第 4 章　函数发生器的设计及 Proteus 仿真

本章导读：

本章将介绍一种基于单片机和 Proteus 的函数发生器设计的方法，其中包括了系统方案的设计和各模块芯片的选型、硬件电路的设计、软件程序的编写及系统的仿真等完整的系统设计步骤。

4.1　设计任务书

在系统开始设计之前，设计者必须要了解本设计的目的、意义，有关的要求以及整个设计的时间安排，这样才能做到有目的有计划地完成系统设计。

4.1.1　设计的目的与意义

设计的目的：在了解单片机与函数发生器原理的基础上，利用单片机与相关芯片完成一款函数发生器的设计，并利用 Proteus 对系统进行仿真，验证设计的正确性与可行性。

设计的意义：函数发生器是测试领域使用极其广泛的工具，它可以为测试提供各种信号源，如正弦波、方波、锯齿波等，随着数字技术与计算机技术的高速发展，有些高档的函数发生器是可编程的，即产生任意的波形。在经过本科或专科的学习后，利用单片机技术设计一款较为简单的函数发生器，将书本上的知识加以利用，不仅可以很好地巩固所学的知识，而且可以培养动手能力，为即将到来的工作生涯做准备。

4.1.2　设计的要求

1. 完成系统的架构设计，在 Proteus 上搭建硬件平台。
2. 编写相关的软件程序。
3. 完成系统调试，使系统能正常工作。

4.1.3　设计及论文的时间安排

第一部分 阅读相关资料（2 周）。
第二部分 设计系统的总体方案并完成芯片的选型工作（3 周）。
第三部分 在 Proteus 下完成硬件平台的搭建（3 周）。
第四部分 编制相应的软件程序（3 周）。
第五部分 系统各部分分开调试以及系统总体联调（3 周）。
第六部分 完成论文写作准备答辩（2 周）。

4.1.4　摘要

本设计是利用 AT89C52 单片机与 DAC0832 设计一款可产生多种波形的函数发生器，并利用 Proteus 仿真软件对设计方案进行仿真，验证方案的可行性与正确性。整个设计中涉及单

片机定时/计数器、中断、按键扫描、液晶显示等知识，能有效地巩固所学知识。

论文首先给出了函数发生器的不同设计方法，分析了它们的优缺点，接着给出了本文的设计方法，详细介绍了硬件电路各部分的原理、软件的代码分析及 Proteus 仿真软件的使用方法。在各个模块的介绍中，都给出了详尽的流程图。

关键词：函数发生器，AT89C52，DAC0832，Proteus

4.2 引言

本课题研究来源于实践经验。本节将介绍研究背景以及本文的主要内容安排。

4.2.1 研究背景

函数发生器是电子信息领域中必不可少的调试设备，为待调试的系统提供精确的信号，如正弦波、三角波、锯齿波、方波等，有的函数发生器还具有调制的功能，可以进行调幅、调频、调相、脉宽调制和 VCO 控制。

虚拟仿真软件是电子测量领域的一个大变革，它通过软件实现对数据的显示、处理与分析，大大缩小了设计的成本。Proteus 就是一款功能强大，使用简便的电子仿真软件，也是目前世界上唯一将电路仿真软件、PCB 设计软件和虚拟模型仿真软件三合一的设计平台。

本设计利用 Proteus 这款 EDA 软件仿真、验证基于单片机与 DAC0832 的函数发生器。

4.2.2 本文研究的主要内容

本课题利用单片机 AT89C52、数模转换芯片 DAC0832 以及运算放大器设计了一个具有产生多种波形的函数发生器，其主要功能有：

- 可以产生正弦波、锯齿波、方波、三角波。
- 频率可以调节。
- 可以显示函数发生器运行的状态。

主要工作有：

- 设计系统方案。
- 在 Proteus 中搭建硬件平台。
- 编写相关软件程序。
- 调试运行，验证方案的正确性与可行性。

4.3 函数发生器方案的分析与设计

函数发生器的方案设计涉及系统的正确性与可行性，是整个设计中最为重要的环节，为了能在已有的条件下完成设计，必须要研究各个设计方法的优缺点，并对它们进行权衡与综合，既要能满足设计要求，又能在已有条件下顺利完成。本章首先介绍了几种函数发生器的设计方法，并对它们的特点进行分析，然后提出一种成本低、性价比高、容易实施的方案。

4.3.1 函数发生器的设计方法

函数发生器的设计一般有以下几种：

第一种：采用分立原件或运算放大器和选频网络组成振荡器，这种方法的优点是成本低，电路简单，但是波形的精度不高、稳定性较差、不容易与数字化设备连接。

第二种：运用专门的函数信号发生 IC，如美国马克西姆公司的高频精密函数发生器 MAX038，如图 4-1 所示。此类 IC 具备精度好、频率高等特点，能够精密的产生正弦波、方波、三角波等信号。一般专用的信号发生器芯片价格较为昂贵，这种方案适合高精度的电子测量与控制场合。如图 4-2 所示为 MAX038 产生的正弦波、三角波。

