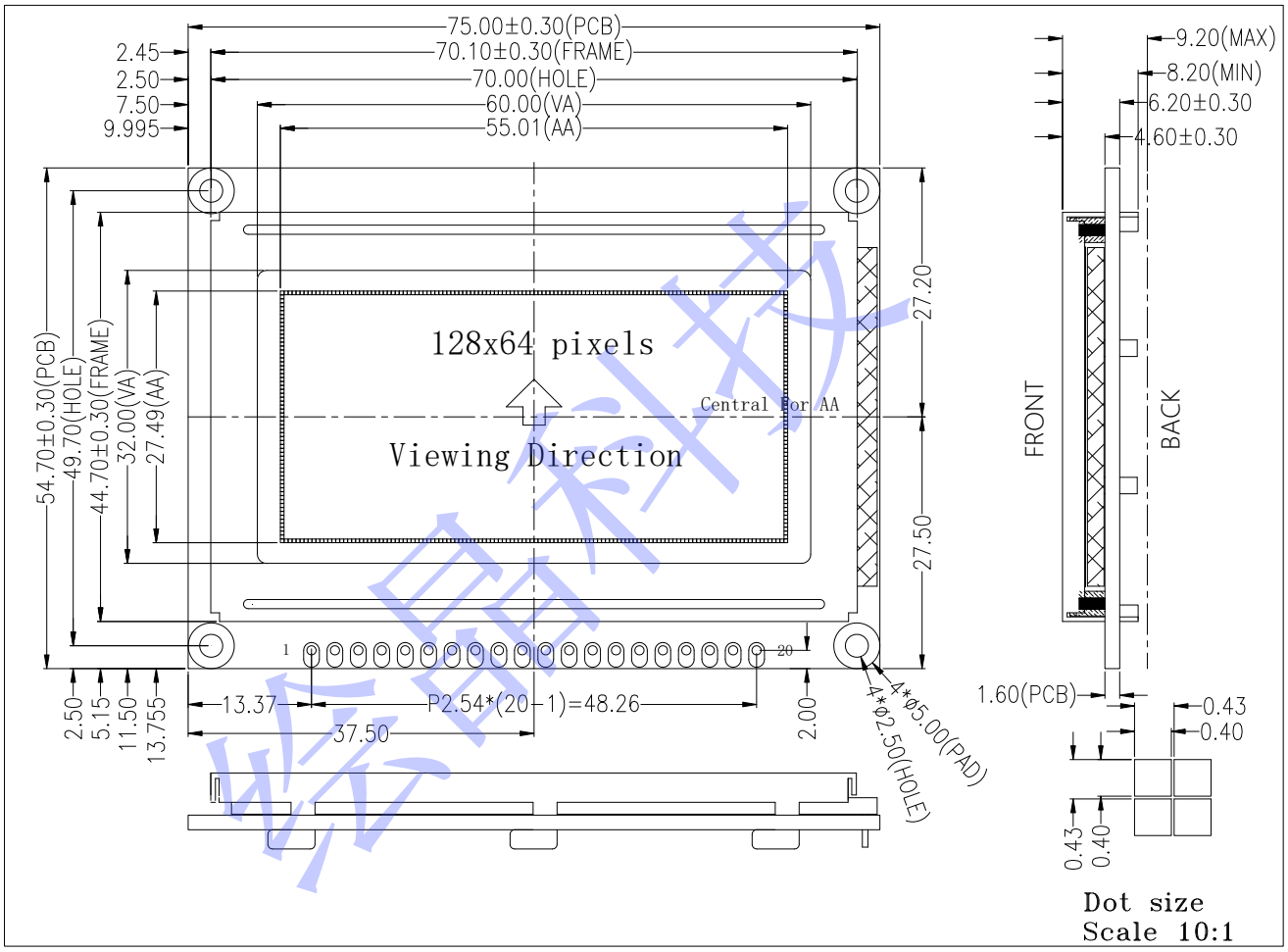
目　录

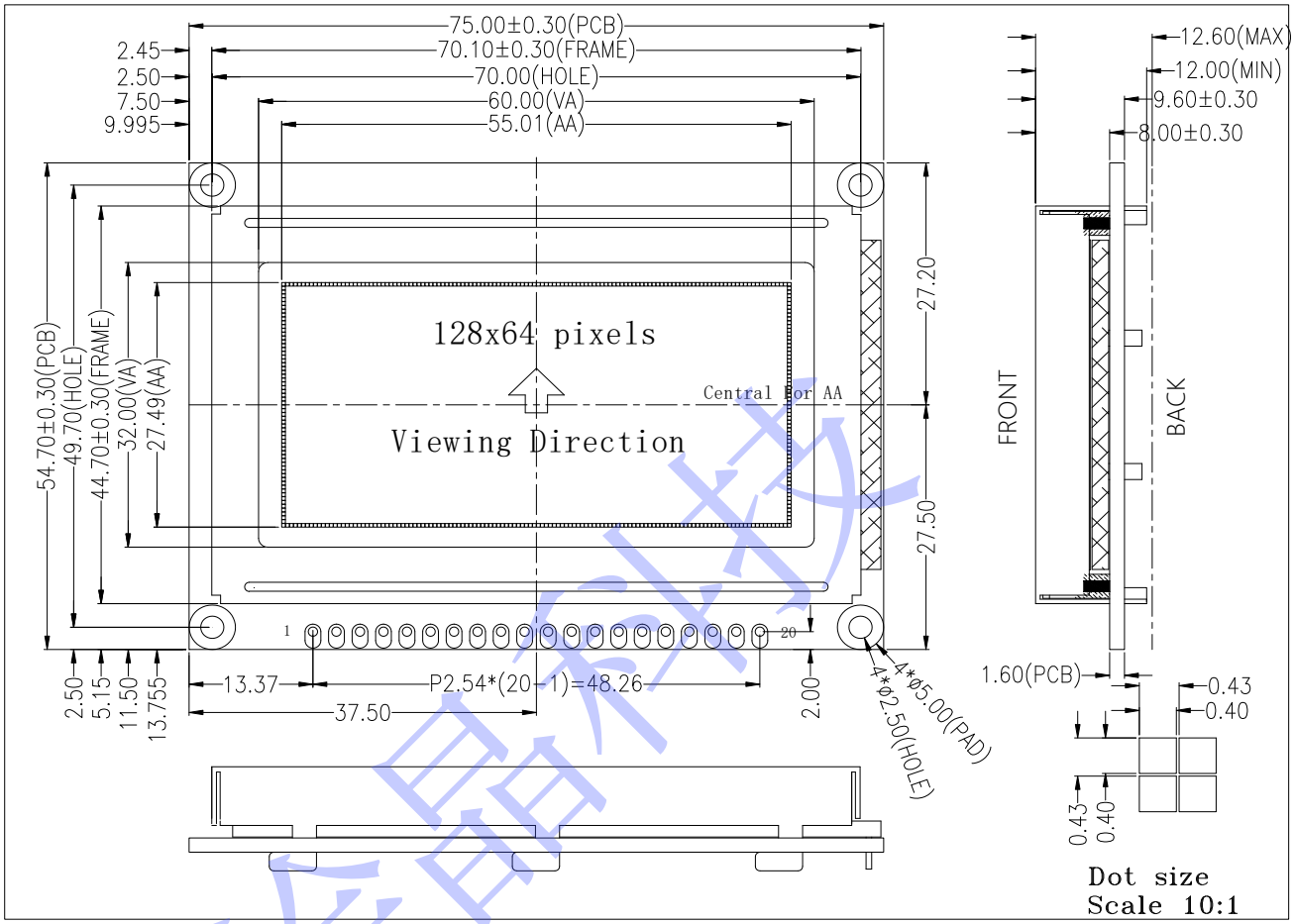
返回目录 Ctrl+Home 或者 wep 里的返回箭头

第一章、 显示器外形结构尺寸图

第一节、 低铁框结构尺寸



第二节、 高铁框结构尺寸



项目	参考值
LCM 尺寸（长×宽×厚）	75.0×54.7×8.2(低铁框)/11.6(高铁框)
可视区域（长×宽）	60.0×32.0
点间距（长×宽）	0.43×0.43
点尺寸（长×宽）	0.40×0.40

第二章、显示器基本功能介绍

- 工作电压 3.3V 或者 5.0V 供电(默认 5.0V)，采购时请和业务说明电压.
- 通讯方式：8 位 6800 并行通讯接口
- 128*64 点阵显示内存，纵向取模，字节倒序 方式
- ASCII 字符图案
  - 本程序例程中有 ASCII 码点阵表 5*8，8*16 方便调用
  - 可以使用竖置横排格式的字库 IC 配合使用
- 低功率省电设计(除背光 45MA/TBD)
  - 正常模式（1mA typ VDD=5.0/3.3V)
  - 待机模式（30uA max VDD=5.0/3.3V)
- 自动上电复位功能
- 指令方面
  - 上电即可使用，没有复杂的初始化设置
  - 纯点阵显示器，不需要太多时间来研读本资料，用户可以专心在软件设计上，本手册提供了比较长的程序例程，用户可以用来移植使用
- 占空比 1/64 偏压比 1/9
- 工作温度-20 度~+70 度
- 储存温度-30 度~+80 度
- 常用显示效果：黄底黑字，蓝底白字，灰底黑字。不常用可以定制，采购时请和业务说明显示效果

第三章、 显示器接口定义说明

第一节、 并口时管脚说明

引脚	名称	方向	说明
1	VDD	--	电源正端(+3.3V 或+5.0V，出厂时设定+5V)
2	VSS	--	电源负端(0V)
3	V0	--	LCD 对比度调节
4-11	DB0 ~ DB7	I/O	单片机与模块之间并口的数据传送通道，DB7 能用作忙标志读出（判忙）
12	/CS1	I	左半屏选取信号(低有效)
13	/CS1	I	左半屏选取信号(低有效)
