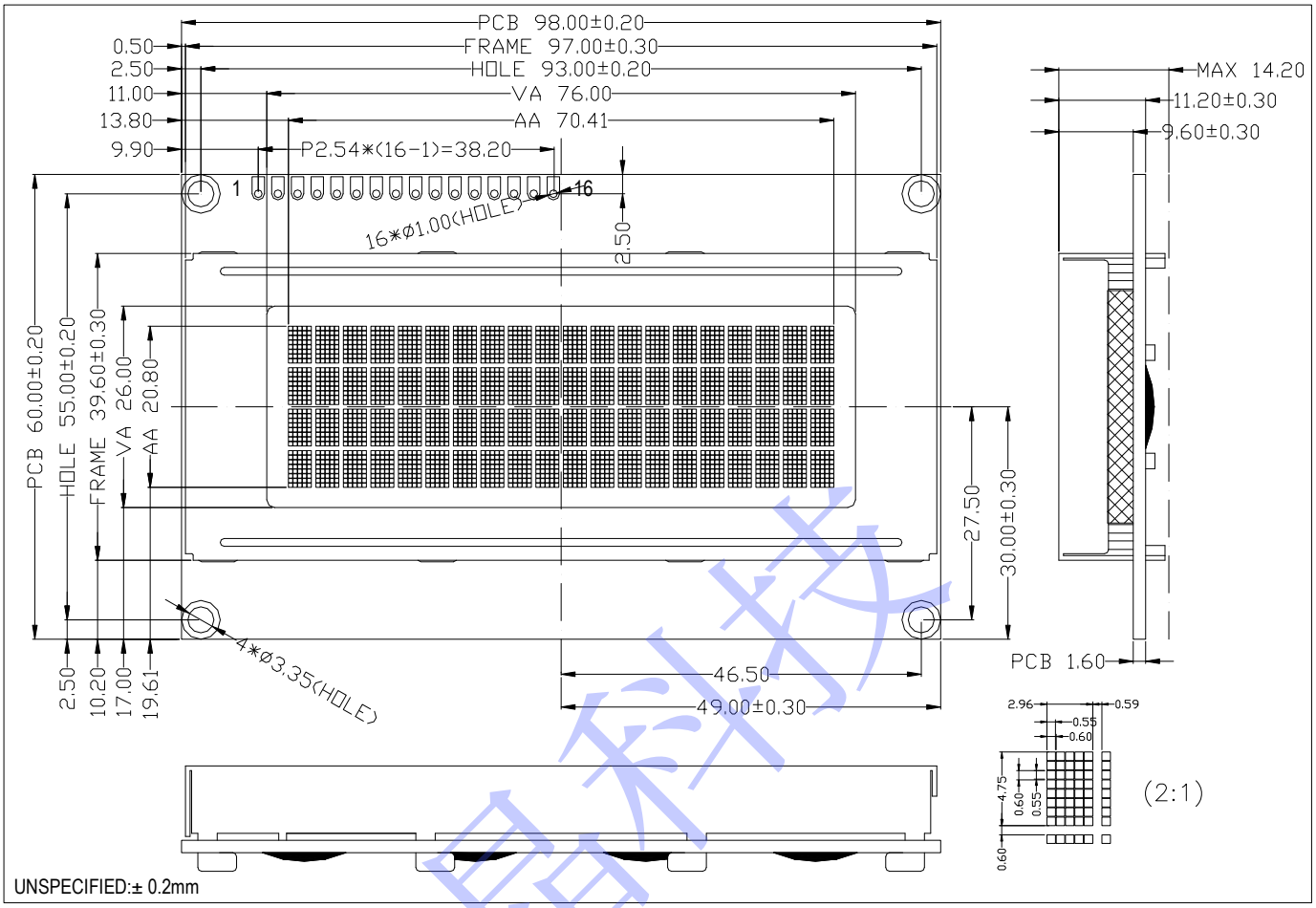
一、模块尺寸



项目	参考值（mm）
LCM 尺寸（长×宽×厚）	98.0×60.0×14.2(MAX)
可视区域（长×宽）	76.0×26.0
点间距（长×宽）	0.60×0.60
点尺寸（长×宽）	0.55×0.55
字符大小（长×宽）	2.96×4.75
字符间距（长×宽）	3.55×5.35

