

JLX12832C-1 使用说明书

目 录

序号	内 容 标 题	页 码
1	概述	2
2	字符型模块的特点	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	6~8
7	指令功能及硬件接口与编程案例	8~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 JLX12832C-1 型液晶模块由于使用方便、带中文字库、显示清晰，广泛应用于各种人机交流面板。

JLX12832C-1 液晶显示模块是 128×32 点阵的汉字图形型液晶显示模块，可显示汉字及图形，内置 8192 个中文汉字（16X16 点阵）、128 个字符（8X16 点阵）及 64X256 点阵显示 RAM（GDRAM）。可与 CPU 直接接口，提供两种界面来连接微处理器：8-位并行及串行两种连接方式。具有多种功能：光标显示、画面移位、睡眠模式等。

2. JLX12832C-1 图像型点阵液晶模块的特性

1.1 结构牢：带 PCB、背光、铁框

1.2 IC 采用矽创公司 ST7920, 功能强大，稳定性好

1.3 功耗低：10 - 100mW（不带背光 10mW, 带背光不大于 100mW）；

1.4 显示内容：

● 128*32 点阵单色图片；或 8 字*2 行汉字、字符、数字；半角字符、数字可显示 16 字*2 行。

● 内置 8192 个中文汉字（16X16 点阵）、128 个字符（8X16 点阵）及 64X256 点阵显示 RAM（GDRAM）。

1.5 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；

1.6 接口简单方便：采用 3 线 SPI 串行接口，可只需 3 位 MPU 的端口。也可选用 8 位并行接口。

1.7 工作温度宽：-10℃ - 60℃；

3. 外形尺寸及接口引脚功能

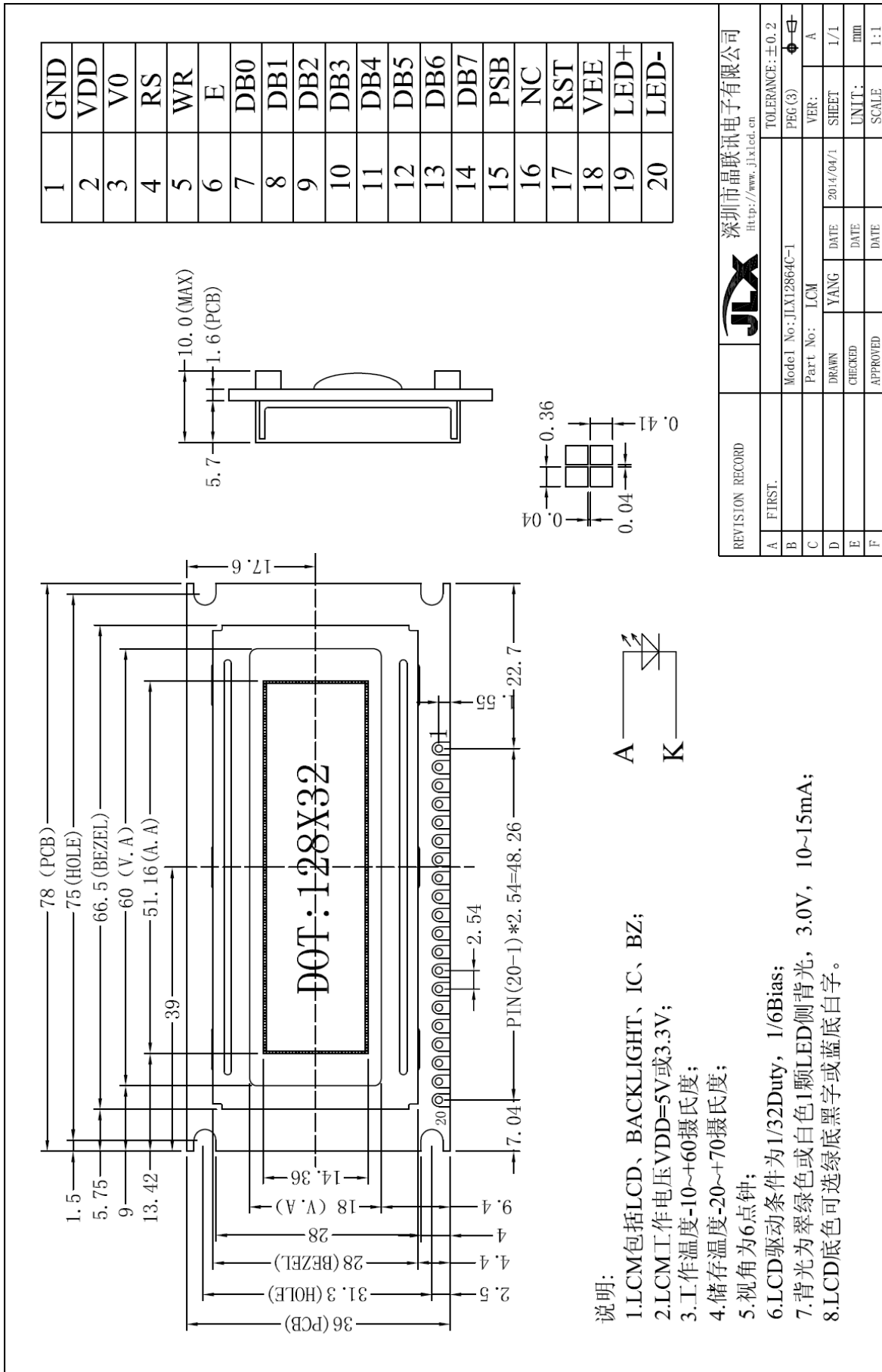


图 1. 外形尺寸

模块的接口引脚功能

引脚	符号	名称	功能
1	VSS	接地	0V
2	VDD	电路电源	5V, 或 3.3V 可选
3	V0	LCD V0 电压输入	5V 和 3.3V 时, 通过此脚对 LCD 驱动电压进行调整
4	RS(CS*)	寄存器选择信号(串行时为片选: CS)	1. 并行接口时: 1:数据寄存器 0:指令寄存器 2. 串行接口时: 片选信号, 高电平有效
5	R/W(SID*)	读写选择(串行时为串行数据: SID)	1. 并行接口时: 0: 写 1:读 2. 串行时为串行数据输入: SID
6	E(SCLK*)	读写使能信号(串行时为串行时钟: SCLK)	1. 并行接口时: 读写使能信号 2. 串行时为串行时钟: SCLK
7~14	D0~D7	数据 DB0~DB7	并行接口时: 数据总线 DB0~DB7 串行接口时: 无效, 空脚 4 位并行接口时, DB4~DB7 作为数据总线, DB0~DB3 不起作用
15	PSB	并行/串行选择	1:选择并行, 0:选择串行; 也可在 PCB 上 R9 焊零欧、R10 空是并行, R10 焊零欧、R9 空是串行, 达到选择并/串接口。
16	NC	空脚	空脚
17	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
18	VEE	升压输出	5V 时此引脚空, 3.3V 时通过此脚对 LCD 驱动电压进行调整
19	BLA	LED 背光正极	LED 背光正极, 5V 或 3.3V 可选
20	BLK	LED 背光负极	LED 背光负极, 0V

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.

