在日常生活中，我们对液晶显示器并不陌生。液晶显示模块已作为很多电子产品的通过器件，如在计算器、万用表、电子表及很多家用电子产品中都可以看到，显示的主要是数字、专用符号和图形。在单片机的人机交流界面中，一般的输出方式有以下几种：发光管、LED 数码管、液晶显示器。发光管和 LED 数码管比较常用，软硬件都比较简单，在前面章节已经介绍过，在此不作介绍，本章重点介绍字符型液晶显示器的应用。

在单片机系统中应用晶液显示器作为输出器件有以下几个优点：

显示质量高

由于液晶显示器每一个点在收到信号后就一直保持那种色彩和亮度，恒定发光，而不像阴极射线管显示器（CRT）那样需要不断刷新新亮点。因此，液晶显示器画质高且不会闪烁。

数字式接口

液晶显示器都是数字式的，和单片机系统的接口更加简单可靠，操作更加方便。

体积小、重量轻

液晶显示器通过显示屏上的电极控制液晶分子状态来达到显示的目的，在重量上比相同显示面积的传统显示器要轻得多。

功耗低

相对而言，液晶显示器的功耗主要消耗在其内部的电极和驱动 IC 上，因而耗电量比其它显示器要少得多。

10．8．1 液晶显示简介

①液晶显示原理

液晶显示的原理是利用液晶的物理特性，通过电压对其显示区域进行控制，有电就有显示，这样即可以显示出图形。液晶显示器具有厚度薄、适用于大规模集成电路直接驱动、易于实现全彩色显示的特点，目前已经被广泛应用在便携式电脑、数字摄像机、PDA 移动通信工具等众多领域。

②液晶显示器的分类

液晶显示的分类方法有很多种，通常可按其显示方式分为段式、字符式、点阵式等。除了黑白显示外，液晶显示器还有多灰度有彩色显示等。如果根据驱动方式来分，可以分为静态驱动（Static）、单纯矩阵驱动（Simple Matrix）和主动矩阵驱动（Active Matrix）三种。

③液晶显示器各种图形的显示原理:

线段的显示

点阵图形式液晶由 M×N 个显示单元组成，假设 LCD 显示屏有 64 行，每行有 128 列，每 8 列对应 1 字节的 8 位，即每行由 16 字节，共 16×8=128 个点组成，屏上 64×16 个显示单元与显示 RAM 区 1024 字节相对应，每一字节的内容和显示屏上相应位置的亮暗对应。例如屏的第一行的亮暗由 RAM 区的 000H——00FH 的 16 字节的内容决定，当（000H）=FFH 时，则屏幕的左上角显示一条短亮线，长度为 8 个点；当（3FFH）=FFH 时，则屏幕的右下角显示一条短亮线；当（000H）=FFH，（001H）=00H，（002H）=00H，……（00EH）=00H，（00FH）=00H 时，则在屏幕的顶部显示一条由 8 段亮线和 8 条暗线组成的虚线。这就是 LCD 显示的基本原理。

字符的显示

用 LCD 显示一个字符时比较复杂，因为一个字符由 6×8 或 8×8 点阵组成，既要找到和显示屏幕上某几个位置对应的显示 RAM 区的 8 字节，还要使每字节的不同位为“1”，其它的为“0”，为“1”的点亮，为“0”的不亮。这样一来就组成某个字符。但由于内带字符发生器的控制器来说，显示字符就比较简单了，可以让控制器工作在文本方式，根据在 LCD 上开始显示的行列号及每行的列数找出显示 RAM 对应的地址，设立光标，在此送上该字符对应的代

码即可。

汉字的显示

汉字的显示一般采用图形的方式，事先从微机中提取要显示的汉字的点阵码（一般用字模提取软件），每个汉字占 32B，分左右两半，各占 16B，左边为 1、3、5……右边为 2、4、6……根据在 LCD 上开始显示的行列号及每行的列数可找出显示 RAM 对应的地址，设立光标，送上要显示的汉字的第一字节，光标位置加 1，送第二个字节，换行按列对齐，送第三个字节……直到 32B 显示完就可以 LCD 上得到一个完整汉字。

