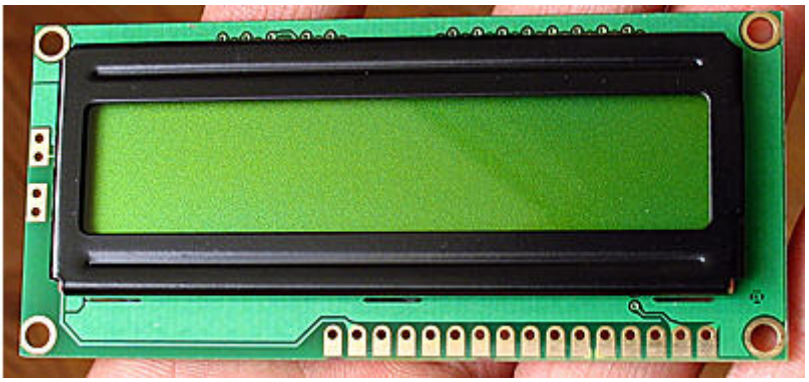
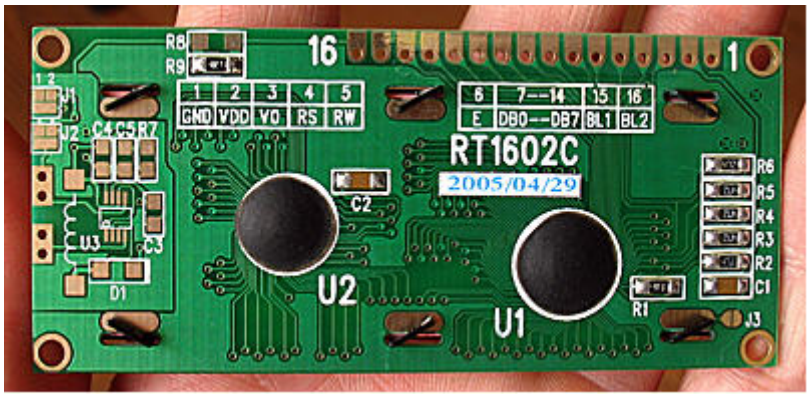
1602 详细资料和实例

1602 字符液晶在实际的产品中运用的也比较多了，前几天留意了一下，发现宿舍门前的自动售水机就是采用的 1602 液晶进行显示的。而且对于单片机的学习而言，掌握 1602 的用法是每一个学习者必然要经历的过程。在此，我将使用 1602 过程中遇到的问题以及感受记录下来，希望能够给初学者带来一点指导，少走一点弯路。

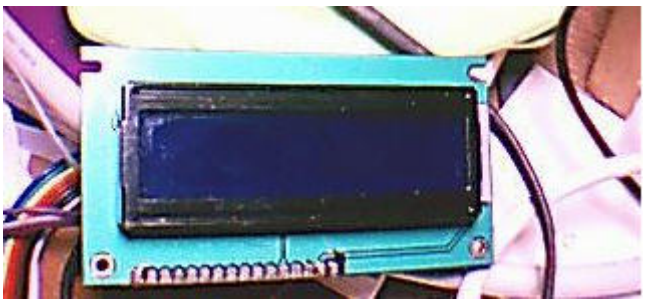
所谓 1602 是指显示的内容为 16*2,即可以显示两行，每行 16 个字符。目前市面上字符液晶绝大多数是基于 HD44780 液晶芯片的，控制原理是完全相同的，因此基于 HD44780 写的控制程序可以很方便地应用于市面上大部分的字符型液晶。



1602 液晶的正面(绿色背光，黑色字体)



1602 液晶背面(绿色背光，黑色字体)



另一种 1602 液晶模块，显示屏是蓝色背光白色字体

字符型 LCD1602 通常有 14 条引脚线或 16 条引脚线的 LCD，多出来的 2 条线是背光电源线 VCC(15 脚)和地线 GND(16 脚)，其控制原理与 14 脚的 LCD 完全一样，引脚定义如下表所示：