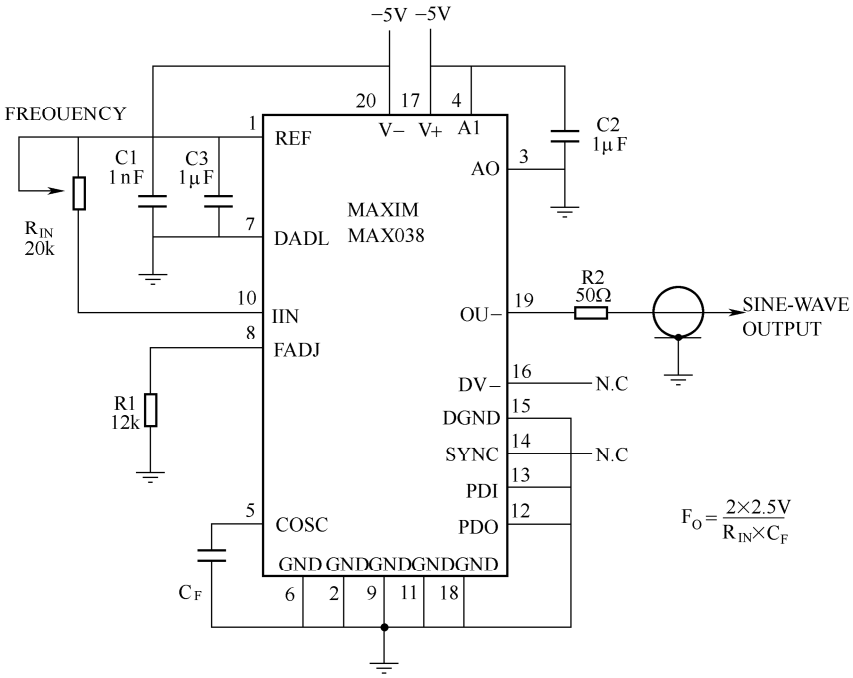

图 4-1　MAX038 应用图

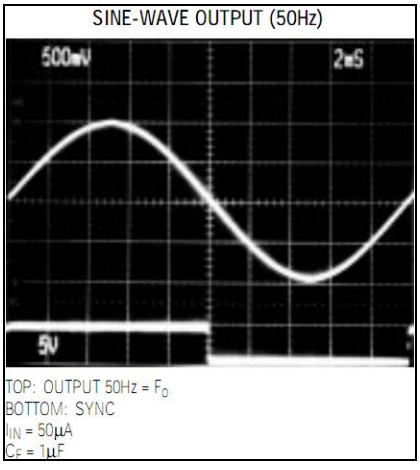

图 4-2　MAX038 产生的波形图

第三种：采用微控制器与 DA 转换芯片产生所需的信号，这种方法的最大特点是系统容易与数字化设备相连，如计算机等，产生的波形精度高、稳定性好。

第四种：20 世纪 70 年代发展起了一种 DDS 技术，它是一种频率合成技术，采用数字信号处理模块，参照一个频率固定且精确的时钟源，产生频率、相位均可调的信号。DDS 技术具有转换速度快、精度高等优点，基于此种技术的函数发生器有逐步取代传统的函数发生器之势。

4.3.2 总体方案设计

系统的架构可以分为两部分：第一、硬件系统；第二、软件系统。硬件系统由微控制器及其外围电路、键盘输入接口、液晶显示接口等组成。软件系统包括键盘识别程序、液晶显示程序、波形形成程序等。系统架构图如图 4-3 所示。

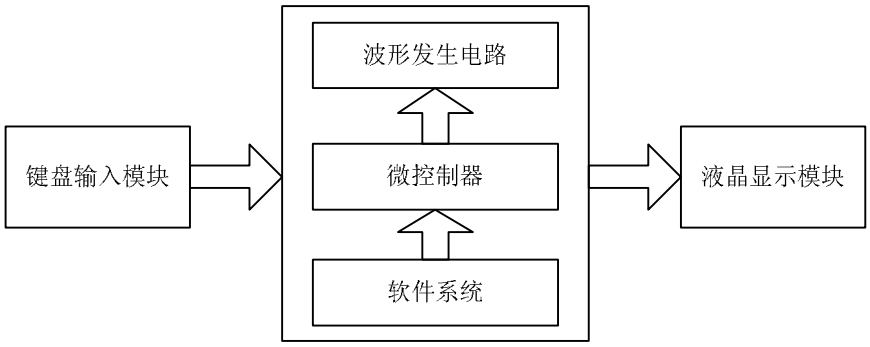

图 4-3 系统架构图

微控制器为 51 单片机，4×4 键盘构成了键盘输入模块，DAC0832 与运算放大器组成了波形发生电路，液晶显示模块由 1602 及其外围电路构成，在软件系统的控制下完成了函数发生器的功能。通过键盘输入所需要产生的波形及频率，波形的种类有正弦波、三角波、方波、锯齿波等，频率有 1kHz、2kHz、3kHz 等，单片机识别出键盘的键值，然后控制液晶显示屏显示相应的信息，并控制波形发生电路输出信号。

4.4 系统硬件各模块设计

硬件平台由单片机最小系统、键盘输入电路、液晶显示电路、D/A 转换电路等部分组成，本节将对各个模块作详细的介绍。

4.4.1 单片机最小系统

单片机、时钟电路、电源电路、复位电路构成了单片机最小系统，如图 4-4 所示。

时钟电路由晶振与电容等器件构成。复位电路由按键、电阻与电容等器件构成，RST 引脚是复位信号输入端，高电平有效，高电平持续时间需要 24 个时钟周期以上，比如时钟的频率为 12MHz，则复位信号至少要持续 2μs 以上。51 单片机的 EA 引脚是访问外部存储器的控制信号，EA=0 时，访问外部 ROM，EA=1 时，CPU 访问内部存储器或访问地址超过存储容量时自动执行外部程序存储器的程序。一般 EA 直接接高电平（+5V）。

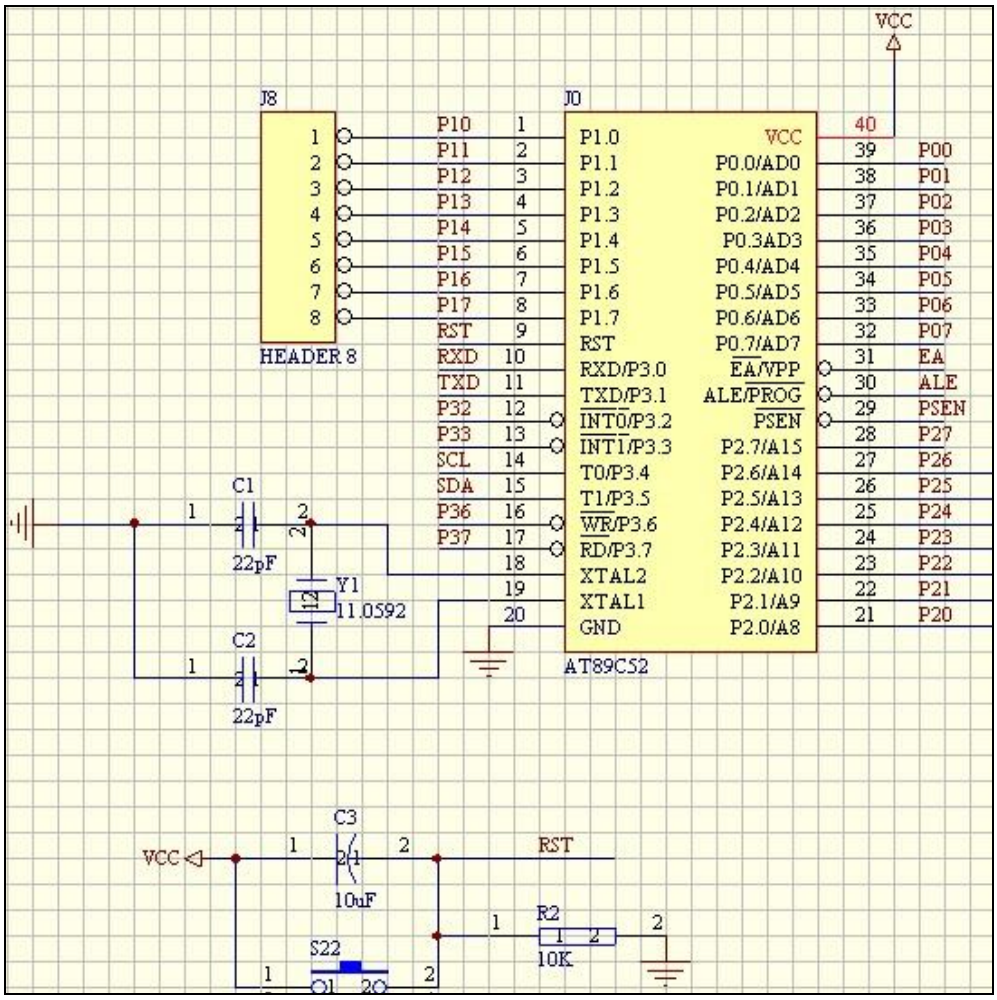

图 4-4　单片机最小系统

4.4.2　键盘输入电路

在一个需要人机交互的电子系统中，键盘是必不可少的输入装置。键盘是有许多按键开关组成的，一旦按键按下，单片机 I/O 口的电平会发生变化，单片机通过判断 I/O 口电平的变化来识别按键。在单片机的接口应用中，键盘接口一般分为两种：一种是独立式键盘，一种是矩阵式键盘。