4.2 工作电路:

图 1 是 JLX12832C-1 图像点阵型模块的电路框图。

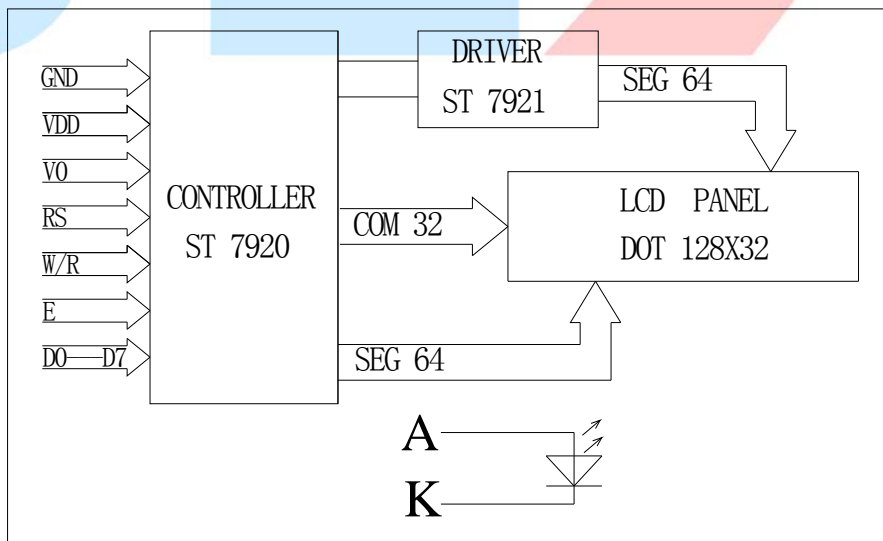


图 2: JLX12832C-1 图像点阵型液晶模块的电路框图

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： $-10\sim+60^{\circ}\text{C}$ ；

存储温度： $-20\sim+70^{\circ}\text{C}$ ；

背光板可显示绿色, 黄绿色, 兰色和白色。背光一般为绿色, 也可为客户设计为其他颜色, 但价格较绿色贵一点。

正常工作电流为： $10\sim 20\text{mA}$ （若 LED 灯数不止一颗, 则乘以相应数量）；

工作电压： 5.0V 或 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	
电路电源	VDD - VSS	-0.3		5.5	V
LCD 驱动电压	VDD - V0	-0.3		7	V
静电电压		-	-	100	V
工作温度		-10		+60	$^{\circ}\text{C}$
储存温度		-20		+70	$^{\circ}\text{C}$

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	5.0V 供电	4.0	5.0	5.2	V
输入高电平	VIH	-	2.2		VDD	V
输入低电平	VIO	-	-0.3		0.6	V
输出高电平	VOH	$I_{OH} = 0.2\text{mA}$	2.4		-	V
输出低电平	VOO	$I_{OO} = 1.2\text{mA}$	-		0.4	V
工作电流	IDD	VDD = 5.0V		2.0		mA
液晶驱动电压	VDD-V0	VDD = 5.0V	1	1.4	1.8	V
液晶驱动电压	VEE-V0	VDD =	1.5	2.0	2.2	V

表 3：直流（DC）参数

6. 读写时序特性

6.1 并行接口时:

从 CPU 写到 LCD 驱动 IC:ST7920 (Writing Data from CPU to ST7920)

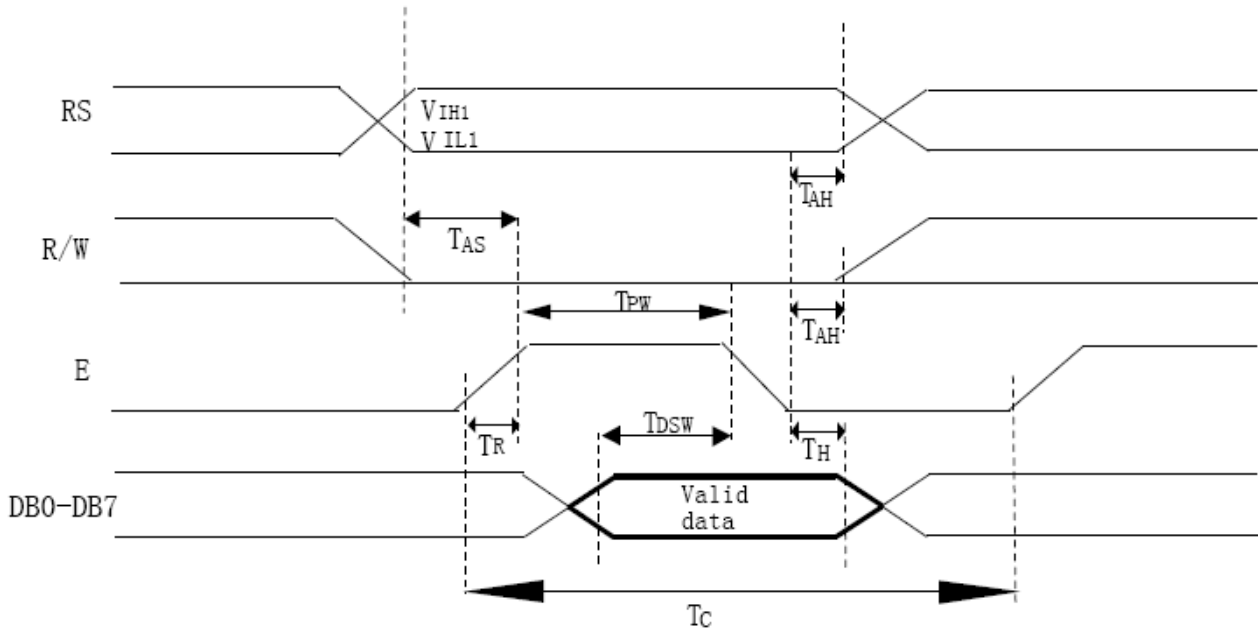
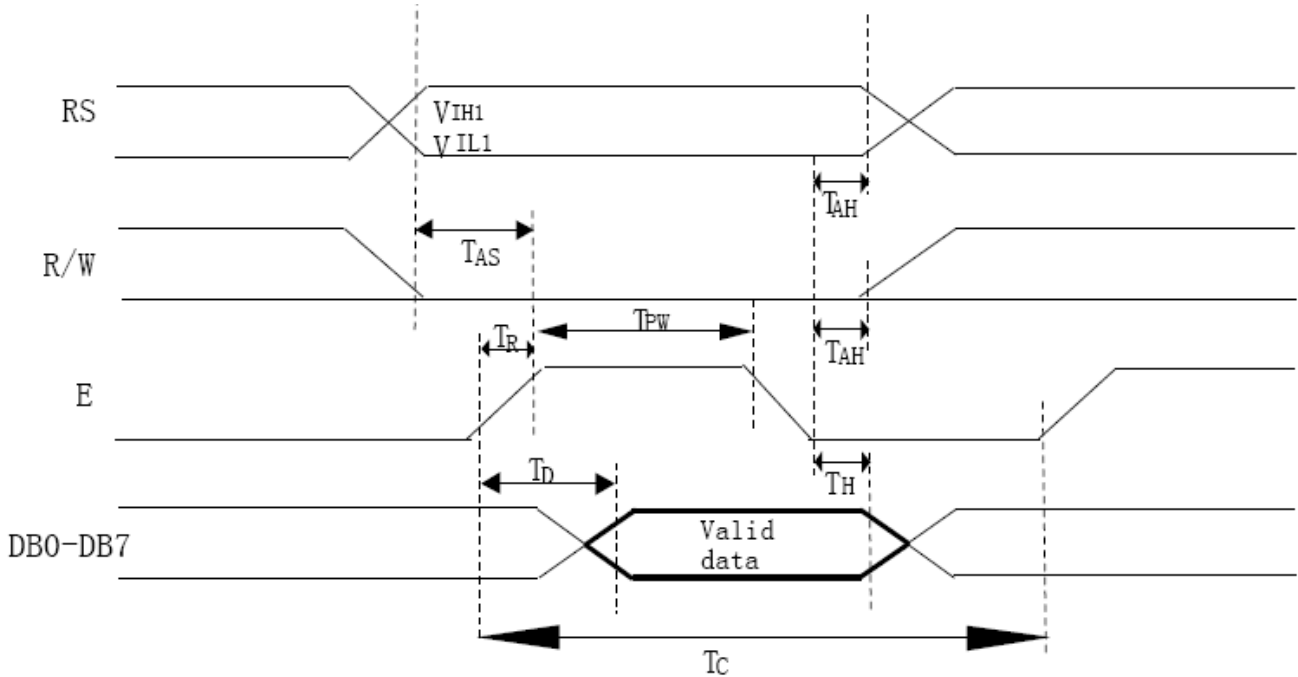