14	/RST	I	复位信号
15	R/W	I	=0，写模式
			=1，读模式
16	RS(CD)	I	=1，写数据
			=0，写指令
17	/E	I	使能信号，低有效
18	VOUT(VEE)	-	LCD 电压输出或者输入（默认为输出）用于调节对比度电压
19	LEDA	-	背光电源的正极（3。3V 或者 5V）出厂 5V
20	LEDK	-	背光电源负极

第四章、 显示器的电性参数

第一节、 直流供电参数

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	-	3.2/4.9	3.3/5.0	3.4/5.1	V
玻璃电压	VLCD	VDD-VO	7.5	9.0	13.5	V
背光工作电压	VLED	-	3.2/4.8	3.3/5.0	3.4/5.2	V
IO 输入高电平	VIH	-	0.7VDD	-	VDD	V
IO 输入低电平	VIL	-	-0.3	-	0.6	V
LCM 输出高电平	VOH	-	0.8VDD	-	-	V
LCM 输出低电平	VOL	-	-	-	0.4	V
模块工作电流	IDD	=VDD	-	-	1.2	MA
模块待机电流	IDO	=VDD	-	-	50	uA
背光工作电流	ILED	=VLED	-	45	60	MA

第二节、 极限参数

参数名称	符号	极限值	单位	备注
输入电压	V_{DD}	-0.3~+7.0	V	(1)
LCD 驱动电压	V_{EE}	V_{DD} -19.0~ V_{DD} +0.3	V	(4)
驱动电源电压	V_B	-0.3~ V_{DD} +0.3	V	(1),(3)
	V_{LCD}	V_{EE} -0.3~ V_{DD} +0.3	V	(2)
工作温度	T_{OPR}	-20~+70	^{0}C	
储存温度	T_{STO}	-30~+80	^{0}C	

注

1、 电压均相对于 VSS=0V

2、 VEE1\VEE2 接相同的电源电压，VLCD=VDD-VEE

3、 对于 M\FRM\CL\RSTB\ADC\CLK1\CLK2\CS1B\CS2B\CS3\E\R/W\RS 和 DB0-DB7

4、 对于 VOL(R),V2L(R),V3L(R)和 V5L(R).