独立式键盘的每一个按键都有一个信号线与单片机相连，每一个键互不影响，如图 4-5 所示。这种键盘的优点是结构简单，使用方便，但是缺点也是显而易见的，那就是占用资源过多，按键数目越多，占用的 I/O 口就越多，所以如果系统需要的按键比较多时，一般采用矩阵式键盘。

矩阵式键盘的按键连接在行、列线构成的矩阵电路的交叉处，每当有按键按下时通过该键将相应的行、列线连通，如图 4-6 所示。获取键值的过程为：CPU 先将某一个行线为低，其余行线为高，比如这里先将 P1.4 置为 0，然后 CPU 读取列线的值，如果 P1.1 为 0，说明 P1.1 与 P1.4 相连了，那么可以确定“1”号键被按下了，依此类推。

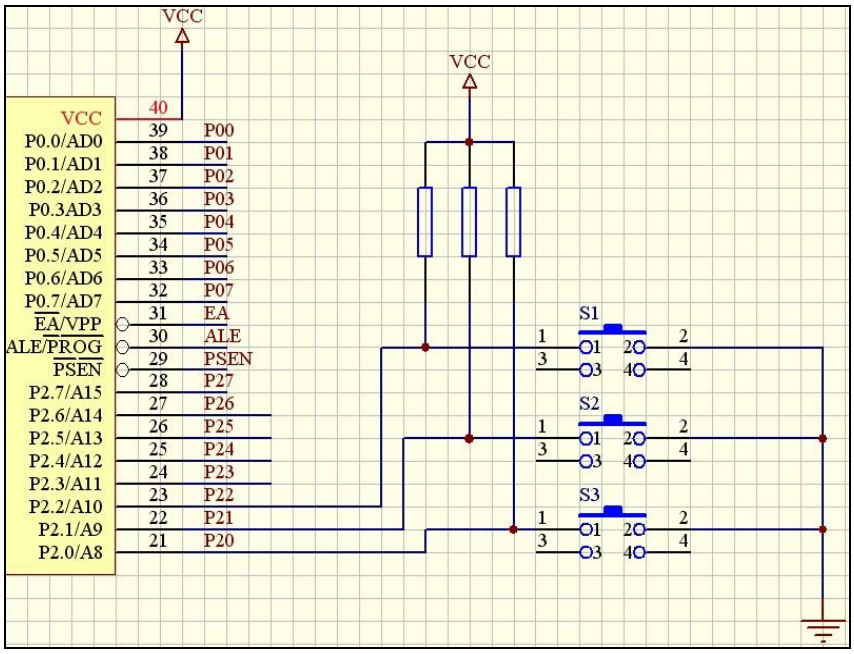

图 4-5　独立式键盘

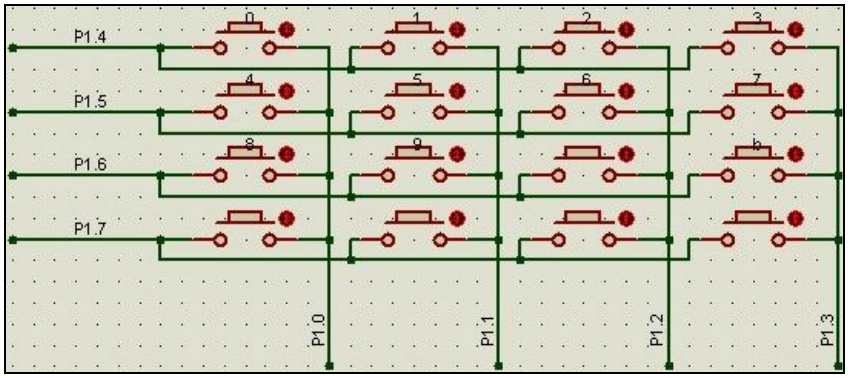

图 4-6　矩阵式键盘

4.4.3　液晶显示电路

1. 液晶显示原理简介

早在 1888 年，人们就发现液晶这一呈液体状的化学物质，像磁场中的金属一样，当受到外界电场影响时，其分子会产生精确的有序排列。如果对分子的排列进行控制，光线可以穿越液晶分子。LCD 显示屏是由不同部分组成的分层结构，最后面的一层是由荧光物质组成的可以发射光线的背光层，该层发出的光线在穿过偏振过滤层之后进入液晶层，液晶层包含了成千上万水晶液滴，水晶液滴都被包含在细小的单元格结构中，一个或多个单元格构成屏幕上的一个像素。当 LCD 中的电极产生电场时，液晶分子就会产生扭曲，从而将穿越其中的光线进行有规律地折射，然后经过过滤层的过滤在显示屏上显示出来。

2. 通用 1602 字符液晶

1602 字符液晶是能够同时显示 16×02 即 32 个字符的液晶显示屏，如图 4-7 所示。

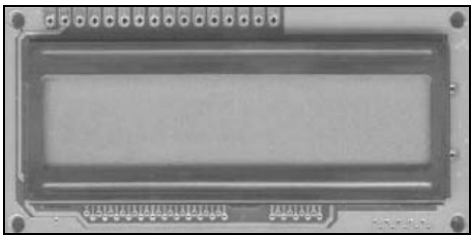

图 4-7　1602 字符液晶

1602 的显示特性如下：

- 5V 电压供电，低功耗、高可靠。
- 内置 160 个 5×7 点阵字符和 32 个 5×10 点阵字符。
- 64 字节的自定义字符 RAM，可自定义 8 个 5×8 点阵字符或 4 个 5×11 点阵字符。
- 显示方式：STN、半透、正显。
- 驱动方式：1/16DUTY，1/5BIAS。
- 视角方向：6 点。
- 背光方式：底部 LED。
- 通讯方式：4 位或 8 位并口。
- 适配 MC51 和 M6800 系列 MPU 的操作时序。

