

JLX12832G-28003-BN 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
5	技术参数	5
6	时序特性	6~10
7	指令功能及硬件接口与编程案例	10~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 JLX12832G-28003 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。

JLX12832G-28003 可以显示 128 列*32 行点阵单色图片，或显示 8 个*2 行 16*16 点阵的汉字，或显示 16 个*2 行 8*16 点阵的英文、数字、符号，或显示 21 个*4 行 5*8 点阵的英文、数字、符号。

2. JLX12864G-28003 图像型点阵液晶模块的特性

2.1 结构牢：背光带有挡墙，插接式 FPC。

2.2 IC 采用矽创公司 ST7567, 功能强大，稳定性好

2.3 功耗低:1~100mW（关掉背光：[0.3mA@3.3V](#), 打开背光不大于 100mW）；

2.4 +显示内容：

- 128*32 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*2 行。

- 按照 12*12 点阵汉字来计算可显示 10 字/行*2 行。

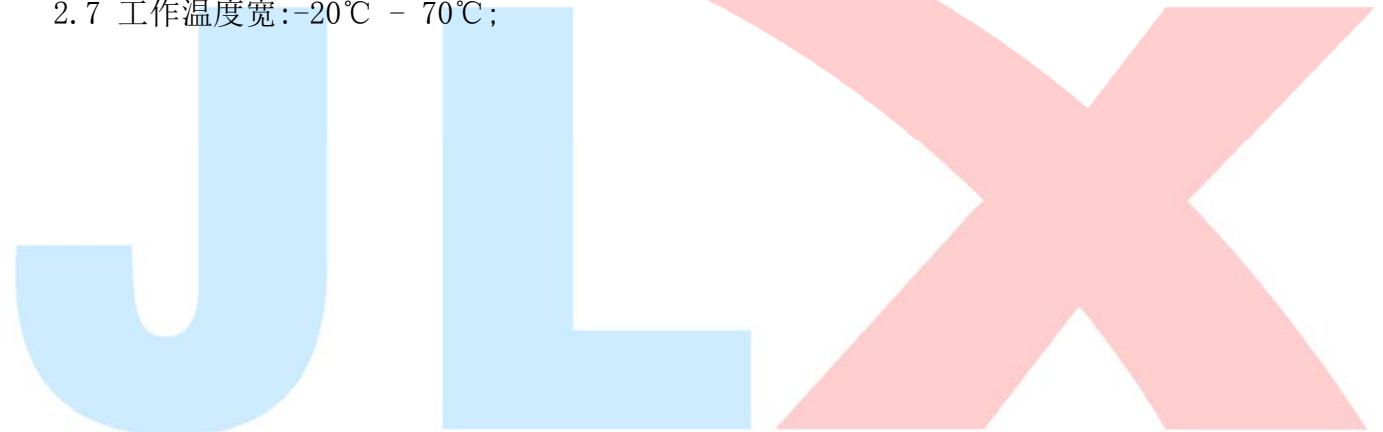
- 可显示 16 个*2 行 8*16 点阵的英文、数字、符号。

- 可显示 21 个*4 行 5*8 点阵的英文、数字、符号。

2.5 指令功能强:可软件调对比度、正显/反显转换、行列扫描方向可改（可旋转 180 度使用）。

2.6 接口简单方便：可采用 4 线 SPI 串口，或选择并口（6800 时序和 8080 时序可选）。

2.7 工作温度宽:-20℃ - 70℃；



3. 外形尺寸及接口引脚功能

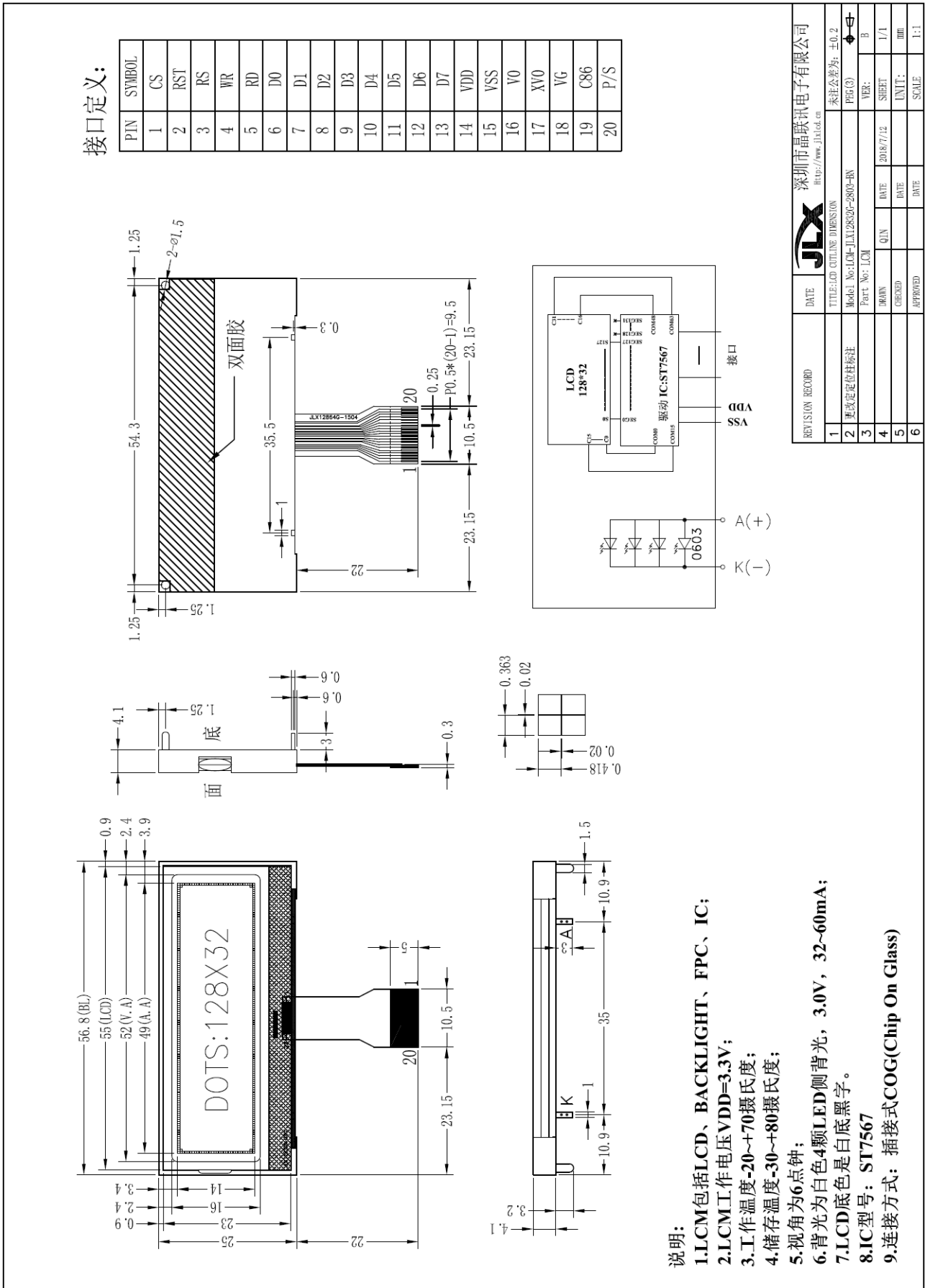


图 1. 外形尺寸

模块的接口引脚功能

引线号	符号	名称	功能
1	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	RS (A0)	寄存器选择信号	H:数据寄存器 0:指令寄存器
4	R/W (/WR)	6800 时序:读/写 8080 时序: 写	并行接口时并且选择 6800 时序时: H:读数据 L:写数据 并行接口时并且选择 8080 时序时: 写数据, 低电平有效. 串行接口时: 接 VDD 或悬空
5	E (/RD)	6800 时序:使能 8080 时序: 读	并行接口时并且选择 6800 时序时: 使能信号, 高电平有效. 并行接口时并且选择 8080 时序时: 读数据, 低电平有效. 串行接口时: 接 VDD 或悬空
6-11	D0-D5	I/O	数据总线 DB0-DB5 串行接口时: 空脚