图 3. 从 CPU 写到 IC:ST7920 (Writing Data from CPU to ST7920)

E 周期	T_C	E	1200	-	-	ns
E 脉冲宽度	T_{PW}	E	140	-	-	ns
E 上升/下降时间	T_R, T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据建立时间	T_{DSW}	DB0~DB7	40	-	-	ns
数据保持时间	T_H	DB0~DB7	20	-	-	ns

从 LCD 驱动 IC:ST7920 读到 CPU



E 周期	T_C	E	1200	-	-	ns
E 脉冲宽度	T_{PW}	E	140	-	-	ns
E 上升/下降时间	T_R, T_F	E	-	-	25	ns
地址建立时间	T_{AS}	RS、RW、E	10	-	-	ns
地址保持时间	T_{AH}	RS、RW、E	20	-	-	ns
数据延迟时间	T_{DDR}	DB0~DB7	-	-	100	ns
数据保持时间	T_H	DB0~DB7	20	-	-	ns

6.2 串行接口时:

从 CPU 写到 LCD 驱动 IC:ST7920

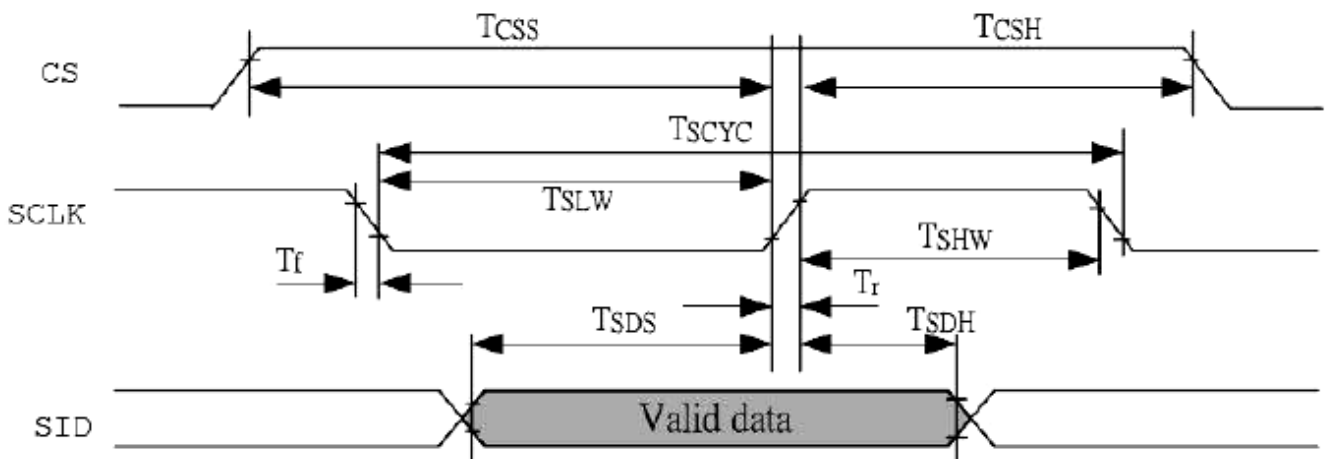


图: 从 CPU 写到 IC:ST7920 (Writing Data from CPU to ST7920)

上升/下降时间	T_R, T_F	-	-	-	0.2	μs
串行时钟周期	T_{SCYC}	E	600	-	-	ns
SCLK 高脉宽	T_{SHW}	E	300	-	-	ns
SCLK 低脉宽	T_{SLW}	E	300	-	-	ns
SID 数据建立时间	T_{SDS}	RW	40	-	-	ns
SID 数据保持时间	T_{SDH}	RW	40	-	-	ns
CS 建立时间	T_{CSS}	RS	60	-	-	ns
CS 保持时间	T_{CSH}	RS	60	-	-	ns

7. 指令功能：

7.1 指令表

● 指令表 1 (RE=0: 基本指令集)