二、功能介绍

- ✧ 工作电压 3.3V 或者 5V，只能选择一种电压使用，下单请注明
- ✧ 通讯方式：4 位或者 8 位并行通讯接口
- ✧ 字符显示器内存
  - ■ 字符发生器 ROM（CGROM）：10880 位（192 个 5*8 点阵格式字符和 64 个 5*11 点阵格式字符）
  - ■ 字符发生器 RAM（CGRAM）：64*8 位（5 个 5*8 点阵格式字符或 4 个 5*11 点阵格式字符，自编特殊字符）
  - ■ 显示数据 RAM：80*8 位*2 行（可以 16*2 个字符）
- ✧ ASCII 字符图案
  - ■ 5X8 点 ASCII 字符图案 192 个
  - ■ 5X11 点 ASCII 字符图案 64 个
  - ■ 特殊字符图案可以对字符发生器 RAM 直接编程得到
  - ■ 用户需要其它“通用字符”图案的，我们可以订制
- ✧ 功能指令
  - ■ 光标闪烁　　显示开关
  - ■ 字符左右移动
- ✧ 低功率省电设计(除背光 30MA)
  - ■ 正常模式（550uA　typ　VDD=5V)
  - ■ 待机模式（30uA　max　VDD=5V)
- ✧ 自动上电复位功能
- ✧ 占空比 1/16　　偏压比 1/5
- ✧ 工作温度-20℃---+70℃
- ✧ 储存温度-30℃---+80℃
- ✧ 可视角度 6 点
- ✧ 显示效果有黄底黑字，蓝底白字，白底黑字可选，下单请注明

三. 接口定义

引脚	名称	方向	说明
1	VSS	--	电源负端(0V)
2	VDD	--	电源正端(+3.3V 或+5.0V，出厂时设定+5.0V)
3	V0	--	LCD 电压调节电压　当接地时，LCD 显示最深
4	RS	I	=1，写数据
			=0，写指令
5	RW	I	=0，写模式
			=1，读模式
6	E	I	使能信号，高电平有效。
7-14	DB0 ~ DB7	I/O	单片机与模块之间并口的数据传送通道， 当用 4 位时，DB0-DB3 不用 DB7 能用作忙标志读出（判忙）
15	LEDA	I	背光电源的正极（3.3V 或者 5V）出厂 5V
16	LEDK	O	背光电源负极

四．电性参数(直流）

名称	符号	测试条件	参数范围			单位
			最小	标准	最大	
模块工作电压	VDD	-	2.4	5.0	5.5	V
玻璃电压	V0	V0-VDD	4.5	5.0	7.0	V
背光工作电压	VLED	-	2.8	5.0	5.5	V
IO 输入高电平	VIH	-	2.2	-	VDD	V
IO 输入低电平	VIL	-	-0.3	-	0.6	V
LCM 输出高电平	VOH	-	2.4	-	-	V
LCM 输出低电平	VOL	-	-	-	0.4	V
模块工作电流	IDD	=VDD	-	-	0.5	MA
模块待机电流	IDO	=VDD	-	-	10	uA
背光工作电流	ILED	=VLED	8	15	20	MA

五．显示原理

a) DDRAM

① 第一三行 00H-27H；第二四行 40H-67H，如下图（DDRAM 与显示屏的关系）

② 移位范围为第一行 00H-27H；第二行 40H-67H，如下图（左移，右移）移位范围取决于模块的占空比，本模块占空比为 1/16

③ 显示原理为，写入地址，然后写入字符码，就能在显示器上显示你需要的字符，例如地址写 87H，字符码写 48H ，在第一行第 7 个字符显示“H”

④ 字符码是标准的 ASCII 编码，可以直接引用字符，程序能翻译出字符内码，你要显示什么字符，直接在源代码内写字符，例如写地址 80H，写显示字符 table[]=“12345678ABCDEF”,地址有自动加 1 的特点，这样在显示屏的第-行就显示 12345678ABCDEF

正常（DDRAM与显示屏的关系）20x4

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53
14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27
54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67

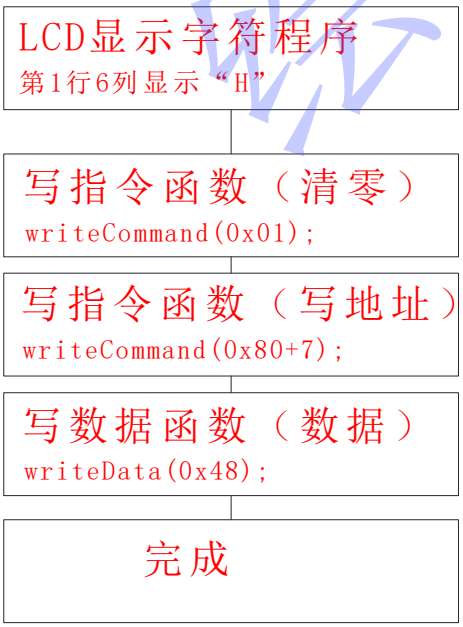
左移

01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14
41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54
15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27	00
55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67	40