12	D6 (SCLK)	I/O	并行接口时: 数据总线 DB6 串行接口时: 串行时钟 (SCLK)
13	D7 (SDA)	I/O	并行接口时: 数据总线 DB7 串行接口时: 串行数据 (SDA)
14	VDD	供电电源正极	供电电源正极
15	VSS	接地	0V
16	V0	倍压电路	
17	XV0	倍压电路	
18	VG	偏置电压	
19	C86	选择 6800 或 8080	并行接口时: H:6800 系统, L:8080 系统。 串行接口时: 接 VDD
20	P/S	选串并控制接口	接 VDD:选择并行接口, 接 VSS:选择串行接口

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 128×32 点阵, 128 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)。

4.2 背光参数

该型号液晶模块带 LED 背光源。它的性能参数如下:

背光板白色。

正常工作电流为: 32~80mA (LED 灯数共 4 颗);

工作电压: 3.0V;

4.2 工作电路：

图 2 是 JLX12832G-28003-BN 图像点阵型模块的电路框图，它由驱动 IC ST7567 及几个电阻电容成。

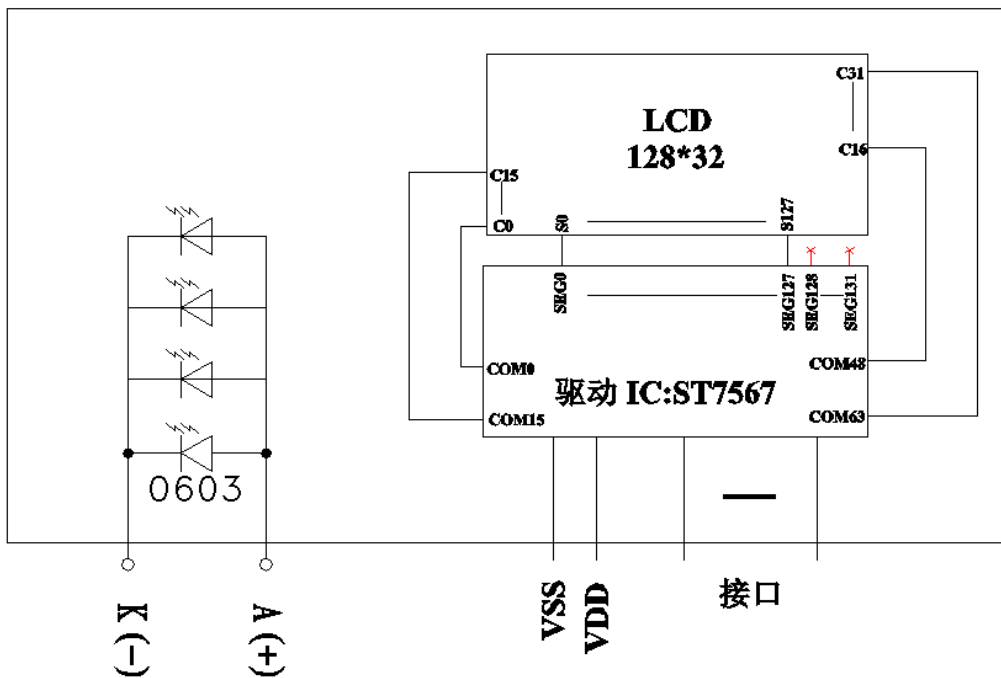


图 2: JLX12832G-280 图像点阵型液晶模块的电路框图

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		3.6	V
LCD 驱动电压	V0、XV0	-0.3		13.5	V
静电电压		-	-	100	V
工作温度		-20		+70	°C
储存温度		-30		+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	VDD		2.4	3.3	3.6	V
背光工作电压	VLED		2.9	3.0	3.1	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} = -0.5mA	0.8xVDD	-	VDD	V
输出低电平	V _{OHC}	I _{OL} = -0.5mA	VSS	-	0.2xVDD	V
模块工作电流	I _{DD}	VDD = 3.3V	-		0.3	mA
背光工作电流	I _{LED}	V _{LED} =3.0V	32	60	80	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 ST7567 (Writing Data from CPU to ST7567)

System Bus Timing for 4-Line Serial Interface

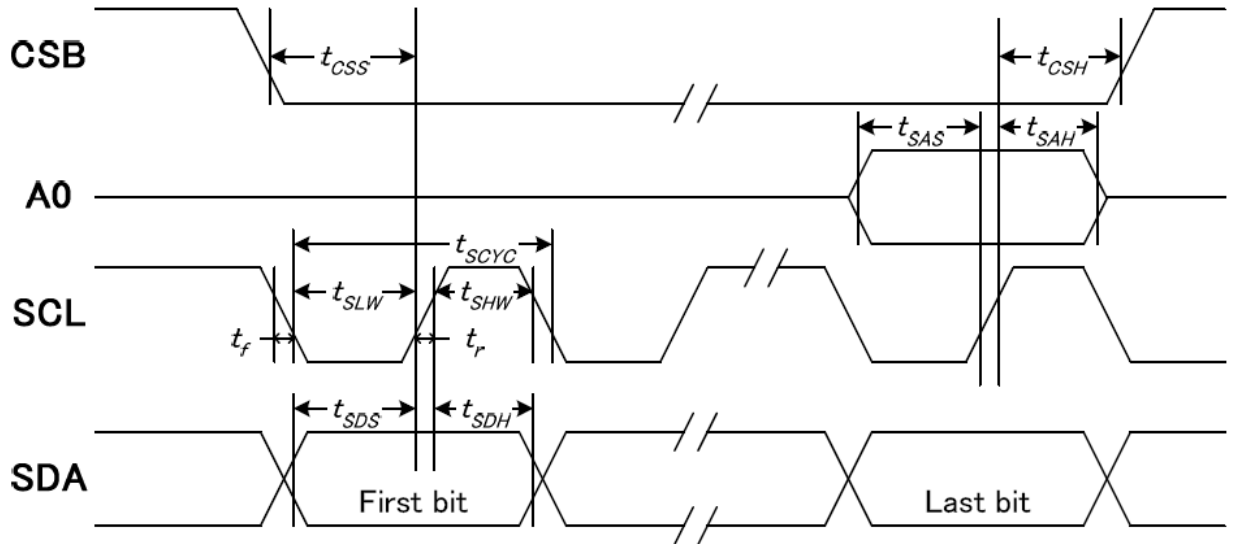


图 4. 从 CPU 写到 ST7567 (Writing Data from CPU to ST7567)

