基于 51 单片机和 DAC0832的信号源(proteus 电路图加程序)

班 级 ___
学 号 ___

西安电子科技大学

自动测试实验报告

题　　目　基于 51 单片机和 DAC0832的信号源

学　　院　机电工程学院

专　　业　测控技术与仪器

导师姓名　吕晓洲

学生姓名 _____

学　　号 _____

摘要

本文介绍了以 8051 和 DAC0832 为核心的信号源， 可以通过按键选择正弦波、方波、三角波、锯齿波和梯形波，也可以通过按键选择 798.6Hz、266.2Hz、88.7Hz、29.6Hz、9.85Hz、3.3Hz、1.1Hz 共九档频率。 波形和频率通过软件改变， 幅值通过硬件放大的放大器改变。本信号源具有结构简单、 功能丰富、 使用方便另外价格实惠等特点。

【关键词】单片机， 8051，DAC0832，信号源，频率，波形

一. 实验要求以及方案选择

1.实验要求：

设计一个信号源，能产生正弦波、三角波、锯齿波、方波等简单的波形，能够方便改变波形和频率。

2. 方案选择：

方案一：完全由硬件电路制作， 使用传统的锁相频率合成方法。通过芯片 IC145152，压控振荡器搭接的锁相环电路输出稳定性极好的正弦波，再利用过零比较器转换成方波，积分电路转换成三角波。

此方案，电路复杂，干扰因素多，不易实现。

方案二：直接利用波形产生芯片，例如，利用 MAX038 芯片组成的电路输出波形。 MAX038 是精密高频波形产生电路，能够产生准确的锯齿波、 三角波、 方波和正弦波四种周期性波形。 但此方案成本高，程序复杂度高。

方案三：通过单片机控制 DAC，输出五种波形。此方案输出的波形分辨率不够高，频率有限。但此方案电路简单、成本低，波形和频率容易选择。

二. 实验元件及原理介绍

1. 80C51 单片机

80C51 单片机属于 MCS-51 系列单片机，由 Intel 公司开发，其结构是 8048 的延伸，改进了 8048 的缺点，增加了如乘（MUL ）、除（DIV ）、减（SUBB）、比较（ CMP ）、16 位数据指针、布尔代数运算等指令， 以及串行通信能力和 5 个中断源。 采用 40 引脚双列直插式 DIP（Dual In Line Package ），内有 128 个 RAM 单元及 4K 的 ROM 。它把构成计算机的中央处理器 CPU、存储器、 寄存器、 I/O 接口制作在一块集成电路芯片中，从而构成较为完整的计算机、而且其价格便宜。