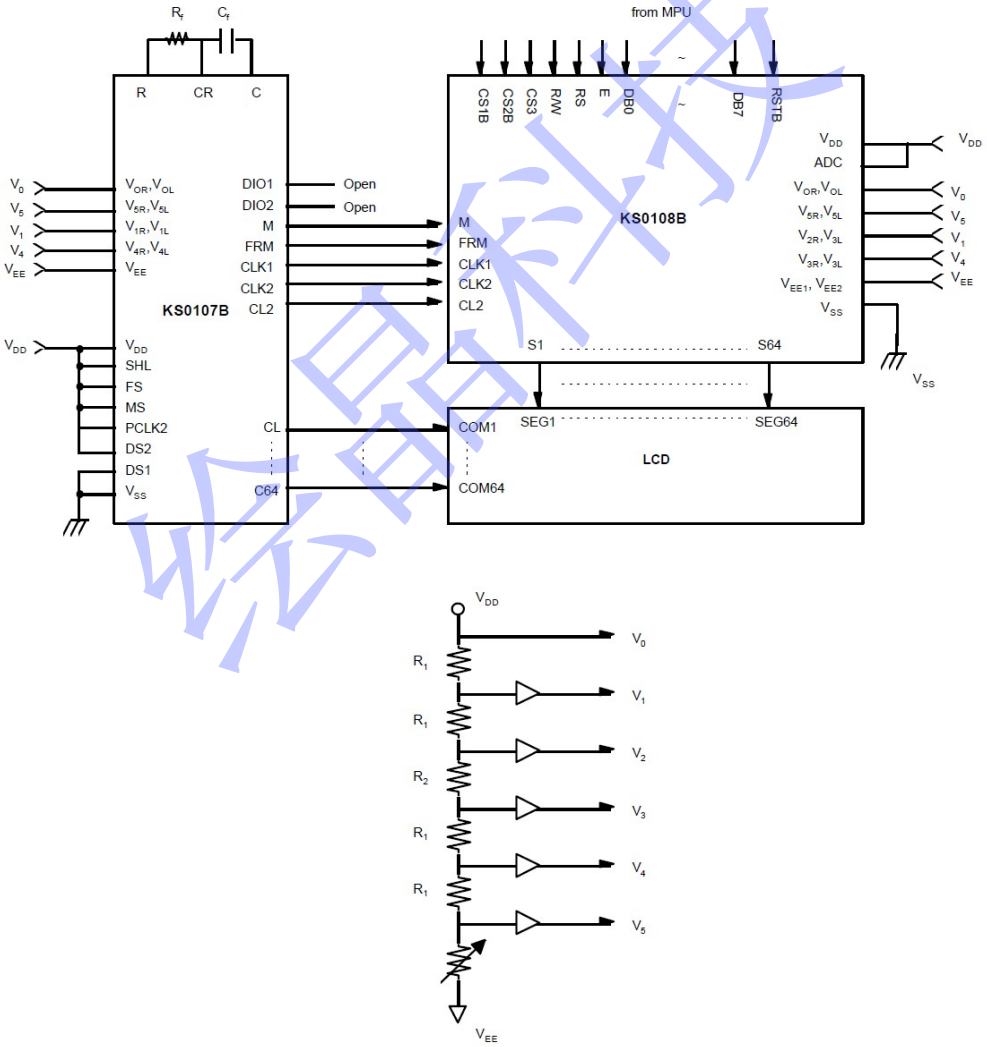
a) VDD>VOL>VOR>V2L=V2R>V3L>V5L>V5R>VEE

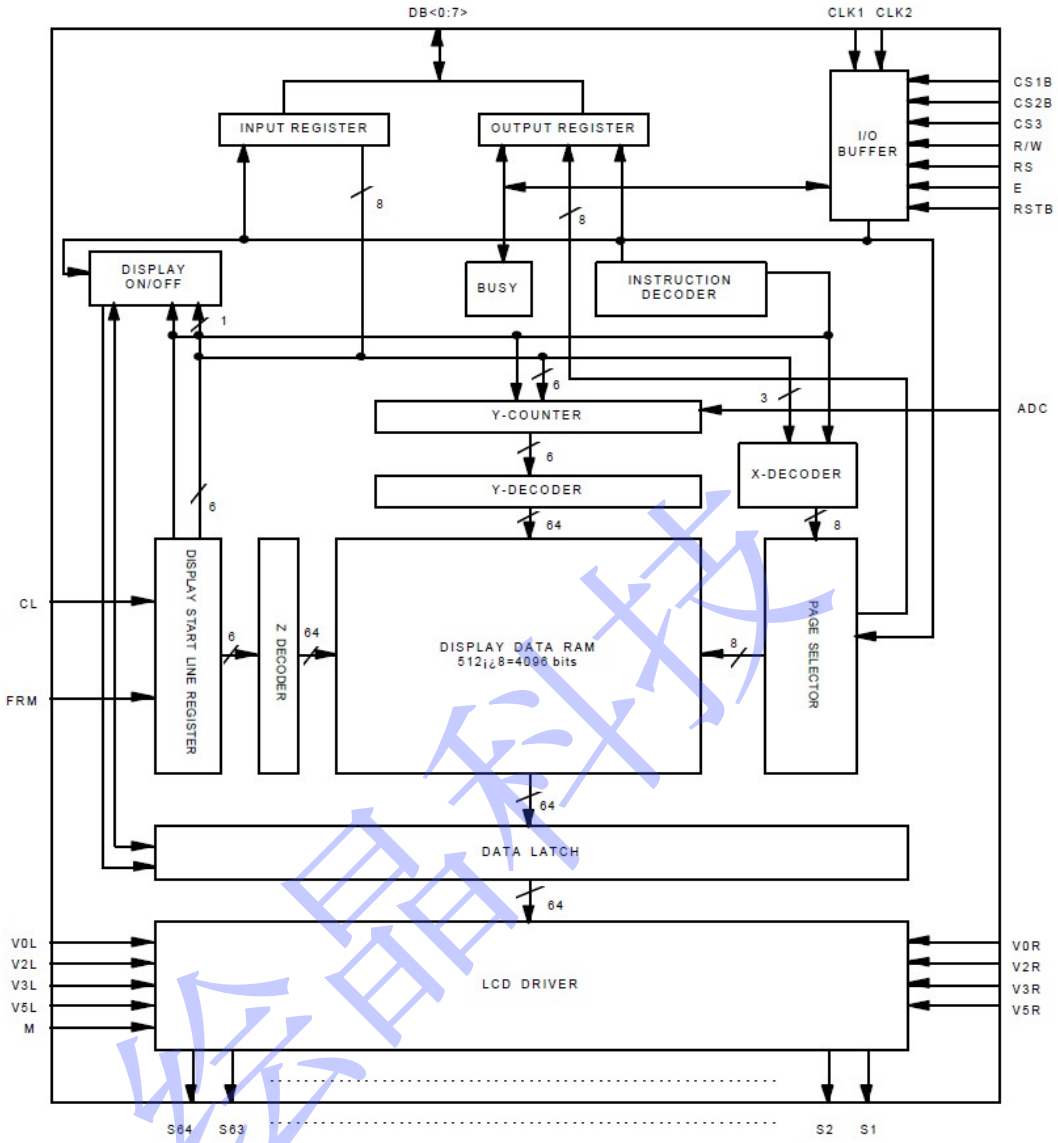
第三节、 液晶屏功耗

类别	条件	参数	符号
模块	5V	1.2	MA
背光	5V	45	MA

第五章、显示器的显示结构原理

第一节、 显示器控制器方框图





第二节、 显示内存映射图

Y＝	左半屏64列					右半屏64列					行号
	0	1	…	62	63	0	1	…	62	63	
X=0	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	0 ↓ 7
↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	DB0 ↓	8 ↓