6.2 串行接口：时序要求 (AC 参数):

写数据到 ST7567 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)	T _{scyc}	引脚: SCK	50	—	—	T _{scyc}
保持SCK高电平脉宽 (SCK "H" pulse width)	T _{shw}	引脚: SCK	25	—	—	T _{shw}
保持SCK低电平脉宽 (SCK "L" pulse width)	T _{slw}	引脚: SCK	25	—	—	T _{slw}
地址建立时间 (Address setup time)	T _{sas}	引脚: RS	20	—	—	T _{sas}
地址保持时间 (Address hold time)	T _{sah}	引脚: RS	10	—	—	T _{sah}
数据建立时间 (Data setup time)	T _{sds}	引脚: SI	20	—	—	T _{sds}
数据保持时间 (Data hold time)	T _{sdh}	引脚: SI	10	—	—	T _{sdh}

片选信号建立时间 (CS-SCL time)	T_{css}	引脚: CS	20	—	—	T_{css}
片选信号保持时间 (CS-SCL time)	T_{csh}	引脚: CS	40	—	—	T_{csh}

VDD = 3.3V, Ta = 25°C

6.3 并行接口:

从 CPU 写到 ST7567 (Writing Data from CPU to ST7567)

System Bus Timing for 6800 Series MPU

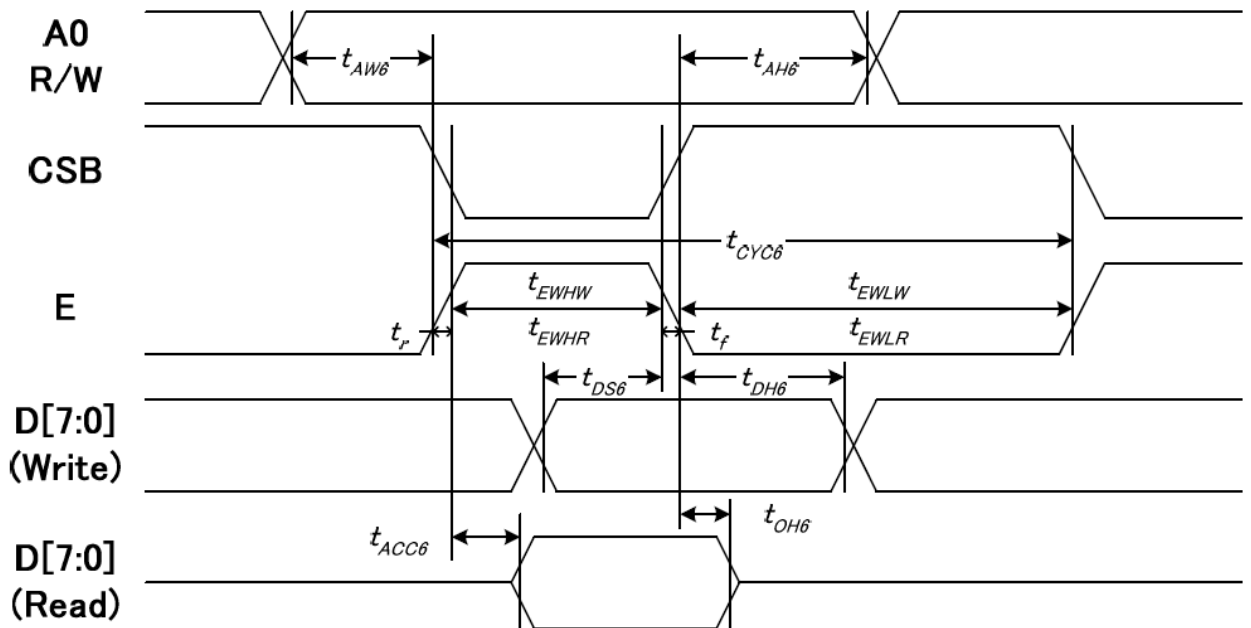


图 5. 从 CPU 写到 ST7567 (Writing Data from CPU to ST7567)

System Bus Timing for 8080 Series MPU

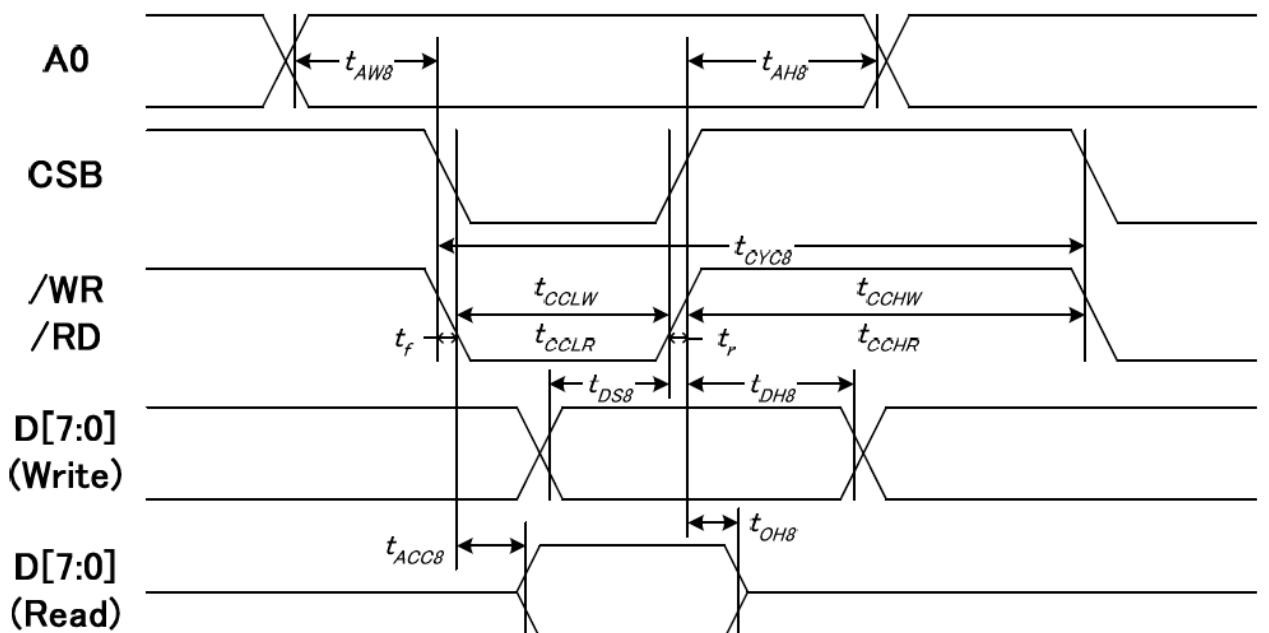


图 6. 从 CPU 写到 ST7567 (Writing Data from CPU to ST7567)

6.4 并行接口: 时序要求 (AC 参数):

写数据到 ST7567 的时序要求: (8080 系列 MPU)

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH6	10	—	—	ns
地址建立时间		tAW6	0	—	—	
系统循环时间		tCYC6	240	—	—	
使能“低”脉冲(写)	WR	tEWLW	80	—	—	
使能“高”脉冲(写)		tEWHW	80	—	—	
使能“低”脉冲(读)	RD	tEWLR	80	—	—	
使能“高”脉冲(读)		tEWHR	140	—	—	
写数据建立时间	D7-D0	tDS6	40	—	—	
写数据保持时间		tDH6	10	—	—	
读时间		tACC6	—	—	70	
读输出允许时间		tOH6	5	—	50	