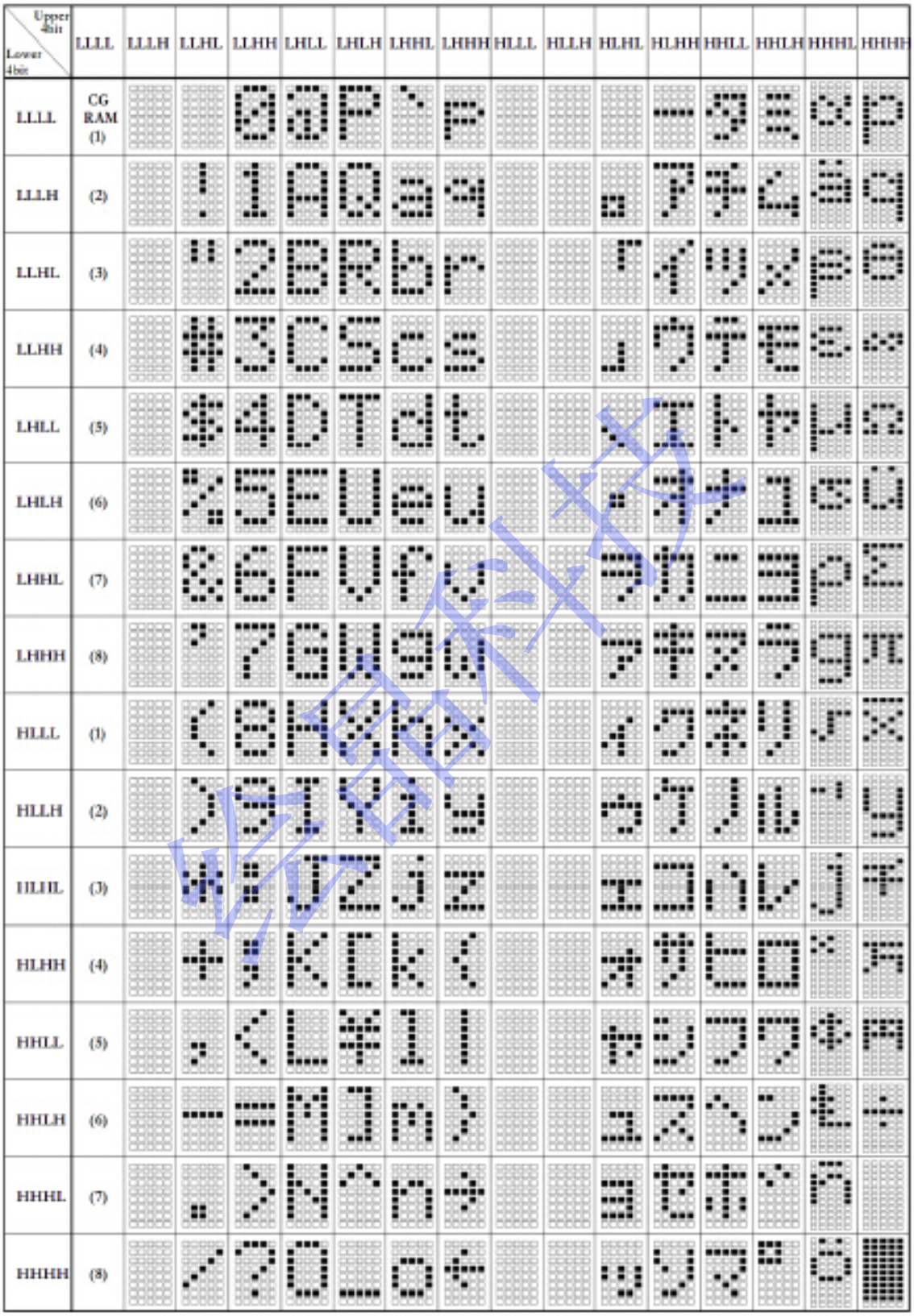
右移

27	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
67	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52
13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26
53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66

写显示数据流程



b) 字符内码查询表


Upper 4bit
Lower 4bit
LLLL LLLH LLHL LLHH LHLL LHLH LHHL LHHH HLLL HLLH HLHL HLHH HHLL HHLH HHHL HHHH
LLLL CG RAM (1)
LLLH (2)
LLHL (3)
LLHH (4)
LHLL (5)
LHLH (6)
LHHL (7)
LHHH (8)
HLLL (1)
HLLH (2)
HLHL (3)
HLHH (4)
HHLL (5)
HHLH (6)
HHHL (7)
HHHH (8)

