LM1095R

液晶显示模块应用参考

深圳市拓普微科技开发有限公司

版本	描述	日期	编者
0.1	新版本	2005-10-28	淮俊霞
0.2	修改 2.4：双图层的灰阶显示 修改参考程序中的错字	2006-01-18	淮俊霞
0.3	修改 2.6：初始化寄存器的设置 修改参考程序：delay 函数中的延迟时间	2006-08-10	淮俊霞

目 录

1 简介

本公司产品 LM1095R 为 192×128 点阵中文/ 图形液晶显示模块，内置 RA8803 控制器。模块不仅可以显示单一的文本、图形，而且可以实现双图层的合成显示（“或”、“异或”、“同或”、“与”四种逻辑关系），此外还能产生四阶灰度的效果。文本模式下能够实现大小字体的混编（最大字体为 64×64），在连续输入资料时，可以自动设定行距。使显示画面更加美观，大大节省用户的开发时间。

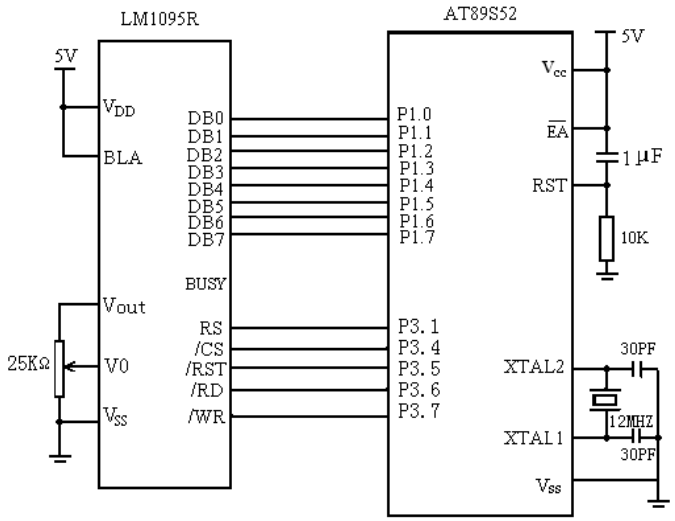
模块的特点有：

- 单电源供电，内置升压电路
- 高对比度，FSTN 型 LCD 屏
- 白色 LED 背光
- 内嵌简体中文字库（7602 个汉字）
- 双图层内存（2×9.6K 显示存储器）
- 可自定义 16 个字符

2 应用

2.1 接口

模块与单片机 AT89S52 的接口，采用 I/O 方式的 8 位并行通信，如下图所示。



模块引脚	名称
1	Vout
2	V0
3	VDD
4	VSS
5	BUSY
6	NC
7	/CS
8	RS
9	/WR
10	/RD
11	DB0
...	...
18	DB7
19	/RST
20	BLA

注：BUSY 空接

2.2 指令操作

操作	控制状态			指令代码							
	RS	/RD	/WR	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
写寄存器命令	0	1	0	寄存器地址码							
				参数							
读寄存器数据命令	0	0	1	寄存器地址码							
数据写操作	1	1	0	写数据							
数据读操作	1	0	1	读数据							