指令	指令码										说明	执行时间 (540KHZ)
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
清除显示	0	0	0	0	0	0	0	0	0	1	将 DDRAM 填满“20H”，并且设定 DDRAM 的地址计数器 (AC) 到“00H”	1.6ms
地址归零	0	0	0	0	0	0	0	0	1	X	设定 DDRAM 的地址计数器 (AC) 到“00H”，并且将光标移到开头原点位置，这个指令并不改变 DDRAM 的内容	72 μs
输入点设定	0	0	0	0	0	0	0	1	I/D	S	指定在数据的读取与写入时，设定光标的移动方向及指定显示的移位	72 μs
显示状态开/关	0	0	0	0	0	0	1	D	C	B	D=1: 整体显示 ON C=1: 光标 ON B=1: 光标位置反白 ON	72 μs
光标或显示移位控制	0	0	0	0	0	1	S/C	R/L	X	X	设定光标的移动与显示的移位控制位；这个指令并不改变 DDRAM 的内容	72 μs
功能设定	0	0	0	0	1	DL	X	0 RE	X	X	DL=1: 8 位控制模式 DL=0: 4 位控制模式 RE=1: 选择扩展指令集 RE=0: 选择基本指令集	72 μs
设定 CGRAM 地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	设定 CGRAM 地址到地址计数器 (AC) 需确认扩展指令中 SR=0 (卷动地址或 RAM 地址选择)	72 μs
设定 DDRAM 地址	0	0	1	0 AC6	AC5	AC4	AC3	AC2	AC1	AC0	设定 DDRAM 地址到地址计数器 (AC) AC6 固定为 0	72 μs
读取忙碌标志 (BF) 和地址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	读取忙碌标志 (BF) 可以确认内部动作是否完成，同时可以读出地址计数器 (AC) 的值	0 μs
写数据到 RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	写入数据到内部 RAM (DDRAM/CGRAM/GDRAM)	72 μs
读出 RAM 的数据	1	1	D7	D6	D5	D4	D3	D2	D1	D0	从内部 RAM 读取数据 (DDRAM/CGRAM/GDRAM)	72 μs

● 指令表 2 (RE=1: 扩展指令集)

指令	指令码										说明	执行时间 (540KHZ)	
	R S	R W	DB 7	DB 6	DB 5	DB 4	DB 3	DB 2	DB 1	DB0			
待机模式	0	0	0	0	0	0	0	0	0	0	1	进入待机模式，执行任何其他指令都可终止待机模式（COM1~32 停止动作）	72ms
滚动地址或RAM地址选择	0	0	0	0	0	0	0	0	0	1	SR	SR=1: 允许输入垂直滚动地址 SR=0: 允许设定 CGRAM 地址 (基本指令)	72μs
反白选择	0	0	0	0	0	0	0	0	1	R1	R0	选择 4 行中的任一行反白显示，并可决定反白与否 R1、R0 初值为“00”，当第一次设定时为反白显示，再一次设定时为正常显示	72μs
扩充功能设定	0	0	0	0	1	DL	X	1	RE	G	0	DL=1: 8 位控制模式 DL=0: 4 位控制模式 RE=1: 选择扩展指令集 RE=0: 选择基本指令集 G=1: 绘图显示 ON G=0: 绘图显示 OFF	72μs
设定 IRAM 地址或滚动地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	AC0	SR=1: AC5~AC0 为垂直滚动地址	72μs
设定绘图RAM地址	0	0	1	0	0	0	AC3	AC2	AC1	AC0	AC0	设定 GDRAM 地址到地址计数器 (AC) 先设垂直地址再设水平地址 (连续写入两个字节的数来完成垂直与水平的坐标地址) 垂直地址范围 AC5~AC0 水平地址范围 AC3~AC0	72μs

请详细参考 IC 资料“ST7920C17.PDF”的第 13~14 页。

备注;当IC1在接受指令前,微处理器必须先确认其内部处于非忙碌状态,即读取BF标志时,BF 需为零,方可接受新的指令;如果在送出一个指令前并不检查BF 标志,那么在前一个指令和这个指令中间必须延长一段较长的时间,即是等待前一个指令确实执行完成。

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

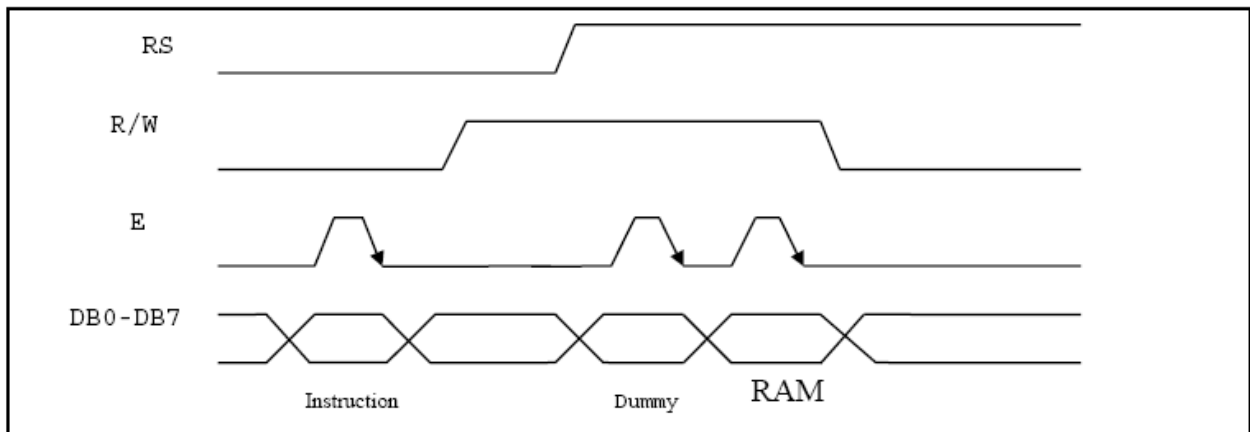
並列介面資料傳輸訊號

當PSB腳接高電位時，ST7920將進入並列模式，在並列模式下可由指令 **DL FLAG** 來選擇8-位元或4-位元介面，主控制系統將配合(RS, RW, E, DB0..DB7)來達成傳輸動作。

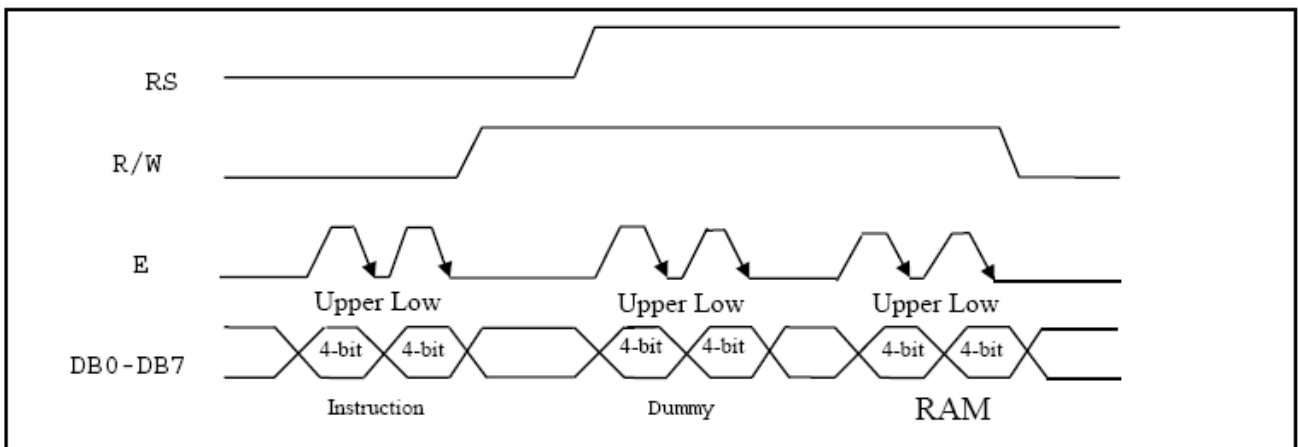
從一個完整的流程來看，當下設定位址指令後(CGRAM, DDRAM, IRAM.....)若要讀取資料時需先 DUMMY READ 一次才會讀取到正確資料第二次讀取時則不需 DUMMY READ 除非又下設定位址指令才需再次 DUMMY READ。

