

# JLX19296G-240-PC 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	3~5
4	基本原理	6
5	技术参数	7
6	时序特性	7~10
7	指令功能及硬件接口与编程案例	11~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 JLX19296G-240 是一款内置中文字库的液晶模块，由于使用方便、显示清晰，广泛应用于各种人机交流面板。

JLX19296G-240 可以显示不大于 192×96 点阵单色图片，或显示 12 个×6 行=72 个的 16\*16 点阵的汉字，或显示 24 个×6 行=144 个的 8\*16 点阵的英文、数字、符号。

## 2. JLX19296-240 图像型点阵液晶模块的特性

1.1 内置中文字库：模块驱动控制器带中文字库，方便使用；

1.2 结构牢：背光带挡墙；

1.3 IC 采用 KS0192, 功能强大，稳定性好

1.4 功耗低:1 - 100mW（不带背光 1mW <3.3V@0.3mA>, 带背光不大于 100mW<3.3V@30mA>）；

1.5 显示内容：

- 192\*96 点阵单色图片。

- 可显示 12 个×6 行=72 个的 16\*16 点阵的汉字。

- 可显示 24 个×6 行=144 个的 8\*16 点阵的 ASCII 码

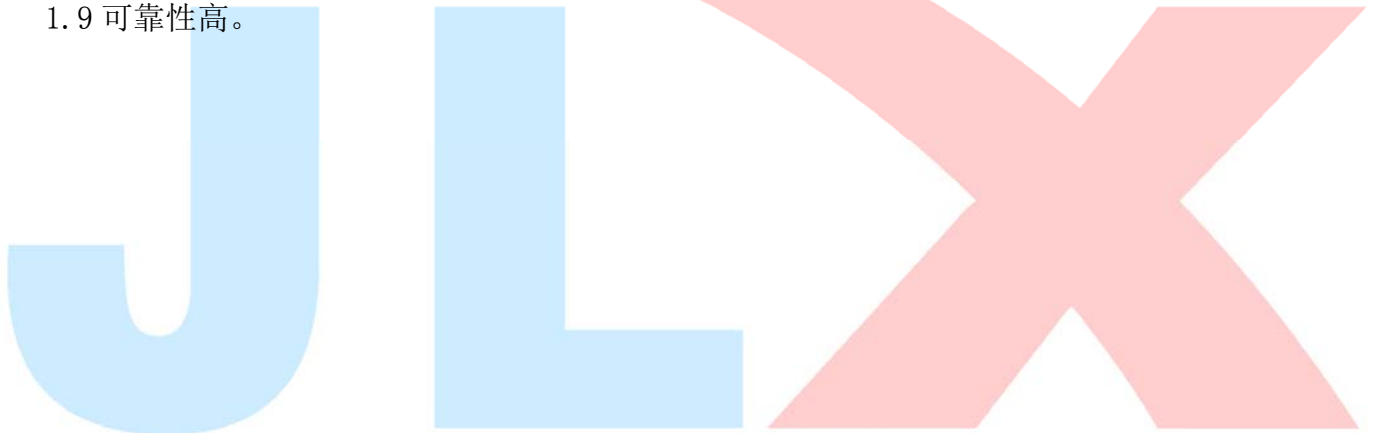
- 可选用 16\*16 点阵或其他点阵的图片来自编汉字。

1.6 指令功能强；

1.7 接口简单方便:可选 I<sup>2</sup>C 总线、4 线 SPI 串口、6800 系列并口。

1.8 工作温度宽:-20℃ - 70℃；

1.9 可靠性高。



3. 外形尺寸及接口引脚功能

3.1 外形尺寸图

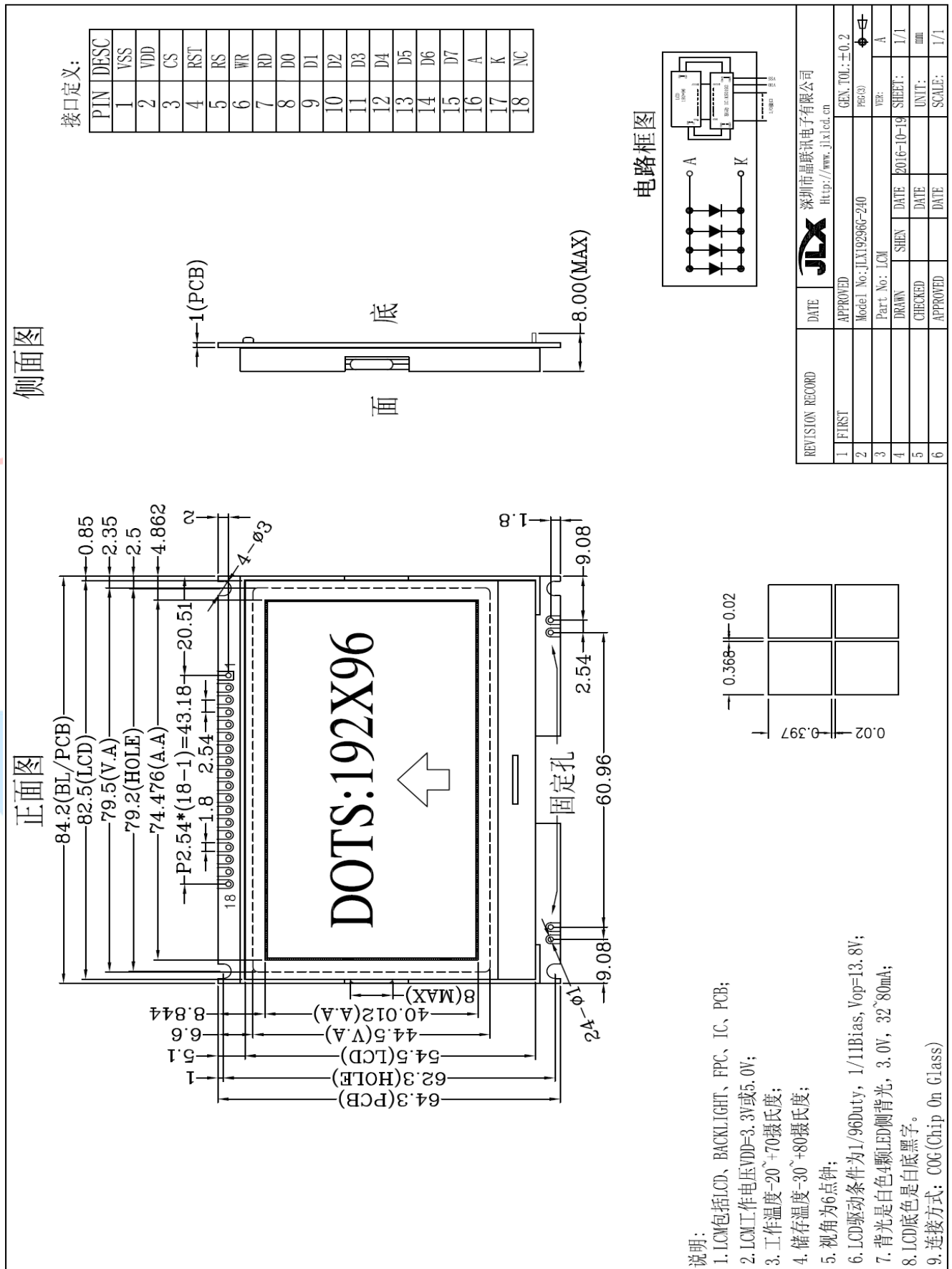


图 1. 外形尺寸

### 3.2 模块的接口引脚功能

#### 3.2.1 并行时接口引脚功能

引线号	符号	名称	功能
1	VSS	接地	0V
2	VDD	供电电源正极	供电电源正极
3	CS	片选	低电平片选
4	RST	复位	低电平复位，复位完成后，回到高电平，液晶屏开始工作
5	RS	寄存器选择信号	H:数据寄存器 0:指令寄存器（IC资料上所写为“CD”）
6	WR	读/写	6800 时序：H:读数据 0:写数据
7	RD	使能信号	6800 时序：使能信号
8~15	DO~D7	I/O	并行接口时，数据总线 DB0~DB7
16	A	背光电源正极	同 VDD 电压即可
17	K	背光电源负极	接 VSS
18	NC	空	空

