JLX16048G-947-PN 使用说明书

目 录

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 JLX16048G-947G-PN 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。

JLX16048-947G-PN 可以显示 160 列*48 行点阵单色图片，或显示 10 个/行*3 行 16*16 点阵的汉字，或显示 20 个/行*6 行 8*8 点阵的英文、数字、符号。

2．JLX16048-947G-PN 图像型点阵液晶模块的特性

2.1 结构轻、薄、带背光、焊接式 FPC、省去购买连接座的成本

2.2 IC 采用 UC1604c,功能强大，稳定性好

2.3 功耗低:当电压为 3.3V 时，功耗低：不带背光 1mW（3.3V*0.3mA）,带背光不大于 100mW（3.3V*30mA）；

2.4 显示内容:

（1）160*48 点阵单色图片，或其它小于 160*48 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 10 字*3 行;

（3）按照 12*12 点阵汉字来计算可显示 13 字*3 行;

（4）按照 8*16 点阵汉字来计算可显示 20 字*3 行;

（5）按照 5*8 点阵汉字来计算可显示 32 字*6 行;

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：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℃;

3. 外形尺寸及接口引脚功能

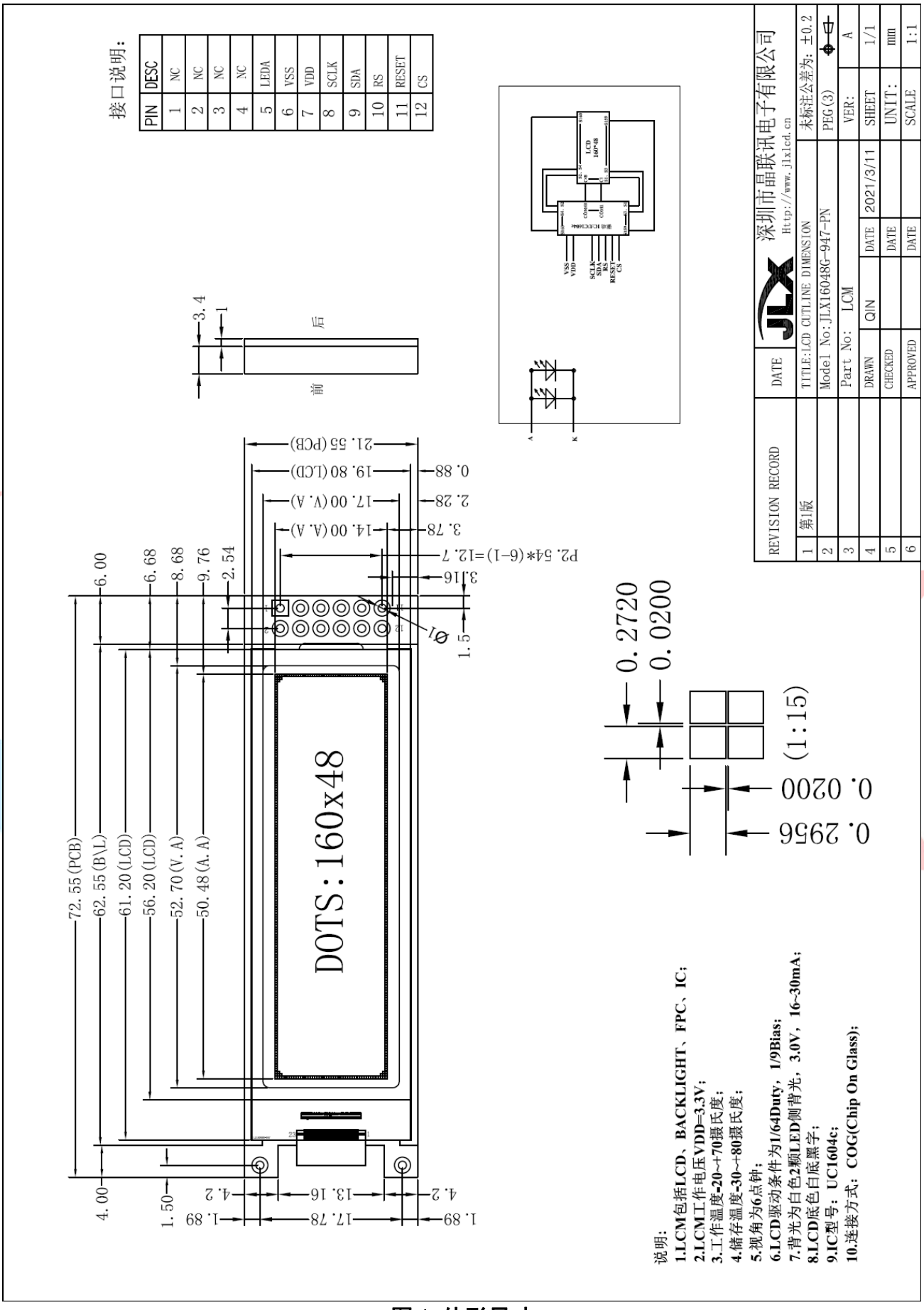


图 1. 外形尺寸