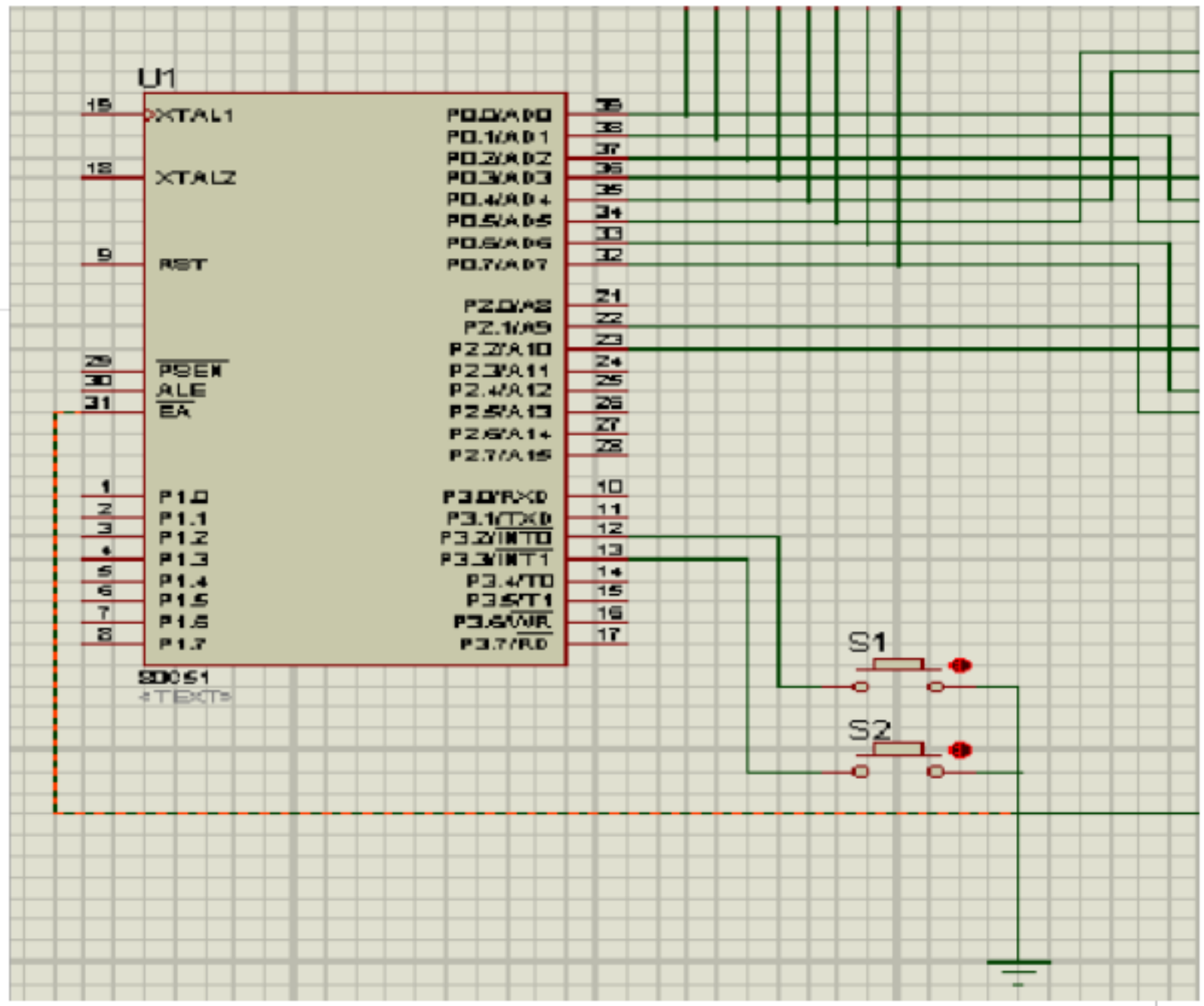

图 1.80C51 单片机及键盘电路

图 1 所示是 80C51 单片机以及接口电路， 由于 Proteus 对单片机的 VCC 和 GND 是默认自动连接的，所以这里就不需要再连接电源和地了。本次试验使用 80C51 单片机内部晶振，所以也无需外接晶振。按键 S1 和 S2 分别连接到 80C51 的 INT0（P3.2）和 INT1（P3.3），按键 S1 控制波形选择，每按一次变换一次波形，分别为正弦波、方波、锯齿波、梯形波和三角波。按键 S2 控制频率，共七档，每按一次频率下降为上一次的 1/3。