引脚号	引脚名	电平	输入/输出	作用
1	Vss			电源地
2	Vcc			电源(+5V)
3	Vee			对比调整电压
4	RS	0/1	输入	0=输入指令 1=输入数据
5	R/W	0/1	输入	0=向LCD写入指令或数据 1=从LCD读取信息
6	E	1,1→0	输入	使能信号，1时读取信息， 1→0(下降沿)执行指令
7	DB0	0/1	输入/输出	数据总线line0(最低位)
8	DB1	0/1	输入/输出	数据总线line1
9	DB2	0/1	输入/输出	数据总线line2
10	DB3	0/1	输入/输出	数据总线line3
11	DB4	0/1	输入/输出	数据总线line4
12	DB5	0/1	输入/输出	数据总线line5
13	DB6	0/1	输入/输出	数据总线line6
14	DB7	0/1	输入/输出	数据总线line7(最高位)
15	A	+Vcc		LCD背光电源正极
16	K	接地		LCD背光电源负极

HD44780 内置了DDRAM、CGROM和CGRAM。

DDRAM 就是显示数据 RAM，用来寄存待显示的字符代码。共 80 个字节，其地址和屏幕的对应关系如下表：

	显示位置	1	2	3	4	5	6	7	……	40
DDRAM 地 址	第一行	00H	01H	02H	03H	04H	05H	06H	……	27H
	第二行	40H	41H	42H	43H	44H	45H	46H	……	67H

也就是说想要在 LCD1602 屏幕的第一行第一列显示一个"A"字,就要向 DDRAM 的 00H 地址写入“A”字的代码（指 A 的字模代码，0x20～0x7F 为标准的 ASCII 码，通过这个代码，在 CGROM 中查找到相应的字符显示）就行了。但具体的写入是要按 LCD 模块的指令格式来进行的，后面我会说到的。那么一行可有 40 个地址呀？是的，在 1602 中我们就用前 16 个就行了。第二行也一样用前 16 个地址。对应如下：

DDRAM 地址与显示位置的对应关系

00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
40H	41H	42H	43H	44H	45H	46H	47H	48H	49H	4AH	4BH	4CH	4DH	4EH	4FH

（事实上我们往 DDRAM 里的 00H 地址处送一个数据，譬如 0x31(数字 1 的代码，见字模关系对照表)并不能显示 1 出来。这是一个令初学者很容易出错的地方，原因就是如果你要想在 DDRAM 的 00H 地址处显示数据，则必须将 00H 加上 80H，即 80H，若要在 DDRAM 的 01H 处显示数据，则必须将 01H 加上 80H 即 81H。依次类推。大家看一下控制指令的的 8 条：DDRAM 地址的设定，即可以明白是怎么样的一回事了），1602 液晶模块内部的字符发生存储器（CGROM)已经存储了 160 个不同的点阵字符图形（无汉字），如下表所示，这些字符有：阿拉伯数字、英文字母的大小写、常用的符号、和日文假名等，每一个字符都有一个固定的代码，比如大写的英文字母“A”的代码是 01000001B（41H），显示时模块把地址 41H 中的点阵字符图形显示出来，我们就能看到字母“A”

上表中的字符代码与我们 PC 中的字符代码是基本一致的。因此我们在向 DDRAM 写 C51 字符代码程序时甚至可以直接用 P1＝'A'这样的方法。PC 在编译时就把“A”先转为 41H 代码了。

字符代码 0x00～0x0F 为用户自定义的字符图形 RAM(对于 5X8 点阵的字符，可以存放 8 组，5X10 点阵的字符，存放 4 组)，就是 CGRAM 了。后面我会详细说的。

0x20～0x7F 为标准的 ASCII 码，0xA0～0xFF 为日文字符和希腊文字符，其余字符码(0x10～0x1F 及 0x80～0x9F)没有定义。

那么如何对 DDRAM 的内容和地址进行具体操作呢，下面先说说 HD44780 的指令集及其设置说明，请浏览该指令集，并找出对 DDRAM 的内容和地址进行操作的指令。共 11 条指令：

HD44780 的指令集

1.清屏指令

指令功能	指令编码										执行时间/ms
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
清屏	0	0	0	0	0	0	0	0	0	1	1.64

功能：<1> 清除液晶显示器，即将 DDRAM 的内容全部填入"空白"的 ASCII 码 20H;
<2> 光标归位，即将光标撤回液晶显示屏的左上方;
<3> 将地址计数器(AC)的值设为 0。