	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	DB7	55
X=7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	DB0 ↓ DB7	56 ↓ 63

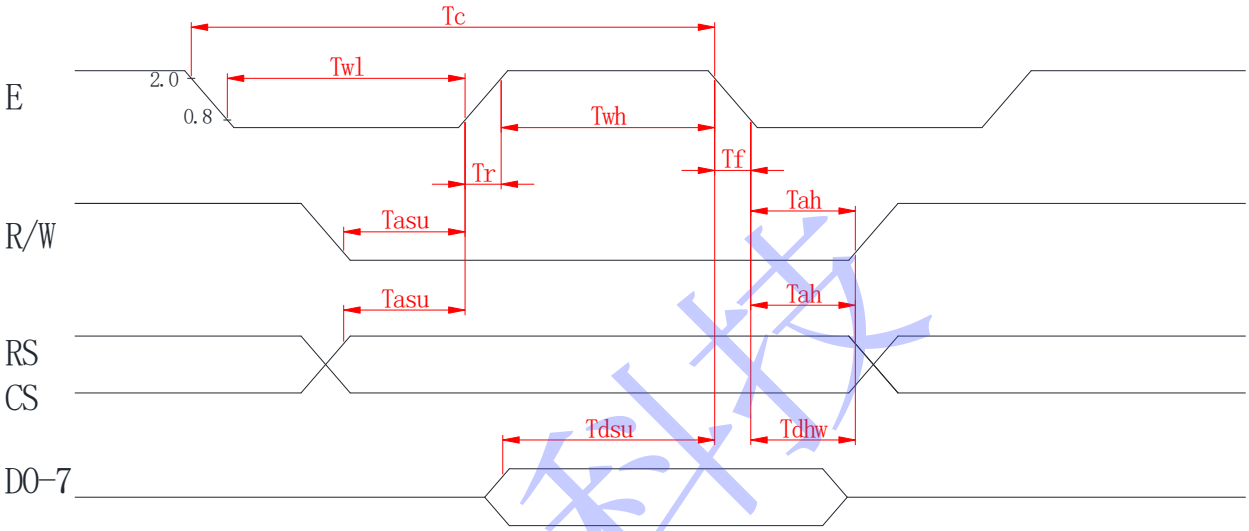
12864 图形点阵显示屏是两个 64*64 点阵显示的屏合在一起，如图分的左半屏，右半屏，是一款纯点阵屏，在编源代码的时候要注意一下（两个 CS 控制左右半屏）

这款纯点阵屏的地址结构如图，列地址（0-63）*2， 列地址范围以 1 个像素单位定义为（0-63）*2 列， 行地址范围以 8 个像素为一个单位定义为（0-7）页，首行地址范围为一个像素单位，设置范围为（0-63）