10．8．2 1602 字符型 LCD 简介

字符型液晶显示模块是一种专门用于显示字母、数字、符号等点阵式 LCD，目前常用 16*1，16*2，20*2 和 40*2 行等的模块。下面以长沙太阳人电子有限公司的 1602 字符型液晶显示器为例，介绍其用法。一般 1602 字符型液晶显示器实物如图 10-53：

图 10-53 1602 字符型液晶显示器实物图

10．8.2.1 1602LCD 的基本参数及引脚功能

1602LCD 分为带背光和不带背光两种，基控制器大部分为 HD44780，带背光的比不带背光的厚，是否带背光在应用中并无差别，两者尺寸差别如下图 10-54 所示：

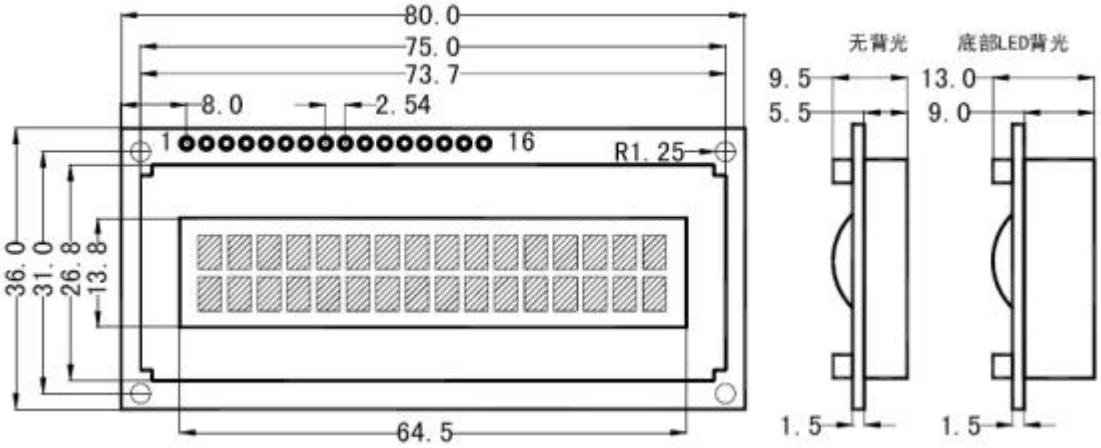

图 10-54 1602LCD 尺寸图

1602LCD 主要技术参数：

显示容量:16×2 个字符

芯片工作电压:4.5—5.5V

工作电流:2.0mA(5.0V)

模块最佳工作电压:5.0V

字符尺寸:2.95×4.35(W×H)mm

引脚功能说明

1602LCD 采用标准的 14 脚（无背光）或 16 脚（带背光）接口，各引脚接口说明如表 10-13 所示:

编号	符号	引脚说明	编号	符号	引脚说明
1	VSS	电源地	9	D2	数据
2	VDD	电源正极	10	D3	数据
3	VL	液晶显示偏压	11	D4	数据
4	RS	数据/命令选择	12	D5	数据
5	R/W	读/写选择	13	D6	数据
6	E	使能信号	14	D7	数据
7	D0	数据	15	BLA	背光源正极
8	D1	数据	16	BLK	背光源负极

表 10-13：引脚接口说明表

第 1 脚：VSS 为地电源。

第 2 脚：VDD 接 5V 正电源。

第 3 脚：VL 为液晶显示器对比度调整端，接正电源时对比度最弱，接地时对比度最高，对比度过高时会产生“鬼影”，使用时可以通过一个 10K 的电位器调整对比度。

第 4 脚：RS 为寄存器选择，高电平时选择数据寄存器、低电平时选择指令寄存器。

第 5 脚：R/W 为读写信号线，高电平时进行读操作，低电平时进行写操作。当 RS 和 R/W 共同为低电平时可以写入指令或者显示地址，当 RS 为低电平 R/W 为高电平时可以读忙信号，当 RS 为高电平 R/W 为低电平时可以写入数据。