模块的接口引脚功能:

引 线 号	符 号	名 称	功 能
1	ROM_IN	NC	NC
2	ROM_OU	NC	NC
3	ROM_SCK	NC	NC
4	ROM_CS	NC	NC
5	LDEA	背光电源	背光电源正极、同 VDD 电压（5V 或 3.3V）
6	VSS	接地	电源输入地
7	VDD	供电电源正极	供电电源正极
8	SCK	I/O	串行接口时：串行时钟(SCLK)
9	SDA	I/O	串行接口时：串行数据(SDA)
10	A0(RS)	寄存器选择信号	H:数据寄存器 0:指令寄存器
11	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
12	CS0(CS)	片选	低电平片选

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 160×48 点阵,160 个列信号与驱动 IC 相连，48 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：

2 是 JLX16048G-947-PN 图像点阵型模块的电路框图,它由驱动 UC1604c 及几个电阻 电容组成。

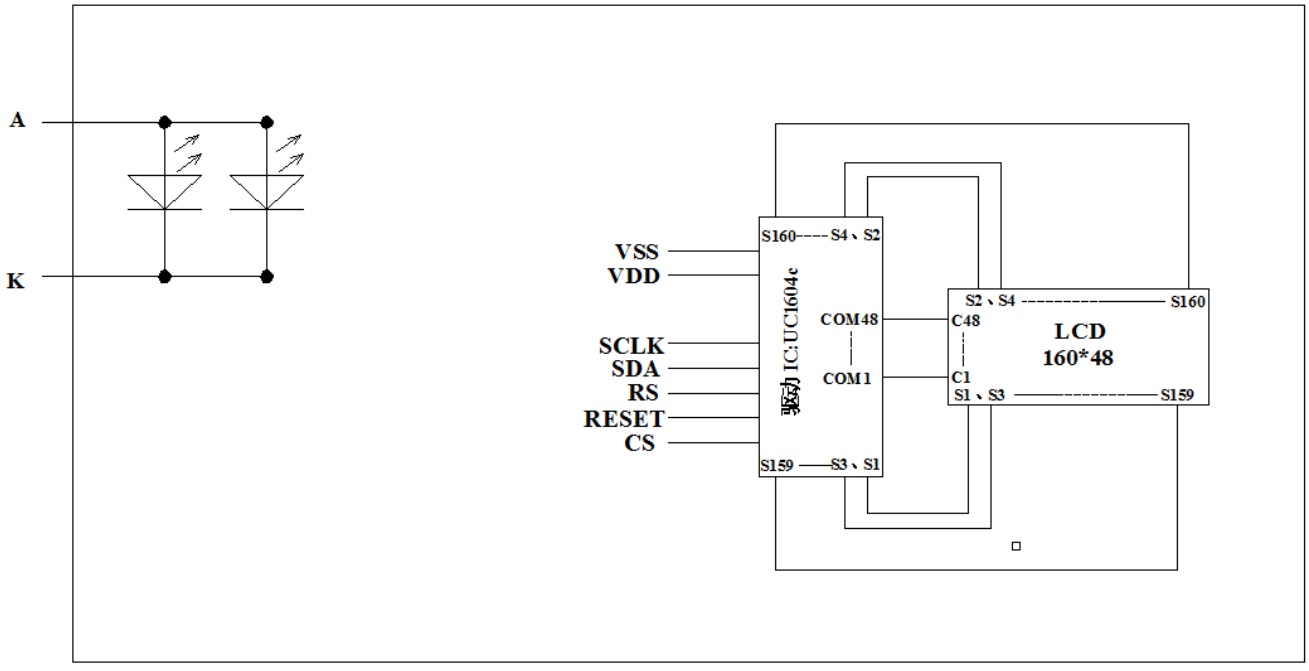


图 2:JLX16048G-947-PN 图像点阵型液晶模块的电路框图

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:
背光板可选择白色。
正常工作电流为：16∽40mA（LED 灯数共 2 颗）;
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		3.6	V
工作温度		-20		+70	℃
储存温度		-30		+80	℃