表 1：模块并行接口引脚功能

#### 3.2.2 四线串行时接口引脚功能

引线号	符号	名称	功能
1	VSS	接地	0V
2	VDD	供电电源正极	供电电源正极
3	CS	片选	低电平片选
4	RST	复位	低电平复位，复位完成后，回到高电平，液晶屏开始工作
5	RS	寄存器选择信号	H:数据寄存器 0:指令寄存器（IC资料上所写为“CD”）
6	WR	读/写	串行接口时，接 VDD
7	RD	使能信号	串行接口时，接 VDD
8~13	DO~D5	I/O	接 VDD
14	D6 (SCK)	串行时钟	串行时钟
15	D7 (SDA)	串行数据	串行时钟
16	A	背光电源正极	同 VDD 电压即可
17	K	背光电源负极	接 VSS
18	NC	空	空

表 2：4 线 SPI 串行接口引脚功能

#### 3.2.3 I<sup>2</sup>C 总线时接口引脚功能

引线号	符号	名称	功能
1	VSS	接地	0V
2	VDD	供电电源正极	供电电源正极
3	CS	片选	低电平片选
4	RST	复位	低电平复位，复位完成后，回到高电平，液晶屏开始工作
5	RS	寄存器选择信号	IIC 接口：RS 接 VSS。
6	WR	读/写	串行接口时，接 VDD
7	RD	使能信号	串行接口时，接 VDD
8~9	DO~D1	I/O	串行接口时，数据总线 DB0~DB1 接 VDD

10~14	D2~D6 (SDA)	I/O	IIC 接口：D2-D6 四根数据线连一起，做串行数据
15	D7 (SCLK)	I/O	IIC 接口：串行时钟
16	A	背光电源正极	同 VDD 电压即可
17	K	背光电源负极	接 VSS
18	NC	空	空

表 3：I<sup>2</sup>C 总线接口引脚功能

#### 4. 基本原理

##### 4.1 液晶屏 (LCD)

在 LCD 上排列着 192×96 点阵, 192 个列信号与驱动 IC 相连, 96 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 工作电图:

图 1 是 JLX19296G-240 图像点阵型模块的电路框图, 它由驱动 IC KS0192 及几个电阻电容组成。

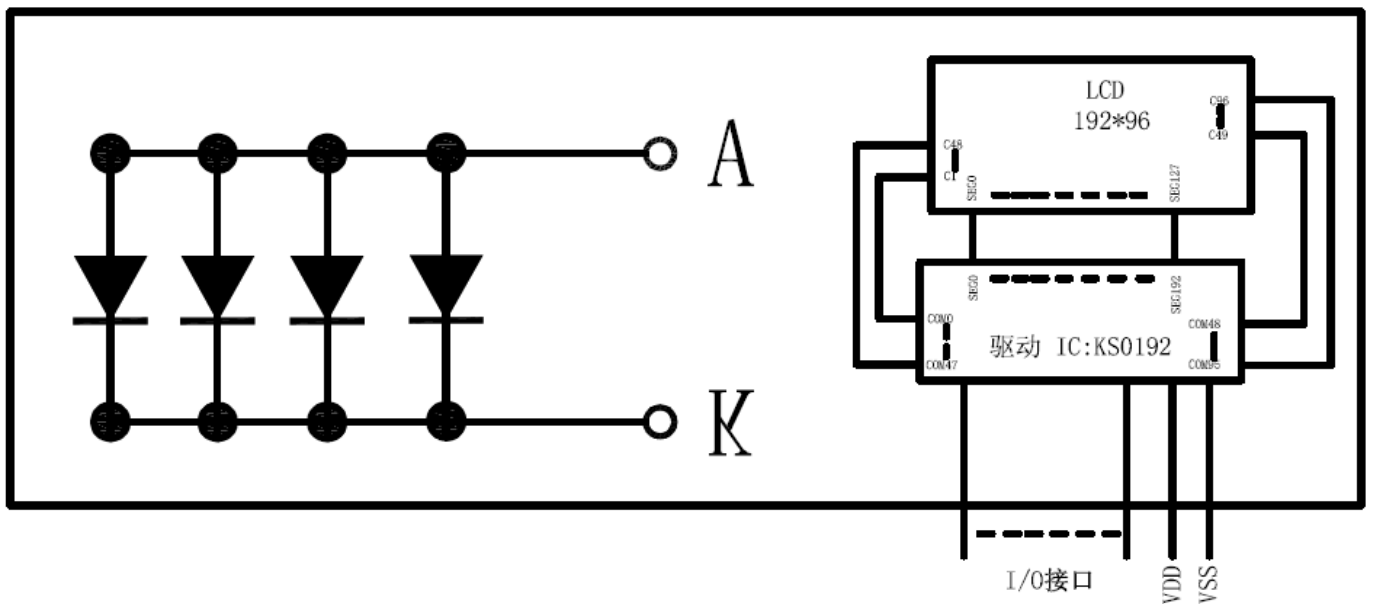


图 2: JLX19296G-240 图像点阵型液晶模块的电路框图

##### 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~+70° C;

存储温度: -30~+80° C;

背光板选用白色;

正常工作电流为: 32~80mA;

工作电压: 3.0V

## 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	°C
储存温度		-30		+80	°C

表 4：最大极限参数

### 5.2 直流（DC）参数

可以选择 3.3V 供电及 5.0V 供电两种方式：

名称	符号	测试条件	标准值			单位
			MIN	TYPE	MAX	
工作电压	VIN	3.3V 供电	1.7	3.3	3.4	V
		5.0V 供电	2.6	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	IOH = 0.2mA	2.4		-	V
输出低电平	VOO	I00 = 1.2mA	-		0.4	V
模块工作电流	IDD	VDD = 3.3V	-		0.3	mA
背光工作电流	ILED	VLED=3.0V	32	60	80	mA

表 5：直流（DC）参数

## 6. 读写时序特性 (AC 参数)

### 6.1 4 线 SPI 串行接口写时序特性 (AC 参数)

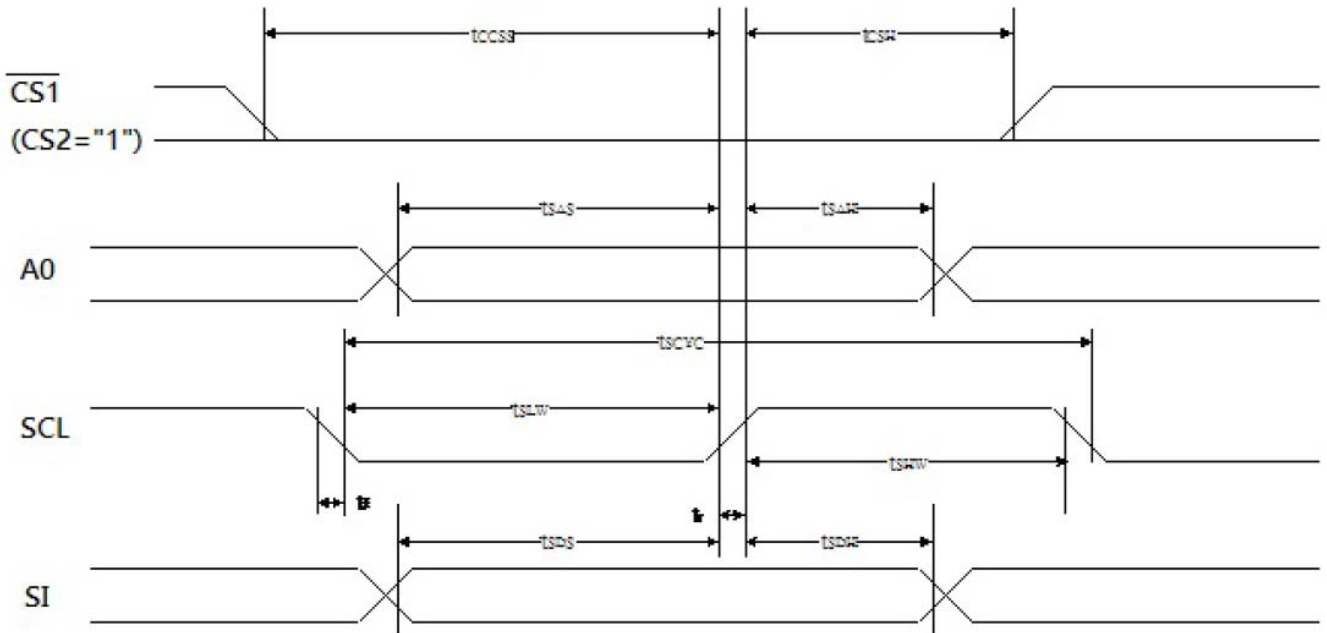


图 3. 从 CPU 写到 KS0192 (Writing Data from CPU to KS0192)