2.3 寄存器及参数表

寄存器	缺省值	指令参数							
		DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
00H	C9H	电源模式(Power Mode) 11： 正常模式 00： 关闭模式		软件复位 1：复位 0：正常模式	0	显示模式 1：文本模式 0：图形模式	显示开关 1： 开 0： 关	屏幕闪烁 1： 是 0： 否	显示类型 1： 正向 0： 负向
03H	80H	1	0	0	0	高低位反转显示数据 1：是 0：否	垂直移动 1： 允许 0： 禁止	水平移动 1： 允许 0： 禁止	平移模式 1：水平移动 0：垂直移动
10H	6FH	读数据时，光标自动移位 1： 是 0： 否	中英文对齐 1： 是 0： 否	正反相存储数据选择 1： 正相 0： 反相	字体设置 1： 粗体 0： 正常	写数据时，光标自动移位 1： 是 0： 否	光标显示 1： 开 0： 关	光标闪烁 1： 是 0： 否	光标宽度 1： 8 或 16 0： 固定 8
11H	22H	光标的高度				行距			
12H	91H	图形模式下，光标自动移位方向选择 1： 向右 0： 向下	图层显示模式选择 001： 只显示 Page 1 (单一上图层显示) 010： 只显示 Page 2 (单一下图层显示) 011： 双图层显示 000： 灰阶显示			在双图层模式下，Page1 与 Page 2 的逻辑关系 00： “OR” 01：“XOR” 10： “NOR” 11：“AND”		选择读写操作的图层 00：Page 0 (512B SRAM) 01：Page 1 (9.6KB SRAM) 10：Page 2 (9.6KB SRAM) 11：同时存取两图层	
20H	27H	0	0	设定工作视窗右边位置					
21H	27H	0	0	设定显示视窗右边位置					
30H	EFH	设定工作视窗底部位置							
31H	EFH	设定显示视窗底部位置							
40H	00H	0	0	设定工作视窗左边位置					
41H	00H	0	0	设定显示视窗左边位置					
50H	00H	设定工作视窗顶部位置							
51H	00H	设定显示视窗顶部位置							
60H	00H	0	0	设定光标 X 地址					
70H	00H	设定光标 Y 地址							
71H	00H	水平移动时，块移动的起始行							
72H	EFH	水平移动时，块移动的结束行							
90H	04H	设定移位时钟							
E0H	00H	(1) 图形模式下，若寄存器[F0]Bit3 为 1，本寄存器的数据会被全部填写到 DDRAM 内，之后[F0]Bit3 被置 0。 (2) 灰阶模式下，用来控制灰阶的显示效果，“0”与“1”的数目代表显示的比率。							
F0H	92H	1	0	字型 ROM 的选择 00： 简体字型(256KB) 01： 繁体字型(512KB) 10： 简体字型(512KB)		写资料到 DDRAM 1：开始写入 0：未动作	文本模式下强制 ASCII 解码 1：是 0：否	4 种 ASCII 区块选择 00：区块 0 01：区块 1 10：区块 2 11：区块 3	
F1H	0FH	字型水平方向的大小 00：一倍 01：二倍 10：三倍 11：四倍		字型垂直方向的大小 00：一倍 01：二倍 10：三倍 11：四倍		1	1	1	1

2.4 参数总结

单页显示	双图层显示

选择页码
[12H]=0x91 或 0x11 // Page 1
[12H]= 0xA2 或 0x22 // Page 2
设定显示模式
[00H]=0xCD // 文本模式
[00H]=0xC5 // 图形模式
定位光标位置
[60H]=0x**
[70H]=0x**
写入数据
(1) 图形模式下，直接写入数据
(2) 文本模式下，先写入代码的高字节，再写入代码的低字节
 	
图形功能：
1）光标自动移位方向：
[12H] Bit7 位
2）自动写入数据：
[E0H]、[F0H]Bit3 位
 	
文本功能：
1）文本的设置
行距选择： [11H] Bit3:0 位
字体选择： [10H] Bit4 位
字型大小：[F1H] Bit7:4 位
中/英文文字对齐：[10H] Bit6 位
字符负向显示：[10H] Bit5 位
字符高低位反转：[03H] Bit3 位
水平移动：[03H] Bit1:0 位；[71H]、[72H] 可以设置任意两 Common 区块之间的移动
垂直移动：[03H] Bit2、0 位
 	
2）光标的设置
定位：[60H]、[70H]
光标显示：[10H] Bit2 位
光标闪烁：[10H] Bit1 位
光标宽度：[10H] Bit0 位
光标高度：[11H] Bit7:4 位
	选择 page1 进行读写操作
	[12H]=0x91 或 0x11
	设定显示模式
	[00H] Bit3 位
	定位光标位置
	[60H]=0x**
	[70H]=0x**
	写入数据
	
	选择 page2 进行读写操作
	[12H]= 0xA2 或 0x22
	设定显示模式
	[00H] Bit3 位
	定位光标位置
	[60H]=0x**
	[70H]=0x**
	写入数据
	