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	VIH	-	0.8xVDD		VDD	V
输入低电平	VIO	-	VSS		0.6	V
输出高电平	VOH	IOH = 0.2mA	0.8xVDD		VDD	V
输出低电平	VOO	IOO = 1.2mA	VSS		0.2xVDD	V
模块工作电流	IDD	VDD = 3.0V	-		0.3	mA
背光工作电流	ILED	VLED=3.0V	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 UC1604c（Writing Data from CPU to UC1604c）

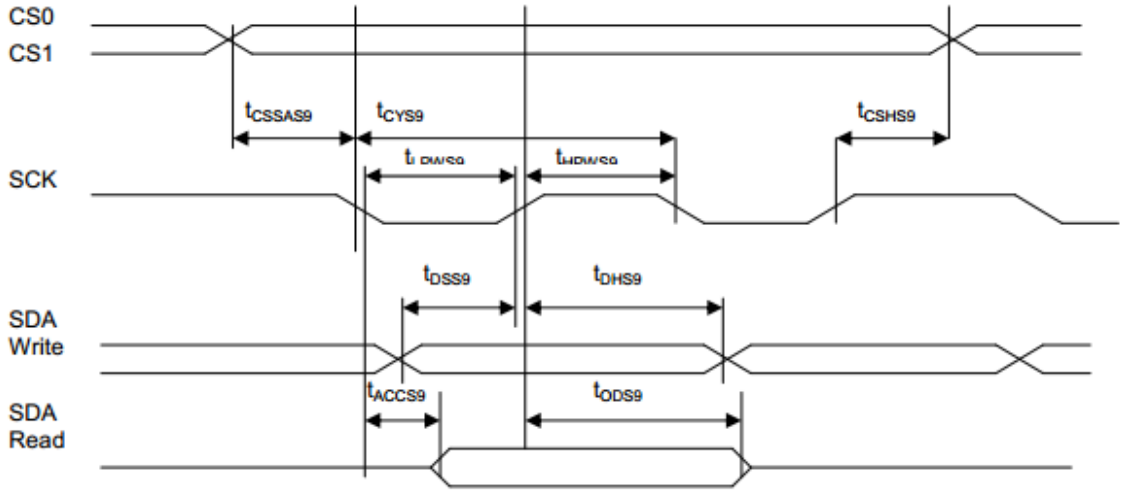


FIGURE 18: Serial Bus Timing Characteristics (for S9)

图 6. 从 CPU 写到 UC1604C（Writing Data from CPU to UC1604C）

6.2 串行接口：时序要求（AC 参数）：

写数据到 UC1604C 的时序要求：（6800 系列 MPU）

Symbol	Signal	Description	Condition	Min.	Max.	Unit
(2.5V ≤ V_{DD} ≤ 3.6V, Ta= –30 to +85°C)				(Read / Write)		
t_{CSSAS9} t_{CSHS9}	CS1, CS0	Chip select setup time Chip select hold time		5 5	–	nS
t_{CYS9} t_{LPWS9} t_{HPWS9}	SCK	System cycle time Low pulse width High pulse width		190 / 70 80 / 20 80 / 20	–	nS
t_{DSS9} t_{DHS9}	SDA (Write)	Data setup time Data hold time		20 10	–	nS
t_{ACC9} t_{OD9}	SDA (Read)	Read access time Output disable time	C_L = 100pF	– –	80 30	nS
(1.7V ≤ V_{DD} < 2.5V, Ta= –30 to +85°C)				(Read / Write)		
t_{CSSAS9} t_{CSHS9}	CS1, CS0	Chip select setup time		10 10	–	nS
t_{CYS9} t_{LPWS9} t_{HPWS9}	SCK	System cycle time Low pulse width High pulse width		230 / 110 100 / 40 100 / 40	–	nS
t_{DSS9} t_{DHS9}	SDA (Write)	Data setup time Data hold time		24 15	–	nS
t_{ACC9} t_{OD9}	SDA (Read)	Read access time Output disable time	C_L = 100pF	– –	100 60	nS