表 6. 写数据到 KS0192 的时序要求

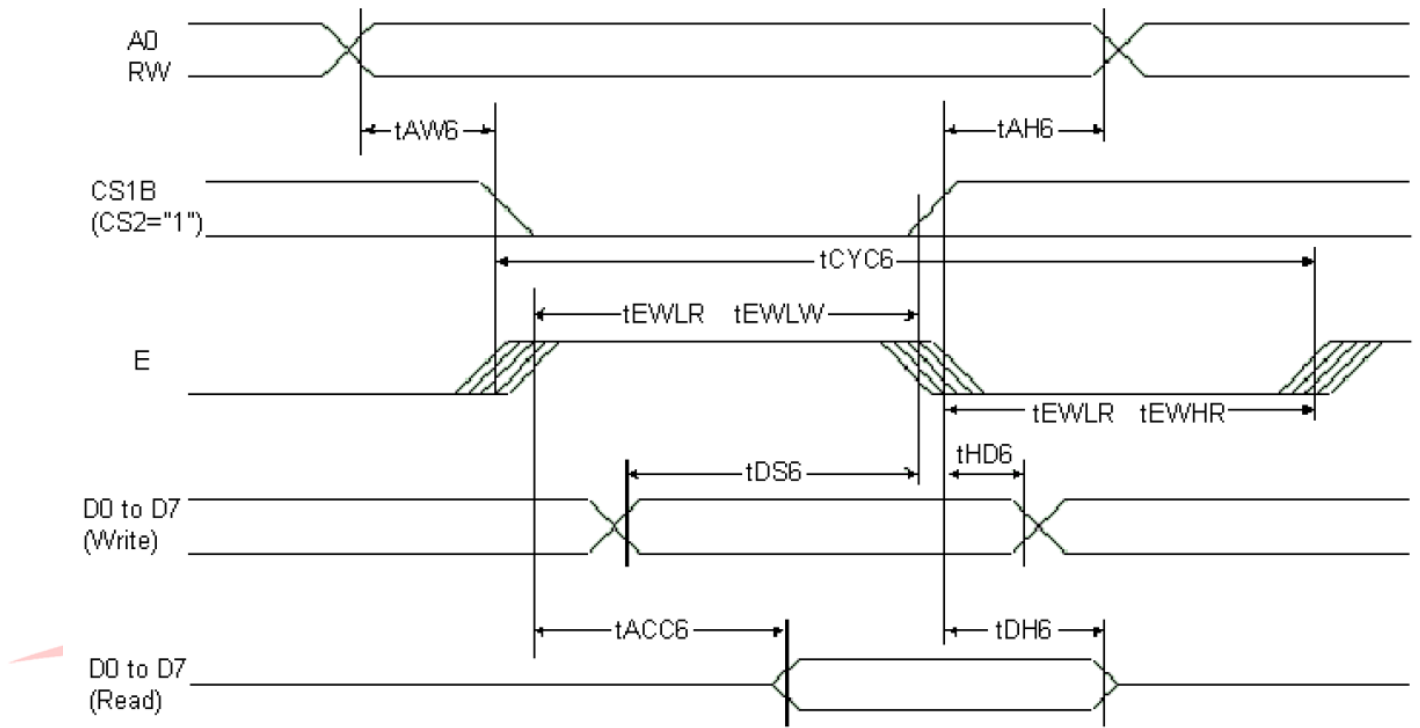
项目	符号	测试条件	极限值			单位
			MIN	TYPE	MAX	
4线 SPI串口时钟周期 (4-line SPI Clock Period)	tSCYC	引脚: SCL	50	---	---	ns
保持SCK高电平脉宽 (SCL "H" pulse width)	tSHW		25	---	---	ns
保持SCLK低电平脉宽 (SCL "L" pulse width)	tSLW		25	---	---	ns
地址建立时间 (Address setup time)	tSAS	引脚: A0	20	---	---	ns
地址保持时间 (Address hold time)	tSAH		10	---	---	ns
数据建立时间 (Data setup time)	tSDS	引脚: SID	20	---	---	ns
数据保持时间 (Data hold time)	tSDH		10	---	---	ns
片选信号建立时间 (CS-SCL time)	tCSS	引脚: CSB	20	---	---	ns
片选信号保持时间 (CS-SCL time)	tCSH		40	---	---	ns

VDD = 1.8~3.3V ± 5%, Ta = -30~85°C

输入信号的上升和下降时间 (TR, TF) 在 15 纳秒或更少的规定。

所有的时间, 用 20% 和 80% 作为标准规定的测定。

6.2 6800 时序并行接口的时序特性 (AC 参数)



1.

从 CPU 写到 KS0192 (Writing Data from CPU to KS0192)

图 4. 写数据到 KS0192 的时序要求 (6800 系列 MPU)

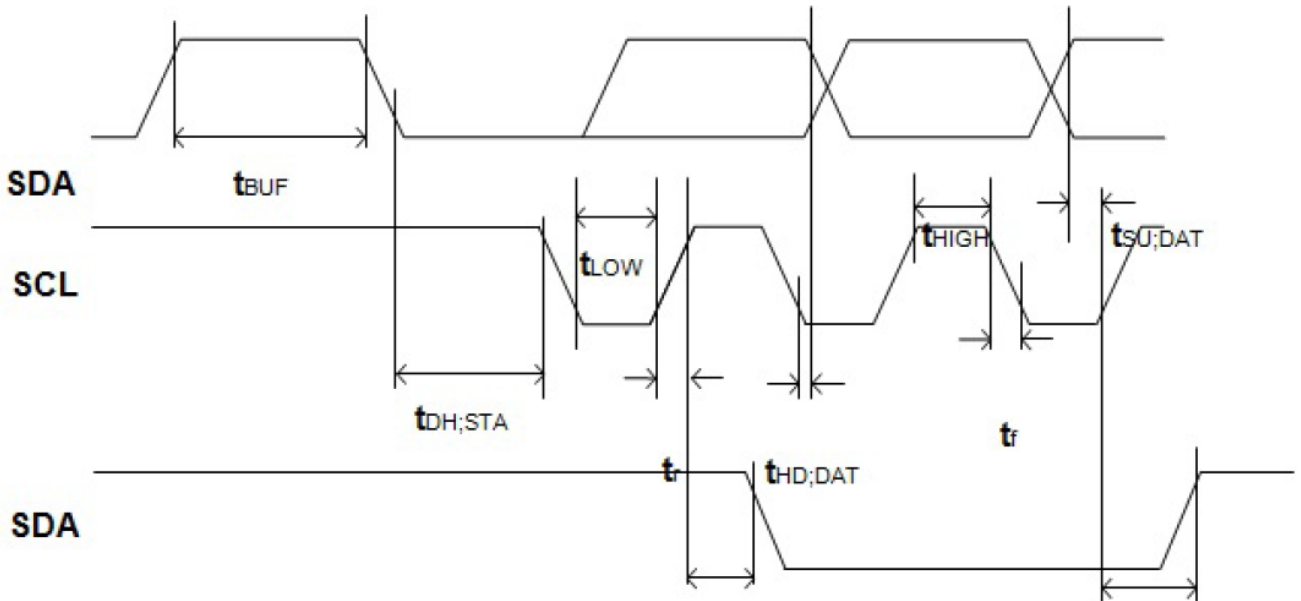
表 7. 读写数据的时序要求

项目	符号	名称	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH6	0		--	ns
地址建立时间		tAW6	0		--	ns
系统循环时间		tCYC6	240			ns
使能“低”脉冲宽度(写)	E	tEHLW	80		--	ns
使能“高”脉冲宽度(写)		tEHLR	80		--	ns
使能“低”脉冲宽度(读)	E	tEHLR	80			
使能“高”脉冲宽度(读)		tEHLR	140			
写数据建立时间	DB[7: 0]	tDS6	40		--	ns
写数据保持时间		tDH6	5		--	ns