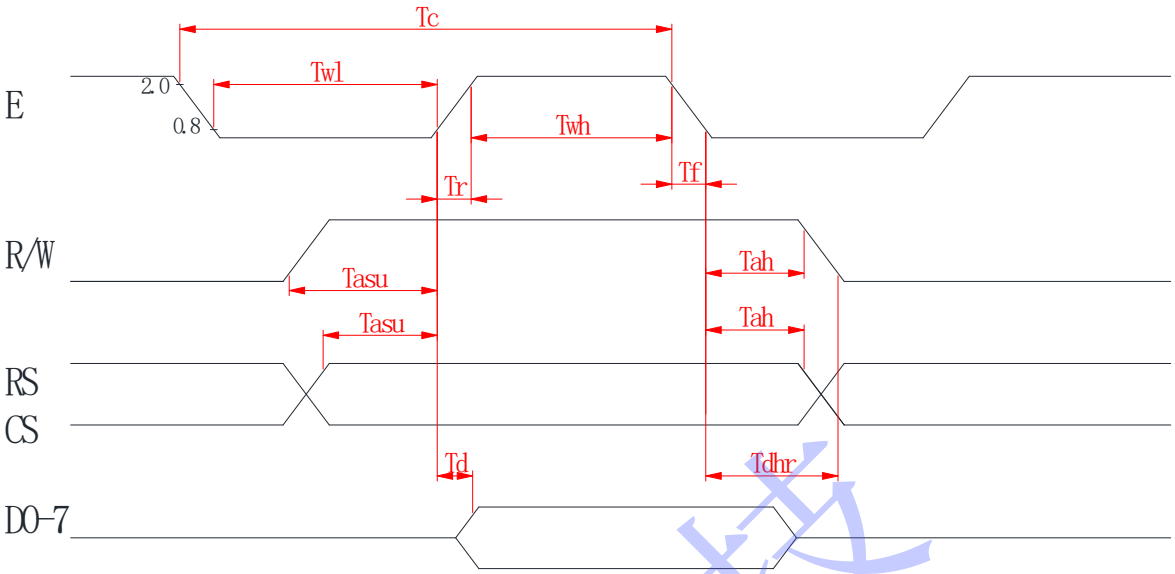
第三节、 写入数据流程图

第六章、 驱动程序时序图说明

第一节、 写时序图



第二节、 读时序图



第三节、 时序参数表

名称	符号	最小值	典型值	最大值	单位
E周期时间	T_C	1000	--	--	ns
E高电平时间	T_{WH}	450	--	--	ns
E低电平时间	T_{WL}	450	--	--	ns
E上升沿时间	T_R	--	--	25	ns
E下降沿时间	T_F	--	--	25	ns
地址建立时间	T_{ASU}	140	--	--	ns
地址保持时间	T_{AH}	10	--	--	ns
数据建立时间	T_{DSU}	200	--	--	ns
数据延迟时间	T_D	--	--	320	ns
写数据保持时间	T_{DHW}	10	--	--	ns
读数据保持时间	T_{DHR}	20	--	--	ns

第七章、 驱动程序的指令说明

第一节、 显示模块指令表：

NO	指令	指令码										HEX	说明
		RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0		
1	显示开关	0	0	0	0	1	1	1	1	1	LH	3F	控制显示器的开关，不影响DDRAM中数据和内部状态。 0：关； 1：开
2	Y地址设置	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	40	列地址设置（共64列）
3	面地址设置	0	0	1	0	1	1	1	AC2	AC1	AC0	B8	页地址设置（共8页）
4	首行设置	0	0	1	1	AC5	AC4	AC3	AC2	AC1	AC0	C0	设置显示屏第一行的位置
5	状态读	0	1	Busy	0	N/F	REST	0	0	0	0		读状态： Busy=0；空闲 Busy=1；工作中 N/F=0；显示开 N/F=1；显示关 REST=0；正常 REST=1；复位
6	写数据	1	0	D7	D6	D5	D4	D3	D2	D1	D0		写数据
7	读数据	1	1	D7	D6	D5	D4	D3	D2	D1	D0		读数据

第二节、 基本指令详细说明

一、 显示开关

RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
0	0	0	0	1	1	1	1	1	D	0x3E+X

显示数据在D=1时显示，在D=0消失。尽管当D=0时显示数据不在屏幕上显示，该数据依然保存在存储器中，因此可以将D=0改变到D=1使其显示。

二、 列地址设置

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	0X40+X

显示数据存储器的Y地址（AC0-AC5）在Y计数器中设置。地址由指令设置并在对显示RAM读或写时自动增1.

三、 页地址设置

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	1	0	0	0	0	AC2	AC1	AC0	0Xb0+X

显示存储器的X地址（AC0-AC2）在X地址寄存器在X地址寄存器中设置，MPU中读、写操作在这一页面执行，直到下一个页被设置。

四、 显示开始行设置

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	0	1	1	AC5	AC4	AC3	AC2	AC1	AC0	0XC0+x

显示存储器的Z地址在显示开始行寄存器中被设置并显示在屏幕顶端。当显示占空比为1/64或其它（1/32-1/64），在LCD显示屏从显示开始指令指定的行开始显示。

五、 读状态

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	0	1	busy	0	ON/OFF	RESET	0	0	0	0	接收

BUSY

BUSY=1，芯片执行内部操作，不接受指令。

BUSY=0，芯片准备好接收指令。

ON/OFF

ON/OFF=1，显示开

ON/OFF=0，显示关

RESET

RESET=1，系统正在被初始化，在这个状态下，除状态读指令外，其余不接收

RESET=0，系统初始化结束，系统可以正常工作

六、写显示数据

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	1	0	D7	D6	D5	D4	D3	D2	D1	D0	发送