Note: tr (Rising time), tf (falling time) : ≤ 15nS

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

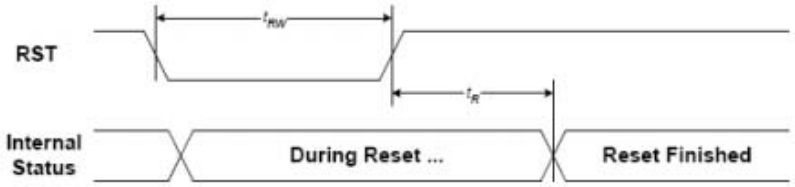


FIGURE 20: Reset Characteristics

(1.7V ≤ V_{DD} ≤ 3.6V, Ta= –30 to +85°C)

Symbol	Signal	Description	Condition	Min.	Max.	Unit
t_{RW}	RST	Reset low pulse width		3	–	μS
t_R	RST, Internal Status	Reset to Internal Status pulse delay		6	–	mS

图 7：**电源启动后复位的时序**

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 0x60
(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，32 行分为 4 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 192 列中的其中一列。比如液晶模块的第 100 列地址十六进制为 0x63，那么此指令由 2 个字节来表达：0x16, 0x03
(4) 列地址低4位设置		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
(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 1	显示列地址增减： 0xA0：常规：列地址从左到右， 0xA1：反转：列地址从右到左

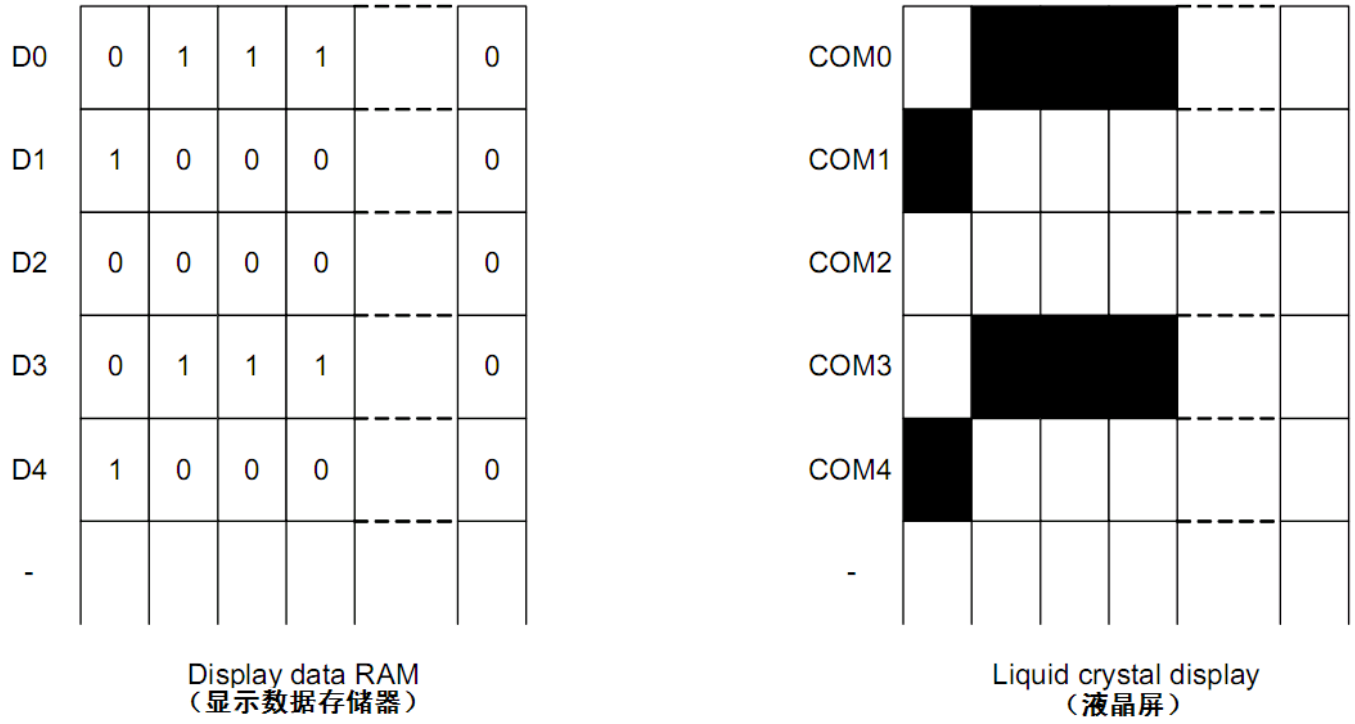
(9)显示正显/反显 (Display normal/reverse)		0	1	0	1	0	0	1	1	0 1	显示正显/反显: **0xA6**：常规：正显 **0xA7**：反显
(10)显示全部点阵 (Display all points)		0	1	0	1	0	0	1	0	0 1	显示全部点阵: **0xA4**：常规 **0xA5**：显示全部点阵
(11)LCD 偏压比设置 (LCD bias set)		0	1	0	1	0	0	0	1	0 1	设置偏压比： **0XA2**：BIAS=1/9 (常用) **0XA3**：BIAS=1/7
(12) 读-改-写 (Read-modify-write)		0	1	1	1	0	0	0	0	0	**0XE0**: “读-改-写” 开始。 列地址的增加： 写入时：列地址+1 读出时：列地址不加 串口时不能用此指令
(13) 退出上述“读-改-写”指令(End)		0	1	1	1	0	1	1	1	0	**0XEE** :上述“读-改-写”指令结束
(14) 软件复位（Reset）		0	1	1	1	0	0	0	1	0	**0XE2** :软件复位。
(15) 行扫描顺序选择 (Common output mode select)			1	1	0	0	0 1	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**
(16) 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。也可以单单写 **0x2F**，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	**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
(18)	内部设置液晶电压模式	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：**0x00～0x3F,**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**						
(19)静态图标显示：开/关		0	1	0	1	0	1	1	0	0 1	静态图标的开关设置： **0xAC**: 关, **0xAD**: 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 (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		