VDD=3.3V, Ta=25°C

写数据到 ST7567 的时序要求: (6800 系列 MPU)

项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH8	0	—	—	ns
地址建立时间		tAW8	10	—	—	
系统循环时间		tCYC8	240	—	—	
使能“低”脉冲(写)	WR	tCCLW	80	—	—	
使能“高”脉冲(写)		tCCHW	80	—	—	
使能“低”脉冲(读)	RD	tCCLR	140	—	—	
使能“高”脉冲(读)		tCCHR	80	—	—	
写数据建立时间	D7-D0	tDS8	40	—	—	
写数据保持时间		tDH8	20	—	—	
读时间		tACC8	—	—	70	
读输出允许时间		tOH8	5	—	50	

VDD=3.3V, Ta=25°C

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

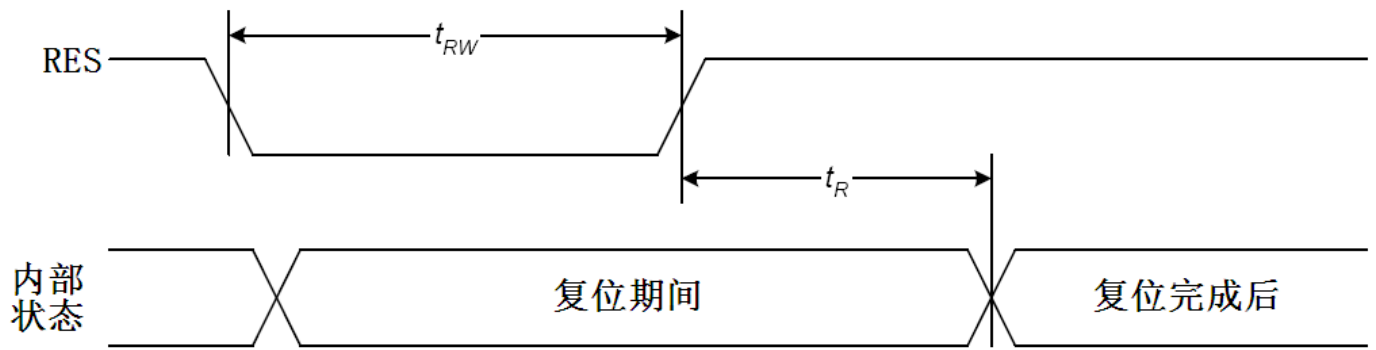
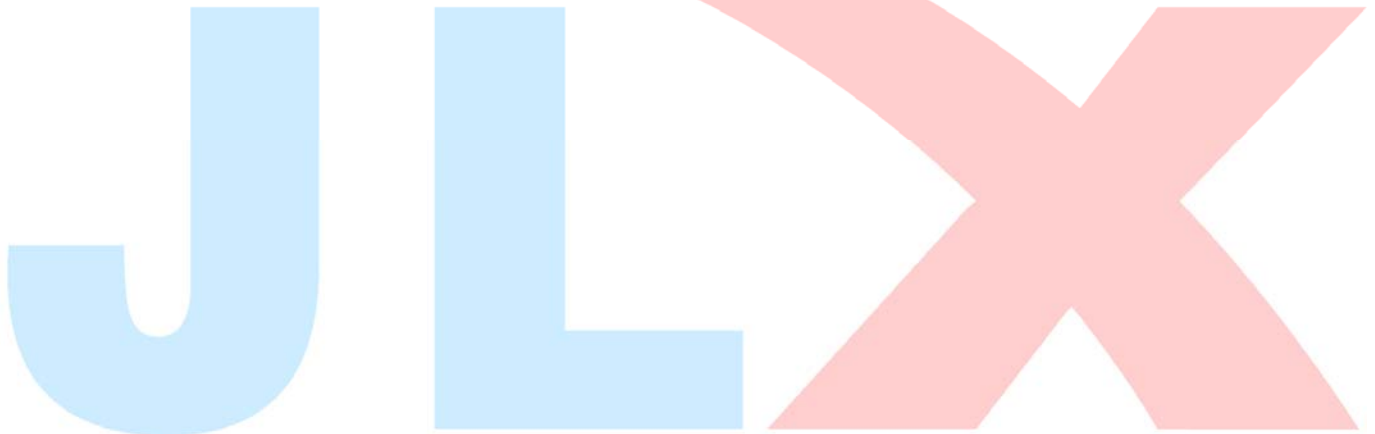


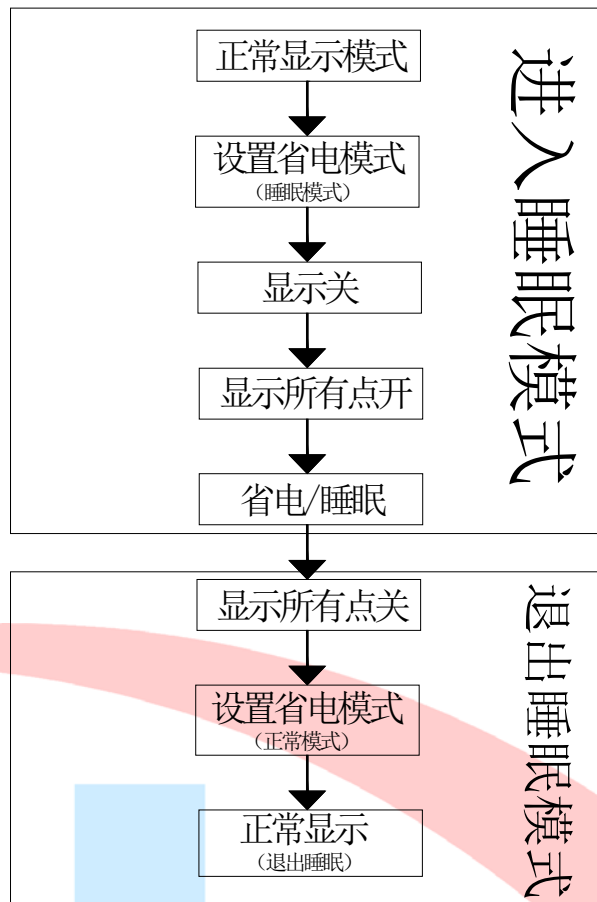
图 7：电源启动后复位的时序

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	1.0	—	—	us



6.6 省电模式设置



7. 指令功能：
7.1 指令表

指令表

表 8.