VDD = 1.8~3.3V ± 5%, Ta = -30~85°C



6.3 I<sup>2</sup>C 接口的时序特性 (AC 参数)



从 CPU 写到 KS0192 (Writing Data from CPU to KS0192)

图 5. 写数据到 KS0192 的时序要求 (I<sup>2</sup>C 系列 MPU)

表 8. 读写数据的时序要求

项目	符号	名称	极限值			单位
			MIN	TYPE	MAX	
SCL时钟频率	CSL	FSCLK	--		400	kUZ
SCL时钟的低周期	CSL	TLOW	1.3		--	us
SCL时钟周期	CSL	THIGH	0.6		--	us
数据保持时间	SDA	TSU;Data	0.1		--	ns
数据建立时间	SDA	THD;Data	0		0.9	us
SCL, SDA 的上升时间	SCL	TR	20+0.1Cb		300	ns
SCL, SDA 下降时间	SCL	TF	20+0.1Cb		300	ns
每个总线为代表的电容性负载		Cb	--		400	pF
一个重复起始条件设置时间	SDA	TSU;SUA	0.6		--	us
启动条件的保持时间	SDA	THD;STA	0.6		--	us
为停止条件建立时间		TSU;STO	0.6		--	us
容许峰值宽度总线		TSW	--		50	ns
开始和停止条件之间的总线空闲时间	SCL	TBUF	1.3	--	1.3	us

所有的时间，用 20%和 80%作为标准规定的测定。

这是推荐的操作 IC 接口与 VDD1 高于 2.6V。

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

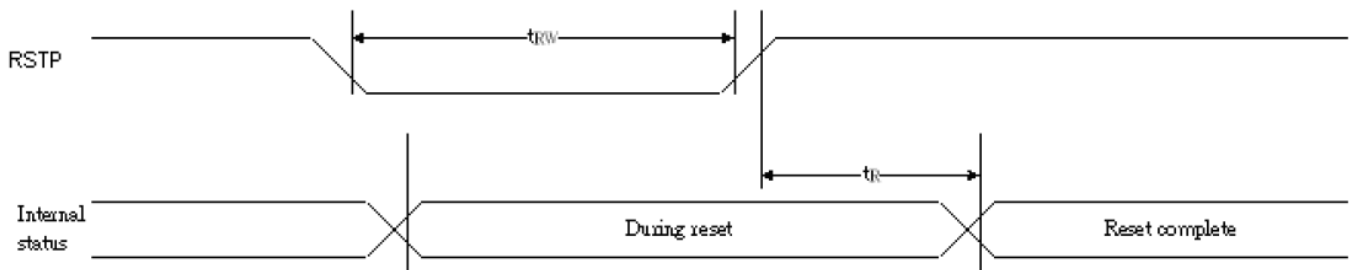
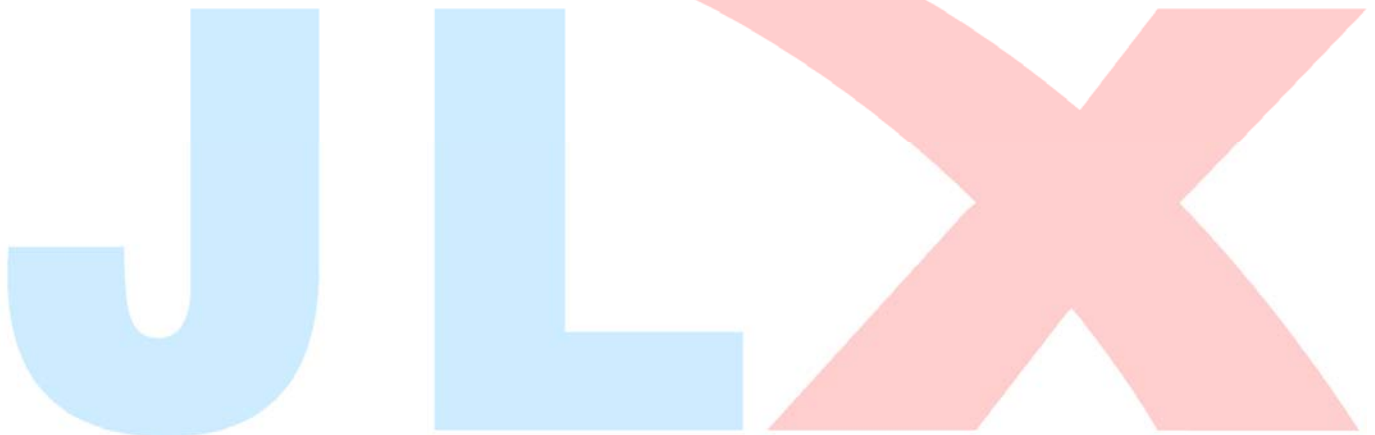


图 6：电源启动后复位的时序

表 9：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	$T_{RW}$		--	--	1	us
复位保持低电平的时间	$T_{RD}$	引脚：RESET, WR	1	--	--	ms



7. 指令功能:

7.1 指令表

KS0192 instruction Table 1 (RE1=0, RE2=0)

Instruction	RE1,RE2	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time
(1) Read display data	X	1	1	Read data								Read data into DDRAM/CGRAM/SEGRAM	72μs
(2) Write display data	X	1	0	Write data								Write data into DDRAM/CGRAM/SEGRAM	72μs
(3) Clear Display	X	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM, and set DDRAM address to "00H" from AC	4.6ms
(4) Return Home	0,0	0	0	0	0	0	0	0	0	1	X	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	72μs
(5) Entry Mode Set	0,0	0	0	0	0	0	0	0	1	I/D	S	Assign cursor moving direction. I/D="1": increment I/D="0": decrement And entire display shift enable bit.	72μs
(6) Instruction Table Set	X	0	0	0	0	0	0	1	X	RE1	RE2	Set Instruction Table RE1,RE2=(0,0), table 1 (default) RE1,RE2=(0,1), table 2 RE1,RE2=(1,0), table 3 RE1,RE2=(1,1), table 4	72μs
(7) Cursor or Display Shift Set	0,0	0	0	0	0	0	1	S/C	R/L	X	X	Cursor or display shift S/C="1": display shift S/C="0": cursor shift R/L="1": shift to right R/L="0": shift to left	72μs
(8) Function Set	0,0	0	0	0	0	1	IF	D	C	B	X	Set Interface Data Length IF="1": 8-bit interface IF="0": 4-bit interface Set Display /Cursor/Blink On/OFF D="1": display on D="0": display off C="1": cursor on C="0": cursor off B="1": blink on B="0": blink off	72μs
(9) Set DDRAM Address	0,0	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter	72μs
(10) Set SEGRAM Address	0,0	0	1	0	1	0	0	AC3	AC2	AC1	AC0	Set ICON RAM address in address counter	72μs
(11) Read Busy Flag and Address	0,0	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Can know internal operation is ready or not by reading BF. The contents of address counter can also be read. BF="1": busy state BF="0": ready state	72μs