(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 规格书第 35 页“POWER SAVE”
(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！

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 160*48 点阵的屏分为 6 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：



Display data RAM
（显示数据存储器）

Liquid crystal display
（液晶屏）

下图摘自 UC1604C IC 资料，可通过“UC1604c_al.3.pdf”之第 36 页获取最佳效果。

PA[3:0]	0	Line Address															Panel Location	MY=0 SL=0	MY=0 SL=16	MY=1 SL=0	MY=1 SL=16
0000	D0	R0	1	0							Page 0						COM1	R0	R16	R63	R15
	D1	R1	1	0													COM2	R1	R17	R62	R14
	D2	R2	1	1													COM3	R2	R18	R61	R13
	D3	R3	1	1													COM4	R3	R19	R60	R12
	D4	R4	1	0													COM5	R4	R20	R59	R11
	D5	R5	0	0													COM6	R5	R21	R58	R10
	D6	R6	0	1													COM7	R6	R22	R57	R9
	D7	R7	0	1													COM8	R7	R23	R56	R8
0001	D0	R8									Page 1						COM9	R8	R24	R55	R7
	D1	R9															COM10	R9	R25	R54	R6
	D2	R10															COM11	R10	R26	R53	R5
	D3	R11															COM12	R11	R27	R52	R4
	D4	R12															COM13	R12	R28	R51	R3
	D5	R13															COM14	R13	R29	R50	R2
	D6	R14															COM15	R14	R30	R49	R1
	D7	R15															COM16	R15	R31	R48	R0
0010	D0	R16									Page 2						COM17	R16	R32	R47	R63
	D1	R17															COM18	R17	R33	R46	R62
	D2	R18															COM19	R18	R34	R45	R61
	D3	R19															COM20	R19	R35	R44	R60
	D4	R20															COM21	R20	R36	R43	R59
	D5	R21															COM22	R21	R37	R42	R58
	D6	R22															COM23	R22	R38	R41	R57
	D7	R23															COM24	R23	R39	R40	R56
0011	D0	R24									Page 3						COM25	R24	R40	R39	R55
	D1	R25															COM26	R25	R41	R38	R54
	D2	R26															COM27	R26	R42	R37	R53
	D3	R27															COM28	R27	R43	R36	R52
	D4	R28															COM29	R28	R44	R35	R51
	D5	R29															COM30	R29	R45	R34	R50
	D6	R30															COM31	R30	R46	R33	R49
	D7	R31															COM32	R31	R47	R32	R48
0100	D0	R32									Page 4						COM33	R32	R48	R31	R47
	D1	R33															COM34	R33	R49	R30	R46
	D2	R34															COM35	R34	R50	R29	R45
	D3	R35															COM36	R35	R51	R28	R44
	D4	R36															COM37	R36	R52	R27	R43
	D5	R37															COM38	R37	R53	R26	R42
	D6	R38															COM39	R38	R54	R25	R41
	D7	R39															COM40	R39	R55	R24	R40
0101	D0	R40									Page 5						COM41	R40	R56	R23	R39
	D1	R41															COM42	R41	R57	R22	R38
	D2	R42															COM43	R42	R58	R21	R37
	D3	R43															COM44	R43	R59	R20	R36
	D4	R44															COM45	R44	R60	R19	R35
	D5	R45															COM46	R45	R61	R18	R34
	D6	R46															COM47	R46	R62	R17	R33
	D7	R47															COM48	R47	R63	R16	R32
0110	D0	R48									Page 6						COM49	R48	R0	R15	R31
	D1	R49															COM50	R49	R1	R14	R30
	D2	R50															COM51	R50	R2	R13	R29
	D3	R51															COM52	R51	R3	R12	R28
	D4	R52															COM53	R52	R4	R11	R27
	D5	R53															COM54	R53	R5	R10	R26
	D6	R54															COM55	R54	R6	R9	R25
	D7	R55															COM56	R55	R7	R8	R24
0111	D0	R56									Page 7						COM57	R56	R8	R7	R23
	D1	R57															COM58	R57	R9	R6	R22
	D2	R58															COM59	R58	R10	R5	R21
	D3	R59															COM60	R59	R11	R4	R20
	D4	R60															COM61	R60	R12	R3	R19
	D5	R61															COM62	R61	R13	R2	R18
	D6	R62															COM63	R62	R14	R1	R17
	D7	R63															COM64	R63	R15	R0	R16
1000	D0	R64									Page 8						CIC	R64	R64	R64	R64