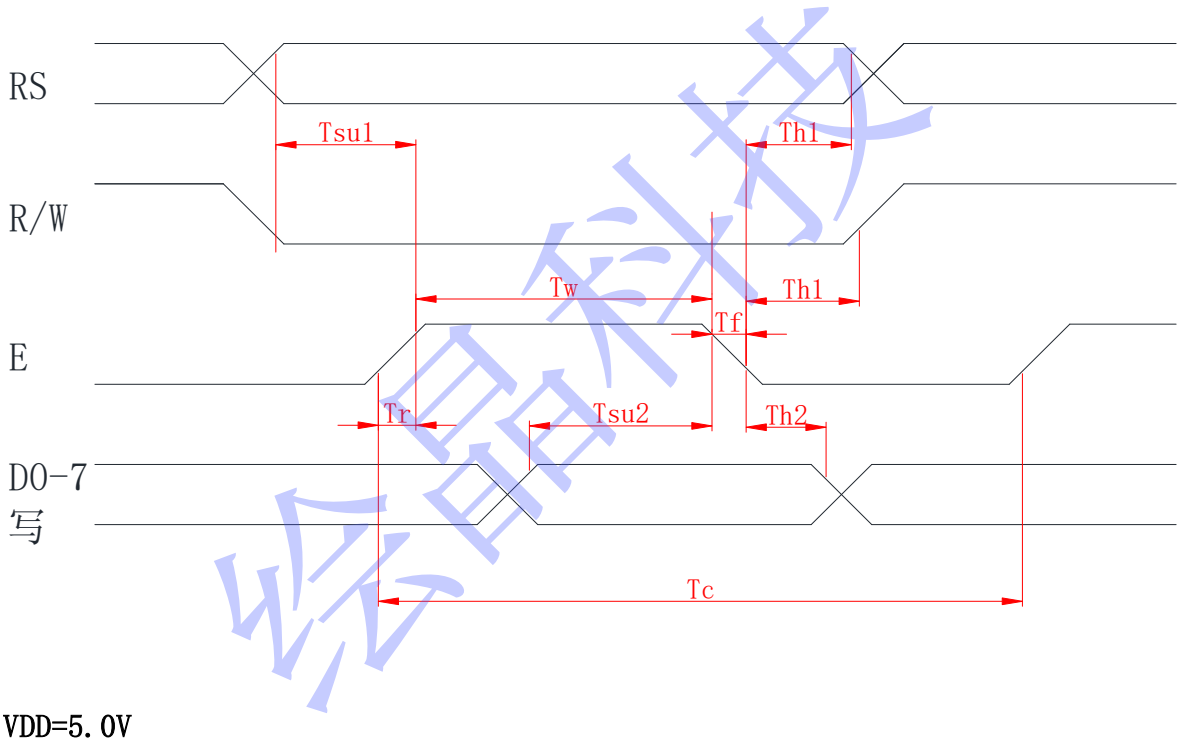
c) CGRAM

自编字符 CGRAM

模块预留了几个自编字符空间，编写一些特殊的字符或者符号，使用原理，进入 CGRAM （ 40H）；选择内码地址 02（00 到 08）；然后写入 8*8 点阵数据（横向取模），要显示自编字符时，写显示 DDRAM 地址，然后写入内码地址 02 就能显示自编字符了。

六．时序图

写数据时序



VDD=5.0V

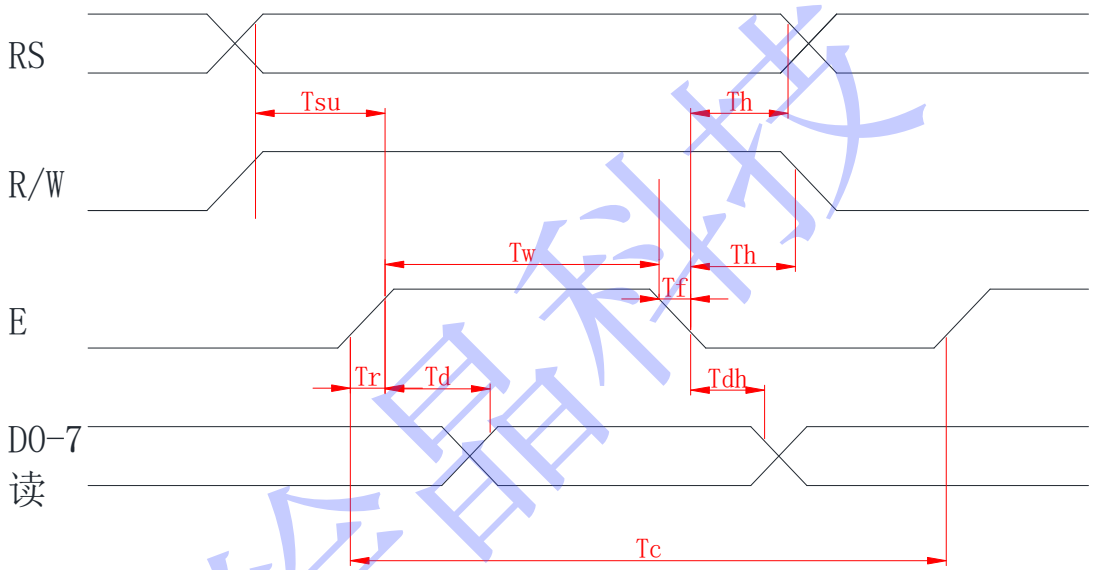
项目	信号	标识	条件	最小	最大	单位
E周期	E	Tc	写模式	500	-	纳秒
E上升/下降时间		Tr,Tf		-	20	
E脉冲宽度（1,0）		Tw		230	-	
R/W和RS建立时间	R/W（RS	Tsu1		40	-	
R/W和RS保持时间		Th1		10	-	
数据建立时间	DB[0-7]	Tsu2		80	-	
数据保持时间		Th2		10	-	