在4-位元傳輸模式中，每一個八位元的指令或資料都將被分為兩個位元組動作：較高4位元 (DB7~DB4) 的資料將會被放在第一個位元組的 (DB7~DB4) 部分，而較低4位元 (DB3~DB0) 的資料則會被放在第二個位元組的 (DB7~DB4) 部分，至於相關的另四位元則在4-位元傳輸模式中DB3~DB0介面未使用。

相關介面傳輸訊號請參考下圖說明：



Timing Diagram of 8-bit Parallel Bus Mode Data Transfer



Timing Diagram of 4-bit Parallel Bus Mode Data Transfer

在接收到同步位元及RW和RS資料的啓始位元組後，每一個八位元的指令將被分為兩個位元組接收到：較高4位元 (DB7~DB4) 的指令資料將會被放在第一個位元組的LSB部分，而較低4位元 (DB3~DB0) 的指令資料則會被放在第二個位元組的LSB部分，至於相關的另四位元則都為0。

串列傳輸訊號請參考下圖說明：

串列介面與串列傳輸資料

當PSB腳接低電位時，ST7920將進入串列模式，在串列模式下將使用兩條資料傳輸線作串列資料的傳送，主控制系統將配合傳輸同步時脈線（SCLK）與接收串列資料線（SID），來達成串列傳輸的動作。

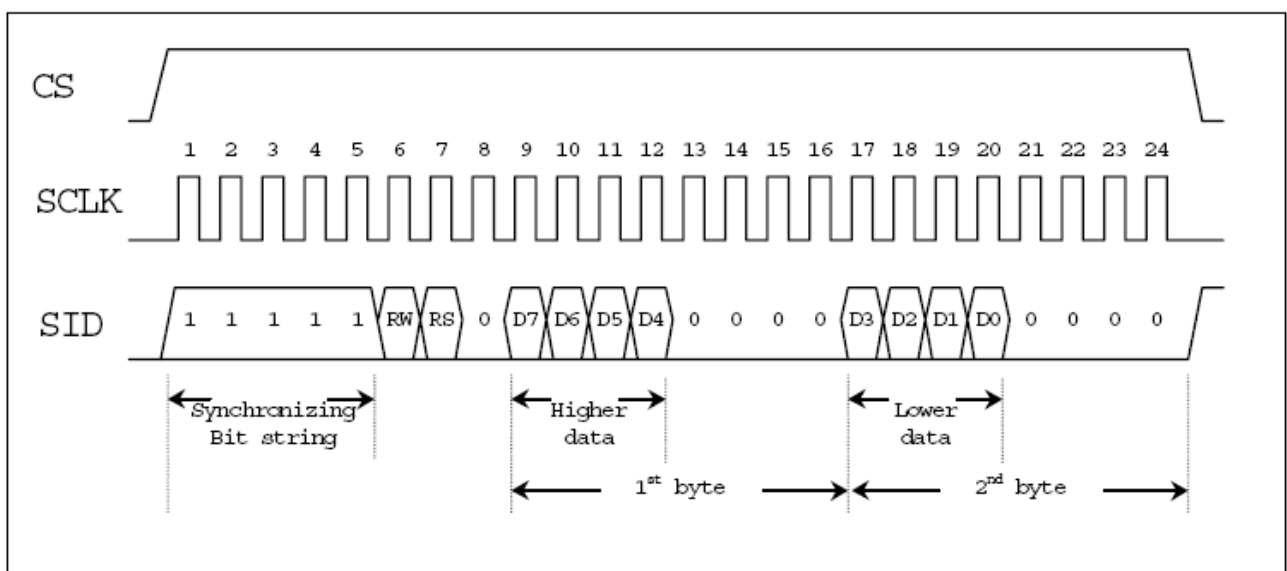
當需要同時連接數顆ST7920晶片時，晶片選擇腳（CS）將要被配合使用，在晶片選擇腳（CS）設為高電位時，同步時脈線（SCLK）輸入的訊號才會被接收，另一方面，當晶片選擇腳（CS）設為低電位時，ST7920的內部串列傳輸計數與串列資料將會被重置，也就是說在此狀態下，傳輸中的資料將被終止清除，並且將待傳輸的串列資料計數重設回第一位元；在一個最小的系統架構下，由一個微處理器連接控制單一個ST7920晶片時，相關的連接介面只需要使用同步時脈線（SCLK）與接收串列資料線（SID）兩隻腳，在這個模式下晶片選擇腳（CS）將被固定接到高電位。

ST7920的同步時脈線（SCLK）具有獨立的操作時脈，但是當有連續多個指令需要被傳送時，指令執行的時間將需要被考慮，必須確實等到前一個指令完全執行完成才能傳送下一筆資料，因為ST7920內部並沒有傳送/接收緩衝區。

從一個完整的串列傳輸流程來看，一開始先傳輸啓始位元組，它需先接收到五個連續的“1”（同步位元字串）在啓始位元組，此時傳輸計數將被重置並且串列傳輸將被同步，再跟隨的兩個位元字串分別指定傳輸方向位元（RW）及暫存器選擇位元（RS），最後第八的位元則為“0”。

在接收到同步位元及RW和RS資料的啓始位元組後，每一個八位元的指令將被分為兩個位元組接收到：較高4位元（DB7~DB4）的指令資料將會被放在第一個位元組的LSB部分，而較低4位元（DB3~DB0）的指令資料則會被放在第二個位元組的LSB部分，至於相關的另四位元則都為0。