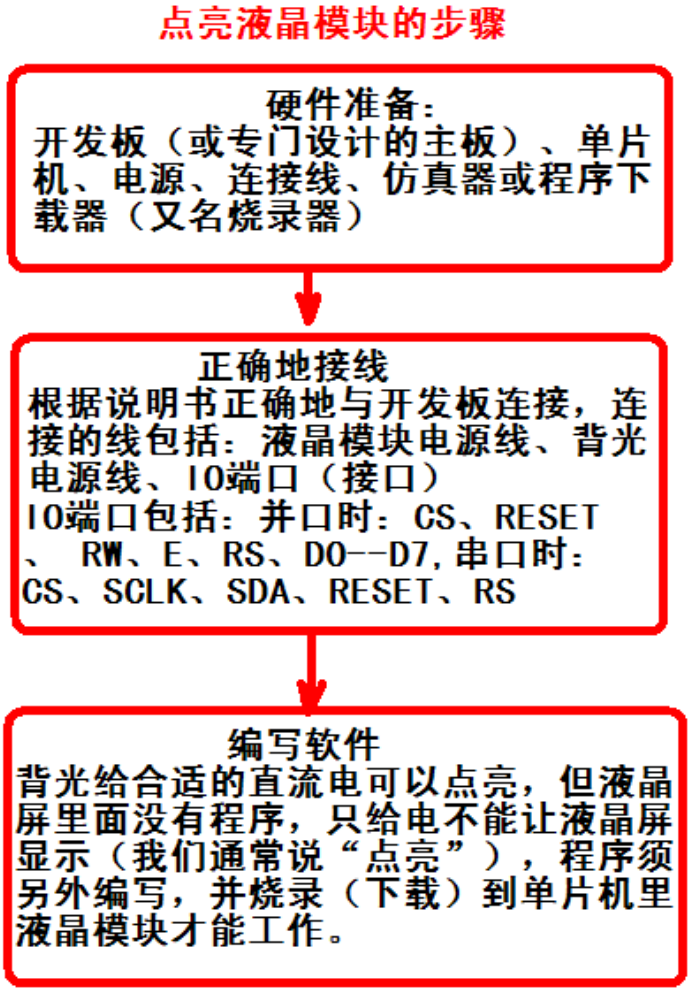
MX=0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8			SEG128	SEG129	SEG130	SEG131	SEG132
MX=1	SEG132	SEG131	SEG130	SEG129	SEG128	SEG127	SEG126	SEG125			SEG5	SEG4	SEG3	SEG2	SEG1

Example for memory mapping: let MX = 0, MY = 0, SL = 0, according to the data shown in the above table:

⇒ Page 0 SEG 1 (D7-D0) : 0001 1111b
⇒ Page 0 SEG 2 (D7-D0) : 1100 1100b

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



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:

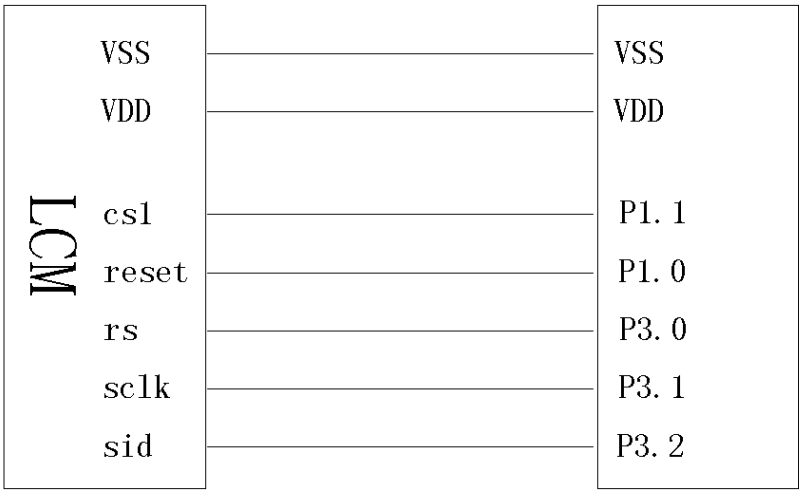


图 8. 串行接口

7.5.1 程序

```
//   驱动 IC 是:UC1604C

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

sbit cs1=P1^1;
sbit reset=P1^0;
sbit rs=P3^0;
sbit sclk=P3^2;
sbit sid=P3^1;
sbit key=P2^0;

void delay_us(int i);
void delay(int i);


//写指令到 LCD 模块
void transfer_command(int data1)
{
    char i;
    cs1=0;
    rs=0;
    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;
}

//写数据到 LCD 模块
void transfer_data(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;
}

//延时 1
void delay(int i)
{
 int j,k;
 for(j=0;j<i;j++)
 for(k=0;k<110;k++);
}

//延时 2
void delay_us(int i)
{
 int j,k;
 for(j=0;j<i;j++)
 for(k=0;k<10;k++);
}

void waitkey()
{
repeat:
    if(key==1)goto repeat;
    else delay(2200);
}

//LCD 模块初始化
//========设定微调对比度值 0x56==========
void initial_lcd()
{
    reset=0;        //低电平复位
    delay(250);
    reset=1;            //复位完毕
    delay(800);
    transfer_command(0xe2);  //软复位
    delay(500);
    transfer_command(0x2f);  //打开内部升压
```

```
    delay(500);
    transfer_command(0x81);  //微调对比度
    transfer_command(0x56);  //微调对比度的值，可设置范围 0x00～0xFF
    transfer_command(0xeb);  //1/9 偏压比（bias）
    transfer_command(0xa0);  //
    transfer_command(0xc4);  //行列扫描顺序：从上到下
    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
}

//全屏清屏
void clear_screen()
{
    unsigned char i,j;
    for(i=0;i<8;i++)
    {
        lcd_address(1+i,1);
        for(j=0;j<192;j++)
        {
            transfer_data(0x00);
        }
    }
}

void display_graphic_160x48(uchar *dp)
{
    uchar i,j;
    for(i=0;i<6;i++)
    {
        lcd_address(i+1,1);
        for(j=0;j<160;j++)
        {
            transfer_data(*dp);
            dp++;
        }
```

```
    }
}

//==================display a piture of 160*48 dots================
void full_display(uchar data_left,uchar data_right)
{
    int i,j;
    for(i=0;i<8;i++)
    {
        lcd_address(i+1,1);
        for(j=0;j<96;j++)
        {
            transfer_data(data_left);
            transfer_data(data_right);
        }
    }
}

//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++;
        }
    }
```

```
}

//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 reverse, uchar *text)
{
    uchar i,j,k,data1;
    uint address;
    j = 0;
    while(text[j] != '\0')
    {
        i=0;
        address=1;
        while(Chinese_text_16x16[i]> 0x7e )
        {
            if(Chinese_text_16x16[i] == text[j])
            {
                if(Chinese_text_16x16[i+1] == text[j+1])
                {
                    address = i*16;
                    break;
                }
            }
            i +=2;
        }
        if(column>191)
        {
            column =0;
            page +=2;
        }
        if(address !=1)
        {
            for(k=0;k<2;k++)
            {
                lcd_address(page+k,column);
                for(i=0;i<16;i++)
                {
                    if(reverse==1) data1=~Chinese_code_16x16[address];
                    else data1=Chinese_code_16x16[address];
                    transfer_data(data1);
                    address++;
                }
            }
            j +=2;
        }
        else
```

```
        {
            for(k=0;k<2;k++)
            {
                lcd_address(page+k,column);
                for(i=0;i<16;i++)
                {
                    if(reverse==0) transfer_data(0x00);