VDD=3.3V

项目	信号	标识	条件	最小	最大	单位
E周期	E	Tc	写模式	1000	-	纳秒

E上升/下降时间		Tr,Tf		-	25	
E脉冲宽度（1,0）		Tw		450	-	
R/W和RS建立时间	R/W（RS	Tsu1		60	-	
R/W和RS保持时间		Th1		20	-	
数据建立时间	DB[0-7]	Tsu2		195	-	
数据保持时间		Th2		10	-	

读数据时序



VDD=5.0V

项目	信号	标识	条件	最小	最大	单位
E周期	E	Tc	读模式	500	-	纳秒
E上升/下降时间		Tr,Tf		-	20	
E脉冲宽度（1,0）		Tw		230	-	
R/W和RS建立时间	R/W（RS	Tsu		40	-	
R/W和RS保持时间		Th		10	-	
数据建立时间	DB[0-7]	Td		-	120	
数据保持时间		Tdh		10	-	

VDD=3.3V

项目	信号	标识	条件	最小	最大	单位
E周期	E	Tc	读模式	1000	-	纳秒
E上升/下降时间		Tr,Tf		-	25	

E脉冲宽度（1,0）		Tw		450	-	
R/W和RS建立时间	R/W（RS	Tsu		60	-	
R/W和RS保持时间		Th		20	-	
数据建立时间	DB[0-7]	Td		-	360	
数据保持时间		Tdh		5	-	