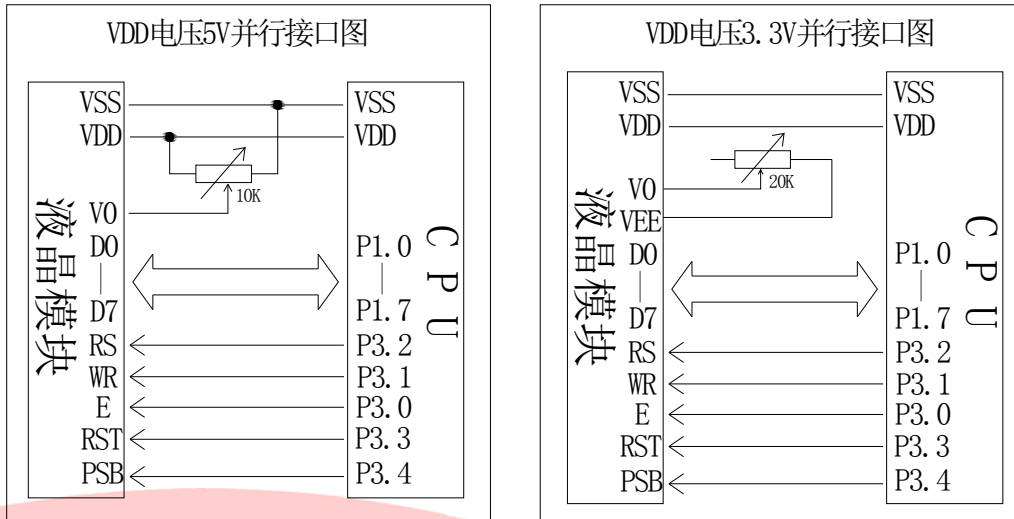
串列傳輸訊號請參考下圖說明：



Timing Diagram of Serial Mode Data Transfer

7.5 程序举例:

液晶模块与 MPU 并行接口图如下(以 8051 系列单片机为例):



并行接口图

```

/* Test program for JLX12832C
   Driver IC is:ST7920(or competible)
   JLX electronic Co.,ltd, http://www.jlxlcd.cn;http://www.jlxlcd.com.cn
   本程序显示内容是在第一行第一位置显示中文字"深圳晶联讯"
*/

```

```

#include <reg51.h>

```

```

sbit r_w=P3^1;
sbit reset=P3^3;
sbit rs=P3^2;
sbit e=P3^0;
sbit psb=P3^4;
#define data_bus P1
#define uchar unsigned char

```

```

char code yun[]=
{
/*-- 文字: 运 --*/
/*-- 宋体 23; 此字体下对应的点阵为: 宽 x 高=31x31 --*/
/*-- 宽度不是 8 的倍数, 现调整为: 宽度 x 高度=32x32 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x01, 0x80,
0x07, 0x0F, 0xFF, 0xC0, 0x03, 0x8F, 0xFF, 0xC0, 0x03, 0x80, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x70, 0x03, 0x3F, 0xFF, 0xF8,
0x7F, 0x80, 0x70, 0x00, 0x33, 0x80, 0xF0, 0x00, 0x03, 0x00, 0xE0, 0x00, 0x03, 0x01, 0xC0, 0x00,
0x03, 0x01, 0x86, 0x00, 0x03, 0x03, 0x07, 0x00, 0x03, 0x07, 0x03, 0x80, 0x03, 0x0E, 0x01, 0xC0,
0x03, 0x1F, 0xFF, 0xE0, 0x03, 0x1F, 0xFC, 0xE0, 0x03, 0x0E, 0x00, 0xE0, 0x0F, 0x80, 0x00, 0x40,
0x3D, 0xC0, 0x00, 0x00, 0x78, 0xF0, 0x00, 0x00, 0x70, 0x7F, 0xC1, 0xFC, 0x20, 0x1F, 0xFF, 0xF8,
0x00, 0x03, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

```

```

char code xing[]={
/*-- 文字: 行 --*/
/*-- 宋体 23; 此字体下对应的点阵为: 宽 x 高=31x31 --*/
/*-- 宽度不是 8 的倍数, 现调整为: 宽度 x 高度=32x32 --*/

```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x00, 0x00, 0x00, 0xE0, 0x00, 0xE0,
0x01, 0xE3, 0xFF, 0xF0, 0x03, 0x83, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00,
0x0C, 0x60, 0x00, 0x00, 0x38, 0x70, 0x00, 0x00, 0x20, 0xE0, 0x00, 0x30, 0x00, 0xEF, 0xFF, 0xF8,
0x01, 0xCF, 0xFF, 0xFC, 0x03, 0x80, 0x0E, 0x00, 0x03, 0xC0, 0x0E, 0x00, 0x07, 0xC0, 0x0E, 0x00,
0x0F, 0x80, 0x0E, 0x00, 0x1D, 0x80, 0x0E, 0x00, 0x31, 0x80, 0x0E, 0x00, 0x61, 0x80, 0x0E, 0x00,
0x01, 0x80, 0x0E, 0x00, 0x01, 0x80, 0x0E, 0x00, 0x01, 0x80, 0x0E, 0x00, 0x01, 0x80, 0x0E, 0x00,
0x01, 0x80, 0x0E, 0x00, 0x01, 0x80, 0x0E, 0x00, 0x01, 0x80, 0x0E, 0x00, 0x01, 0x81, 0xFE, 0x00,
0x01, 0x80, 0x7C, 0x00, 0x01, 0x00, 0x3C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_R[]={
/*- 文字: R --*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=16x32 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7F, 0xF0, 0x3C, 0x78,
0x18, 0x1C, 0x18, 0x1C, 0x18, 0x0C, 0x18, 0x0C, 0x18, 0x1C, 0x18, 0x1C, 0x18, 0x78, 0x1F, 0xF0,
0x19, 0xC0, 0x19, 0xE0, 0x18, 0xE0, 0x18, 0xE0, 0x18, 0x70, 0x18, 0x70, 0x18, 0x78, 0x18, 0x38,
0x18, 0x38, 0x3C, 0x3C, 0x7E, 0x1E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_U[]={
/*- 文字: U --*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=16x32 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFC, 0x3E, 0x78, 0x1C,
0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08,
0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x30, 0x08, 0x38, 0x18,
0x38, 0x18, 0x1E, 0x70, 0x0F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_N[]={
/*- 文字: N --*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=16x31 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xF8, 0x3E, 0x78, 0x1C,
0x3C, 0x08, 0x3C, 0x08, 0x2E, 0x08, 0x2E, 0x08, 0x2F, 0x08, 0x27, 0x08, 0x27, 0x88, 0x23, 0x88,
0x23, 0xC8, 0x21, 0xC8, 0x21, 0xE8, 0x20, 0xE8, 0x20, 0xF8, 0x20, 0x78, 0x20, 0x78, 0x20, 0x38,
0x20, 0x38, 0x70, 0x18, 0xF8, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_I[]={
/*- 文字: I --*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=16x31 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1F, 0xF8, 0x03, 0xC0,
0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80,
0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80,
0x01, 0x80, 0x03, 0xC0, 0x1F, 0xF8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_G[]={
/*- 文字: G --*/
/*- 宋体 23; 此字体下对应的点阵为: 宽 x 高=16x31 --*/
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xF8, 0x0E, 0x78,
0x1C, 0x38, 0x38, 0x18, 0x30, 0x18, 0x70, 0x08, 0x70, 0x00, 0x70, 0x00, 0x60, 0x00, 0x60, 0x00,
0x60, 0x00, 0x60, 0x00, 0x60, 0x7E, 0x70, 0x3C, 0x70, 0x18, 0x70, 0x18, 0x30, 0x18, 0x38, 0x18,
0x18, 0x38, 0x1E, 0x78, 0x07, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
char code char_JLX[]={
/*- 调入了一幅图像: E:\work\图片收藏夹\JLX12832c.bmp --*/
/*- 宽度 x 高度=128x32 --*/
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
0x8F, 0xF0, 0x06, 0x63, 0x03, 0x00, 0xC0, 0x00, 0x30, 0x0C, 0x03, 0xFF, 0x81, 0xB0, 0x00, 0x19,
0x8C, 0x31, 0xFF, 0xC1, 0xFF, 0x80, 0xC0, 0x3F, 0xF8, 0x0C, 0x63, 0xF1, 0x81, 0xB0, 0x3F, 0xFD,
```

