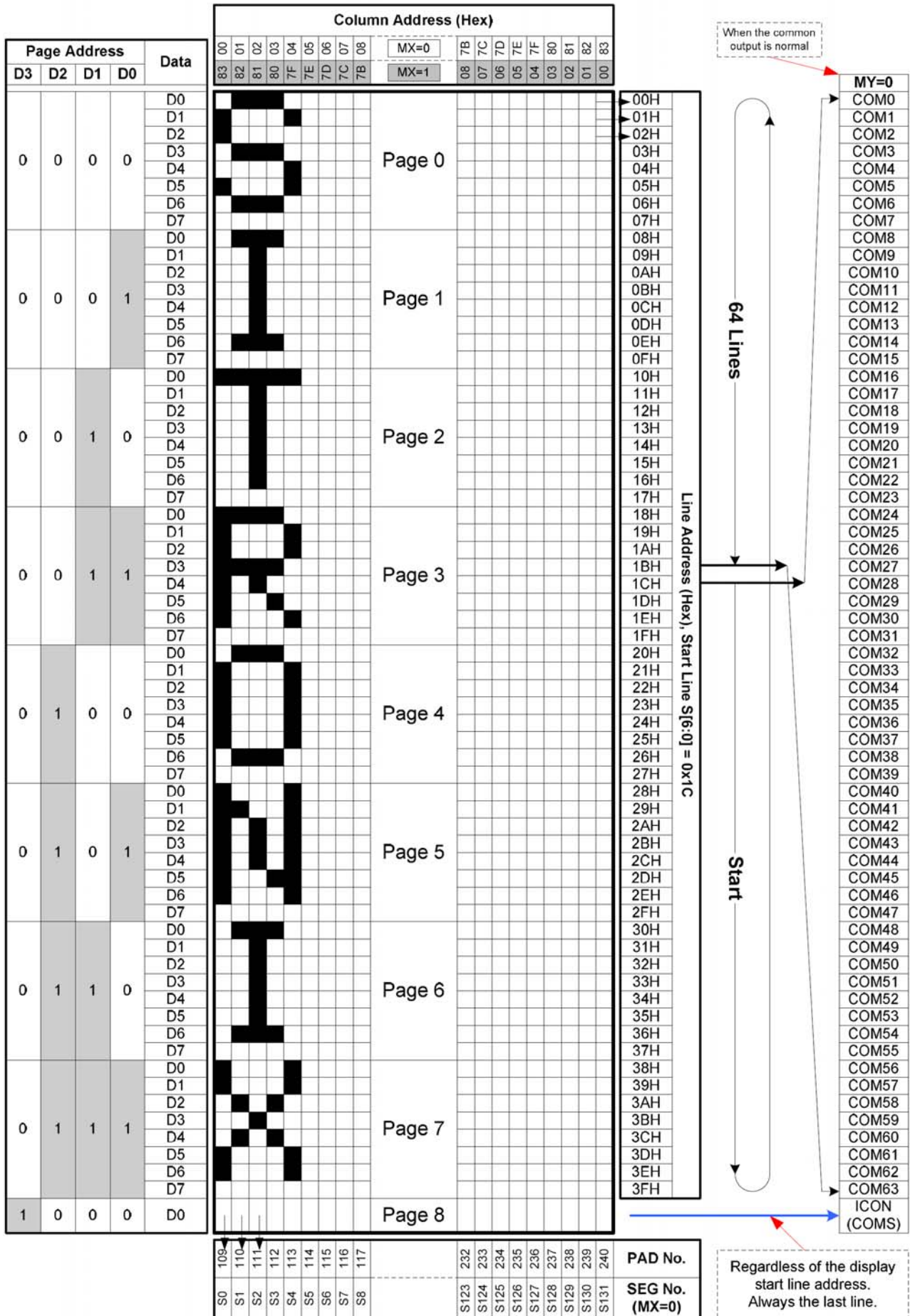
请详细参考 IC 资料”ST7567A\_V1.1a.PDF”的第 37~49 页。