2.光标归位指令

指令功能	指令编码										执行时间/ms
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
光标归位	0	0	0	0	0	0	0	0	1	X	1.64

功能：<1> 把光标撤回到显示器的左上方;
<2> 把地址计数器(AC)的值设置为 0;
<3> 保持 DDRAM 的内容不变

3.进入模式设置指令

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
进入模式设置	0	0	0	0	0	0	0	1	I/D	S	40

功能：设定每次定入 1 位数据后光标的移位方向，并且设定每次写入的一个字符是否移动。参数设定的情况如下所示：

位名	设置
I/D	0=写入新数据后光标左移
	1=写入新数据后光标右移
S	0=写入新数据后显示屏不移动
	1=写入新数据后显示屏整体右移 1 个字

4.显示开关控制指令

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
显示开关控制	0	0	0	0	0	0	1	D	C	B	40

功能：控制显示器开/关、光标显示/关闭以及光标是否闪烁。参数设定的情况如下：

位名	设置	
D	0=显示功能关	1=显示功能开
C	0=无光标	1=有光标
B	0=光标闪烁	1=光标不闪烁

5.设定显示屏或光标移动方向指令

指令功能	指令编码										执行时间 /us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
设定显示屏或光标移动方向	0	0	0	0	0	1	S/C	R/L	X	X	40

功能：使光标移位或使整个显示屏幕移位。参数设定的情况如下：

S/C	R/L	设定情况
0	0	光标左移 1 格，且 AC 值减 1
0	1	光标右移 1 格，且 AC 值加 1
1	0	显示器上字符全部左移一格，但光标不动
1	1	显示器上字符全部右移一格，但光标不动

6.功能设定指令

指令功能	指令编码										执行时间 /us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
功能设定	0	0	0	0	1	DL	N	F	X	X	40

功能：设定数据总线位数、显示的行数及字型。参数设定的情况如下：

位名	设置
DL	0=数据总线为 4 位
	1=数据总线为 8 位
N	0=显示 1 行
	1=显示 2 行
F	0=5×7 点阵/每字符
	1=5×10 点阵/每字符

7.设定 CGRAM 地址指令

指令功能	指令编码										执行时间 /us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
设定CGRAM地址	0	0	0	1	CGRAM的地址(6位)						40

功能：设定下一个要存入数据的 CGRAM 的地址。

8.设定 DDRAM 地址指令

指令功能	指令编码										执行时间 /us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
设定DDRAM地址	0	0	1	CGRAM的地址(7位)							40

功能：设定

下一个要存入数据的 CGRAM 的地址。
(注意这里我们送地址的时候应该是 0x80+Address，这也是前面说到写地址命令的时候要加上 0x80 的原因)

9.读取忙信号或 AC 地址指令

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
读取忙碌信号或AC地址	0	1	FB	AC内容(7位)							40

功能：<1>
读取忙碌信号 BF 的内容，BF=1 表示液晶显示器忙，暂时无法接收单片机送来的数据或指令;
当 BF=0 时，液晶显示器可以接收单片机送来的数据或指令;
<2> 读取地址计数器(AC)的内容。

10.数据写入 DDRAM 或 CGRAM 指令一览

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
数据写入到DDRAM或CGRAM	1	0	要写入的数据 D7~D0								40

功能：<1>
将字符码写入 DDRAM，以使液晶显示屏显示出相对应的字符;
<2> 将使用者自己设计的图形存入 CGRAM。

11.从 CGRAM 或 DDRAM 读出数据的指令一览

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
从CGRAM或DDRAM读出数据	1	1	要读出的数据 D7~D0								40