七、用户指令集说明：

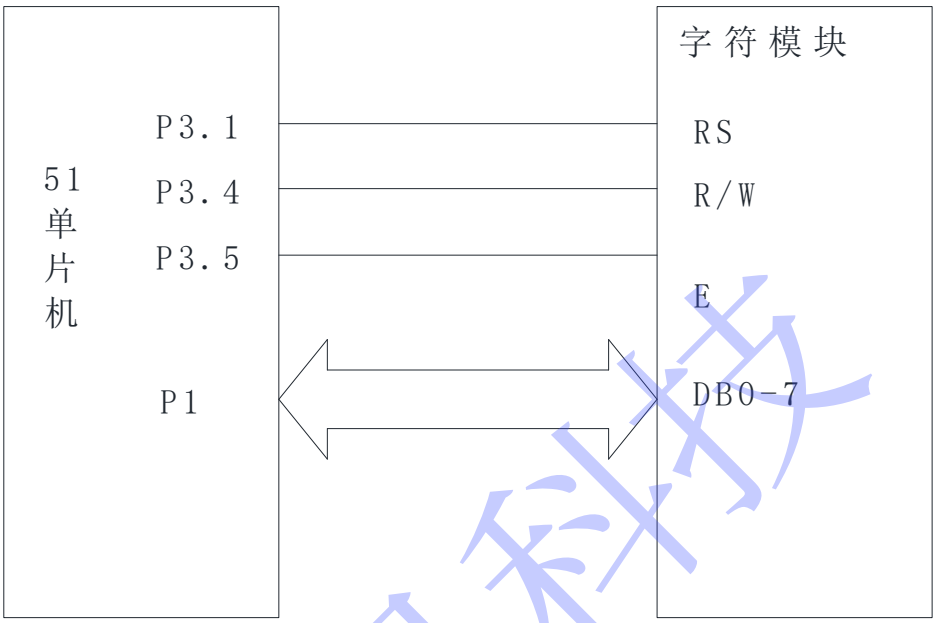
1、指令表：

NO	指令	指令码										HEX	执行时间	说明
		RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0			
1	清除显示	0	0	0	0	0	0	0	0	0	1	01	1.53ms	将20H（空格代码）写入DDRAM，将地址计数器中的地址00H设置为DDRAM地址（可以清除数据，光标初始化，输出模式为递增
2	返回	0	0	0	0	0	0	0	0	1	0	02	1.53ms	将地址计数器中的地址00H设置为DDRAM地址，并将光标恢复至初始位置DDRAM的内容保持不变
3	输入模式设置	0	0	0	0	0	0	0	1	I/D	SH	04	39US	设置光标移动方向，并允许整个显示移动 I/D=1；光标闪烁右移，地址自增 I/D=0；光标闪烁左移，地址自减 SH=1，允许移动， SH=0，不允许移动
4	显示开关	0	0	0	0	0	0	1	D	C	B	08	39US	设置显示，光标，光标的闪烁控制位 D=1；显示开，D=0；显示关 C=1；光标开，C=0；光标关 D=1；光标闪，D=0；光标闪关
5	移位	0	0	0	0	0	1	S/C	R/L	-	-	10	39US	设置光标移动，显示移动方向的控制位，DDRAM数据保持不变 SL， R/R 操作 0，0光标向左移，地址自减1 0，1光标向右移，地址自增1 1，0所有显示左移，光标跟随移位 1，1所有显示右移，光标跟随移位
6	功能设置	0	0	0	0	1	DL	N	F	-	-	20	39US	设置接口数据长度 DL=1；8位并口， DL=0；4位并口 显示行数 N=1； 2行 ， N=0；1行 显示字符 F=1；5*11 ， F=0 ；5*8点阵
7	设置CGRAM地址	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	40	39US	在地址计数器内设置CGRAM地址
8	设置DDRAM地址	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	80	39US	在地址计数器内设置DDRAM地址
9	读忙标志	0	0	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0		0US	判忙（BF=1为忙，BF=0不忙，可以接受指令）
10	写数据	1	0	D7	D6	D5	D4	D3	D2	D1	D0	00	43S	写数据