接口定义如表 4-1 所示。

表 4-1　通用 1602 接口定义

管脚号	符号	功能
1	Vss	电源地
2	Vdd	电源电压
3	V0	LCD 驱动电压
4	RS	寄存器选择输入端，输入 MPU 选择模块内部寄存器类型信号：RS=0，当 MPU 进行写模块操作，指向指令寄存器；当 MPU 进行读模块操作，指向地址计数器；RS=1，无论 MPU 读操作还是写操作，均指向数据寄存器
5	R/W	读写控制输入端，输入 MPU 选择读/写模块操作信号：R/W=0 读操作；R/W=1 写操作
6	E	使能信号输入端，输入 MPU 读/写模块操作使能信号；读操作时，高电平有效；写操作时，下降沿有效
7	DB0	数据输入/输出口
8	DB1	数据输入/输出口
9	DB2	数据输入/输出口
10	DB3	数据输入/输出口
11	DB4	数据输入/输出口
12	DB5	数据输入/输出口
13	DB6	数据输入/输出口
14	DB7	数据输入/输出口
15	A	背光的正端+5V
16	K	背光的负端 0V

3. 通用 1602 液晶与单片机的连接

将单片机的 P3.5 与使能信号输入端 E 相连，P3.6 与读写控制输入端 R/W 相连，P3.7 与寄存器选择输入端 RS 相连，P0 口作为数据传输线，如图 4-8 所示。

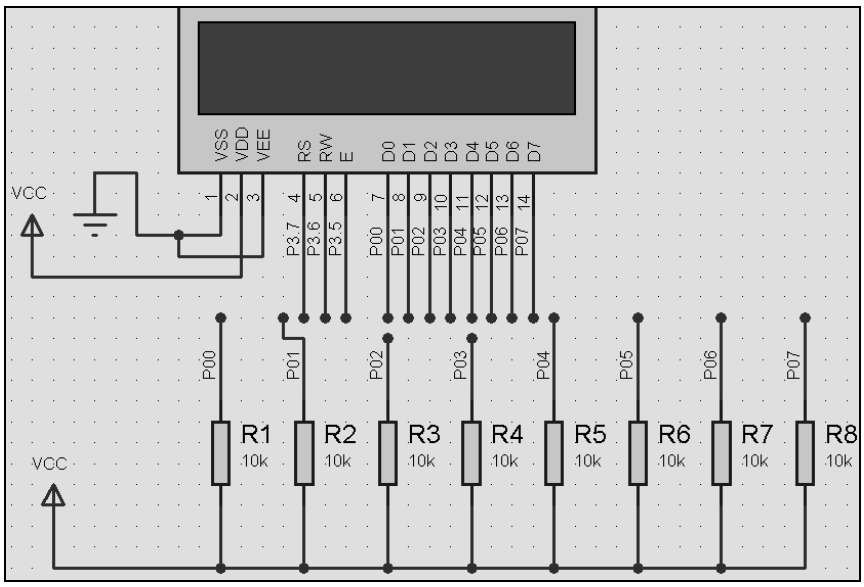

图 4-8　液晶与单片机连接图

4.4.4　D/A 转换电路

1. DAC0832 数模转换芯片简介

DAC0832 是集成数模转换芯片，可将输入的数字信号转换为模拟信号输出，基本原理是：输入为二进制数字量时，输出的模拟量与数字量成一定的比例关系，其内部由恒流源（恒压源）、模拟开关等组成。DAC0832 采用的是倒 T 型电阻网络，使用时需要外接运算放大器，若运算放大器的增益不够，还需外接反馈电阻。DAC0832 的引脚图如图 4-9 所示，功能说明如表 4-2 所示。

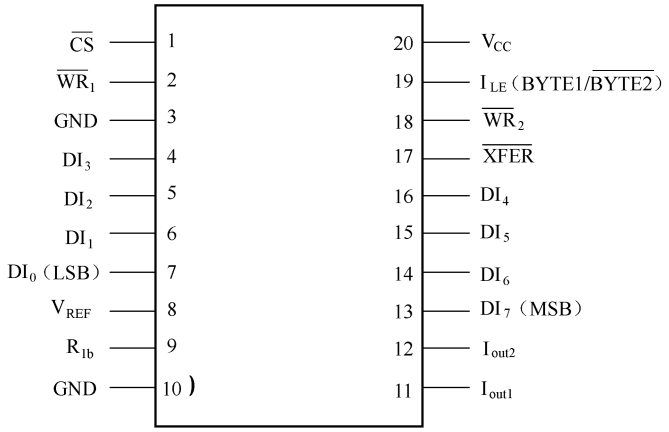

图 4-9　DAC0832 引脚图

表 4-2　DAC0832 功能表

管脚号	符号	功能
4-7，13-16	DI0～DI7	数据输入线
1	CS	片选信号输入线，低电平有效
2，18	WR1, WR2	输入寄存器的写选通信号， DAC 寄存器写选通输入线
3	GND	模拟地、摸拟信号和基准电源的参考地
8	Vref	基准电压输入线
9	Rfb	反馈信号输入线，芯片内部有反馈电阻
10	GND	数字地，两种地线在基准电源处共地比较好
11，12	Iout1,Iout2	电流输出线，两值之和为一常数
17	XFER	数据传送控制信号输入线，低电平有效
19	ILE	数据锁存允许控制信号输入线，高电平有效
20	Vcc	电源输入线

2. 单片机与 DAC0832 的连接

这里将单片机的 P2 口与 DAC0832 的数据线相连，Iout1、Iout2 接运算放大器，第一个运放输出的为负电压，第二个运放输出的为正电压，连接图如图 4-10 所示。Iout1、Iout2 的值与输入数据的关系为：$I_{out1}=\frac{V_{REF}}{15k\Omega}\times\frac{Input}{256}$；$I_{out2}=\frac{V_{REF}}{15k\Omega}\times\frac{255-Input}{256}$，通过运算放大器将电流转换为电压输出。