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						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	显示列地址增减： 1 0xA0 : 常规：列地址从左到右， 0xA1 : 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： 1 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	显示全部点阵： 1 0xA4 : 常规 0xA5 : 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： 1 0xA2 : BIAS=1/9 (常用) 0xA3 : BIAS=1/7
(12) 软件复位 (Reset)	0	1	1	1	0	0	0	1	0	0XE2 :软件复位。
(13) 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： 0XC0 :普通扫描顺序：从上到下 0XC8 :反转扫描顺序：从下到上

(14) 电源控制 (Power control set)		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 0x2C,0x2E,0x2F 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 0x2F ，一次性打开三部分电路。		
(15) 选择内部电阻比例	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： 0x20~0x27 ，数值越大对比度越浓，越小越淡		
(16)	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 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： 0x00~0x3F ，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)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： 1 0xAC : 关， 0xAD : 开。 此指令在进入及退出睡眠模式时起作用
(18) 升压倍数选择 (Booster ratio set)	0	1	1	1	1	1	0	0	0	选择升压倍数： 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
			0	0	0	0	0	0		2 位数设置 升压倍数

(19) 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 规格书第 47 页“POWER SAVE”
(20)空指令 (NOP)	0	1	1	1	0	0	0	1	1	空操作
(21) 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“ST7567_V1.7.PDF”的第 21~28 页。

7.3 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 128*32 点阵的屏分为 4 个“页”，从第 0“页”到第 3“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：

D0	0	1	1	1		0
D1	1	0	0	0		0
D2	0	0	0	0		0
D3	0	1	1	1		0
D4	1	0	0	0		0
-						

Display data RAM
(显示数据存储器)

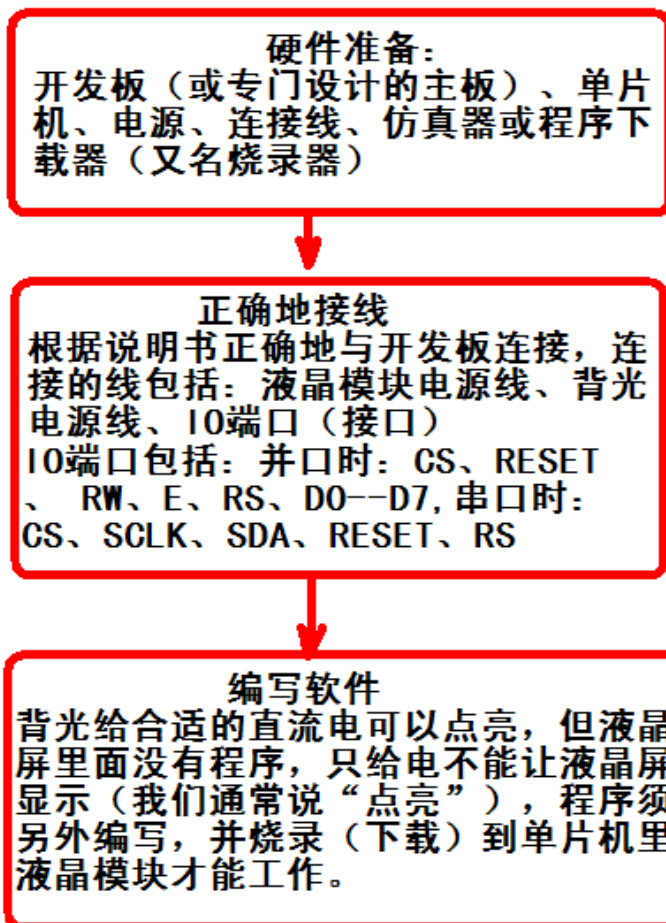
COM0						
COM1						
COM2						
COM3						
COM4						
-						

Liquid crystal display
(液晶屏)

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

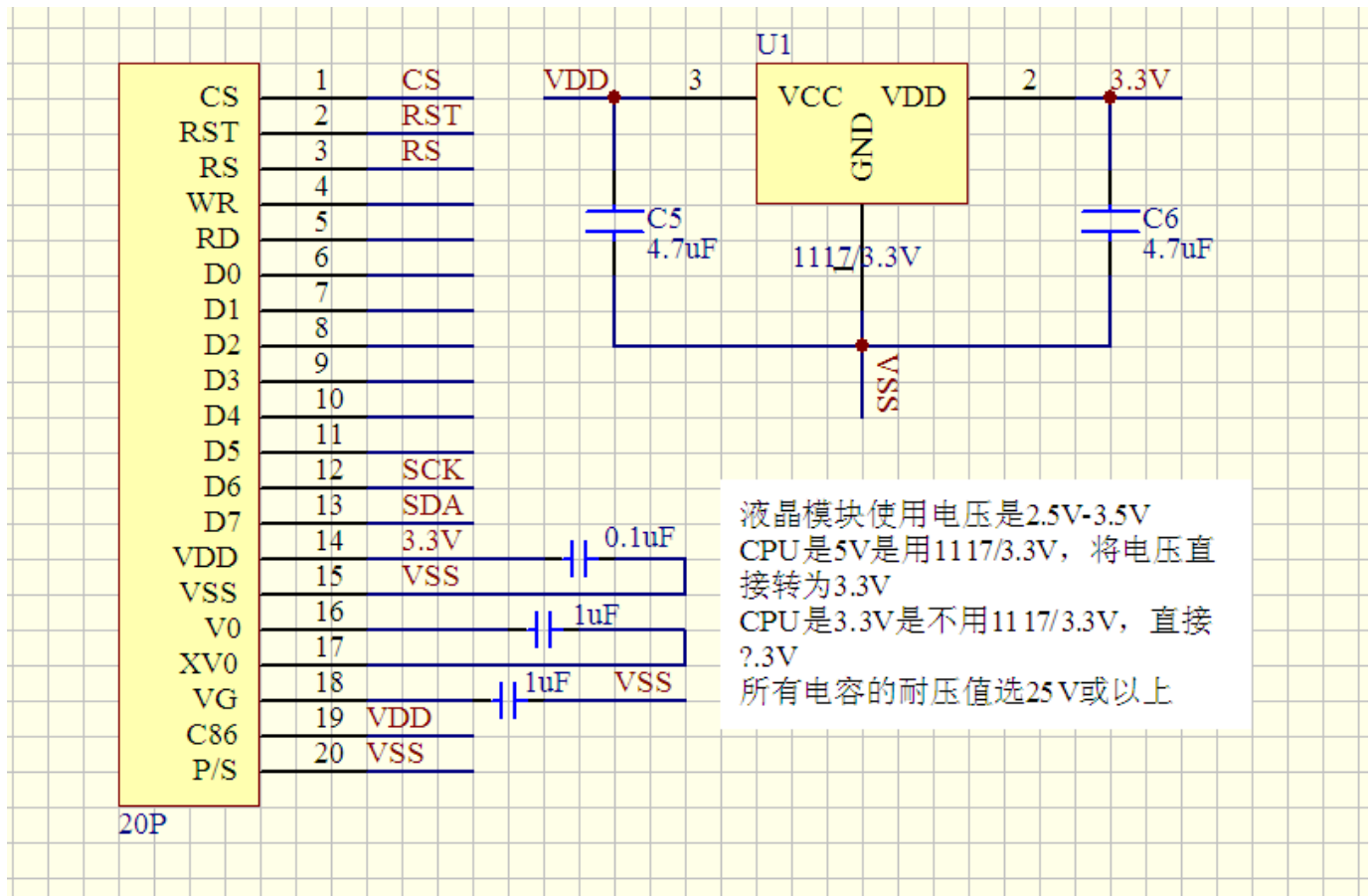


7.5 程序举例:

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



串行程序：

```

/* 针对液晶模块型号：JLX12832G-28003
   串行接口，
   驱动 IC 是:ST7567(or compatible)
   晶联讯电子：网址 http://www.jlxlcd.cn;
*/
#include <reg52.H>
#include <intrins.h>

sbit rs=P3^0; /*接口定义:lcd_rs 就是 LCD 的 rs*/
sbit sclk=P3^1; /*接口定义:lcd_sclk 就是 LCD 的 sclk*/
sbit sid=P3^2; /*接口定义:lcd_sid 就是 LCD 的 sid*/
sbit reset=P1^0; /*接口定义:lcd_reset 就是 LCD 的 reset*/
sbit cs1=P1^1; /*接口定义:lcd_cs1 就是 LCD 的 cs1*/
sbit key=P2^0; /*按键接口，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 bmp_128x16[];
uchar code zhuang[];
uchar code tai[];
uchar code shi[];
uchar code yong[];
    
```

```
uchar code mon[];
```

```
void transfer_command_lcd(int data1)
{
    char i;
    cs1=0;
    rs=0;
    for(i=0;i<8;i++)
    {
        sclk=0;
        delay_us(1);
        if(data1&0x80) sid=1;
        else sid=0;
        sclk=1;
        delay_us(1);
        data1=data1<<=1;
    }
    cs1=1;
}
```

```
/*写数据到LCD模块*/
```

```
void transfer_data_lcd(int data1)
{
    char i;
    cs1=0;
    rs=1;
    for(i=0;i<8;i++)
    {
        sclk=0;
        if(data1&0x80) sid=1;
        else sid=0;
        sclk=1;
        data1=data1<<=1;
    }
    cs1=1;
}
```

```
void clear_screen()
{
    uchar i, j;
    for(j=0;j<4;j++)
    {
        transfer_command_lcd(0xb0+j);
        transfer_command_lcd(0x10);
        transfer_command_lcd(0x00);
        for(i=0;i<132;i++)
        {
            transfer_data_lcd(0x00);
        }
    }
}
```

```
//长延时
```

```
void delay(int i)
{
    uint j, k;
```



```
    for(j=0;j<i;j++)
    for(k=0;k<500;k++);
}

//短延时
void delay_us(int i)
{
    uint j,k;
    for(j=0;j<i;j++)
    for(k=0;k<10;k++);
}

void waitkey()
{
    repeat:
        if(P2&0x01) goto repeat;
        else delay(60);
        if(P2&0x01) goto repeat;
        else delay(400);
}

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

void display_test(uchar data_left,uchar data_right)
{
    int i,j;
    for(j=0;j<4;j++)
    {
        lcd_address(j,0);
        for(i=0;i<128;i++)
        {
            transfer_data_lcd(data_left);
            transfer_data_lcd(data_right);
        }
    }
}

void display_graphic_128x32(uint page,uint column,uchar *dp)
{
    uint i,j;
    for(j=0;j<4;j++)
    {
        lcd_address(page+j,column);
        for(i=0;i<128;i++)
        {
            transfer_data_lcd(*dp);
            dp++;
        }
    }
}
```

```

void display_graphic_128x16(uint page,uint 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_lcd(*dp);
            dp++;
        }
    }
}

```

```

void display_graphic_8x16(uint page,uint column,uint reverse,uchar *dp)
{
    uint i,j;
    for(j=0;j<2;j++)
    {
        lcd_address(page+j, column);
        for(i=0;i<8;i++)
        {
            if(reverse==1) transfer_data_lcd(~*dp);
            else transfer_data_lcd(*dp);
            dp++;
        }
    }
}

```

```

void display_graphic_16x16(uint page,uint column,uint reverse,uchar *dp)
{
    uint i,j;
    for(j=0;j<2;j++)
    {
        lcd_address(page+j, column);
        for(i=0;i<16;i++)
        {
            if(reverse==1) transfer_data_lcd(~*dp);
            else transfer_data_lcd(*dp);
            dp++;
        }
    }
}

```

```

//=====initinal=====
void initial_lcd()
{
    reset=0;           //Reset the chip when reset=0
    delay(20);
    reset=1;
    transfer_command_lcd(0xe2);    /*软复位*/
    delay(20);
    transfer_command_lcd(0x2c);    /*升压步聚 1*/
    delay(5);
    transfer_command_lcd(0x2e);    /*升压步聚 2*/
    delay(5);
}

```

```

transfer_command_lcd(0x2f);    /*升压步聚 3*/
delay(5);
transfer_command_lcd(0x23);    /*粗调对比度，可设置范围 20~27*/
transfer_command_lcd(0x81);    /*微调对比度*/
transfer_command_lcd(0x10);    /*微调对比度的值，可设置范围 0x00~0x3f*/
transfer_command_lcd(0xa2);    /*1/9 偏压比 (bias) */
transfer_command_lcd(0xc8);    /*行扫描顺序：从上到下*/
transfer_command_lcd(0xa0);    /*列扫描顺序：从左到右*/
transfer_command_lcd(0x40);    /*起始行：从第一行开始*/
transfer_command_lcd(0xaf);    /*开显示*/
}

```

```

void main(void)
{
    initial_lcd();
    while(1)
    {
        clear_screen();
        display_graphic_128x32(1, 1, bmp1);
        waitkey();
        clear_screen();
        display_graphic_128x32(1, 1, bmp2);
        waitkey();
        clear_screen();
        display_graphic_16x16(1, 16*0, 1, zhuang);
        display_graphic_16x16(1, 16*1, 1, tai);
        display_graphic_8x16(1, 32, 1, mon);
        display_graphic_16x16(1, 40, 0, shi);
        display_graphic_16x16(1, 56, 0, yong);
        display_graphic_128x16(3, 1, bmp_128x16);
        waitkey();
        clear_screen();
        display_test(0xff, 0xff);
        waitkey();
        clear_screen();
        display_test(0x55, 0xaa);
        waitkey();
        clear_screen();
        display_test(0xaa, 0x55);
        waitkey();
    }
}

```

```

uchar code zhuang[]={
/*-- 文字： 状 --*/
/*-- 宋体 12； 此字体下对应的点阵为： 宽 x 高=16x16 --*/
0x08, 0x30, 0x00, 0xFF, 0x20, 0x20, 0x20, 0x20, 0xFF, 0x20, 0xE1, 0x26, 0x2C, 0x20, 0x20, 0x00,
0x04, 0x02, 0x01, 0xFF, 0x40, 0x20, 0x18, 0x07, 0x00, 0x00, 0x03, 0x0C, 0x30, 0x60, 0x20, 0x00};

```

```

uchar code tai[]={
/*-- 文字： 态 --*/
/*-- 宋体 12； 此字体下对应的点阵为： 宽 x 高=16x16 --*/
0x00, 0x04, 0x04, 0x04, 0x84, 0x44, 0x34, 0x4F, 0x94, 0x24, 0x44, 0x84, 0x84, 0x04, 0x00, 0x00,
0x00, 0x60, 0x39, 0x01, 0x00, 0x3C, 0x40, 0x42, 0x4C, 0x40, 0x40, 0x70, 0x04, 0x09, 0x31, 0x00};