第 6 脚：E 端为使能端，当 E 端由高电平跳变成低电平时，液晶模块执行命令。

第 7～14 脚：D0～D7 为 8 位双向数据线。

第 15 脚：背光源正极。

第 16 脚：背光源负极。

10．8．2．3 1602LCD 的指令说明及时序

1602 液晶模块内部的控制器共有 11 条控制指令，如表 10-14 所示：

序号	指令	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
1	清显示	0	0	0	0	0	0	0	0	0	1
2	光标返回	0	0	0	0	0	0	0	0	1	*
3	置输入模式	0	0	0	0	0	0	0	1	I/D	S
4	显示开/关控制	0	0	0	0	0	0	1	D	C	B
5	光标或字符移位	0	0	0	0	0	1	S/C	R/L	*	*
6	置功能	0	0	0	0	1	DL	N	F	*	*
7	置字符发生存贮器地址	0	0	0	1	字符发生存贮器地址					
8	置数据存贮器地址	0	0	1	显示数据存贮器地址						
9	读忙标志或地址	0	1	BF	计数器地址						
10	写数到 CGRAM 或 DDRAM）	1	0	要写的数据内容							
11	从 CGRAM 或 DDRAM 读数	1	1	读出的数据内容							

表 10-14：控制命令表

1602 液晶模块的读写操作、屏幕和光标的操作都是通过指令编程来实现的。（说明：1 为高电平、0 为低电平）

指令 1：清显示，指令码 01H,光标复位到地址 00H 位置。

指令 2：光标复位，光标返回到地址 00H。

指令 3：光标和显示模式设置 I/D：光标移动方向，高电平右移，低电平左移 S:屏幕上所有文字是否左移或者右移。高电平表示有效，低电平则无效。

指令 4：显示开关控制。 D：控制整体显示的开与关，高电平表示开显示，低电平表示关显示 C：控制光标的开与关，高电平表示有光标，低电平表示无光标 B：控制光标是否闪烁，高电平闪烁，低电平不闪烁。

指令 5：光标或显示移位 S/C：高电平时移动显示的文字，低电平时移动光标。

指令 6：功能设置命令 DL：高电平时为 4 位总线，低电平时为 8 位总线 N：低电平时为单行显示，高电平时双行显示 F：低电平时显示 5x7 的点阵字符，高电平时显示 5x10 的点阵字符。

指令 7：字符发生器 RAM 地址设置。

指令 8：DDRAM 地址设置。

指令 9：读忙信号和光标地址 BF：为忙标志位，高电平表示忙，此时模块不能接收命令或者数据，如果为低电平表示不忙。

指令 10：写数据。

指令 11：读数据。

与 HD44780 相兼容的芯片时序表如下：

读状态	输入	RS=L，R/W=H，E=H	输出	D0—D7=状态字
写指令	输入	RS=L，R/W=L，D0—D7=指令码，E=高脉冲	输出	无
读数据	输入	RS=H，R/W=H，E=H	输出	D0—D7=数据
写数据	输入	RS=H，R/W=L，D0—D7=数据，E=高脉	输出	无