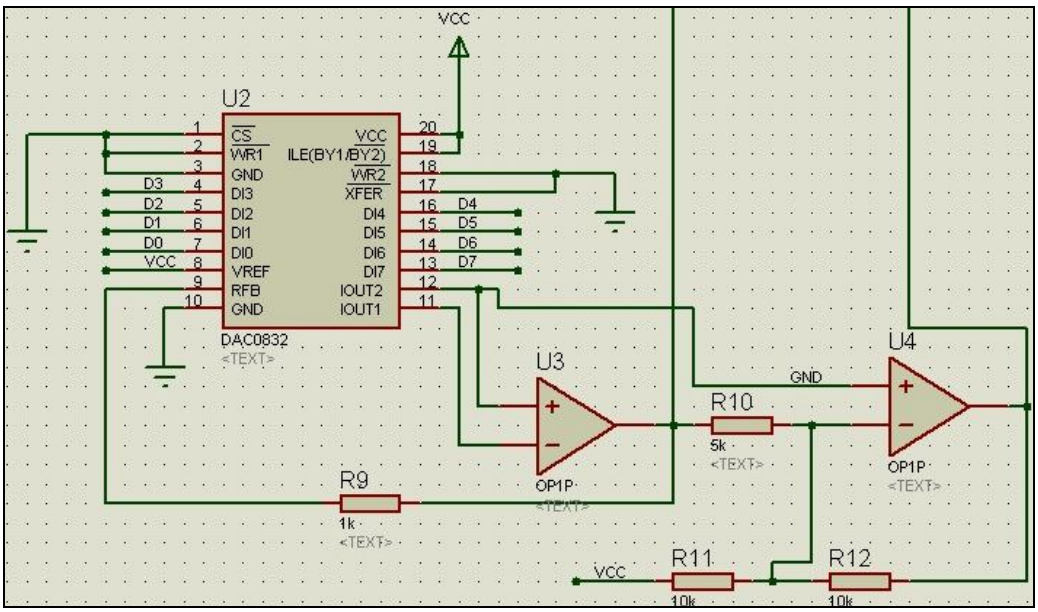

图 4-10　单片机与 DAC0832 连接图

4.4.5　Proteus 仿真平台的建立

1. Proteus 简介

Proteus 软件是英国 Labcenter Electronics 公司出版的 EDA 工具软件，它不仅具有其他 EDA

工具软件的仿真功能，还能仿真单片机及外围器件，是目前最好的仿真单片机及外围器件的工具之一。Proteus 是目前世界上唯一将电路仿真软件、PCB 设计软件和虚拟模型仿真软件三合一的设计平台，其处理器模型支持 8051、HC11、PIC10/12/16/18/24/30/DSPIC33、AVR、ARM、8086 和 MSP430 等，2010 年增加了 Cortex 和 DSP 系列处理器，并持续增加其他系列处理器模型。在编译方面，它也支持 IAR、Keil 和 MPLAB 等多种编译器。

课程设计、毕业设计是学生走向就业的重要实践环节。由于 Proteus 提供了实验室无法相比的大量的元器件库，提供了修改电路设计的灵活性，提供了实验室在数量、质量上难以相比的虚拟仪器、仪表，因而也提供了培养学生实践精神、创造精神的平台，所以如果掌握 Proteus 的使用方法，对于设计将会有极大的帮助。这里简要介绍 Proteus 在单片机仿真中的使用方法。

运行 ISIS 6 Professional，出现如图 4-11 所示的窗口界面。

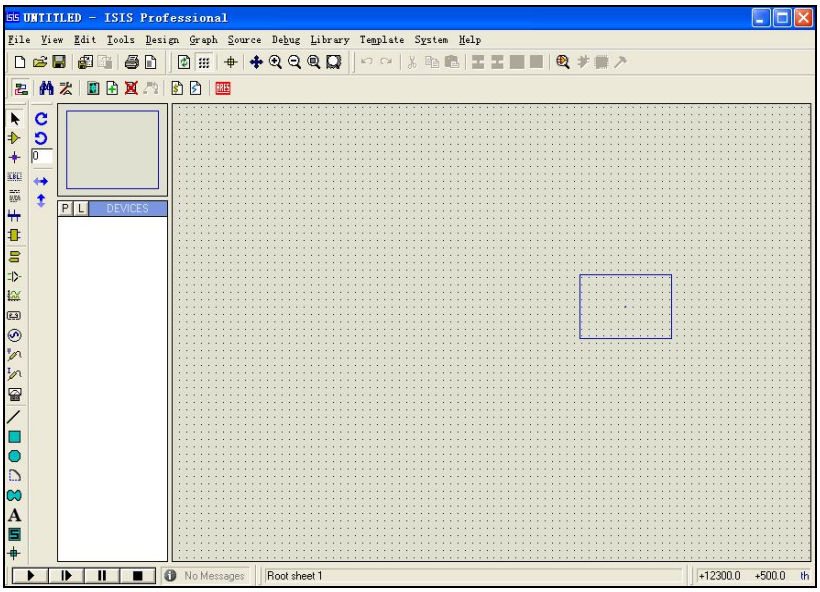

图 4-11　Proteus 界面图

窗口各部分主要有以下几个功能：

（1）原理图编辑窗口（The Editing Window）：用来绘制原理图。蓝色方框内为可编辑区，元件要放到它里面。

（2）预览窗口（The Overview Window）：窗口中显示两个内容，一个是当你在元件列表中选择一个元件时，它会显示该元件的预览图；另一个是当你的鼠标焦点落在原理图编辑窗口时它会显示整张原理图的缩略图，并会显示一个绿色的方框，绿色的方框里面的内容就是当前原理图窗口中显示的内容。

（3）模型选择工具栏（Mode Selector Toolbar）：

1）主要模型（Main Modes）：

- 选择元件。
- 放置连接点。
- 放置标签。
- 放置文本。
- 用于绘制总线。

- 用于放置子电路。
- 用于即时编辑元件参数。

2）配件（Gadgets）：

- 终端接口（terminals）：有 VCC、地、输出、输入等接口。
- 器件引脚：用于绘制各种引脚。
- 仿真图表（graph）：用于各种分析，如 Noise Analysis。
- 信号发生器（generators）。
- 电压探针。
- 电流：使用仿真图表时要用到。
- 虚拟仪表：示波器、频率计等。

3）2D 图形（2D Graphics）：

- 画各种直线。
- 画各种方框。
- 画各种圆。
- 画各种圆弧。
- 画各种多边形。
- 画各种文本。
- 画符号。
- 画原点。

（4）元件列表（The Object Selector）：用于挑选元件（components）、终端接口（terminals）、信号发生器（generators）、仿真图表（graph）等。

（5）仿真工具栏：仿真控制按钮如图 4-12 所示。

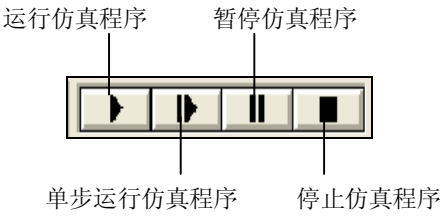

图 4-12　仿真控制按钮

2. Proteus 下系统平台的建立

（1）器件的选择。开始运行 Proteus 后单击 P 按钮，如图 4-13 所示，在 Keywords 输入框中输入需要的元器件，如图 4-14 所示。本设计涉及的元器件清单如表 4-3 所示。

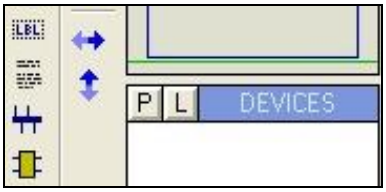

图 4-13　器件选择按钮

图 4-14　器件搜索框

表 4-3　元器件清单

器件名	Proteus 代号	数量
单片机	AT89C51	1 个
10k 电阻	3WATT10K	12 个
运算放大器	OP1P	2 个
1602 液晶	LM16L	1 个
按键	BUTTON	16 个
数模转换器	DAC0832	1 个