```

0x8F, 0xF0, 0xD9, 0x80, 0x03, 0x0F, 0xFC, 0x00, 0x61, 0xFF, 0xF3, 0xF1, 0x83, 0x18, 0x00, 0x19,
0x8C, 0x30, 0xDF, 0xE0, 0x1B, 0x0C, 0xCC, 0x01, 0xC0, 0x18, 0x03, 0xFF, 0x83, 0x18, 0x7F, 0xF9,
0x8F, 0xF0, 0xF9, 0x87, 0x9B, 0x0C, 0xCC, 0x01, 0x80, 0x3F, 0xC3, 0xB1, 0x86, 0x0C, 0x00, 0x19,
0x80, 0x00, 0xD9, 0x81, 0xFF, 0x0F, 0xFC, 0x7F, 0xFC, 0x70, 0xC3, 0xFF, 0x8C, 0xC6, 0x1F, 0x99,
0xBF, 0xFC, 0xFF, 0xF1, 0x9B, 0x0C, 0xCC, 0x01, 0x80, 0xFF, 0xC3, 0xFD, 0x98, 0xC3, 0x19, 0x99,
0xB3, 0xCC, 0xD9, 0x81, 0x9B, 0x0C, 0xCC, 0x01, 0x81, 0xB0, 0xC3, 0xFF, 0x01, 0x80, 0x19, 0x99,
0xBF, 0xFC, 0xFF, 0xC1, 0xFB, 0x0F, 0xFC, 0x01, 0x80, 0x3F, 0xC3, 0xF6, 0x01, 0xB0, 0x1F, 0x99,
0xB3, 0xCD, 0xFB, 0xC1, 0xDB, 0xC0, 0xC3, 0x01, 0x80, 0x30, 0xC3, 0x33, 0x03, 0x18, 0x19, 0x99,
0xBF, 0xFC, 0x1E, 0x61, 0x99, 0xC0, 0xC3, 0x01, 0x80, 0x30, 0xC3, 0x3D, 0x87, 0xFC, 0x00, 0x79,
0xB3, 0xCC, 0x1C, 0x30, 0x18, 0xC0, 0x7F, 0x03, 0x80, 0x31, 0xC3, 0x38, 0xC3, 0x0C, 0x00, 0x31,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x80, 0x00, 0x00, 0x00, 0x0F, 0xDE, 0x1F, 0xC3, 0x07, 0x87, 0x87, 0x87, 0x87, 0xC0, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x0D, 0x87, 0x0C, 0xCC, 0xCC, 0xCC, 0xCC, 0xC0, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x0D, 0x83, 0x0C, 0xCC, 0xC0, 0xCC, 0xD8, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x07, 0x03, 0x00, 0xC7, 0x83, 0x80, 0xD8, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x07, 0x03, 0x01, 0x87, 0x80, 0xC1, 0x98, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x07, 0x03, 0x03, 0x0C, 0xC0, 0xC3, 0x18, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0x0D, 0x83, 0x06, 0x0C, 0xC0, 0xC6, 0x18, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x03, 0x0C, 0xCD, 0x83, 0x0C, 0x0C, 0xCC, 0xCC, 0x0C, 0xC0, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x1B, 0x1F, 0xDF, 0xC7, 0x8F, 0xC7, 0x87, 0x8F, 0xC7, 0x80, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x1E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,

```

```

};

/*=====延时=====*/
void delay(int i)
{
    int j,k;
    for(j=0;j<i;j++)
    for(k=0;k<990;k++);
}

//=====delay time=====
void delay1(int i)
{
    int j,k;
    for(j=0;j<i;j++)
    for(k=0;k<2;k++);
}

/*====等待按键：P2.0 与地相接时按键生效=====*/
void Switch()
{
    repeat:
        if (P2&0x01) goto repeat;
        else delay(8);
        if (P2&0x01) goto repeat;
        else;
}

//====检查忙标志位 BF=====
void check_busy1()
{
    rs=0;
    data_bus=0xff;
}

```

```
    if(P1^7==1)
    {
        r_w=1;
        e=1;
        //delay(5);
        e=0;
    }
else;
}

/*====送指令到液晶模块驱动 IC=====*/
void transfer_command(int data1)
{
    check_busy1();
    rs=0;
    r_w=0;
    data_bus=data1;
    e=1;
    delay1(5);
    e=0;
    delay1(5);
}

/*=====传送数据到液晶模块驱动 IC=====*/
void transfer_data(int data1)
{
    check_busy1();
    rs=1;
    r_w=0;
    data_bus=data1;
    e=1;
    //delay1(1);
    e=0;
    //delay1(1);
}

void clear_screen()
{
    int i, j;
    transfer_command(0x36); /*选用扩展指令集*/
    //transfer_command_ic2(0x36); /*选用扩展指令集*/
    //delay(10);
    for(j=0; j<32; j++)
    {
        transfer_command(0x80+j); /*垂直地址*/
        transfer_command(0x80); /*水平地址*/

        for(i=0; i<32; i++)
        {
            transfer_data(0x00);
        }
    }
}

void clear_ddram()
{
    transfer_command(0x30);
}
```

```
    transfer_command(0x01);
    delay(10);
}

void display_3232(int y, int x, char code *p)
{
    int i, j;
    transfer_command(0x36); /*选用扩展指令集*/
    delay(10);
    transfer_command(0x80); /*选用扩展指令集*/
    delay(10);
    for(j=0; j<32; j++)
    {
        transfer_command(0x80+y+j); /*垂直地址*/

        transfer_command(0x80+x); /*水平地址*/

        for(i=0; i<4; i++)
        {
            transfer_data(*p);
            p++;
        }
    }
}
```

```
void display_1632(int y, int x, char code *p)
{
    int i, j;
    transfer_command(0x36);
    delay(10);
    transfer_command(0x80);
    delay(10);
    for(j=0; j<32; j++)
    {
        transfer_command(0x80+y+j);

        transfer_command(0x80+x);

        for(i=0; i<2; i++)
        {
            transfer_data(*p);
            p++;
        }
    }
}
```