冲

表 10-15：基本操作时序表

读写操作时序如图 10-55 和 10-56 所示：

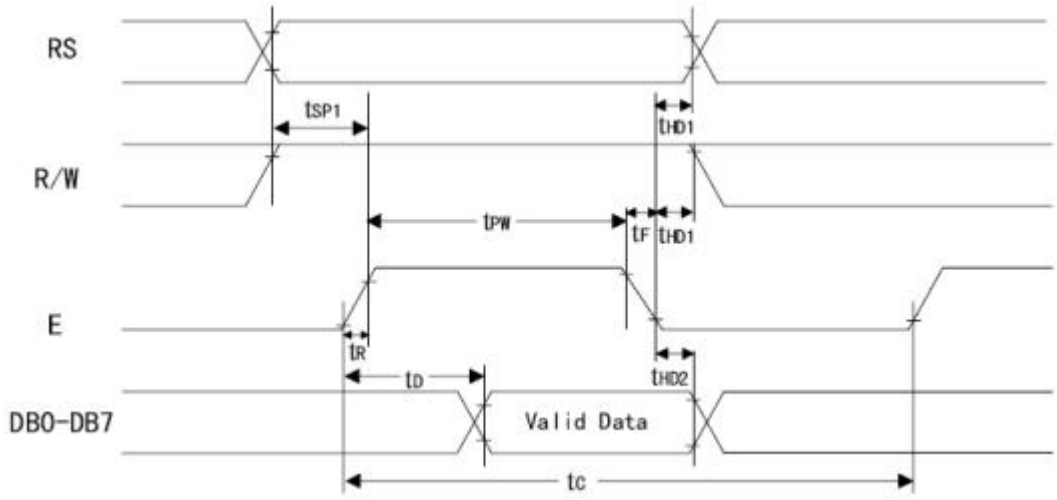

图 10-55 读操作时序

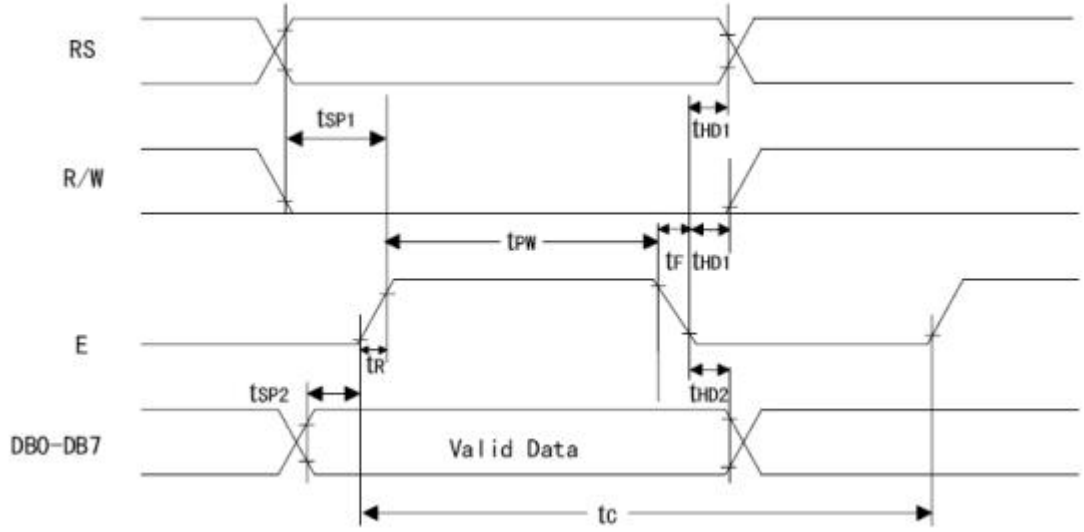

图 10-56 写操作时序

10．8．2．4 1602LCD 的 RAM 地址映射及标准字库表

液晶显示模块是一个慢显示器件，所以在执行每条指令之前一定要确认模块的忙标志为低电平，表示不忙，否则此指令失效。要显示字符时要先输入显示字符地址，也就是告诉模块在哪里显示字符，图 10-57 是 1602 的内部显示地址。