表 10. 指令表

KS0192 instruction Table 2 (RE1=0, RE2=1)

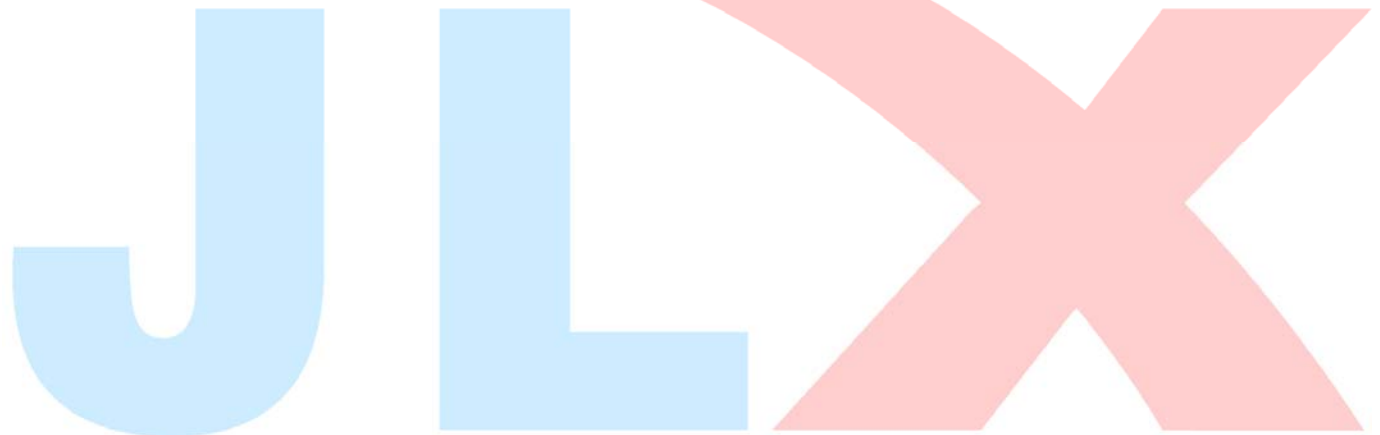
Instruction	RE1,RE2	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time
(1) Read display data	X	1	1	Read data							Read data into DDRAM/CGRAM/SEGRAM	72μs	
(2) Write display data	X	1	0	Write data							Write data into DDRAM/CGRAM/SEGRAM	72μs	
(3) Clear Display	X	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM, and set DDRAM address to "00H" from AC	72μs
(12) Power Down Mode	0,1	0	0	0	0	0	0	0	0	1	PD	Set power down mode bit. PD="1": power down mode set PD="0": power down mode disable	72μs
(13) COM and SEG Scan Direction Set	0,1	0	0	0	0	0	0	0	1	SHL	BID	Segment bi-direction function BID="1": Seg 191 --> Seg 0 BID="0": Seg 0--> Seg 191 Common scan direction SHL="1": reverse SHL="0": normal	72μs
(6) Instruction Table Set	X	0	0	0	0	0	0	1	X	RE1	RE2	RE1,RE2=(0,0), table 1(default) RE1,RE2=(0,1), table 2 RE1,RE2=(1,0), table 3 RE1,RE2=(1,1), table 4	72μs
(14) Cursor Blink and Display Reverser Set	0,1	0	0	0	0	1	0	0	0	REV	B/W	Assign black/write inverting of cursor REV="1": display reverse REV="0": display normal B/W="1": black/write inverting of cursor enable. B/W="0": black/write inverting of cursor disable.	72μs
(15) Display Shift Set	0,1	0	0	0	1	DS6	DS5	DS4	DS3	DS2	DS1	(When DC="1") Determine the line for display shift DS6="1/0": 6th line display shift enable/disable DS5="1/0": 5th line display shift enable/disable DS4="1/0": 4th line display shift enable/disable DS3="1/0": 3rd line display shift enable/disable DS2="1/0": 2nd line display shift enable/disable DS1="1/0": 1st line display shift enable/disable	72μs
(16) Set Data Length for 3-SPI	0,1	0	0	1	SD6	SD5	SD4	SD3	SD2	SD1	SD0	Set data length for 3-line Serial Interface	72μs

表 11. 指令表

KS0192 instruction Table 3 (RE1=1, RE2=0)

Instruction	RE1,RE2	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time
(1) Read display data	X	1	1	Read data								Read data into DDRAM/CGRAM/SEGRAM	72μs
(2) Write display data	X	1	0	Write data								Write data into DDRAM/CGRAM/SEGRAM	72μs
(3) Clear Display	X	0	0	0	0	0	0	0	0	0	1		72μs
(6) Instruction Table Set	X	0	0	0	0	0	0	1	X	RE1	RE2	RE1,RE2=(0,0) , table 1(default) RE1,RE2=(0,1) , table 2 RE1,RE2=(1,0) , table 3 RE1,RE2=(1,1) , table 4	72μs
(17) Set HCGROM Address Byte 1	1,0	0	0	0	0	1	0	AC7	AC6	AC5	AC4	high byte of HCGROM code	72μs
(18) Set HCGROM Address Byte 0	1,0	0	0	0	0	1	1	AC3	AC2	AC1	AC0	low byte of HCGROM code	72μs
(19) Set CGROM Address Byte 3	1,0	0	0	0	1	0	0	AC15	AC14	AC13	AC12	AL15~AL12	72μs
(20) Set CGROM Address Byte 2	1,0	0	0	0	1	0	1	AC11	AC10	AC9	AC8	AL12~AL8	72μs
(21) Set CGROM Address Byte 1	1,0	0	0	0	1	1	0	AC7	AC6	AC5	AC4	AL7~AL4	72μs
(22) Set CGROM Address Byte 0	1,0	0	0	0	1	1	1	AC3	AC2	AC1	AC0	AL3~AL0	72μs
(23) Set CGRAM Address Byte 1	1,0	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	set CGRAM address	72μs

表 12. 指令表



KS0192 instruction Table 4 (RE1=1, RE2=1)

Instruction	RE1,RE2	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time
(1) Read display data	X	1	1	Read data								Read data into DDRAM/CGRAM/SEGRAM	72μs
(2) Write display data	X	1	0	Write data								Write data into DDRAM/CGRAM/SEGRAM	72μs
(3) Clear Display	X	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM, and set DDRAM address to "00H" from AC	72μs
(6) Instruction Table Set	1,1	0	0	0	0	0	0	1	X	RE1	RE2	RE1,RE2=(0,0), table 1(default) RE1,RE2=(0,1), table 2 RE1,RE2=(1,0), table 3 RE1,RE2=(1,1), table 4	72μs
(24) Select DC-DC Step up	1,1	0	0	0	0	1	0	0	DC2	DC1	DC0	Select the step-up of the internal voltage converter 00*:x7 01*:x2 100:x3 101:x4 110:x5 (default) 111:x6	72μs
(25) Select LCD Bias	1,1	0	0	0	0	1	0	1	BS2	BS1	BS0	Select LCD bias 1/4bias ~ 1/11 bias 000:1/4 bias 001:1/5 bias 010: 1/6 bias 011:1/7 bias 100: 1/8 bias 101:1/9 bias (default) 110:1/10 bias 111:1/11 bias	72μs
(26) Select Regulator Resistor	1,1	0	0	0	0	1	1	0	R2	R1	R0	Select Internal resistance ratio of the regulator resistor default: 100	72μs
(27) Select Electronic Volume Register	1,1	0	0	0	1	EC5	EC4	EC3	EC2	EC1	EC0	Set the V0 output voltage electronic volume register default: 100000	72μs
(28) Power Control Set	1,1	0	0	1	0	0	0	SW ON	VR	VB	VF	control power circuit on/off default: OFF	72μs
(29) Frame Frequency Adjustment	1,1	0	0	1	0	0	1	DF3	DF2	DF1	DF0	default:0000, 0001(Slow) -->1111 (Fast)	72μs
(30) Double Frame Frequency	1,1	0	0	1	0	1	1	0	0	0	1	FRR*2=1: frame frequency *2 FRR*2=0: frame frequency normal	72μs

Note:

1. When an MPU program with Busy Flag(DB7) checking is made, 1/2 FOSC ( is necessary ) for executing the next instruction by the " E " signal after the Busy Flag ( DB7) goes to " Low ".