	两图层“或”关系
	[12H]=0xB3 //Page1 RAM “OR” Page2 RAM
	两图层“异或”关系
	[12H]=0xB7 //Page1 RAM “XOR” Page2 RAM
	两图层“同或”关系
	[12H]=0xBB //Page1 RAM “NOR” Page2 RAM
	两图层“与”关系
	[12H]=0xBF //Page1 RAM “AND” Page2 RAM
	
	灰阶显示：
	提高系统时钟
	[01H] Bit1:0 位
	灰阶对比设置
	[E0H]=0x**
	提高帧频
	[90H]=0x**
	显示灰阶
	[12H]=0x00
	关显示
	[00H] Bit2 位
	开显示
	[00H] Bit2 位
	延迟 1 秒

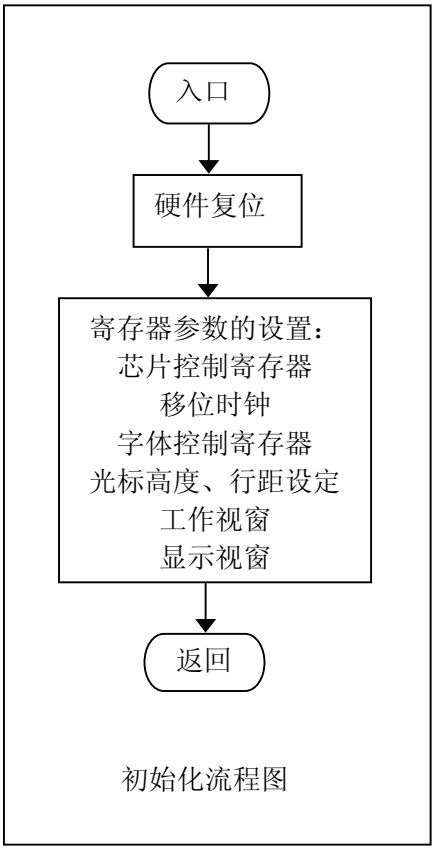
2.5 显示屏幕与视窗

X 坐标 / Y 坐标	00H			01H			……			17H		
	DB7	…	DB0	DB7	…	DB0	DB7	…	DB0	DB7	…	DB0
00H	192×128 像素											
01H												
⋮												
7FH												

用户有两种视窗的选择：一种是显示视窗(Display Window)，另一种是工作视窗(Active Window)。显示视窗是实际的 LCD 面板的大小，而工作视窗是在实际的显示视窗内设定比显示视窗小的子视窗。显示图形或者字符(左上角)的位置可以用光标寄存器[60H]、[70H]来定义，在连续写入数据时可以设置光标自动移位（寄存器[10H]的 Bit3 置 1），光标的移动会以寄存器[20H]、[30H]、[40H]、[50H]所设置的工作视窗为边界，自动换行或换页。

2.6 复位和初始化

上电后，模块需要进行硬件复位和初始化的设置，参见下面的流程图及程序。初始化结束后，模块的状态为：选择 Page 1 ，文本模式，正常字体，光标闪烁处于屏幕的右上角。



初始化流程图

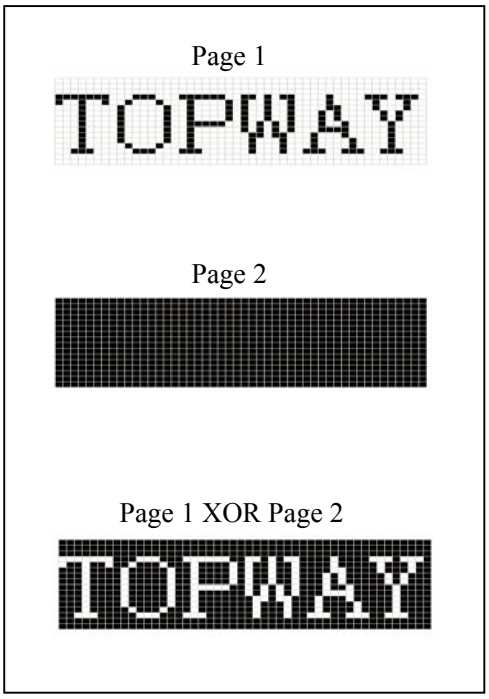
```
_RST=0;                  //复位开始
delay(time);             //延迟 250ms 以上
_RST=1;                  //复位结束
delay(time);             //延迟 50ms

CmdWrite (0x00,0xcd); //文本模式，开显示
CmdWrite (0x90,0x0d); //移位时钟的设定
CmdWrite (0xf0,0xa0);  //简体中文 ROM，ASCII 区块 0
CmdWrite (0x11,0x00); //光标高度为 1，行距为 0

CmdWrite (0x20,0x17); //工作视窗参数的设定
CmdWrite (0x30,0x7f);
CmdWrite (0x40,0x00);
CmdWrite (0x50,0x00);

CmdWrite (0x21,0x17); //显示视窗参数的设定
CmdWrite (0x31,0x80);
CmdWrite (0x41,0x00);
CmdWrite (0x51,0x00);
```