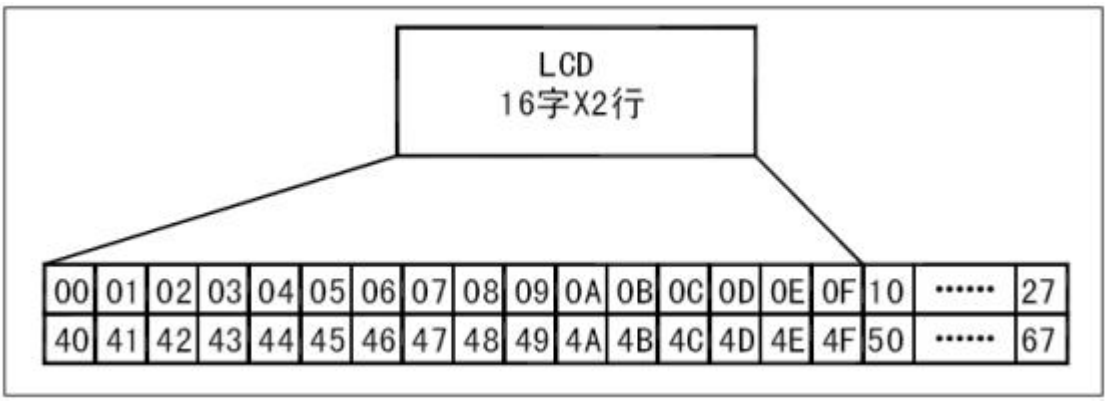

图 10-57 1602LCD 内部显示地址

例如第二行第一个字符的地址是 40H，那么是否直接写入 40H 就可以将光标定位在第二行第一个字符的位置呢？这样不行，因为写入显示地址时要求最高位 D7 恒定为高电平 1 所以实际写入的数据应该是 01000000B（40H）+10000000B(80H)=11000000B(C0H)。

在对液晶模块的初始化中要先设置其显示模式，在液晶模块显示字符时光标是自动右移的，无需人工干预。每次输入指令前都要判断液晶模块是否处于忙的状态。

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，如图 10-58 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”

表 13-4 CGROM 和 CGRAM 中字符代码与字符图形对应关系

高位 / 低位	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
××××0000	CGRAM (1)		0	@	P	\	p		ー	タ	三	α	P
××××0001	(2)	!	1	A	Q	a	q	口	ア	チ	ム	ä	q
××××0010	(3)	"	2	B	R	b	r	r	イ	川	メ	β	θ
××××0011	(4)	#	3	C	S	c	s	⌋	ウ	ラ	モ	ε	∞
××××0100	(5)	$	4	D	T	d	t	\	エ	ト	セ	μ	Ω
××××0101	(6)	%	5	E	U	e	u	ロ	オ	ナ	ユ	B	0
××××0110	(7)	&	6	F	V	f	v	テ	カ	ニ	ヨ	P	Σ
××××0111	(8)	>	7	G	W	g	w	ア	キ	ヌ	ラ	g	π
××××1000	(1)	(	8	H	X	h	x	イ	ク	ネ	リ	∫	X̄
××××1001	(2)	)	9	I	Y	i	y	ウ	ケ	⌋	ル	-1	y
××××1010	(3)	*	:	J	Z	j	z	エ	コ	リ	レ	j	千
××××1011	(4)	+	;	K	[	k	{	オ	サ	ヒ	ロ	x	万
××××1100	(5)	フ	<	L	¥	l	|	セ	シ	フ	ワ	¢	円
××××1101	(6)	-	=	M	]	m	}	ユ	ス	ヘ	ソ	ŧ	÷
××××1110	(7)	.	>	N	^	n	→	ヨ	セ	ホ	ハ	n̄	
××××1111	(8)	/	?	O	-	o	←	ッ	ソ	マ	ロ	ö	

图 10-58 字符代码与图形对应图 www.lovebjb.com

10．8．2．5 1602LCD 的一般初始化（复位）过程

延时 15mS

写指令 38H（不检测忙信号）

延时 5mS

写指令 38H（不检测忙信号）
延时 5mS
写指令 38H（不检测忙信号）
以后每次写指令、读/写数据操作均需要检测忙信号
写指令 38H：显示模式设置
写指令 08H：显示关闭
写指令 01H：显示清屏
写指令 06H：显示光标移动设置
写指令 0CH：显示开及光标设置

10．8．3 1602LCD 的软硬件设计实例

在 1602LCD 第一行显示网站名：www.hificat.com 在第二行显示联系电话：0571-85956028。
实验前应先将显示切换开关切换到 LCD 工作状态。