2. " X " Don't care

表 13. 指令表

请详细参考 IC 资料”KS0192.PDF”。

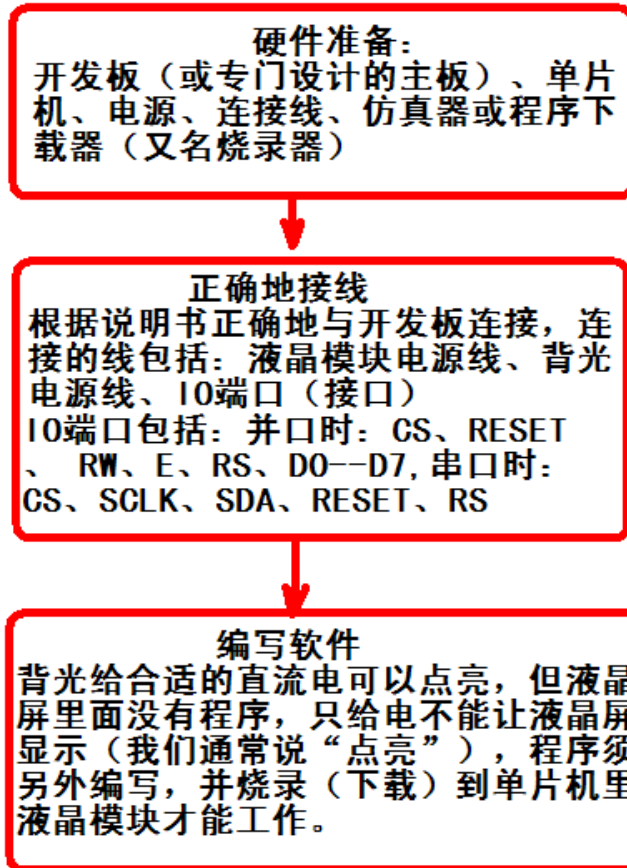
### 7.2 点阵与 DD RAM 地址的对应关系

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 192\*96 点阵的屏分为 12 个“页”，从第 0 “页” 到第 11 “页”。

### 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



### 7.4 程序举例:

#### 7.4.1 串行接口

液晶模块与 MPU (以 8051 系列单片机为例) 接口图如下:

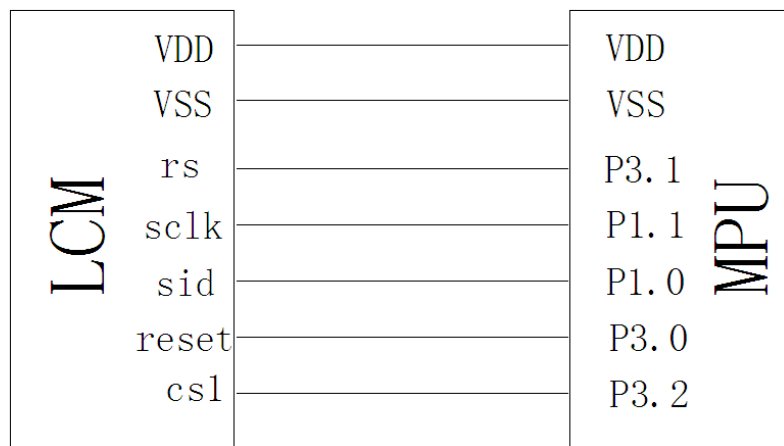


图 8. 串行接口

#### 7.5.2 以下是串行接口例程序

/\* 液晶模块型号: JLX19296G-240-PC-S

驱动 IC 是：  
版权所有：晶联讯电子：网址 <http://www.jlxlcd.cn>;

```
*/  
#include <reg52.H>  
#include <intrins.h>  
#include <chinese_code.h> //找销售人员索要，也可以注释掉相关函数，里面只是一些取模数组  
  
sbit cs1=P3^2;  
sbit reset=P3^0;  
sbit rs=P3^1;  
sbit sclk=P1^1;  
sbit sid=P1^0;  
sbit key=P2^0;  
  
#define uchar unsigned char  
#define uint unsigned int  
  
uint CGRAM_order=0;  
uint CGRAM_ADDRESS=0x80;  
  
/*延时：1 毫秒的 i 倍*/  
void delay(int i)  
{  
    int j,k;  
    for(j=0;j<i;j++)  
        for(k=0;k<110;k++);  
}  
  
/*延时：10us 的 i 倍*/  
void delay_us(int i)  
{  
    int j,k;  
    for(j=0;j<i;j++)  
        for(k=0;k<10;k++);  
}  
  
/*等待一个按键，我的主板是用 P2.0 与 GND 之间接一个按键*/  
void waitkey()  
{  
    repeat:  
        if (key==1) goto repeat;  
        else delay(2000);  
}  
  
//写指令到 LCD 模块  
  
void transfer_command_lcd(int data1)  
{  
    char i;  
    cs1=0;  
    rs=0;  
    for(i=0;i<8;i++)  
    {  
        sclk=0;  
        delay_us(8); //延时很重要  
        if(data1&0x80) sid=1;  
        else sid=0;
```



```

        sclk=1;
        delay_us(8);           //延时很重要
        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;
        // delay_us(1);
        data1=data1<<=1;
    }
    cs1=1;
}

void initial_lcd()
{
    delay(30);
    reset=0;
    delay(30);
    reset=1;
    delay(30);
    delay(30);
    //////////////////////////////////////
    transfer_command_lcd(0x08); //指令表 1
    delay_us(10);
    transfer_command_lcd(0x30); //关显示
    delay_us(50);

    transfer_command_lcd(0x08); //指令表 1
    delay_us(10);
    transfer_command_lcd(0x30); //关显示
    delay_us(10);

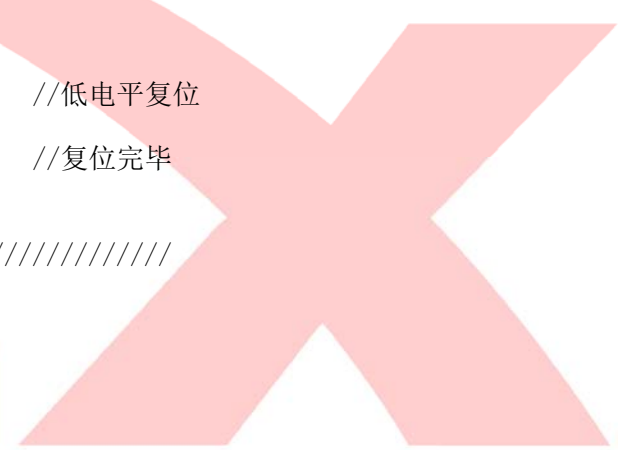
    transfer_command_lcd(0x08); //指令表 1
    delay_us(10);
    transfer_command_lcd(0x30); //关显示
    delay_us(10);

    transfer_command_lcd(0x08); //指令表 1
    delay_us(10);
    transfer_command_lcd(0x01); //清屏
    delay_us(10);
    delay_us(10);
    delay_us(10);
    delay_us(10);
    delay_us(10);

    //////////////////////////////////////
    transfer_command_lcd(0x08); //指令表 1