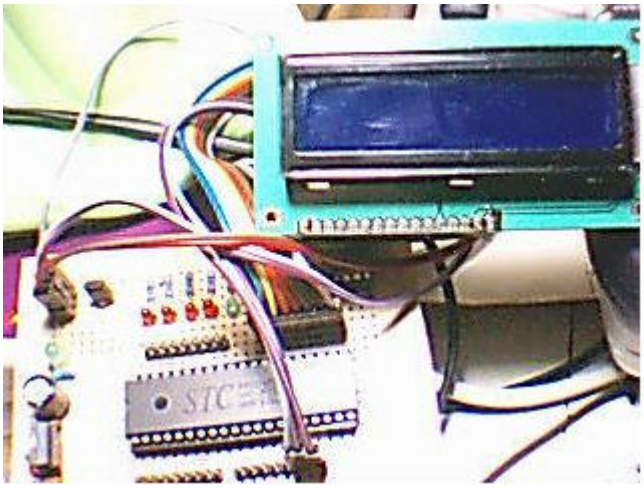
功能：读取 DDRAM 或 CGRAM 中的内容。

基本操作时序：

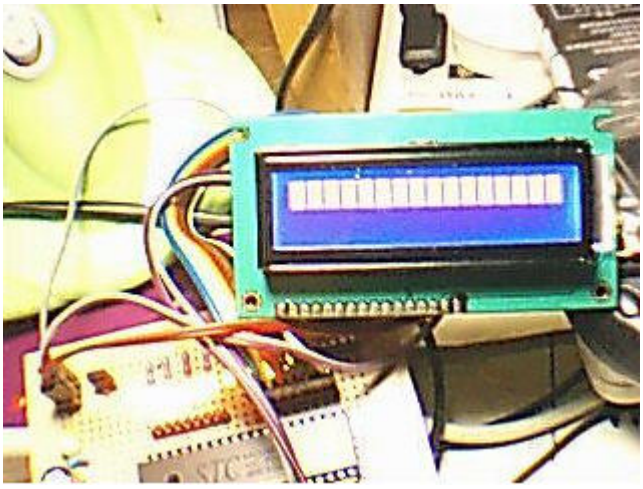
读状态　　输入：RS=L，RW=H，E=H
　　输出：DB0～DB7=状态字
写指令　　输入：RS=L，RW=L，E=下降沿脉冲，DB0～DB7=指令码
　　输出：无
读数据　　输入：RS=H，RW=H，E=H
　　输出：DB0～DB7=数据
写数据　　输入：RS=H，RW=L，E=下降沿脉冲，DB0～DB7=数据
　　输出：无

呵呵，看到这么多的控制指令希望你没有头晕。其实这么多的指令刚开始的时候没有必要全部掌握，随着学习的深入可以再尝试去用更复杂的控制指令。下面让我们一起驱动 1602 的液晶吧。下面是我的液晶的连接图，用的是那种蓝底白字的液晶，其实蓝底白字和那种绿底黑字的液晶唯一的区别就是颜色的问题，至于用哪种液晶，就看各位自己的喜好咯。

这就是我做测试用的最小系统，单片机是 STC89C516，晶振为 12M。液晶为蓝底白字的那种 1602。



当我们硬件连接错误，或者程序错误时就会出现下图这种情况，就是上排显示 16 的白色的块（蓝底黑字的液晶则显示的是 16 个黑块）。



下面我们来驱动 1602 吧在 1602 的上排显示“LCD1602 check ok”下排显示“study up”程序中没有用到忙检测，而是用的是延时函数来替代忙检测

```
#include<reg52.h>          //包含头文件，这个嘛，就不用多说了～～
#define uint unsigned int  //预定义一下
#define uchar unsigned char
sbit rs=P3^5;          //1602 的数据/指令选择控制线
sbit rw=P3^6;          //1602 的读写控制线
sbit en=P3^7;          //1602 的使能控制线
```

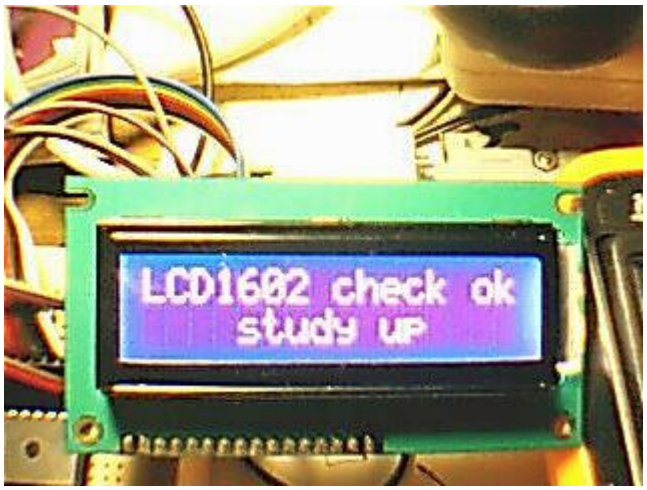
```
/*P2 口接 1602 的 D0~D7，注意不要接错了顺序，我以前可在这上面吃过亏～*/
uchar code table[]="LCD1602 check ok"; //要显示的内容 1 放入数组 table
uchar code table1[]="study up";              //要显示的内容 2 放入数组 table1
void delay(uint n)       //延时函数
{
   uint x,y;
   for(x=n;x>0;x--)
      for(y=110;y>0;y--);
}
/*****************************************************************************/
```