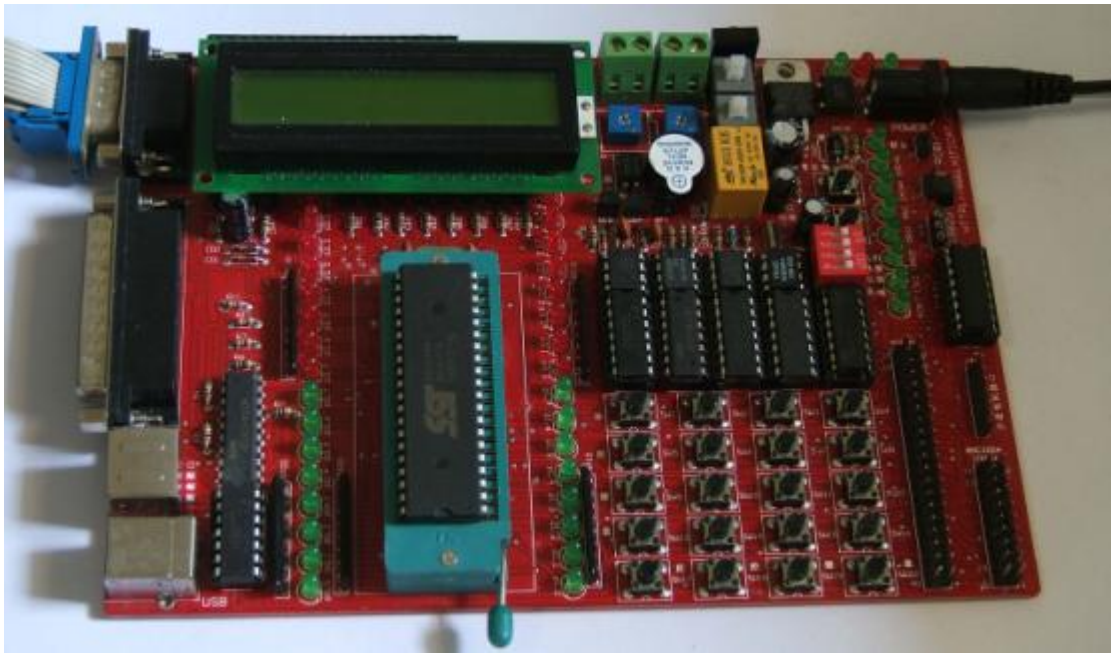

图 10-59 1602LCD 实验演示图

10．8．3．1 硬件原理图

1602 液晶显示模块可以和单片机 AT89C51 直接接口，电路如图 10-60 所示。

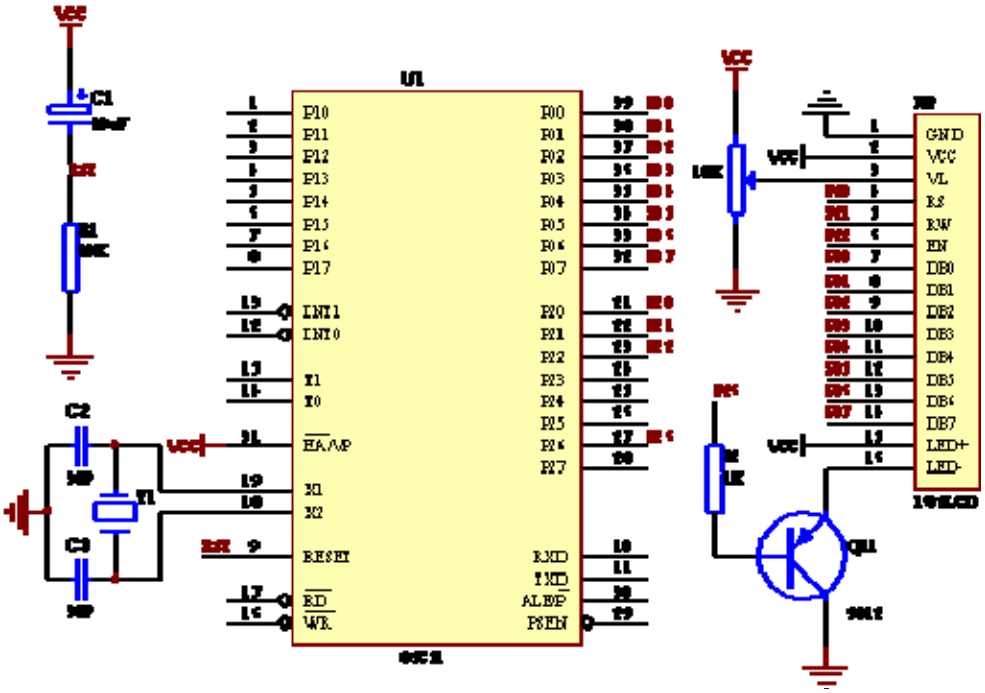

图 10-60 硬件原理图

10．8．3．2 程序流程图

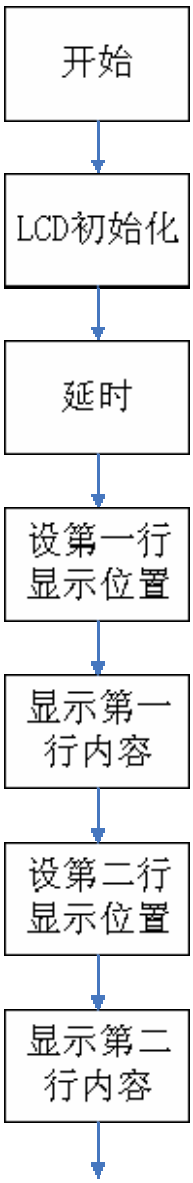

图 10-61 软件流程图

10．8．3．3 软件代码

```
#include <reg51.h>
#include <intrins.h>

sbit rs= P2^0;
```

```
sbit rw = P2^1;
sbit ep = P2^2;

unsigned char code dis1[] = {"www.hificat.com"};
unsigned char code dis2[] = {"0571-85956028"};

void delay(unsigned char ms)
{
unsigned char i;
while(ms--)
{
for(i = 0; i< 250; i++)
{
_nop_();
_nop_();
_nop_();
_nop_();
}
}
}

bit lcd_bz()
{
bit result;
rs = 0;
rw = 1;
ep = 1;
_nop_();
_nop_();
_nop_();
_nop_();
result = (bit)(P0 & 0x80);
ep = 0;
return result;
}

void lcd_wcmd(unsigned char cmd)
{
while(lcd_bz());//判断 LCD 是否忙碌