```

```

uchar code shi[]={
/*-- 文字： 使 --*/

```

```

/*-- 宋体 12; 此字体下对应的点阵为：宽 x 高=16x16  --*/
0x40, 0x20, 0xF0, 0x1C, 0x07, 0xF2, 0x94, 0x94, 0x94, 0xFF, 0x94, 0x94, 0x94, 0xF4, 0x04, 0x00,
0x00, 0x00, 0x7F, 0x00, 0x40, 0x41, 0x22, 0x14, 0x0C, 0x13, 0x10, 0x30, 0x20, 0x61, 0x20, 0x00};

uchar code yong[]={
/*-- 文字： 用  --*/
/*-- 宋体 12; 此字体下对应的点阵为：宽 x 高=16x16  --*/
0x00, 0x00, 0x00, 0xFE, 0x22, 0x22, 0x22, 0x22, 0xFE, 0x22, 0x22, 0x22, 0x22, 0xFE, 0x00, 0x00,
0x80, 0x40, 0x30, 0x0F, 0x02, 0x02, 0x02, 0x02, 0xFF, 0x02, 0x02, 0x42, 0x82, 0x7F, 0x00, 0x00};

uchar code mon[]={
/*-- 文字：  :  --*/
/*-- 宋体 12; 此字体下对应的点阵为：宽 x 高=8x16  --*/
0x00, 0x00, 0x00, 0xC0, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x30, 0x30, 0x00, 0x00, 0x00,
};

uchar code bmp_128x16[]={
/*-- 调入了一幅图像： E:\work\图片收藏夹\黑白屏图片\12832 日期. bmp  --*/
/*-- 宽度 x 高度=128x16  --*/
0x00, 0xE0, 0x10, 0x10, 0x10, 0x10, 0xE0, 0x00, 0x00, 0xC0, 0x20, 0x10, 0x10, 0x20, 0xC0, 0x00,
0x00, 0x20, 0x20, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x40, 0x20, 0xF0, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x20, 0xF0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x20, 0xF0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x70, 0x10, 0x10, 0x90, 0x70, 0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x7C, 0x54, 0x54, 0x54, 0xD4, 0x54, 0x54, 0x54, 0x54, 0x7C, 0x00, 0x00, 0x00,
0x00, 0x08, 0xFE, 0xA8, 0xA8, 0xA8, 0xFE, 0x08, 0x00, 0xFC, 0x44, 0x44, 0x44, 0xFC, 0x00, 0x00,
0x00, 0x04, 0x04, 0x04, 0x04, 0x04, 0xFC, 0x04, 0x04, 0x04, 0x84, 0x04, 0x04, 0x00, 0x00, 0x00,
0x00, 0x60, 0x50, 0x48, 0x44, 0x43, 0x60, 0x00, 0x00, 0x1F, 0x20, 0x40, 0x40, 0x20, 0x1F, 0x00,
0x00, 0x40, 0x40, 0x7F, 0x40, 0x40, 0x00, 0x00, 0x00, 0x0E, 0x09, 0x48, 0x48, 0x7F, 0x48, 0x00,
0x00, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x00, 0x40, 0x40, 0x7F, 0x40, 0x40, 0x00, 0x00,
0x00, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x00, 0x40, 0x40, 0x7F, 0x40, 0x40, 0x00, 0x00,
0x00, 0x00, 0x00, 0x7E, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x90, 0x8C, 0x83, 0x92, 0x92, 0x92, 0xFF, 0x92, 0x92, 0x92, 0x92, 0x92, 0x82, 0x80, 0x00,
0x84, 0x44, 0x37, 0x04, 0x04, 0x14, 0x67, 0xC4, 0x30, 0x0F, 0x04, 0x44, 0x84, 0x7F, 0x00, 0x00,
0x40, 0x40, 0x41, 0x41, 0x41, 0x7F, 0x41, 0x41, 0x41, 0x41, 0x7F, 0x41, 0x40, 0x60, 0x40, 0x00,
};

uchar code bmp1[]={
/*-- 调入了一幅图像： E:\显示图案收藏\复件 12832-大熊猫. bmp  --*/
/*-- 宽度 x 高度=128x32  --*/
0x00, 0x00, 0x00, 0x00, 0x7E, 0x2A, 0x2A, 0x2A, 0x2A, 0x2A, 0x7E, 0x00, 0x00, 0x00, 0x00,
0x02, 0xFE, 0x92, 0x92, 0x92, 0xFE, 0x12, 0x11, 0x12, 0x1C, 0xF0, 0x18, 0x17, 0x12, 0x10, 0x00,
0x20, 0x21, 0x2E, 0xE4, 0x00, 0x42, 0x42, 0xFE, 0x42, 0x42, 0x42, 0x02, 0xFE, 0x00, 0x00, 0x00,
0x10, 0x61, 0x06, 0xE0, 0x18, 0x84, 0xE4, 0x1C, 0x84, 0x65, 0xBE, 0x24, 0xA4, 0x64, 0x04, 0x00,
0x00, 0x00, 0x00, 0x00, 0x7E, 0x2A, 0x2A, 0x2A, 0x2A, 0x2A, 0x7E, 0x00, 0x00, 0x00, 0x00,
0x10, 0xD0, 0xFF, 0x50, 0x90, 0x04, 0xF4, 0x54, 0x5F, 0x54, 0x54, 0x5F, 0xF4, 0x04, 0x00, 0x00,
0x10, 0x10, 0xFF, 0x10, 0x10, 0x00, 0x08, 0x08, 0xFF, 0x08, 0x08, 0x08, 0xF8, 0x00, 0x00, 0x00,
0x00, 0x00, 0xFE, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02,
0x00, 0x7F, 0x25, 0x25, 0x25, 0x25, 0x7F, 0x00, 0x00, 0x7F, 0x25, 0x25, 0x25, 0x25, 0x7F, 0x00,
0x08, 0x1F, 0x08, 0x08, 0x04, 0xFF, 0x05, 0x81, 0x41, 0x31, 0x0F, 0x11, 0x21, 0xC1, 0x41, 0x00,
0x00, 0x00, 0x00, 0x7F, 0x20, 0x10, 0x00, 0x7F, 0x00, 0x00, 0x00, 0x00, 0x3F, 0x40, 0x38, 0x00,
0x04, 0x04, 0xFF, 0x00, 0x01, 0x00, 0xFF, 0x41, 0x21, 0x12, 0x0C, 0x1B, 0x61, 0xC0, 0x40, 0x00,
0x00, 0x7F, 0x25, 0x25, 0x25, 0x25, 0x7F, 0x00, 0x00, 0x7F, 0x25, 0x25, 0x25, 0x25, 0x7F, 0x00,
0x03, 0x00, 0xFF, 0x00, 0x00, 0x84, 0x85, 0x45, 0x35, 0x0F, 0x15, 0x25, 0x65, 0xC4, 0x44, 0x00,
0x08, 0x18, 0x0F, 0x04, 0x85, 0x41, 0x31, 0x0D, 0x03, 0x05, 0x09, 0x11, 0x31, 0x61, 0x21, 0x00,
0x40, 0x30, 0x0F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x08, 0xF8, 0x08, 0x08, 0x08, 0x08,
0x08, 0xF8, 0x08, 0x00, 0x00, 0x00, 0x00, 0x08, 0x18, 0x68, 0x80, 0x80, 0x68, 0x18, 0x08,
0x00, 0x10, 0x10, 0xF8, 0x00, 0x00, 0x00, 0x00, 0x70, 0x08, 0x08, 0x08, 0x88, 0x70, 0x00,
0x00, 0x70, 0x88, 0x08, 0x08, 0x88, 0x70, 0x00, 0x00, 0x30, 0x08, 0x88, 0x88, 0x48, 0x30, 0x00,
0x00, 0x70, 0x08, 0x08, 0x08, 0x88, 0x70, 0x00, 0xC0, 0x30, 0x08, 0x08, 0x08, 0x38, 0x00, 0x00,