（2）器件的放置。在原件列表中左键选取器件，然后在原理图编辑窗口中单击，即可放置好原件，如图 4-15 所示。

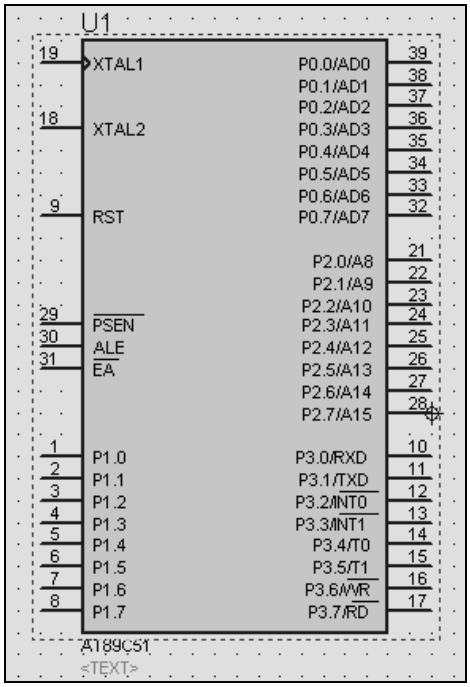

图 4-15　放置器件

（3）器件的连接。Proteus 中器件的连接与 Protel 中很相似，既可以用导线直接相连，也可以使用网络标号，如图 4-16 所示，点击网络标号（LBL）后，再点击器件引脚的延长线，即可放置标号，相同网络标号的导线，在物理上是相连的。