2. DAC0832 数模转换器

DAC0832 是 8 分辨率的 D/A 转换集成芯片。与微处理器完全兼容。这个 DA 芯片以其价格低廉、接口简单、转换控制容易等优点，

在单片机应用系统中得到广泛的应用。 D/A 转换器由 8 位输入锁存器、 8 位 DAC 寄存器、 8 位 D/A 转换电路及转换控制电路构成。

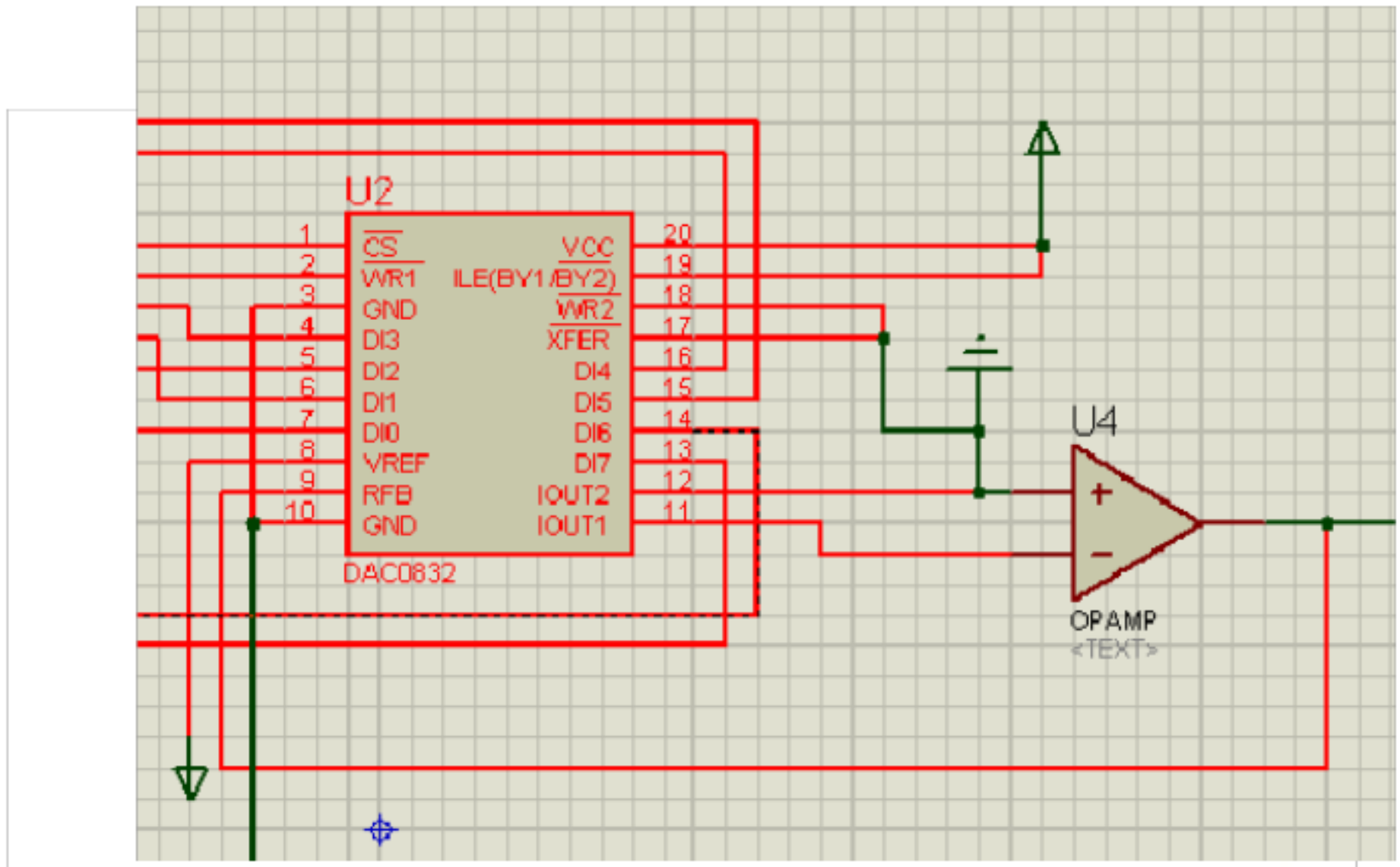

图 2.DAC0832 器件以及 I/V 变换电路

图 2 所示是 DAC0832 器件以及 I/V 变换电路。DAC0832 有两级缓冲，本次试验采用直通模式， 即把 ILE 、CS、WR1、WR2、XFER 都设置为有效，使两个寄存器都处于开放状态，无需控制信号，DAC0832 的输出随时跟谁输入数字的变化而变化，这样只要输入的八位数字量变化，就直接进行 DA 转换。

图中的运算放大器是实现 I/V 变化功能，由于 DAC0832 输出的是电流信号，且 Iout1 和 Iout2 的电流之和为一常数，在 DAC 寄存器各位都是 1 时， Iout1 输出最大，一般在单极性输出时把 Iout2 接地，双极性输出时接运放。 如果要调整放大系数， 只需要在运放输出端与反馈端串联一可调电阻即可。

三.系统软件介绍

本信号源主程序主要有三部分构成： 主程序模块、 外部中断一模块、外部中断 2 模块。

1. 主程序介绍

主程序开始后首先进行初始化， 然后根据波形标志 a,b,c,d,e 的值进入相应的 while 循环，这样写的好处是指令简洁， 输出的波形频率可达近 800Hz。在 while 循环中，单片机根据地址标志位不停低查表，然后把查得的值赋给 DAC0832 的数据口，然后地址标志位加一，并判断地址标志位是否等于 64，如果是就置 0 再往下执行，如果不是直接往下执行。然后根据频率标志位进行相应的延时。