- void lcd_wcom(uchar com) //1602 写命令函数 (单片机给 1602 写命令)

```
    {                                  //1602 接收到命令后，不用存储，直接由 HD44780 执行并产生相应动作
       rs=0;               //选择指令寄存器
       rw=0;               //选择写
       P2=com;             //把命令字送入 P2
       delay(5);           //延时一小会儿，让 1602 准备接收数据
       en=1;               //使能线电平变化，命令送入 1602 的 8 位数据口
       en=0;
    }
    void lcd_wdat(uchar dat)       //1602 写数据函数
    {
       rs=1;            //选择数据寄存器
       rw=0;           //选择写
       P2=dat;         //把要显示的数据送入 P2
       delay(5);       //延时一小会儿，让 1602 准备接收数据
       en=1;           //使能线电平变化，数据送入 1602 的 8 位数据口
       en=0;
    }
    void lcd_init()             //1602 初始化函数
    {
       lcd_wcom(0x38);       //8 位数据，双列，5*7 字形
       lcd_wcom(0x0c);       //开启显示屏，关光标，光标不闪烁
       lcd_wcom(0x06);       //显示地址递增，即写一个数据后，显示位置右移一位
       lcd_wcom(0x01);       //清屏
    }
    void main()          //主函数
    {
       uchar n,m=0;
       lcd_init();               //液晶初始化
       lcd_wcom(0x80);   //显示地址设为 80H（即 00H，）上排第一位（也是执行一条命令）
       for(m=0;m<16;m++)     //将 table[]中的数据依次写入 1602 显示
       {
            lcd_wdat(table[m]);
            delay(200);
```

```
    }
    lcd_wcom(0x80+0x44); //重新设定显示地址为 0xc4,即下排第 5 位
    for(n=0;n<8;n++)   //将 table1[]中的数据依次写入 1602 显示
    {
        lcd_wdat(table1[n]);
        delay(200);
    }
    while(1);       //动态停机
}
```

程序写好后烧写进单片机，现在让我们看看效果吧



这就是显示的效果。

下面让我们来看看如何显示一个自定义的字符吧

我们从 CGROM 表上可以看到，在表的最左边是一列可以允许用户自定义的 CGRAM，从上往下看着是 16 个，实际只有 8 个字节可用。它的字符码是 00000000－00000111 这 8 个地址，表的下面还有 8 个字节，但因为这个 CGRAM 的字符码规定 0－2 位为地址，3 位无效，4－7 全为零。因此 CGRAM 的字符码只有最后三位能用也就是 8 个字节了。等效为 0000X111，X 为无效位，最后三位为 000－111 共 8 个。

如果我们要想显示这 8 个用户自定义的字符，操作方法和显示 CGROM 的一样，先设置 DDRAM 位置，再向 DDRAM 写入字符码，例如“A”就是 41H。现在我们要显示 CGRAM 的第一个自定义字符，就向 DDRAM 写入 00000000B(00H),如果要显示第 8 个就写入 00000111(08H),简单吧！

好！现在我们来看怎么向这八个自定义字符写入字模。有个设置 CGRAM 地址的指令大家还记得吗？赶快再找出来看看。

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
设定CGRAM地址	0	0	0	1	CGRAM的地址(6位)						40

从这个指令可以看出指令数据的高 2 位已固定是 01，只有后面的 6 位是地址数据，而这 6 位中的高 3 位就表示这八个自定义字符，最后的 3 位就是字模数据的八个地址了。例如第一个自定义字符的字模地址为 01000000－01000111 八个地址。我们向这 8 个字节写入字模数据，让它能显示出“℃”