```
void display_12832(int y, int x, char code *p)
{
    int i, j;
    transfer_command(0x36); /*选用扩展指令集*/
    delay(10);
    transfer_command(0x80); /*选用扩展指令集*/
    delay(10);
    for(j=0; j<32; j++)
    {
        transfer_command(0x80+y+j); /*垂直地址*/

        transfer_command(0x80+x); /*水平地址*/

        for(i=0; i<16; i++)
```



```

        {
            transfer_data(*p);
            p++;
        }
    }

}

/*初始化*/
void Initial_ic()
{
    transfer_command(0x30);    /*选用基本指令集*/
    delay(10);
    transfer_command(0x01);    /*清屏*/
    delay(10);
    transfer_command(0x06);    //
    delay(10);
    transfer_command(0x0c);    /*开显示，关光标*/
    delay(10);
}

/*调用中文字库里的汉字*/
void display_char(int y,int x,int char_length,uchar *dp)
{
    uchar i=0;
    transfer_command(0x80+(y-1)*(0x10)+(x-1));
    for(i=0;i<char_length;i++)
    {
        transfer_data(*dp);
        dp++;
        transfer_data(*dp);/* 以上两行数据合起来显示一个汉字*/
        dp++;
    }
}

/*主程序*/
void main(void)
{
    psb=1;                /*选择并口*/
    Initial_ic();        /*初始化*/
    while(1)
    {
        display_char(1,1,8,"深圳市晶联讯电子"); /*（在1行，1列，8个汉字）调用显示汉字子程序*/
        display_char(2,1,8,"JLX12832,128*32"); /*（在2行，1列，8个汉字）调用显示汉字子程序*/
        Switch();        /*等待按键 P2.0 口*/
        clear_ddram();
        clear_screen();
        display_12832(0,0,char_JLX);
        Switch();        /*等待按键 P2.0 口*/
        clear_ddram();
        clear_screen();
        display_3232(0,2,yun); /*第一个是垂直地址，第二个是水平地址，第三个是图案数据*/
        display_3232(0,4,xing);
        Switch();
        clear_screen();
        display_1632(0,1,char_R);
        display_1632(0,2,char_U);
        display_1632(0,3,char_N);
        display_1632(0,4,char_N);
        display_1632(0,5,char_I);
    }
}

```

```

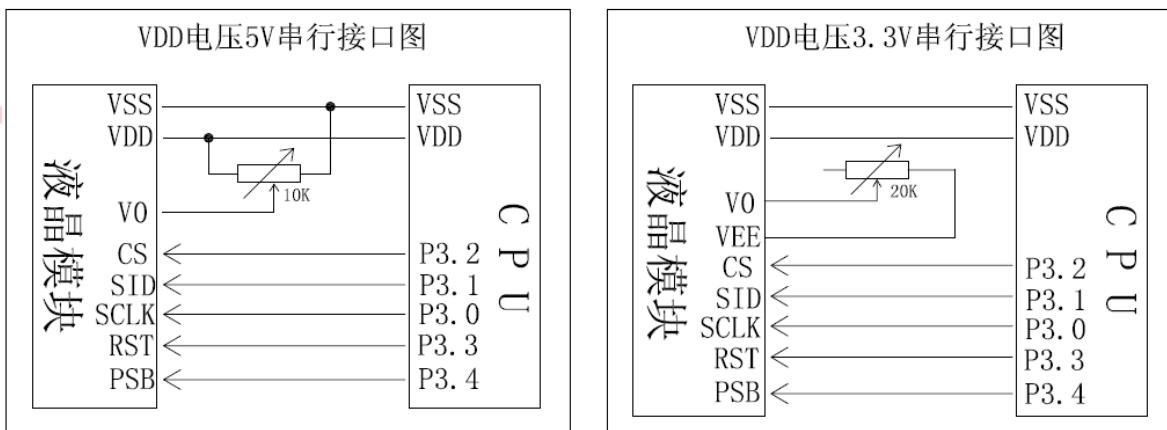
display_1632(0, 6, char_N);
display_1632(0, 7, char_G);
Switch();          /*等待按键 P2.0 口*/
clear_ddram();
clear_screen();
}
}

```

7.5.2、以下为串行接口方式范例程序

与并行方式相比较，只需改变接口顺序以及传送数据、传送命令这两个函数即可：

串行接口：



串行接口图

```

sbit cs=P3^2;
sbit sid=P3^1;
sbit sclk=P3^0;
sbit psb=P3^4;
sbit reset=P3^3;

```

```

//=====transfer command to LCM=====
void transfer_command(int data1)
{
    int i;
    cs=1;
    for(i=0;i<5;i++)
    {
        sid=1;
        sclk=0;
        sclk=1;
    }
    sid=0;    //r_w=0
    sclk=0;
    sclk=1;
    sid=0;    //rs=0
    sclk=0;
    sclk=1;
    sid=0;
}

```

```

sclk=0;
sclk=1;
for(i=0;i<4;i++)
{
    if(data1&0x80) sid=1;
    else sid=0;
    sclk=0;
    sclk=1;
    data1=data1<<=1;
}

for(i=0;i<4;i++)
{
    sid=0;
    sclk=0;
    sclk=1;
}
for(i=0;i<4;i++)
{
    if(data1&0x80) sid=1;
    else sid=0;
    sclk=0;
    sclk=1;
    data1=data1<<=1;
}

for(i=0;i<4;i++)
{
    sid=0;
    sclk=0;
    sclk=1;
}
cs=0;
}
//-----transfer data to LCM-----
void transfer_data(int data1)
{
    int i;
    cs=1;
    for(i=0;i<5;i++)          //SID=1, 5个脉冲，同步信号
    {
        sid=1;
        sclk=0;
        sclk=1;
    }
    sid=0;                    //这一位为 R/W 设置，R/W=0 表示写
    sclk=0;
    sclk=1;
    sid=1;                    //这一位为 RS 设置，RS=1 表示数据寄存器
    sclk=0;
    sclk=1;
    sid=0;
    sclk=0;
    sclk=1;
    for(i=0;i<4;i++)          //传数据的高 4 位
    {

        if(data1&0x80) sid=1;
        else sid=0;
    }
}

```

```

    sclk=0;
    sclk=1;
    data1=data1<<=1;
}

for(i=0;i<4;i++)          //SID=0, 过 4 个脉冲
{
    sid=0;
    sclk=0;
    sclk=1;
}
for(i=0;i<4;i++)          //传数据的低 4 位
{

    if(data1&0x80) sid=1;
    else sid=0;
    sclk=0;
    sclk=1;
    data1=data1<<=1;
}

for(i=0;i<4;i++)          //SID=0, 过 4 个脉冲
{
    sid=0;
    sclk=0;
    sclk=1;
}
cs=0;
}

```