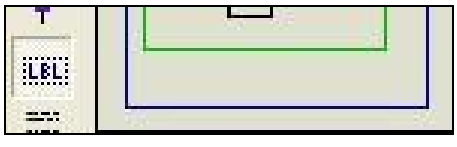

图 4-16　网络标号

不管使用哪种方法，连接后的原理图应该清晰明了。器件的连接图如图 4-17 至图 4-20 所示。

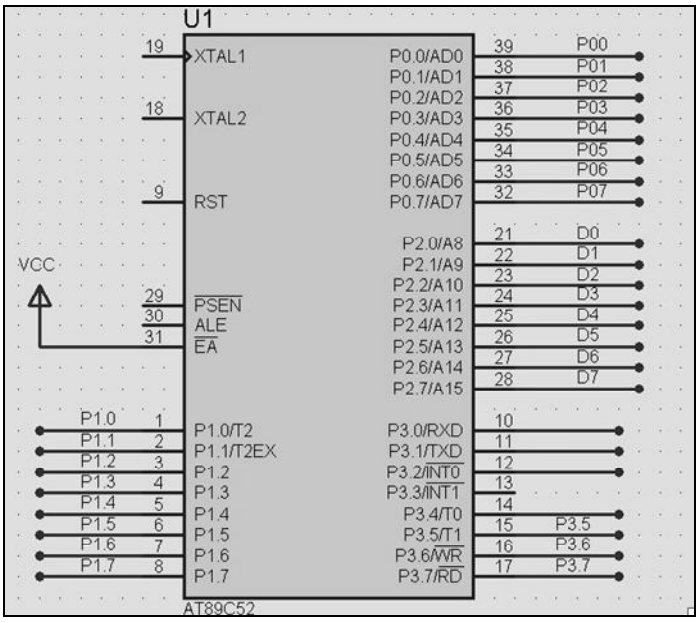

图 4-17　单片机连接图

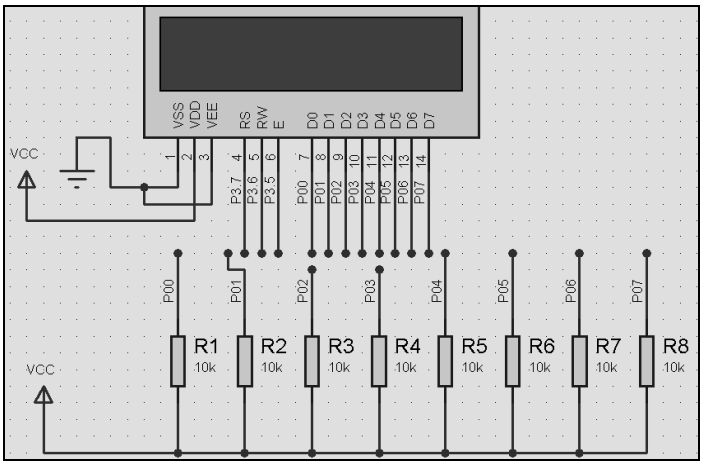

图 4-18　1602 液晶连接图

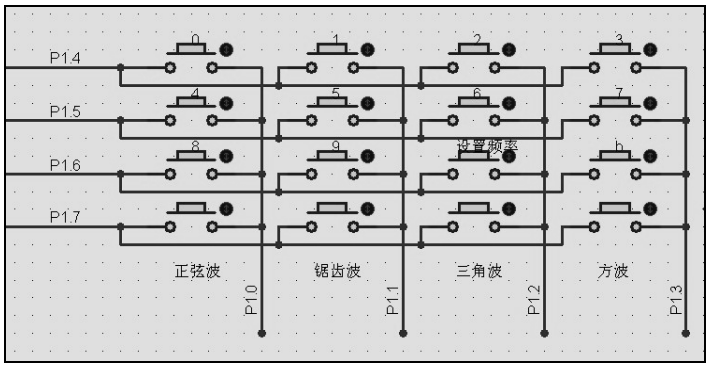

图 4-19　键盘连接图

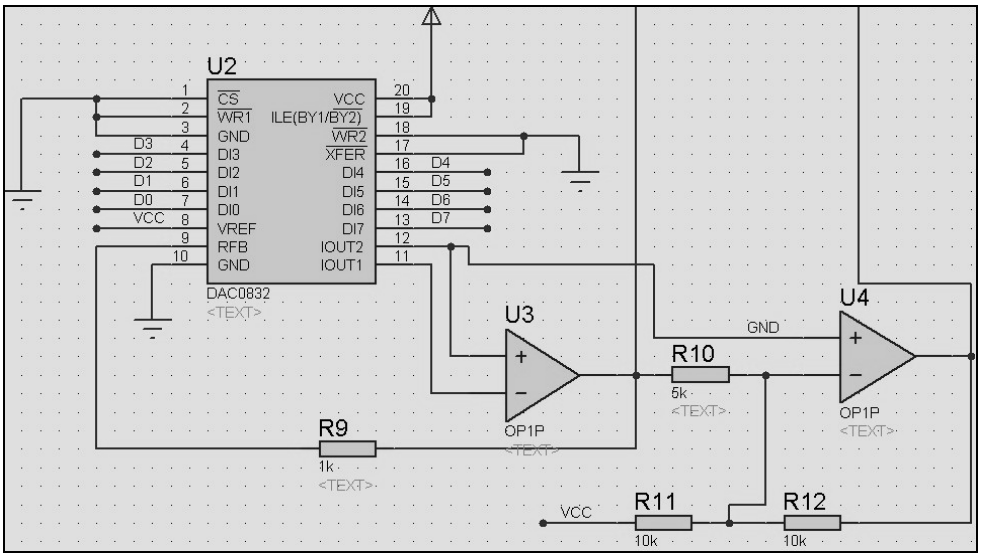

图 4-20　数模转换模块连接图

4.5　系统软件各模块设计

本节详细介绍了系统软件设计的详细过程，包括软件设计流程图、各模块程序的设计等。

4.5.1　软件设计流程图

软件设计流程图如图 4-21 所示。

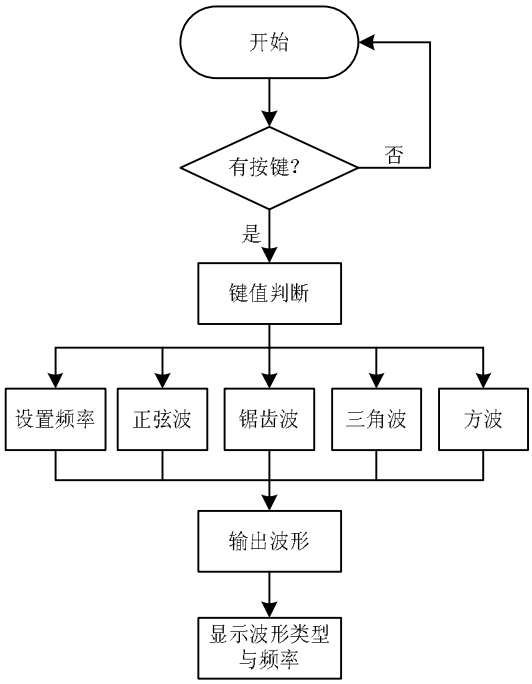

图 4-21　软件设计流程图

4.5.2　键值判断程序

在键值判断的程序中有两个函数，分别是 keydown()与 keyscan()，keydown()的功能是判断是否有按键按下，keyscan()的功能是通过扫描键盘来得出键值。实现代码如下：

键扫描子程序：

```
void  keyscan(void)
{
   P1=0x0F;                          //低四位输入
   delay(1);
   temp=P1;                          //读 P1 口
   temp=temp&0x0F;
   temp=~(temp|0xF0);
   if(temp==1)
      key=0;
   else
   if(temp==2)
      key=1;
   else
   if(temp==4)
      key=2;
   else
   if(temp==8)
      key=3;
   P1=0xF0;                          //高四位输入
   delay(1);
   temp=P1;                          //读 P1 口
   temp=temp&0xF0;
   temp=~((temp>>4)|0xF0);
   if(temp==1)
      key=key+0;
   else
   if(temp==2)
      key=key+4;
   else
   if(temp==4)
      key=key+8;
   else
   if(temp==8)
      key=key+12;                    //计算得到键值
}
```

判断键是否按下：

```
void  keydown(void)
{
   P1=0xF0;                          //行置 1，列置 0
   if(P1!=0xF0)                      //如果有键按下，则扫描键盘得出键值
   {
```

```
      keyscan();
      while(P1!=0xF0);                    //等待键释放
   }
}
```

4.5.3 波形产生与转换程序

前面介绍了 DAC0832 的工作原理，单片机应从 P2 口发送数据，这里将数据传输指令放在定时器 0 的中断函数中，通过定时器的定时时间来控制周期。

```
void Time0(void) interrupt 1 using 0
{
   EA=0;
   TR0=0;
   TH0=a;
   TL0=b;
   TR0=1;
   if(key==12)   P2=sin_tab[n]; if(n>256) {n=0;} else n++;     //产生正弦波
   if(key==13)   P2=m; if(m==0xff)  m=0; else m++;             //产生锯齿波
   if(key==14)   P2=tri_tab[j]; if(j>256) {j=0;} else j++;     //产生三角波
   if(key==15)   P2=fangbo; if(h>128) {h=0;fangbo=~fangbo;} else h++;
                                                               //产生方波
   EA=1;
}
```

4.5.4 液晶显示程序

```
sbit lcden=P3^5;
sbit lcdrw=P3^6;
sbit lcdrs=P3^7;
```

液晶显示的基本函数包括以下三个：

（1）液晶“忙”检测函数。

```
bit busytest()
{
   bit busy;
   lcdrs=0;
   lcdrw=1;
   lcden=1;
   for(time=110;time>0;time--);
   busy=lcdbf;
   _nop_();
   _nop_();
   lcden=0;
   return (busy);
}
```

（2）写控制字函数。

```
void writecom(uchar com)
{
   lcdrs=0;
```

```
   lcdrw=0;
   lcden=0;
   for(time=110;time>0;time--);
   P0=com;
   for(time=110;time>0;time--);
   lcden=1;
   for(time=110;time>0;time--);
   lcden=0;
}
```

（3）写数据函数。

```
void writedat(uchar dat)
{
   lcden=0;
   lcdrs=1;
   lcdrw=0;
   for(time=110;time>0;time--);
   P0=dat;
   for(time=110;time>0;time--);
   lcden=1;
   for(time=110;time>0;time--);
   lcden=0;
}
```

LCD 初始化设定

```
void lcdinit()                          //液晶初始化
{
   delayms(15);
   writecom(0x38);                      //16*2 显示，5*7 点阵，8 位数据
   delayms(5);
   writecom(0x38);
   delayms(5);
   writecom(0x38);
   delayms(5);
   writecom(0x0c);                      //显示开，关光标
   delayms(5);
   writecom(0x06);                      //移动光标
   delayms(5);
   writecom(0x01);                      //清除 LCD 的显示内容
   delayms(5);
   writecom(0x80);
   delayms(5);
   writecom(0x80|0x02);
   delayms(5);
   writecom(0x80|0x07);
   delayms(5);
   for(i=0;i<4;i++)                     //显示字符“Fre:”
   {
      writedat(str5[i]);
```

```
      delayms(5);
   }
   writecom(0x80|0x0d);
   delayms(5);
   for(i=0;i<4;i++)
   {
      writedat(str6[i]);                    //显示字符“Khz”
      delayms(5);
   }
}
```

4.5.5 频率设置函数

频率设置函数的功能是通过改变定时计数器的定时时间来改变输出函数的周期，从而改变频率。write_Fre()函数的功能是在液晶上显示当前频率的值。函数的周期约等于 256 个定时时间，所以可以算出 TH0、TL0 的值。

```
void SetFre(void)
{
   keydown();
   if(key==1){   a=0xf0;b=0xc4;write_Fre(1); }
   if(key==2){   a=0xf8;b=0x5f;write_Fre(2); }
   if(key==3){   a=0xfa;b=0xea;write_Fre(3); }
   if(key==4){   a=0xfc;b=0x2f;write_Fre(4); }
   if(key==5){   a=0xfc;b=0xf2;write_Fre(5); }
   if(key==6){   a=0xfd;b=0x75;write_Fre(6); }
}
```