## 7.2 点阵与 DD RAM(显示数据存储器)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 96\*64 点阵的屏分为 8 个“页”, 从第 0“页”到第 7“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:





### 7.3 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。

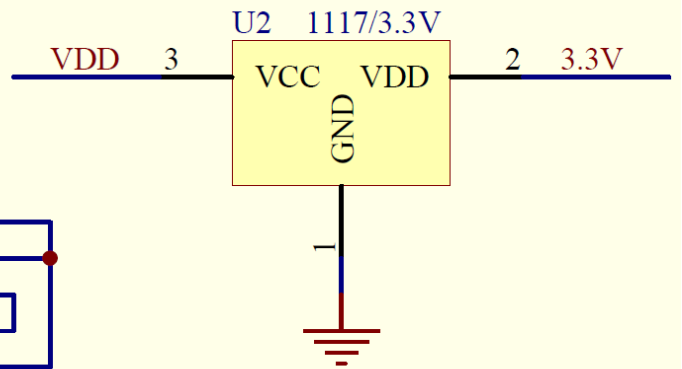
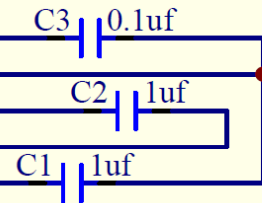
#### 7.4 程序举例:

液晶模块与 MPU (以 8051 系列单片机为例) 接口图如下:

12PIN

CS	1	CS
RST	2	RST
RS	3	RS
SCK	4	SCK
SDA	5	SDA
VDD	6	3.3V
VSS	7	VSS
V0	8	V0
XV0	9	XV0
VG	10	VG
LEDA	11	A
LEDK	12	VSS

12PIN-ST7567



所有电容的耐压值均选耐压为25V以上  
 液晶模块使用电压: 2.7V--3.5V  
 CPU是5V时, 用1117/3.3把电压转为3.3V  
 CPU是3.3V时, 直接使用

### 点亮液晶模块的编程步骤



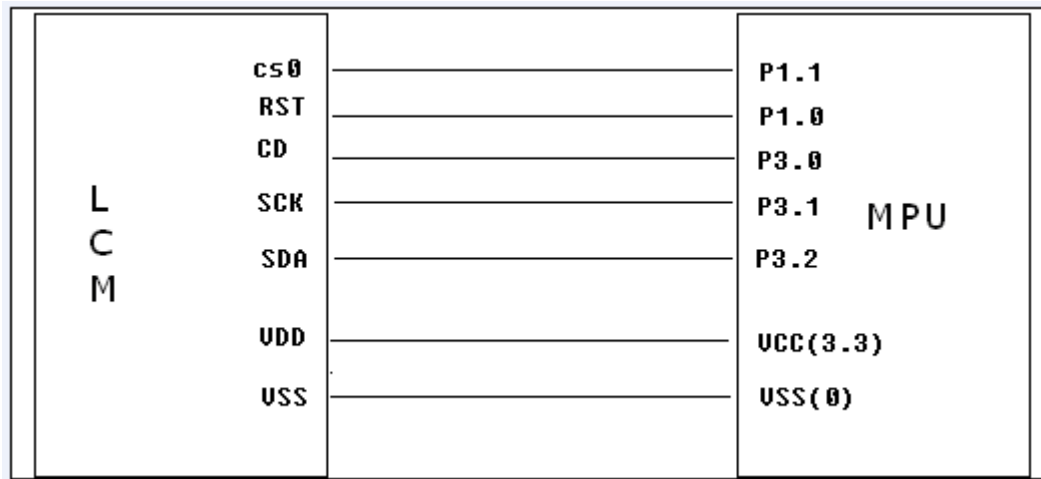


图 5. 串行接口和 MPU 对接示意图

/\* Test program for JLX9664G-959-BN, 串行接口

驱动 IC 是:ST7567A(or compatible)