写数据（D0-D7）至显示存储器，写指令结束扣，Y地址自动增1

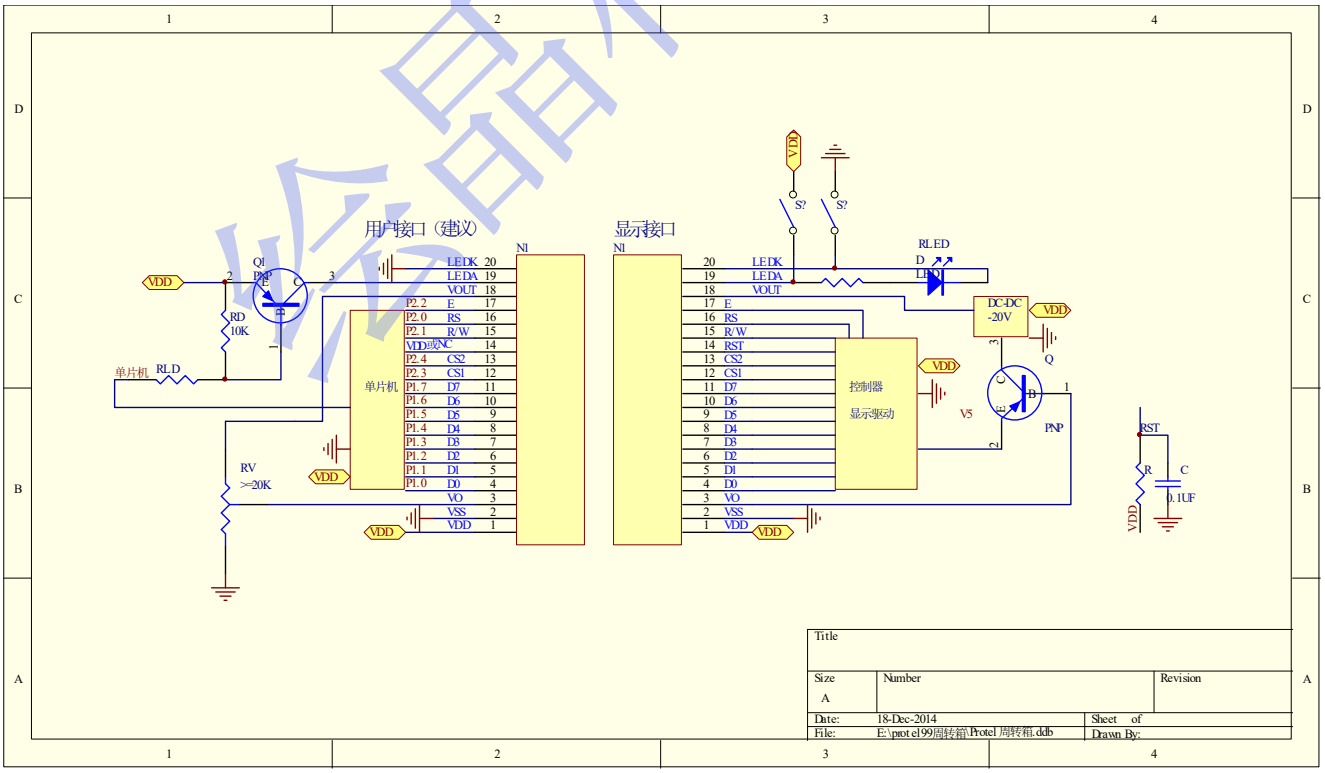
七、读显示数据

	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	HEX
代码	1	1	D7	D6	D5	D4	D3	D2	D1	D0	接收

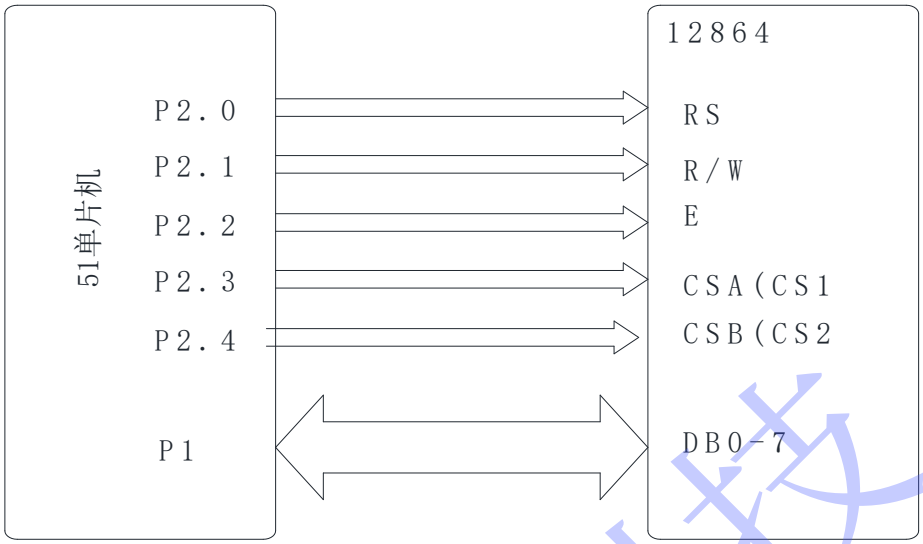
从显示存储器读数据（D0-D7），读指令结束后，Y地址自动增1

第八章、单片机与显示器连接说明

第一节、 接口并口八位应用原理图



第二节、 单片机并口连接



第九章、单片机驱动程序源代码

第一节、 源代码解释定义声明

```
//,单片机:89S52,STC完全兼容
//晶振:12M,编译软件:KEIL 7.06
//并行连接方式,P2.0-RS,P2.1-RW,P2.2-E

#include<reg52.h>
#include <intrins.h>
sbit RS=P2^0; sbit RW=P2^1; sbit E=P2^2;
sbit stop=P3^2;
sbit CSB=P2^4;   //右半屏片选(高有效)
sbit CSA=P2^3;   //左半屏片选(高有效)

typedef unsigned int uint;
typedef unsigned char uchar;
uchar m1,z,z1,d,d1,s,s1,s10,s100;
uchar code ascii_table_8x16[95][16];
uchar code ascii_table_5x8[95][5];

uchar code hui1[]={/*--  文字:  绘  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=16x16   --*/
```

```
0x20,0x30,0xAC,0x63,0x10,0x20,0x10,0x48,0x44,0x43,0x44,0x48,0x10,0x20,0x20,0x00,
0x22,0x67,0x22,0x12,0x12,0x40,0xE2,0x52,0x4A,0x46,0x42,0x52,0x62,0xC2,0x00,0x00,
};
uchar code jing1[]={/*--  文字:  晶  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=16x16   --*/
0x00,0x00,0x00,0x00,0x7F,0x49,0x49,0x49,0x49,0x49,0x7F,0x00,0x00,0x00,0x00,0x00,
0x00,0xFF,0x49,0x49,0x49,0x49,0xFF,0x00,0xFF,0x49,0x49,0x49,0x49,0xFF,0x00,0x00,
};
uchar code ke1[]={/*--  文字:  科  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=16x16   --*/
0x24,0x24,0xA4,0xFE,0xA3,0x22,0x00,0x22,0xCC,0x00,0x00,0xFF,0x00,0x00,0x00,0x00,
0x08,0x06,0x01,0xFF,0x00,0x01,0x04,0x04,0x04,0x04,0x04,0xFF,0x02,0x02,0x02,0x00,
};
uchar code ji1[]={/*--  文字:  技  --*/
/*--  宋体12;  此字体下对应的点阵为:宽x高=16x16   --*/
0x10,0x10,0x10,0xFF,0x10,0x90,0x08,0x88,0x88,0x88,0xFF,0x88,0x88,0x88,0x08,0x00,
0x04,0x44,0x82,0x7F,0x01,0x80,0x80,0x40,0x43,0x2C,0x10,0x28,0x46,0x81,0x80,0x00,
};

const uchar delay=250;  //延时时间常数
static void Wait1ms(void)//延迟1 ms
{
  uchar cnt=0;
  while (cnt<delay) cnt++;
}
//延迟n ms
void WaitNms(int n)
{
    uchar i;
    for(i=1;i<=n;i++)
    Wait1ms();
}
```