11	读数据	1	1	D7	D6	D5	D4	D3	D2	D1	D0	00	43S	读数据

“-”不考虑

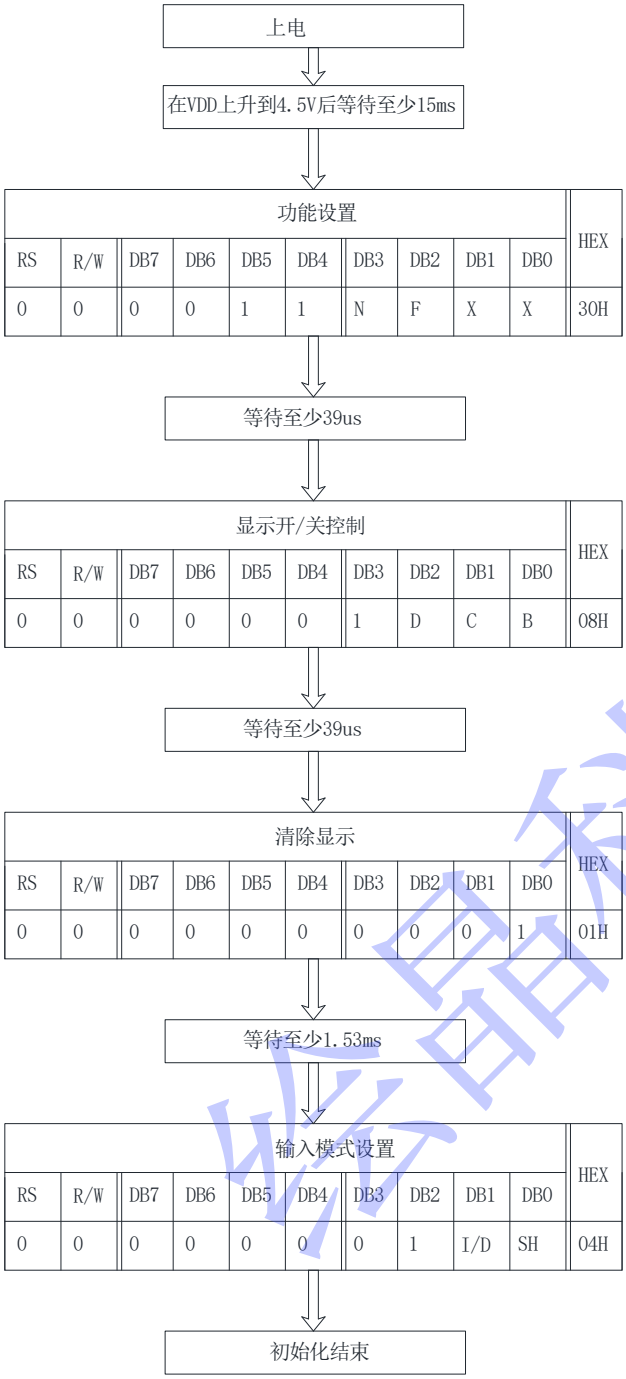
八、单片机接线图



九、初始化程序

a) 在写程序之前都要先对模块的硬件软件初始化，

8位接口模式



N	0	1行显示模式
	1	2行显示模式

F	0	显示字符5*8
	1	显示字符5*11

D	0	显示关
	1	显示开

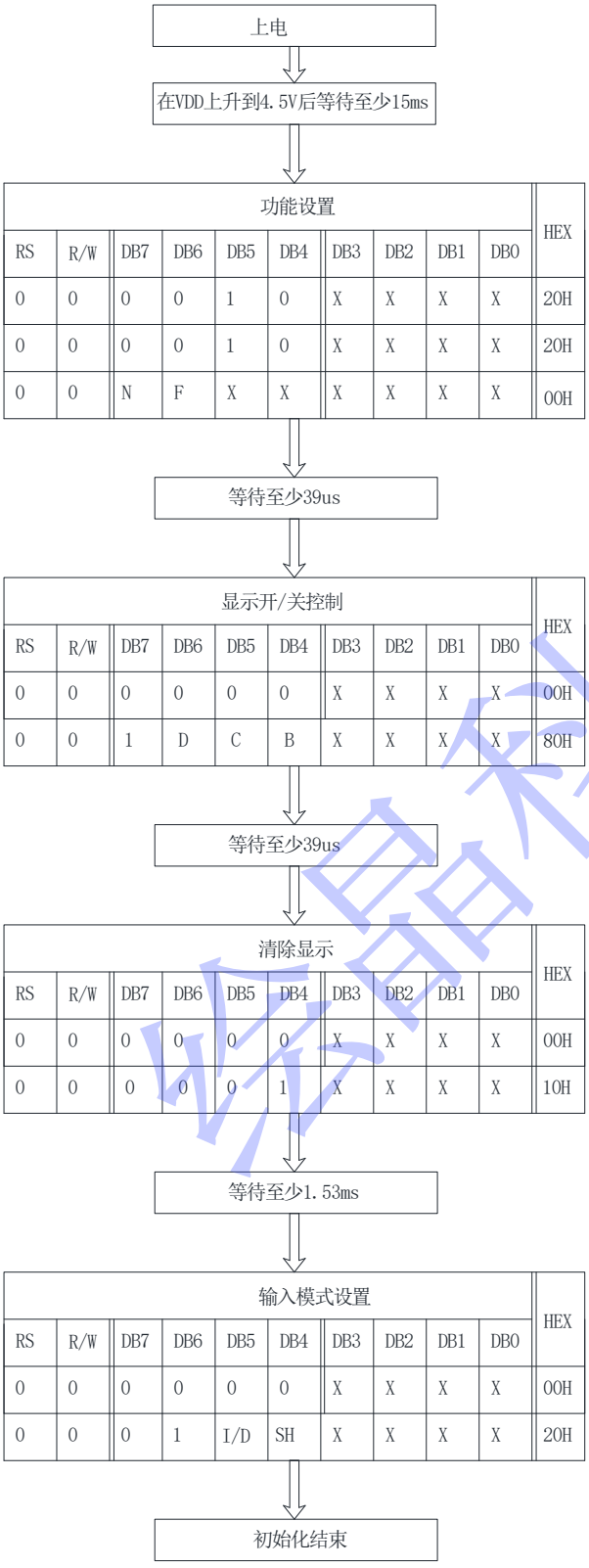
C	0	光标关
	1	光标开

D	0	闪烁关
	1	闪烁开

I/D	0	减少
	1	增加

C	0	显示不移位
	1	显示移位

4位接口模式



N	0	1行显示模式
	1	2行显示模式

F	0	显示字符5*8
	1	显示字符5*11

D	0	显示关
	1	显示开

C	0	光标关
	1	光标开

D	0	闪烁关
	1	闪烁开

I/D	0	减少
	1	增加

C	0	显示不移位
	1	显示移位

九、程序例程

```
//单片机:ATMEL:89S52, STC单片机完全兼容
//204
#include<reg52.h>
#include <intrins.h>
sbit RS=P3^1;
sbit RW=P3^4;
sbit E=P3^5;
sbit stop=P3^2;
//DB7-DB0 : P1

typedef unsigned int uint;
typedef unsigned char uchar;
unsigned char command,LCDdata,j;
unsigned int m,i,k;
uchar num,ii,z,z1,d,d1,s,s1,s10,s100;

void WaitNms(unsigned int x)//延时 x ms
{
unsigned char j;
while(x--)
{for(j=0;j<125;j++)
{;}
 }
}

void WaitNus(unsigned int x)//延时 x us
{
unsigned char j;
while(x--)
{for(j=0;j<12;j++)
{;}
 }
}

 void INTI () interrupt 0 using 1

{
 for(;;);
}

void BUSYFLAG(void)
{
```

```
uchar temp;
P1=0xff;
RS=0;
RW=1;
    while(1)
  {
    E=1;
    temp=P1;                        //读状态字
    E=0;
    if ((temp&0x80)==0)
    break;                      //判断忙标志是否为0
  }
}

void writeCommand(uchar command)
{
BUSYFLAG();
RS=0;
RW=0;

E=1;
P1=command;
E=0;
}
void writeData(uchar DATA)
{
BUSYFLAG();
RS=1;
RW=0;

E=1;
P1=DATA;
E=0;
}

LCDINT(void)
{
WaitNms(15);////延时 x ms
writeCommand(0x30);//8位
WaitNms(4);////延时 x ms
writeCommand(0x30);
WaitNus(100);////延时 x us
writeCommand(0x30);
```

```
writeCommand(0x38);//两行显示模式
writeCommand(0x01);//清屏
writeCommand(0x06);//画面不动
writeCommand(0x0c);//光标设置
writeCommand(0x80);//显示首址
}

void storeCGRAM(uchar d1,uchar d2)
    {
        for(i=0;i<4;i++)
             {
              writeData(d1);
              writeData(d2);
             }
     }


void writetest(uchar dd)
    {
        for(i=0;i<80;i++)
          {
         writeData(dd);
          }
    }

void writeALLcharacter(void)
{
for(i=0x20;i<(0x20+80);i++)//写ASCLL码
  {
 writeData(i);
  }
 WaitNms(500);////延时 x ms
writeCommand(0x01);//清屏
for(i=0xA0;i<(0xA0+80);i++)//写ASCLL码
  {
 writeData(i);
  }
}

void display_string_5x8(uint column,uchar *text)
{
uint i=0,j;
    writeCommand(0x80+column);
    while(text[i]>0x00)
```

```
    {
    if((text[i]>=0x20)&&(text[i]<0x7e))
    {
        j=text[i];
        writeData(j);
        WaitNms(100);////延时 x ms
        i++;
    }
    else
    i++;
    }
}

void time_disp(uchar i,uchar z,uchar z1,uchar d,uchar d1,uchar s,uchar s1,uchar
s10,uchar s100)
{
uchar j; j=0x30;
writeCommand(0x80+i);
writeData(z+j);writeData(z1+j);writeData(':');