                    else transfer_data(0xff);
                }
            }
            j++;
        }
        column +=16;
    }
}
```

```
//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_2(uchar page,uchar column,uchar reverse, uchar *text)
{
    uchar data1;
    uint i=0,j,k,n;

    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);
                }
                if(reverse==0)  transfer_data(0x00);
                else    transfer_data(0xff);
            }
            i++;
            column+=8;
        }
        else
        i++;

        if(column>127)
        {
            column=0;
            page+=2;
        }
    }
}



void display_string_8x16(uint page,uint column,uchar *text)
{
    uint i=0,j,k,n;
    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]);//显示 5x7 的 ASCII 字到 LCD 上，y 为页地址，x 为列地址，最后为数据
                }
            }
            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++;
    }
}

void display_string_5x8_1(uint page,uint column,uchar *text)
{
    uint i=0,j,k;
    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++)
            {
                transfer_data(ascii_table_5x8[j][k]);//显示 5x7 的 ASCII 字到 LCD 上，y 为页地址，x 为
```

```
列地址，最后为数据
            }
            i++;
            column+=6;
        }
        else
        i++;
    }
}



void main(void)
{
    while(1)
    {
        initial_lcd();
        clear_screen();
        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(3,68,1,"1.Graphic       ");
        display_string_5x8(4,68,0,"2.Chinese       ");
        display_string_5x8(5,68,0,"3.Movie         ");
        display_string_5x8(6,68,0,"4.Contrast      ");
        waitkey();
        clear_screen();                          //clear all dots
        display_graphic_160x48(bmp2);
        waitkey();
        clear_screen();                          //clear all dots
        display_graphic_160x48(bmp1);
        waitkey();

        full_display(0xff,0xff);
        waitkey();
        full_display(0x55,0xaa);
        waitkey();
        full_display(0xaa,0x55);
        waitkey();
        full_display(0xff,0x00);
        waitkey();
        full_display(0x00,0xff);
        waitkey();

        clear_screen();                                  //clear all dots
        display_string_8x16(1,1,"UART is not connected");//在第 1 页，第 1 列显示字符串
```

```
        waitkey();
        clear_screen();                                  //clear all dots
        display_string_8x16(4,1,"Board test Box Open");//在第 1 页，第 1 列显示字符串
        waitkey();

        clear_screen();
        display_string_8x16_2(1,1,1,"--");
        display_string_16x16(1,17,1,"→粉尘测试      ");
        display_string_16x16(3,33,0,"一般测试");
        display_string_16x16(5,33,0,"校准模式");
        display_string_16x16(7,33,0,"充电模式 ↓");
        waitkey();
        clear_screen();
        display_graphic_32x32(1,49,cheng1);              //在第 1 页，第 49 列显示单个汉字"成"
        display_graphic_32x32(1,89,gon);               //在第 1 页，第 49 列显示单个汉字"成"
        waitkey();
        clear_screen();                                  //clear all dots
        display_string_8x16(1,1,"(<\"0123456abt~`!@#$%^\">)");//在第 1 页，第 1 列显示字符串
        display_string_8x16(3,1,"{[(<\" '&*|\\@#_-+=' \">)]}");//在第*页，第*列显示字符串
        display_string_5x8_1(5,1,"[!#$%&'()*+,-./0123456789:;<=>?]");
        display_string_5x8_1(6,1,"[ABCDEFGHIJKLMNOPQRSTUVWXYZabcd]");
        display_string_5x8_1(7,1,"(abcdefghijklmnopqrstuvwxyzabcd)");
        display_string_5x8_1(8,1,"{[(<\" '&*|\\@abcde012#_-+=' \">)]}");
        waitkey();

    }
}
```