第二节、 接口时序函数

```
/*判断是否为忙?*/
void Busy(void)
{
    uchar temp;
    CSA=0;
```

```
    CSB=0;
    RS=0;
    RW=1;
    P1=0xff;
    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 wrcomL(uchar command)    //写左半屏指令
{
    Busy();

    CSA=0;
    CSB=1;
    RW=0;
    RS=0;
    P1=command;
    E=1;
    E=0;

}
void wrcomR(uchar command) //写右半屏指令
{
    Busy();
    CSA=1;
    CSB=0;

    RW=0;
    RS=0;
    P1=command;
    E=1;
    E=0;

}
void wrdataL(uchar DATA) //写左半屏数据
{
    Busy();
```

```
    CSA=0;
    CSB=1;
    RW=0;
    RS=1;
    P1=DATA;
    E=1;
    E=0;

}
void wrdataR(uchar DATA) //写右半屏数据
{
    Busy();

    CSA=1;
    CSB=0;
    RW=0;
    RS=1;
    P1=DATA;
    E=1;
    E=0;

}
```

第三节、 应用函数

```
void dispsameone(uchar k)//把一个字节内容填满全屏
{
  uchar i;
  uchar j;
  wrcomL(0xc0);//设置C0为首行
  wrcomL(0xb8);//设置第一页
  wrcomL(0x40);//设置第一列

  wrcomR(0xc0);//设置C0为首行
  wrcomR(0xb8);//设置第一页
  wrcomR(0x40);//设置第一列
  for(j=0;j<8;j++)//再写下一行
 {wrcomR(j+0xB8);
  wrcomL(j+0xB8);
  for(i=0;i<64;i++)//写满一行
```

```
  {
  wrdataL(k);//指针自动加一
  wrdataR(k);
  }
 }
}
void dispsametow(uchar k,q)//把两个紧靠的字节内容填满全屏
{
  uchar i;
  uchar j;
  wrcomL(0xc0);
  wrcomL(0xb8);
  wrcomL(0x40);

  wrcomR(0xc0);
  wrcomR(0xb8);
  wrcomR(0x40);
  for(j=0;j<8;j++)//再写下一行
 {wrcomR(j+0xB8);
  wrcomL(j+0xB8);
  for(i=0;i<32;i++)//写满一行
  {
  wrdataL(k);//指针自动加一
  wrdataL(q);//指针自动加一
  wrdataR(k);
  wrdataR(q);
  }
 }
}

void display_string_8x16(uchar hh,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++)
            {
                wrcomL(0xc0);wrcomR(0xc0);
                wrcomL(0xb8+page+n);wrcomR(0xb8+page+n);
                if(column<64)
```

```
                {
                    wrcomL(0x40+column);
                    for(k=0;k<8;k++)
                    if(hh==0)
                    {
                    wrdataL(ascii_table_8x16[j][k+8*n]);
                    }
                    else
                    {
                    wrdataL(~(ascii_table_8x16[j][k+8*n]));
                    }
                }
                else
                {
                    wrcomR(0x40+(column-64));
                    for(k=0;k<8;k++)
                    if(hh==0)
                    {wrdataR(ascii_table_8x16[j][k+8*n]);
                    }
                    else
                    {wrdataR(~(ascii_table_8x16[j][k+8*n]));
                    }
                }
            }
            i++;
            column+=8;
        }
    else
    i++;
    }
}

void display_string_8x16_t(uchar hh,uint page,uint column,uchar text)
{
    uint j,k,n;
    j=text+16;
    for(n=0;n<2;n++)
    {
        wrcomL(0xc0);wrcomR(0xc0);
        wrcomL(0xb8+page+n);wrcomR(0xb8+page+n);
        if(column<64)
        {
            wrcomL(0x40+column);
```

```
            for(k=0;k<8;k++)
            if(hh==0)
            {
            wrdataL(ascii_table_8x16[j][k+8*n]);
            }
            else
            {
            wrdataL(~(ascii_table_8x16[j][k+8*n]));
            }
        }
        else
        {
            wrcomR(0x40+(column-64));
            for(k=0;k<8;k++)
            if(hh==0)
            {wrdataR(ascii_table_8x16[j][k+8*n]);
            }
            else
            {wrdataR(~(ascii_table_8x16[j][k+8*n]));
            }
        }
    }

}


void display_string_5x8(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;
        wrcomL(0xc0);wrcomR(0xc0);
        wrcomL(0xb8+page);wrcomR(0xb8+page);
        if(column<64)
        {
            wrcomL(0x40+column);
            for(k=0;k<5;k++)
            {wrdataL(ascii_table_5x8[j][k]);
            }
        }
```

```
        else
        {
            wrcomR(0x40+(column-64));
            for(k=0;k<5;k++)
            {wrdataR(ascii_table_5x8[j][k]);
            }
        }
        i++;
        column+=6;
    }
    else
    i++;
    }
}

//显示128x64 点阵边框
void display_bk()
{
uint i,j;
//左框
for(j=0;j<8;j++)
    {
    wrcomL(0xb8+j);
    wrcomL(0x40+0);
    wrdataL(0xff);
    wrdataL(0xff);
    }
    //右框
    for(j=0;j<8;j++)
    {
    wrcomR(0xb8+j);
    wrcomR(0x40+62);
    wrdataR(0xff);
    wrdataR(0xff);
    }
    //上框
    wrcomL(0xb8+0);wrcomR(0xb8+0);
    wrcomL(0x40+2);wrcomR(0x40+0);
    for (i=0;i<62;i++)
    {
    wrdataL(0x03);
    wrdataR(0x03);
    }
```

```
    //下框
    wrcomL(0xb8+7);wrcomR(0xb8+7);
    wrcomL(0x40+2);wrcomR(0x40+0);
    for (i=0;i<62;i++)
    {
    wrdataL(0xC0);
    wrdataR(0xC0);
    }
}

//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 hh,uchar page,uchar column,uchar *dp )
{
uint i,j;
    for(j=0;j<2;j++)
    {
        wrcomL(0xb8+page+j);wrcomR(0xb8+page+j);
        if(column<64)
        {
        wrcomL(0x40+column);
            for (i=0;i<16;i++)
            {
            if(hh==0) wrdataL(*dp);
            else wrdataL(~(*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++;
            }
        }
        else
        {
        wrcomR(0x40+(column-64));
            for (i=0;i<16;i++)
            {
            if(hh==0) wrdataR(*dp);
            else wrdataR(~(*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++;
            }
        }
    }
}
```