rs = 0;
rw = 0;
ep = 0;
_nop_();
_nop_();
```

```
P0 = cmd;
_nop_();
_nop_();
_nop_();
_nop_();
ep = 1;
_nop_();
_nop_();
_nop_();
_nop_();
ep = 0;
}

void lcd_pos(unsigned char pos)
{
lcd_wcmd(pos | 0x80);
}

void lcd_wdat(unsigned char dat)
{
while(lcd_bz());//判断 LCD 是否忙碌
rs = 1;
rw = 0;
ep = 0;
P0 = dat;
_nop_();
_nop_();
_nop_();
_nop_();
ep = 1;
_nop_();
_nop_();
_nop_();
_nop_();
ep = 0;
}

void lcd_init()
{
lcd_wcmd(0x38);
delay(1);
lcd_wcmd(0x0c);
delay(1);
lcd_wcmd(0x06);
```

```
delay(1);
lcd_wcmd(0x01);
delay(1);
}

void main(void)
{
unsigned char i;
lcd_init();// 初始化 LCD
delay(10);
lcd_pos(0x01);//设置显示位置
i = 0;
while(dis1[i] != '\0')
{
lcd_wdat(dis1[i]);//显示字符
i++;
}
lcd_pos(0x42);// 设置显示位置
i = 0;
while(dis2[i] != '\0')
{
lcd_wdat(dis2[i]);// 显示字符
i++;
}
```