2.7 应用举例



1）选择 Page1, 文本模式下写入字符“TOPWAY”

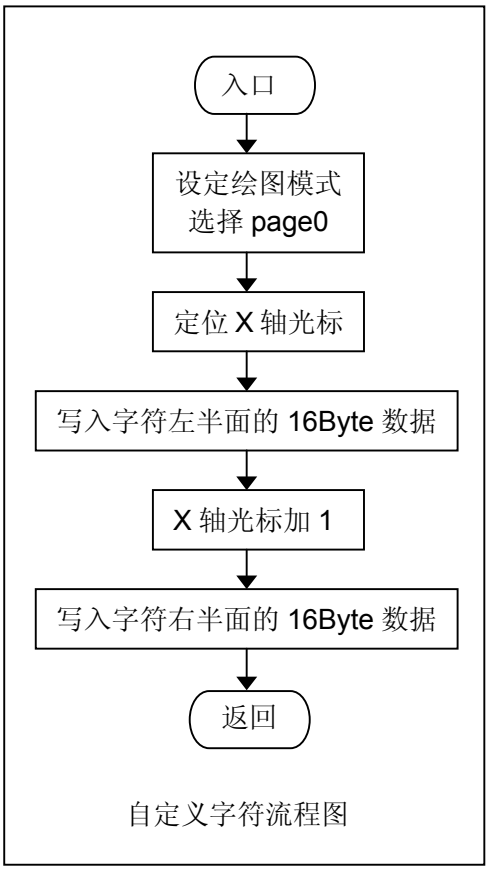
```
CmdWrite (0x12,0x91); //选择 Page1 进行读写
CmdWrite (0x00,0xcd); //文本模式，显示开
CmdWrite (0x60,0x09); //定义 X 轴光标
CmdWrite (0x70,0x38); //定义 Y 轴光标
Printstr ("TOPWAY");  // 调用子函数，写入字符串
delay (time);         //延迟一段时间
```

2）选择 Page2, 图形模式下写入数据“0xff”

```
CmdWrite (0x12,0xa2); //选择 Page2 进行读写
CmdWrite (0x00,0xc5); //图形模式，显示开
CmdWrite (0x60,0x00); //定义 X 轴光标
CmdWrite (0x70,0x00); //定义 Y 轴光标
FullScreenFill(0xff);  // 调用子函数，全屏点亮
delay (time);         //延迟一段时间
```

3）两图层合成，Page1 “XOR” Page2

```
CmdWrite (0x12,0xb7); //Page1“XOR”Page2
delay (time);         //延迟一段时间
```



自定义字符流程图

4）自定义字符

本模块提供给用户 16 个自定义字符，可以写入 SRAM(Page0)，代码为 FFF0H－FFFFH。写入过程见流程图，其中 X 轴光标为代码低 4 位的偶数倍。在文本模式下先写入代码高字节，再写入代码低字节，就可以显示出所定义的字符。

```
CmdWrite (0x00,0xc5); //选择图形模式
CmdWrite (0x12,0x10); //选择 Page0 SRAM
CmdWrite (0x60,0x00); //定义 X 轴光标
DataWrite (data);     //写入前 16 个数据
……
CmdWrite (0x60,0x01); //X 轴光标加 1
DataWrite (data);     //写入后 16 个数据
……

CmdWrite (0x00,0xcd); //文本模式，显示开
CmdWrite (0x12,0x91); //选择 Page1 进行读写
CmdWrite (0x60,0x00); //定义 X 轴光标
CmdWrite (0x70,0x00); //定义 Y 轴光标
DataWrite(0xff);      //写入自定义字符的高 8 位
DataWrite(0xf0);      //写入自定义字符的低 8 位
delay (time);         //延迟一段时间
```