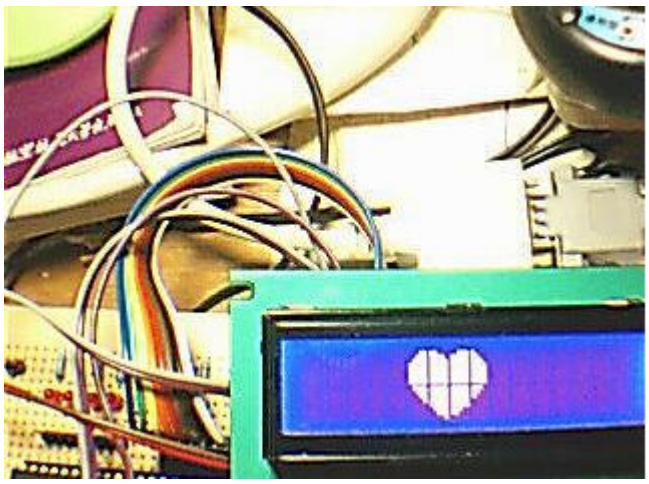
地址：01000000	数据：00010000	图示:○○○■○○○○
01000001	00000110	○○○○○■■○
01000010	00001001	○○○○■○○■
01000011	00001000	○○○○■○○○
01000100	00001000	○○○○■○○○
01000101	00001001	○○○○■○○■
01000110	00000110	○○○○○■■○
01000111	00000000	○○○○○○○○

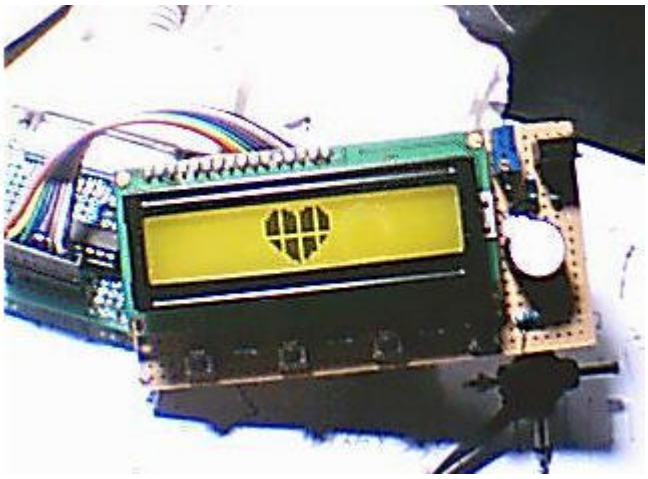
下面我们写一段程序让这 8 个自定义字符显示出一个心的图案：（由于上面那个显示程序已经有很详细的注释了，因此这个程序只对与上个程序不同的地方写注释）

```
#include<reg52.h>
#define uint unsigned int
#define uchar unsigned char
sbit rs=P3^5;
sbit rw=P3^6;
sbit en=P3^7;
uchar code table[]={0x03,0x07,0x0f,0x1f,0x1f,0x1f,0x1f,0x1f,
                    0x18,0x1E,0x1f,0x1f,0x1f,0x1f,0x1f,0x1f,
                    0x07,0x1f,0x1f,0x1f,0x1f,0x1f,0x1f,0x1f,
                    0x10,0x18,0x1c,0x1E,0x1E,0x1E,0x1E,0x1E,
                    0x0f,0x07,0x03,0x01,0x00,0x00,0x00,0x00,
                    0x1f,0x1f,0x1f,0x1f,0x1f,0x0f,0x07,0x01,
                    0x1f,0x1f,0x1f,0x1f,0x1f,0x1c,0x18,0x00,
                    0x1c,0x18,0x10,0x00,0x00,0x00,0x00,0x00};//心图案
/*uchar  code table1[]={0x10,0x06,0x09,0x08,0x08,0x09,0x06,0x00};//字符℃  */
void delay(uint n)
{
   uint x,y;
   for(x=n;x>0;x--)
         for(y=110;y>0;y--);
}
void lcd_wcom(uchar com)
{
   rs=0;
   rw=0;
   P2=com;
   delay(5);
   en=1;
   en=0;
}
```

```
void lcd_wdat(uchar dat)
{
   rs=1;
   rw=0;
   P2=dat;
   delay(5);
   en=1;
   en=0;
}
void lcd_init()
{
        lcd_wcom(0x38);
        lcd_wcom(0x0c);
        lcd_wcom(0x06);
        lcd_wcom(0x01);
}
void main()
{
                char m=0;
   lcd_init();
   lcd_wcom(0x40);             //设定 CGRAM 地址
   for(m=0;m<64;m++)    //将心型代码写入 CGRAM 中
   {
        lcd_wdat(table[m]);
   }
   lcd_wcom(0x85);       //设定上排的显示位置
   for(m=0;m<4;m++)       //显示心型图案的上半部分
   {
        lcd_wdat(m);
   }
   lcd_wcom(0xc5);       //将显示坐标转移到下排和上排相对应的地方
   for(m=4;m<8;m++)       //显示心型图案的下半部分
   {
        lcd_wdat(m);
   }
   while(1);
}
```