第四节、 液晶模块初始化

```
void lcdint(void)
{
wrcomR(0x3e);//显示关
wrcomR(0x3f);//显示开
wrcomL(0x3e);
wrcomL(0x3f);
wrcomL(0xC0);//首行
wrcomR(0xC0);//首行
}

void ini_int1(void)
{
EA=1;
EX0=1;//允许外部INT0的中断
IT0=1;// 允许中断
}

 int scankey1() interrupt 0 using 3 //使用外部中断1,寄存器组3
{
while(P3^2==0){for(;;);}
   IE1=0;//中断标志清零
}
```

第五节、 主调用函数

```
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~程序从这里开始
void main(void)
{
ini_int1();//开中断
for(m1=0;m1<50;m1++)
    {
    lcdint();
    dispsameone(0x00);//清屏
    display_string_8x16(0,0,0,"0123456789ABCDEF");
    display_string_8x16(0,2,0,"HUIJINGKEJI12864");
```

```
display_string_5x8(4,4,"0123456789ABCDEFGHIJ");
display_string_5x8(5,4,"**TEL:0755-23146001*");
display_string_5x8(6,4,"**FAX:0755-23146002*");
display_string_5x8(7,4,".www.huijinglcm.com.");
WaitNms(250);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~1,（5*8和8*16字符显示）

dispsameone(0x00);//清屏
display_graphic_16x16(0,2,32,hui1);
display_graphic_16x16(0,2,48,jing1);
display_graphic_16x16(0,2,64,ke1);
display_graphic_16x16(0,2,80,ji1);
display_string_8x16(0,4,40,"128*64");
display_bk();
WaitNms(250);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~2,（自编边框/16*16汉字显示 ）

dispsameone(0x00);//清屏
dispsameone(0xFF);//清屏
display_graphic_16x16(1,2,32,hui1);
display_graphic_16x16(1,2,48,jing1);
display_graphic_16x16(1,2,64,ke1);
display_graphic_16x16(1,2,80,ji1);
display_string_8x16(1,4,40,"128*64");
WaitNms(250);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~3,（灵活的反白显示演示）

dispsameone(0x55);//显示横扫描1
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~4,扫横1
dispsameone(0xaa);//显示横扫描2
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~5,扫横2
dispsametow(0xff,0x00);//显示竖扫描1
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~6,扫竖1
dispsametow(0x00,0xff);//显示竖扫描2
```

```
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~7,扫竖2
dispsametow(0xaa,0x55);//显示点扫描1
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~8,扫点1
dispsametow(0x55,0xaa);//显示点扫描2
WaitNms(150);
WaitNms(150);
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~9,扫点2
}
```

第六节、 动态显示函数

```
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~10,时钟演示
dispsameone(0x00);//清屏
display_graphic_16x16(1,0,0,hui1);
display_graphic_16x16(1,2,0,jing1);
display_string_8x16(1,5,0,"TEL:755-23146001");
display_string_5x8(7,4,".www.huijinglcm.com.");
display_string_5x8(0,40,"<<<<<<<<<<<<<<");
display_string_5x8(3,40,"<<<<<<<<<<<<<<");
for(z=0;z<10;z++)
{
    display_string_8x16_t(0,1,40,z);
    for(z1=0;z1<10;z1++)
    {
        display_string_8x16_t(0,1,48,z1);
        for(d=0;d<6;d++)
        {
            display_string_8x16_t(0,1,56,10);
            display_string_8x16_t(0,1,64,d);
            for(d1=0;d1<10;d1++)
            {
                display_string_8x16_t(0,1,72,d1);
                for(s=0;s<6;s++)
                {
                    display_string_8x16_t(0,1,80,10);
                    display_string_8x16_t(0,1,88,s);
                    for(s1=0;s1<10;s1++)
```

```
                    {
                        display_string_8x16_t(0,1,96,s1);
                        WaitNms(5);////延时 x ms
                        display_string_8x16_t(0,1,104,10);
                        for(s10=0;s10<10;s10++)
                        {
                            display_string_8x16_t(0,1,112,s10);

                            WaitNms(5);////延时 x ms
                            for(s100=0;s100<10;s100++)
                            {
                            display_string_8x16_t(0,1,120,s100);
                            WaitNms(5);////延时 x ms
                            }
                        }
                    }
                }
            }
        }
    }

    }
```

第七节、 标准字符数据表

uchar code ascii_table_8x16[95][16]={
篇幅大，需要请找公司业务

第十章、版本信息

更新说明

更新日期	更新内容说明	版本号
2023-11-06	第一版	V1.0