```

//低电平复位  
//复位完毕



```

delay_us(100);
transfer_command_lcd(0x06); //Entry Mode Set
delay_us(10);
/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x08); //指令表 1
delay_us(10);
transfer_command_lcd(0x80); //Set DDRAM Address
delay_us(10);

/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x09); //指令表 2
delay_us(10);
transfer_command_lcd(0x06); //行列扫描方向
delay_us(10);
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x2D); //1/11 偏压比(Bais) 0x2F
delay_us(10);
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x26); //升压倍压, 5X
delay_us(10);
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x37); //Select Regulator Volume Register
delay_us(10);
transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x4e); //设置 LCD 对比度值 0x62 0x5a
delay_us(10);
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x82); //升压步骤
delay(10);

transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x86); //升压步骤 2
delay(10);

transfer_command_lcd(0x0b); //指令表 4
delay_us(10);
transfer_command_lcd(0x87); //升压步骤 3
delay(10);

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
transfer_command_lcd(0x08); //指令表 1
delay_us(10);
transfer_command_lcd(0x38); //开显示
delay_us(10);
}

```

```

/*清屏*/
void clear_screen()
{
    transfer_command_lcd(0x01);
}

/*调用中文字库里的汉字*/
void display_char1(int y,int x,uchar *dp)
{
    transfer_command_lcd(0x80); //Set DDRAM Address
    transfer_command_lcd(0x10); //Set SEGRAM Address
    transfer_command_lcd(0x80+(y-1)*(0x10)+(x-1));
    while(*dp>1)
    {
        transfer_data_lcd(*dp);
        dp++;
        transfer_data_lcd(*dp);/* 以上两行数据合起来显示一个汉字*/
        dp++;
    }
}

/*写 16x16 自编字体*/
void write_CGRAM_16x16(uchar *dp)
{
    uint i;
    // uint CGRAM_ADDRESS=0x80;
    transfer_command_lcd(0x0a); //表 3
    CGRAM_ADDRESS++;
    transfer_command_lcd(CGRAM_ADDRESS); //Set CGRAM Address
    for(i=0;i<32;i++)
    {
        transfer_data_lcd(*dp);
        dp++;
    }
}

/*写 192x96 自编图*/
void write_CGRAM_192x96(uint *dp)
{
    uint i,j,k,m;
    uint CGRAM_ADDRESS=0x80;
    transfer_command_lcd(0x0a); //指定<指令表 3>
    CGRAM_ADDRESS=0x80;
    for(m=0;m<6;m++)
    {
        for(i=0;i<12;i++)
        {
            transfer_command_lcd(CGRAM_ADDRESS+m*12+i); //指定 CGRAM 地址
            // dp=dp_temp+i*2+m*384;
            for(j=0;j<16;j++)
            {
                for(k=0;k<2;k++)
                {
                    transfer_data_lcd( *(dp+i*2+m*384+k+24*j) );
                }
            }
        }
    }
}

```

```

}
}

//显示自编字体
void Display_CGRAM_192x96(uint *dp)
{
    uint i, j;
    uint CGRAM_order=0;
    write_CGRAM_192x96(dp);
    transfer_command_lcd(0x08); //指定<指令表 1>

    for(j=0; j<6; j++)
    {
        for(i=0; i<12; i++)
        {
            transfer_command_lcd(0x80+0x10*j+i); //指定 DDRAM 地址
            transfer_data_lcd( ((CGRAM_order+0xa0a0)>>8)&0x00ff );//写 DDRAM 数据的高位
            transfer_data_lcd( (CGRAM_order+0xa0a0)&0x00ff );//写 DDRAM 数据的低位
            CGRAM_order++;
        }
    }
}

```

```

//显示自编字体
void Display_CGRAM_16x16(int y, int x, uchar *dp)
{
    //
    uint CGRAM_order=0;
    write_CGRAM_16x16(dp);
    CGRAM_order++;
    transfer_command_lcd(0x08); //指令表 1
    transfer_command_lcd(0x80+(y-1)*(0x10)+(x-1)); //Set CGRAM Address
    transfer_data_lcd( ((CGRAM_order+0xa0a0)>>8)&0x00ff );//H
    transfer_data_lcd( (CGRAM_order+0xa0a0)&0x00ff );//L
}

```

```

void Display_N_CGRAM_16x16()
{
    Display_CGRAM_16x16(1, 1, Number1); //晶
    Display_CGRAM_16x16(1, 2, Number2); //联
    Display_CGRAM_16x16(1, 3, Number3); //讯
    Display_CGRAM_16x16(1, 4, Number4); //液
    Display_CGRAM_16x16(1, 5, Number1); //晶
    Display_CGRAM_16x16(1, 6, Number5); //模
    Display_CGRAM_16x16(1, 7, Number6); //块
    Display_CGRAM_16x16(1, 8, Number7); //自
    Display_CGRAM_16x16(1, 9, Number8); //编
    Display_CGRAM_16x16(1, 10, Number9); //文
    Display_CGRAM_16x16(1, 11, Number10); //字
    Display_CGRAM_16x16(1, 12, Number11); //体

    Display_CGRAM_16x16(2, 1, Number12);
    Display_CGRAM_16x16(2, 2, Number13);
    Display_CGRAM_16x16(2, 3, Number14);
    Display_CGRAM_16x16(2, 4, Number15);
    Display_CGRAM_16x16(2, 5, Number16);
    Display_CGRAM_16x16(2, 6, Number17);
    Display_CGRAM_16x16(2, 7, Number18);
}

```

```
Display_CGRAM_16x16(2, 8, Number19);  
Display_CGRAM_16x16(2, 9, Number20);  
Display_CGRAM_16x16(2, 10, Number21);  
Display_CGRAM_16x16(2, 11, Number22);  
Display_CGRAM_16x16(2, 12, Number23);
```

```
Display_CGRAM_16x16(3, 1, Number24);  
Display_CGRAM_16x16(3, 2, Number25);  
Display_CGRAM_16x16(3, 3, Number26);  
Display_CGRAM_16x16(3, 4, Number27);  
Display_CGRAM_16x16(3, 5, Number28);  
Display_CGRAM_16x16(3, 6, Number29);  
Display_CGRAM_16x16(3, 7, Number30);  
Display_CGRAM_16x16(3, 8, Number31);  
Display_CGRAM_16x16(3, 9, Number32);  
Display_CGRAM_16x16(3, 10, Number33);  
Display_CGRAM_16x16(3, 11, Number34);  
Display_CGRAM_16x16(3, 12, Number35);
```

```
Display_CGRAM_16x16(5, 1, wangluo);  
Display_CGRAM_16x16(5, 3, xinhao);  
Display_CGRAM_16x16(5, 5, wifi);  
Display_CGRAM_16x16(5, 7, laba);  
Display_CGRAM_16x16(5, 9, xinxi);  
Display_CGRAM_16x16(5, 11, dianchi);  
// Display_CGRAM_16x16(5, 7, Number30);  
// Display_CGRAM_16x16(5, 8, Number31);  
// Display_CGRAM_16x16(5, 9, Number32);  
// Display_CGRAM_16x16(5, 10, Number33);  
// Display_CGRAM_16x16(5, 11, Number34);  
// Display_CGRAM_16x16(5, 12, Number35);
```

```
}
```

```
//-----  
void main ()  
{
```

```
    uchar i=0;  
    initial_lcd();
```