4.5.6 主函数程序

```
main()
{
   TMOD=0x01;
   TH0=0xff;                                //设置定时器初值
   TL0=0x00;
   a=0xff;
   b=0x00;
   TR0=1;
   EA=1;
   ET0=1;                                   //开中断
   k=0;
   lcdinit() ;                              //LCD 初始化
   while(1)
   {
      keydown();
      if(key==10)                           //如果是频率设置键则改变频率
      {
         TR0=0;
         EA=0;
```

```
            ET0=0;                              //暂时关闭定时中断
            SetFre();                           //设置频率
            TR0=1;
            EA=1;
            ET0=1;                              //开中断
        }
        if(key==12)  write_hanshu(1);           //显示正弦波波形名
        if(key==13)  write_hanshu(2);           //显示锯齿波波形名
        if(key==14)  write_hanshu(3);           //显示三角波波形名
        if(key==15)  write_hanshu(4);           //显示方波波形名
    }
}
```

4.6　系统调试方法及性能测试

掌握调试系统的方法对于设计的效率至关重要，本节详细介绍了基于 Proteus 仿真平台的系统调试方法，并对性能进行了测试。

4.6.1　系统调试方法

在 keil 中将代码编译，产生 hex 文件。打开 Proteus 文件，双击单片机，在 Program Files 中选择对应的 hex 文件加载到单片机中，为了能看到输出的波形，需要添加虚拟示波器，选择 Virtual Instruments Mode 中的 OSCILLOSCOPE，Proteus 中的模拟示波器是 4 通道的，使用方法与实际的示波器基本相似，单击仿真工具栏的 play 按钮，即可开始仿真。模拟示波器的放置如图 4-22 所示。

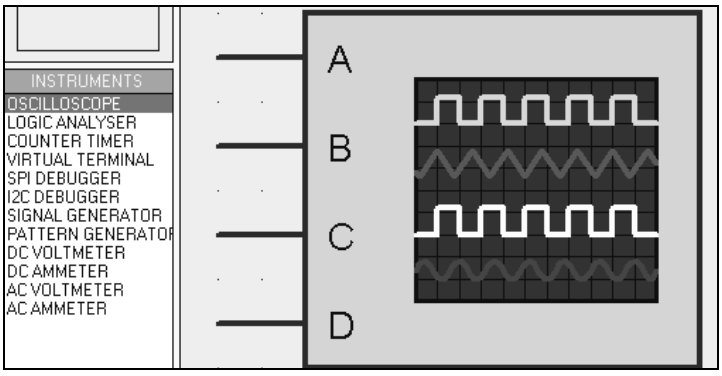

图 4-22　添加示波器

4.6.2　系统性能测试

系统运行后，单击相应的按键，系统即可输出正弦波（如图 4-23 所示）、锯齿波（如图 4-24 所示）、三角波（如图 4-25 所示）、方波（如图 4-26 所示），如图 4-23 至图 4-26 所示，图中可以看出有两道波形，第一道是输出的负电压波形，电压范围为：-5.47016～0V；第二道为双极性电压波形，电压范围为：-4.99998～5.94029V。按下“设置频率”键，进入设置频率模式，按下“1-6 号”键，频率即为 1kHz～6kHz。

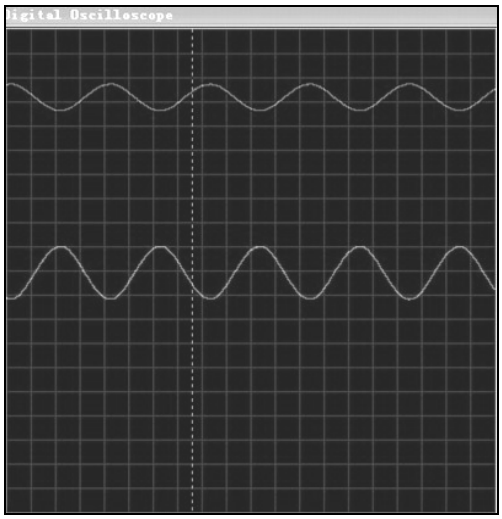

图 4-23　正弦波

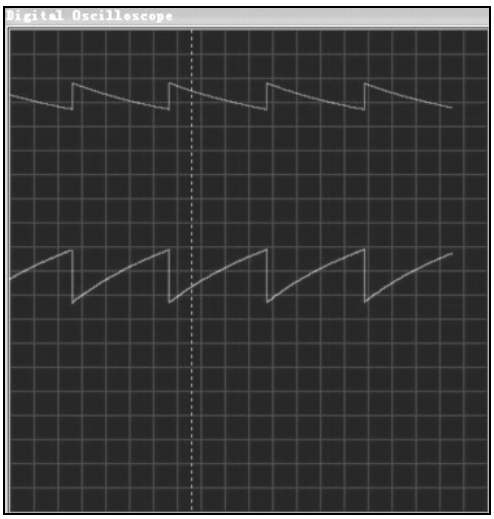

图 4-24　锯齿波

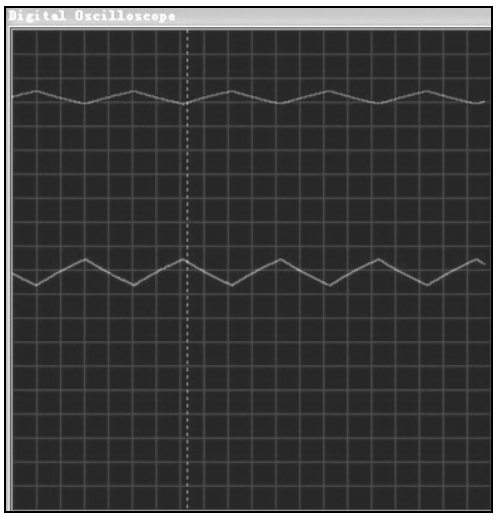

图 4-25　三角波

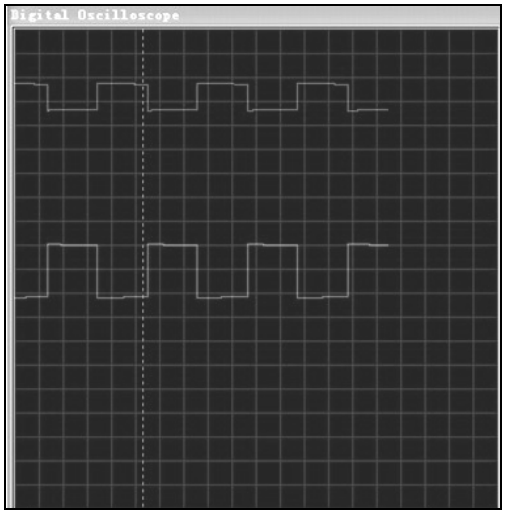

图 4-26　方波

从示波器中可以看出，方波的波形有些失真，实际频率与理论上相比存在着误差，误差的主要原因是，定时器的计数速率为震荡周期的 1/12，当采用 12M 晶体时，计数速率为 1M，计数速率是有限的。另外，比如 3kHz 的周期，1/3 不能除尽，此处也存在误差。

4.7　本章小结

本章介绍了一种基于单片机的函数发生器的设计，并利用 Proteus 仿真平台对设计进行了仿真，在设计过程中包括了器件的选择、硬件设计、软件设计和系统的仿真调试等完整的开发设计的过程，希望对读者的毕业设计有所帮助。