让我们一起来看看显示的效果吧～～
在绿底黑字液晶模块和蓝底白字液晶模块上分别显示的效果。

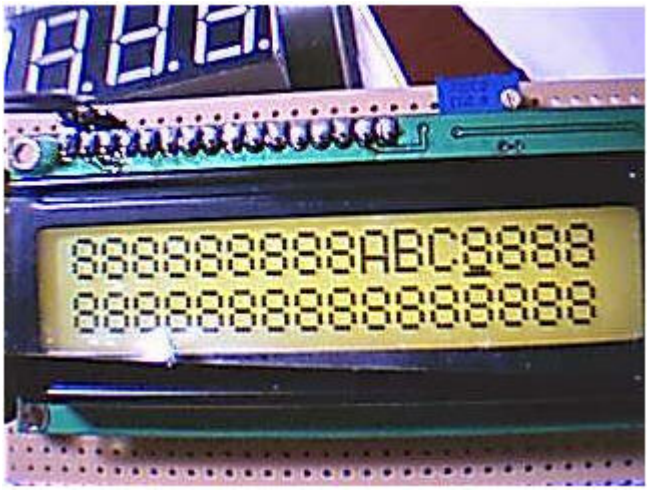




下面再为大家展示几种可能出现的问题

1：通电之后，程序也烧写进去了，但是 1602 就是不显示，只显示一排黑块（一般都是在上排 8 个小黑块，记得刚开始用 1602 液晶的时候，被这个整怕了～～），怎么样，你郁闷了吧，其实出现这种问题的原因无非以下几种：硬件连线上的错误，这种错误一般用万用表仔细检查后很容易找出来。第二种情况就是硬件连接上是正确的，那么此时出问题最大的就是程序上了，如果你用的是忙检测，看一下忙检测函数写对了没，如果用的是延时函数，那么看看延时的时间是否够长。再就是看看时序图，这点很重要的哦。如果硬件和软件都没有错，那么就要考虑 1602 是否坏了，但是出现这种情况的几率很小，如果遇到这种情况，你可以考虑去买彩票了～～

下面这种情况你遇到过吗？我遇到过了的，搞得我很是郁闷～～



我做的实验是要液晶显示 ABC 这三个字母，并且开光标，光标闪烁。大家可以在第一排的最后几位看到 ABC 和光标都已经显示出来了。但是为什么其它位会显示这么多 8 呢？嘿嘿～～郁闷吧。出现这种情况的原因就是在初始化液晶的时候，要把清屏指令放在最后面，否则就会出现上图这种情况。怎么样，第一次听说吧～不过，我不知道其它的液晶是否也有这个问题出现，至少我用的这块就有这种情况，但是我的另一个液晶则没有这种情况出现，不管是在一开始就清屏还是最后清屏。大家注意下就可以了，万一出现了这种情况，就会处理了～～

上面这张图是用 1602 作为显示的温度电子钟～～上面的年月日三个字就是用自定义字符的方法显示的。呵呵，怎么样～～到此 1602 的驱动基本上结束了，剩下的就靠大家自己去发挥了