//对液晶模块进行初始化设置

```
    while(1)  
    {
```

```
        clear_screen(); //清屏  
        Display_CGRAM_192x96(bmp3);  
        waitkey();  
        clear_screen(); //清屏  
        Display_CGRAM_192x96(bmp2);  
        waitkey();
```

```
        clear_screen(); //清屏  
        Display_CGRAM_192x96(bmp8);  
        waitkey();
```

```
        clear_screen(); //清屏  
        Display_N_CGRAM_16x16();  
        waitkey();
```

```
clear_screen(); //清屏
display_char1(1, 1, "深圳市晶联讯电子有限公司");
display_char1(2, 1, "1. 型号: JLX19296G-240");
display_char1(3, 1, "2. 点阵: 192x96, 12 字 x6 行");
display_char1(4, 1, "3. 视区: 79.5x44.5mm");
display_char1(5, 1, "4. 内置: ASCII 字符及汉字库");
display_char1(6, 1, "5. 可自编生僻字及图形");
waitkey();
```

```
clear_screen(); //清屏
transfer_command_lcd(0x08);
transfer_command_lcd(0x80);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xb0);
    transfer_data_lcd(0xa1+i);
}
```

```
transfer_command_lcd(0x08);
transfer_command_lcd(0x90);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xb0);
    transfer_data_lcd(0xc0+i);
}
```

```
transfer_command_lcd(0x08);
transfer_command_lcd(0xa0);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xb0);
    transfer_data_lcd(0xd0+i);
}
```

```
transfer_command_lcd(0x08);
transfer_command_lcd(0xb0);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xb0);
    transfer_data_lcd(0xe0+i);
}
```

```
transfer_command_lcd(0x08);
transfer_command_lcd(0xc0);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xb0);
    transfer_data_lcd(0xf0+i);
}
```

```
transfer_command_lcd(0x08);
transfer_command_lcd(0xd0);
for(i=0; i<12; i++)
{
    transfer_data_lcd(0xf7);
    transfer_data_lcd(0xf3+i);
}
```

```
waitkey();
```

```
}
}
```

并行接口：

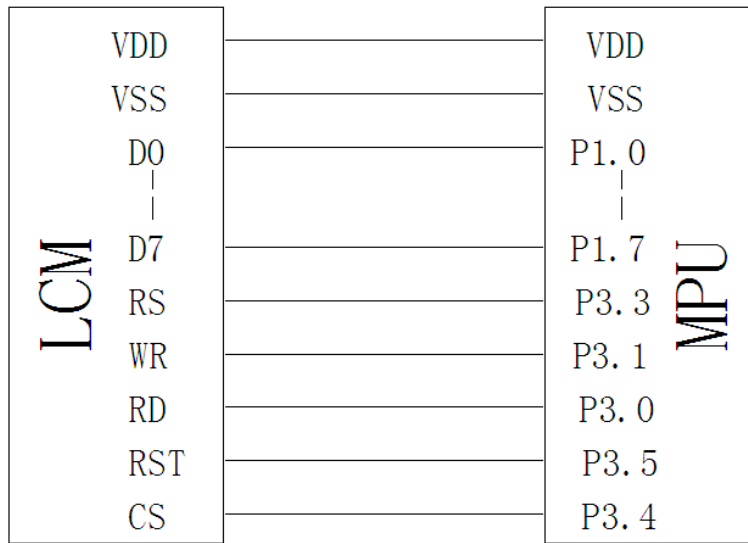


图 9. 并行接口

### 7.5.3、以下为并行接口方式范例程序

与串行方式相比较，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：

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

sbit cs1=P3^4; /*3.4 接口定义*/
sbit reset=P3^5; /*3.3 接口定义*/
sbit rs=P3^3; /*接口定义*/
sbit rd=P3^0; /*接口定义*/
sbit wr=P3^1; /*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*/

//=====transfer command to LCM=====
void transfer_command_lcd(int data1)
{
    cs1=0;
    rs=0;
    rd=0;
    delay_us(1);
    wr=0;
    P1=data1;
    rd=1;
}
```

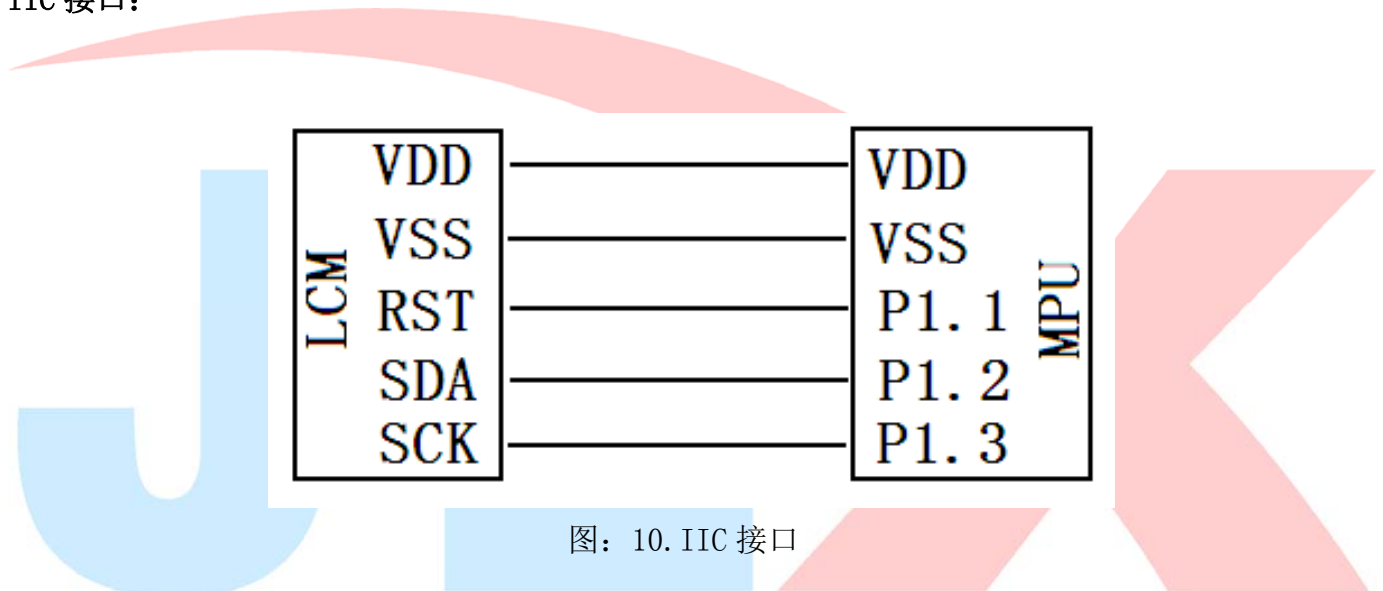
```

    delay_us(200);    //延时很重要
    cs1=1;
    rd=0;
}

//-----transfer data to LCM-----
void transfer_data_lcd(int data1)
{
    cs1=0;
    rs=1;
    rd=0;
    delay_us(1);
    wr=0;
    P1=data1;
    rd=1;
    delay_us(2);
    cs1=1;
    rd=0;
}

```

IIC 接口:



图：10. IIC 接口

#### 7.5.4、以下为 IIC 接口方式范例程序

与串行方式相比较，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：

```

/* 液晶模块型号：JLX19296G-240
   IIC 接口
   驱动 IC 是：KS0192
   版权所有：晶联讯电子；网址 http://www.jlxlcd.cn;
*/
#include <reg52.H>
#include <intrins.h>
#include <chinese_code.h>

sbit reset=P1^1;
sbit scl=P1^3;
sbit sda=P1^2;

```



```
sbit key=P2^0;

#define uchar unsigned char
#define uint unsigned int

//传 8 位指令或数据到 OLED 显示模块
void transfer(uchar data1)
{
    unsigned char j;
    for(j=0;j<8;j++)
    {
        scl=0;
        if(data1&0x80) sda=1;
        else          sda=0;
        data1<<=1;
        scl=1;
        delay_us(20); //延时很重要
    }
    scl=0;
    scl=1;
}

void start_flag()
{
    scl=1; /*START FLAG*/
    sda=1; /*START FLAG*/
    sda=0; /*START FLAG*/
}

void stop_flag()
{
    scl=1; /*STOP FLAG*/
    sda=0; /*STOP FLAG*/
    sda=1; /*STOP FLAG*/
}

//写指令到 LCD 显示模块
void transfer_command_lcd(uchar com)
{
    start_flag();
    transfer(0x78);
    transfer(0x00);
    transfer(com);
    stop_flag();
}

//写数据到 LCD 显示模块
void transfer_data_lcd(uchar dat)
{
    start_flag();
    transfer(0x78);
    transfer(0x40);
    transfer(dat);
    stop_flag();
}
```