writeData(d+j);writeData(d1+j);writeData(':');
writeData(s+j);writeData(s1+j);writeData(':');
writeData(s10+j);writeData(s100+j);
}

main(void)//////////////////////////////
{
uchar m1;
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//跑测试程序
for(m1=0;m1<50;m1++)
    {

      EA=1;
      EX0=1;
      IT0=1;

    LCDINT();
//  WaitNms(350);////延时 x ms
    writeCommand(0x01);//清屏
    writeALLcharacter();//调ASCLL码写入LCD
    WaitNms(400);////延时 x ms
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//1.2屏 调用ASCII
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//测试芯片的ROM
    writeCommand(0x01);
    writeCommand(0x40);       //自编图形（横竖点）
```

```
    storeCGRAM(0xff,0xff);
    storeCGRAM(0xff,0x00);
    storeCGRAM(0x00,0xFF);
    storeCGRAM(0xAA,0xAA);
    storeCGRAM(0x55,0x55);
    storeCGRAM(0xAA,0x55);
    storeCGRAM(0x55,0xAA);

    writeCommand(0x80);
    for(ii=0;ii<7;ii++)
        {
        writetest(ii);
        WaitNms(350);
        }
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//自编横竖点显示
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//3,4,5,6,7,8屏
    writeCommand(0x01);//清屏
    display_string_5x8(4,"HUIJINGKEJI");
    display_string_5x8(0X40+4,"TEL:23146001");     //20*4
    display_string_5x8(20,"welcome to HuiJing !");
    display_string_5x8(0X40+20,"Y works hard! thanks");
    WaitNms(400);////延时 x ms
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//调字符演示
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//左右移动演示
    }

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~老化测试99小时
    writeCommand(0x01);//清屏
    display_string_5x8(0,"HuiJing,23146001;welcome to HuiJing Co.,");
    display_string_5x8(0X40,"Time:");display_string_5x8(0X40+17,"you works hard!
thanks!");
    for(z=0;z<10;z++)
    {
        for(z1=0;z1<10;z1++)
        {
            for(d=0;d<6;d++)
            {
                for(d1=0;d1<10;d1++)
                {
                    for(s=0;s<6;s++)
                    {
                        for(s1=0;s1<10;s1++)
                        {   writeCommand(0x18);
                            WaitNms(10);////延时 x ms
```

```
                        for(s10=0;s10<10;s10++)
                        {
                            WaitNms(10);////延时 x ms
                            for(s100=0;s100<10;s100++)
                            {
                            time_disp(0X40+5,z,z1,d,d1,s,s1,s10,s100);

                            WaitNms(10);////延时 x ms
                            }
                        }
                    }
                }
            }
        }
    }
}


//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

完