晶联讯电子: 网址 <http://www.jlxlcd.cn>; <http://www.jlxlcd.com.cn>

\*/

```
#include <reg51.h>
```

```
#include <chinese.h> //取模数据存放处
```

```
sbit key=P2^0; //对应我司主板按键接口
```

```
sbit cs0=P1^1; //对应LCD的CS引脚
```

```
sbit RST=P1^0; //对应LCD的RST引脚
```

```
sbit CD=P3^0; //对应LCD的RS引脚
```

```
sbit SCK=P3^1; //对应LCD的SCK引脚
```

```
sbit SDA=P3^2; //对应LCD的SDA引脚
```

/\*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++);
}
```

```
void waitkey()
```

```
{
//repeat: if(key==1) goto repeat;
```

```
// else
    delay(2000);
}
```

```
/*写指令到 LCD 模块*/
```

```
void transfer_command(int data1)
```

```
{
    char i;
    cs0=0;
    CD=0;
    for(i=0;i<8;i++)
    {
        SCK=0;
        if(data1&0x80) SDA=1;
        else SDA=0;
        SCK=1;
        delay_us(1);
        data1=data1<<=1;
    }
    cs0=1;
```

```
/*写数据到 LCD 模块*/
```

```
void transfer_data(int data1)
```

```
{
    char i;
    cs0=0;
    CD=1;
    for(i=0;i<8;i++)
    {
        SCK=0;
        if(data1&0x80) SDA=1;
        else SDA=0;
        SCK=1;
        delay_us(1);
        data1=data1<<=1;
    }
    cs0=1;
}
```

```
/*LCD 模块初始化*/
```

```
//-----对比度值设置, 粗度 0x24, 微调 0x1e-----//
```

```
void initial_lcd()
```

```
{
    RST=0;    /*低电平复位*/
    delay(100); //100ms
    RST=1;    /*复位完毕*/
```

```

delay(200); //200ms

transfer_command(0xe2); //软复位*/
delay(5); //5ms

transfer_command(0x2c); //升压步聚 1*/
delay(5); //5ms

transfer_command(0x2e); //升压步聚 2*/
delay(5); //5ms

transfer_command(0x2f); //升压步聚 3*/
delay(5); //5ms

transfer_command(0x24); //粗调对比度, 可设置范围 0x20~0x27*/
transfer_command(0x81); //微调对比度*/
transfer_command(0x21); //微调对比度的值, 可设置范围 0x00~0x3f*/
transfer_command(0xa2); //1/9 偏压比 (bias) */

transfer_command(0xc8); //行扫描顺序: 从上到下*/
transfer_command(0xa0); //列扫描顺序: 从左到右*/
    
```

```

transfer_command(0x40); //起始行: 第一行开始*/
transfer_command(0xaf); //开显示*/
}

//=====设置 LCD 地址函数=====//
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
}

/*全屏清屏*/
void clear_screen()
{
    unsigned char i, j;
    for(i=0; i<9; i++)
    {
        lcd_address(1+i, 1);
        for(j=0; j<132; j++)
        {
            transfer_data(0x00);
        }
    }
}
}
    
```

```
//=====显示 96*64 点阵图像=====//
```

```
void display_graphic(uchar *dp)
{
    uchar i, j;
    for(j=0; j<8; j++)
    {
        lcd_address(j+1, 1);
        for(i=0; i<96; i++)
        {
            transfer_data(*dp);
            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<31; 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 reverse, uchar *dp)
```

```
{
    uchar i, j;
    for(j=0; j<2; j++)
    {
        lcd_address(page+j, column);
        for (i=0; i<16; i++)
        {
            if(reverse==1)
            {
                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*/
            }
            else
                transfer_data(*dp);
            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++;
        }
    }
}
```

```
//显示一串 8x16 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)

```

```
void display_string_8x16(uint page,uint column,uchar reverse,uchar *text)

```