参考程序

```
//---------------------------------------
//本例程的演示结果为:“拓普微 LM1095R”
//---------------------------------------
#include <reg52.h>
#include <intrins.h>
#define uchar unsigned char
#define uint unsigned int
#define LCD_BUS P1 //MCU P1<------> LCM

sbit _RD=P3^6;          //读信号
sbit _WR=P3^7;          //写信号
sbit RS=P3^1;           //寄存器选择
sbit CS=P3^4;           //片选信号
sbit _RST=P3^5;         //复位信号

//---------------------------------------
//延迟子程序
//---------------------------------------
void delay(uint t)
{
   uint i;
   uint j;
   for(j=0;j<t;j++);
      for(i=0;i<109;i++)
          _nop_();
}
//---------------------------------------
//写寄存器命令
//---------------------------------------
void CmdWrite(uchar cmdreg,uchar cmddata)
{
  LCD_BUS=cmdreg;     //写入寄存器的地址
  CS=0;
  _RD=1;
  RS=0;
  _WR=0;
  _nop_();
  _WR=1;
  _nop_();
  RS=1;
  CS=1;

  LCD_BUS=cmddata;    //写入参数
  CS=0;
  _RD=1;
  RS=0;
  _WR=0;
  _nop_();
  _WR=1;
  _nop_();
  RS=1;
  CS=1;
}

//---------------------------------------
//数据写操作
//---------------------------------------
void DataWrite(unsigned char wrdata)
{
  _RD=1;
  _WR=1;
  RS=1;
  LCD_BUS=wrdata;     //写入数据
  CS=0;
  _WR=0;
  _nop_();
  _WR=1;
  _nop_();
  RS=0;
  CS=1;
}

//---------------------------------------
//写入字符串
//---------------------------------------
void Printstr(uchar code *pstr)
{
    while(*pstr>0)
    {
        DataWrite(*pstr);
        pstr++;
        delay(1);
    }
}
//---------------------------------------
//整屏幕写入数据
//---------------------------------------
void FullScreenFill(uchar fill_data)
{
  uchar i,j;
  CmdWrite(0x60,0x00);
  CmdWrite(0x70,0x00);
  for(i=0;i<24;i++)
      for(j=0;j<128;j++)
          DataWrite(fill_data);
}
//---------------------------------------
//模块复位与初始化
//---------------------------------------
void LCD_Initial()
{
  _RST=0;                  //复位开始
  delay(250);
  _RST=1;
  delay(50);              //复位结束

  CmdWrite(0x00,0xcd);
  CmdWrite(0x90,0x0d);
  CmdWrite(0xf0,0xa0);
  CmdWrite(0x11,0x00);

  CmdWrite(0x20,0x17);    //工作视窗参数的设定
  CmdWrite(0x30,0x7f);
  CmdWrite(0x40,0x00);
  CmdWrite(0x50,0x00);

  CmdWrite(0x21,0x17);    //显示视窗参数的设定
  CmdWrite(0x31,0x80);
  CmdWrite(0x41,0x00);
  CmdWrite(0x51,0x00);

 }
//---------------------------------------
//主程序
//---------------------------------------
void main()
{
  _RD=1;
  _WR=1;
  RS=1;
  CS=1;
  LCD_BUS=0xff;
  LCD_Initial();

  CmdWrite(0x12,0x91);    //选择 page1 进行读写
  CmdWrite(0x00,0xcd);    //文本模式，开显示
  FullScreenFill(0x00);   //清屏
  CmdWrite(0x60,0x06);
  CmdWrite(0x70,0x38);
  Printstr("拓普微 LM1095R");
  while(1)
  {};
}                         //end of program
```