```

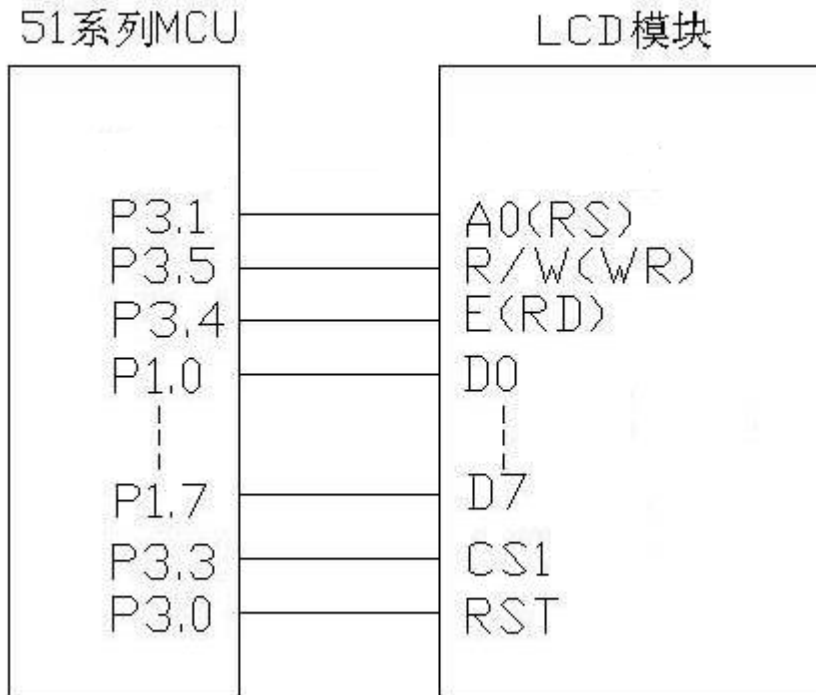
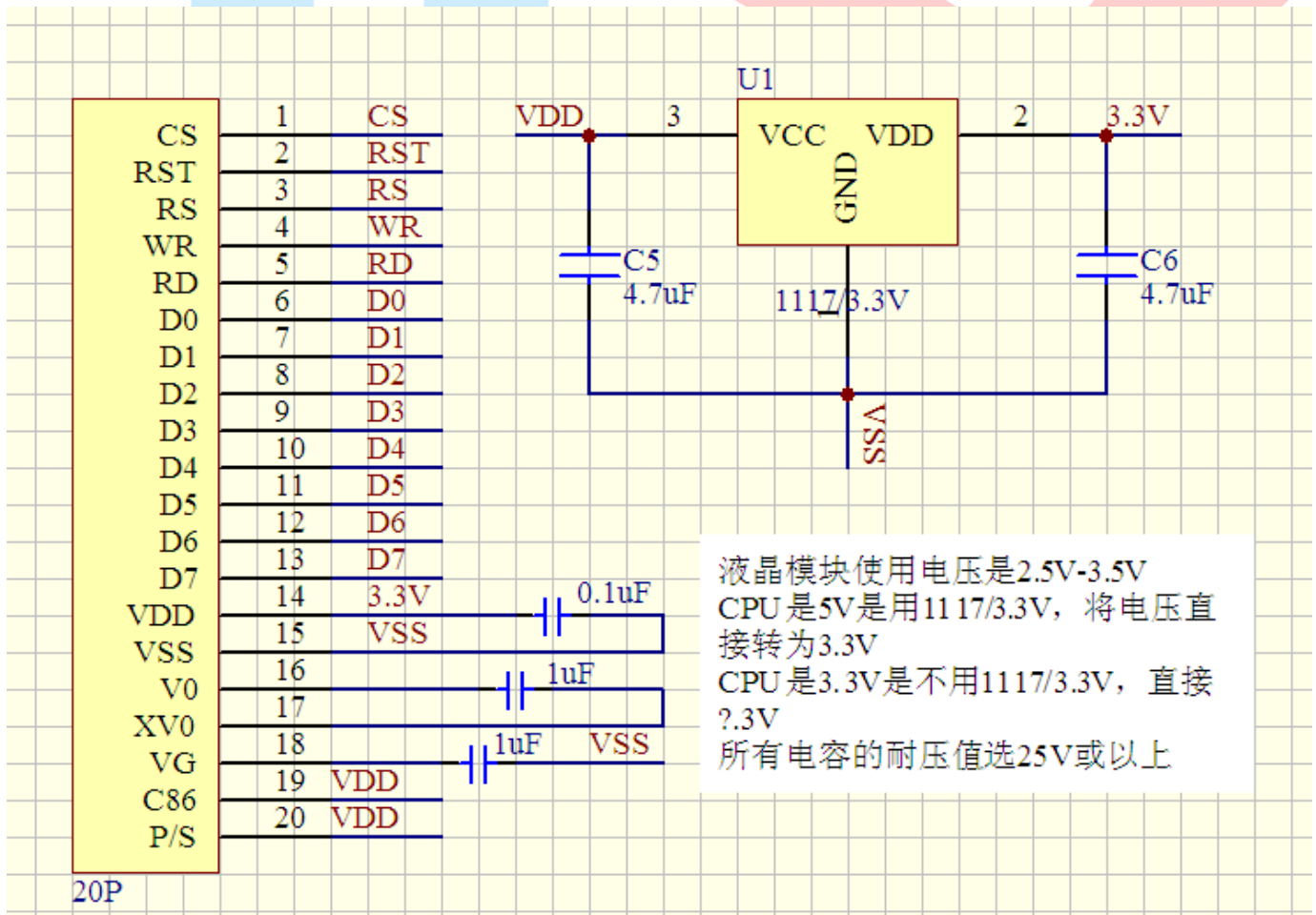



图 8. 并行接口



```
#include <reg52.h>
#include <intrins.h>

sbit cs1=P3^3;    /*3.4 接口定义*/
sbit reset=P3^0; /*3.3 接口定义*/
sbit rs=P3^1;    /*接口定义*/
sbit e=P3^4;     /*接口定义*/
sbit rw=P3^5;   /*接口定义。另外 P1.0~1.7 对应 DB0~DB7*/
sbit key=P2^0;  /*按键接口，P2.0 口与 GND 之间接一个按键*/

/*写指令到 LCD 模块*/
void transfer_command(int data1)
{
    lcd_cs1=0;
    lcd_rs=0;
    lcd_rd=0;
    lcd_wr=0;
    P1=data1;
    lcd_rd=1;
    lcd_cs1=1;
    lcd_rd=0;
}

//写指令到 LCD 模块
void transfer_command_lcd(int data1)
{
    cs1=0;
    rs=0;
    rw=0;
    P1=data1;
    e=1;
    delay_us(1);
    e=0;
    cs1=1;
    e=0;
}

//写数据到 LCD 模块
void transfer_data_lcd(int data1)
{
    cs1=0;
    rs=1;
    rw=0;
    P1=data1;
    e=1;
    delay_us(2);
    e=0;
    cs1=1;
    e=0;
}
```