```
{
    uint i=0, j, k, n, data1;
    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++)
                {
                    if(reverse==1) data1=~ascii_table_8x16[j][k+8*n];
                    else data1=ascii_table_8x16[j][k+8*n];
                    transfer_data(data1);
                }
            }
            i++;
            column+=8;
        }
        else
            i++;
    }
}
```

```
//显示一串 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)

```



```
void display_string_5x8(uint page,uint column,uchar reverse,uchar *text)
```

```
{
    uchar i=0,j,k,data1;
    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) data1=~ascii_table_5x8[j][k];
                else data1=ascii_table_5x8[j][k];
                transfer_data(data1);
            }
            if(reverse==1) transfer_data(0xff);
            else transfer_data(0x00);
            i++;
            column+=6;
```

```
        }
        else
        i++;
    }
}

//睡眠模式
void sleep() //进入睡眠
{
    transfer_command(0xac);/*静态图标关闭*/
    transfer_command(0x00);/*静态图标寄存器设置：关闭。此指令与上述指令一起完成静态图标关闭*/
    transfer_command(0xae);/*显示：关*/
    transfer_command(0xa5);/*全屏显示：开*/
}

void wake() //退出睡眠
{
    transfer_command(0xa4);/*全屏显示：关。进入正常模式*/
    transfer_command(0xad);/*静态图标开启*/
    transfer_command(0x03);/*静态图标寄存器设置：开。此指令与上述指令一起完成静态图标开启*/
    transfer_command(0xaf);/*显示：开*/
}
}
```

```
//主程序
```

```
void main(void)
{
    initial_lcd(); //LCD 初始化
    while(1)
    {
```

```

clear_screen(); //clear all dots
display_graphic(bmp1); //显示 96*64 的图片
waitkey(); //按键可用延时代替
clear_screen(); //clear all dots
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"); //同上
display_string_5x8(3, 43, 1, "1. Graphic"); //同上
display_string_5x8(4, 43, 0, "2. Chinese"); //同上
display_string_5x8(5, 43, 0, "3. Movie"); //同上
display_string_5x8(6, 43, 0, "4. Contrast"); //同上
display_string_5x8(7, 43, 0, "5. Mirror"); //同上
display_string_5x8(8, 1, 0, "PRE USER DEL NEW"); //同上
waitkey(); //按键可用延时代替
clear_screen(); //clear all dots
display_graphic_32x32(1, 17, chengl); //在第 1 页, 第 17 列显示单个汉字"成"*/
display_graphic_32x32(1, 49, gong); //在第 1 页, 第 49 列显示单个汉字"功"*/
display_graphic_16x16(6, 1, 1, zhuang1); //在第 6 页, 第 1 列显示单个汉字"状"
display_graphic_16x16(6, (1+16), 1, tail); //在第 6 页, 第 17 列显示单个汉字"态"
display_string_8x16(6, 33, 0, ":"); //显示":"
display_graphic_16x16(6, 41, 0, shi1); //在第 6 页, 第 41 列显示单个汉字"使"
display_graphic_16x16(6, (1+16*3+8), 0, yong1); //在第 6 页, 第 57 列显示单个汉字"用"
display_string_8x16(6, 79, 0, "00"); //显示 8x16 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)
waitkey(); //按键可用延时代替
clear_screen(); //clear all dots
display_string_8x16(1, 1, 0, "0123456789abcdef"); //显示 8x16 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_8x16(3, 1, 0, "~!@#$%^&*()_+="); //同上
display_string_5x8(5, 1, 1, "!#$%&'()*+,-./01234"); //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(6, 1, 0, "56789.:;<=>?@ABCDEFGHI"); //同上
display_string_5x8(7, 1, 0, "JKLMNOPQRSTUVWXYZ[\]^`"); //同上
display_string_5x8(8, 1, 0, "`_abcdefghijklmnopqrs"); //同上
waitkey(); //按键可用延时代替
}
}

```



**-END-**