其中波形是取一个周期内的 64 个点进行描绘， 波形 ROM 表是将信号一个周期等间距地分离成 64 个点，储存在单片机得 RON 内。具体 ROM 表是通过 MATLAB 生成的，例如正弦表， MATLAB 生成的程序如下： x=0:2*pi/64:2*pi; y=round(sin(x)*127)+128

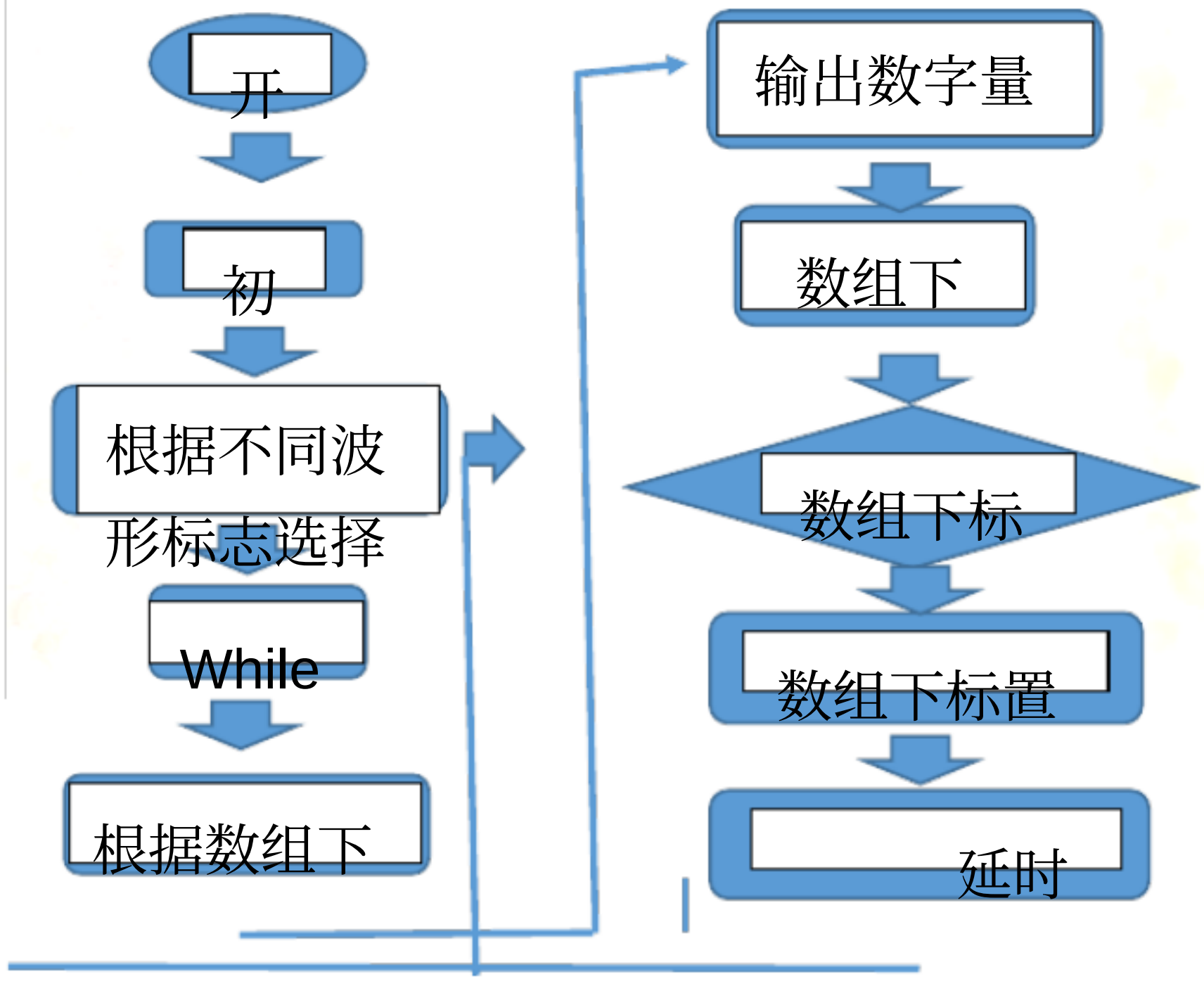

图 3 主程序流程图

2.中断服务子程序

本程序有两个中断，中断 1 控制波形选择，中断 2 控制频率。

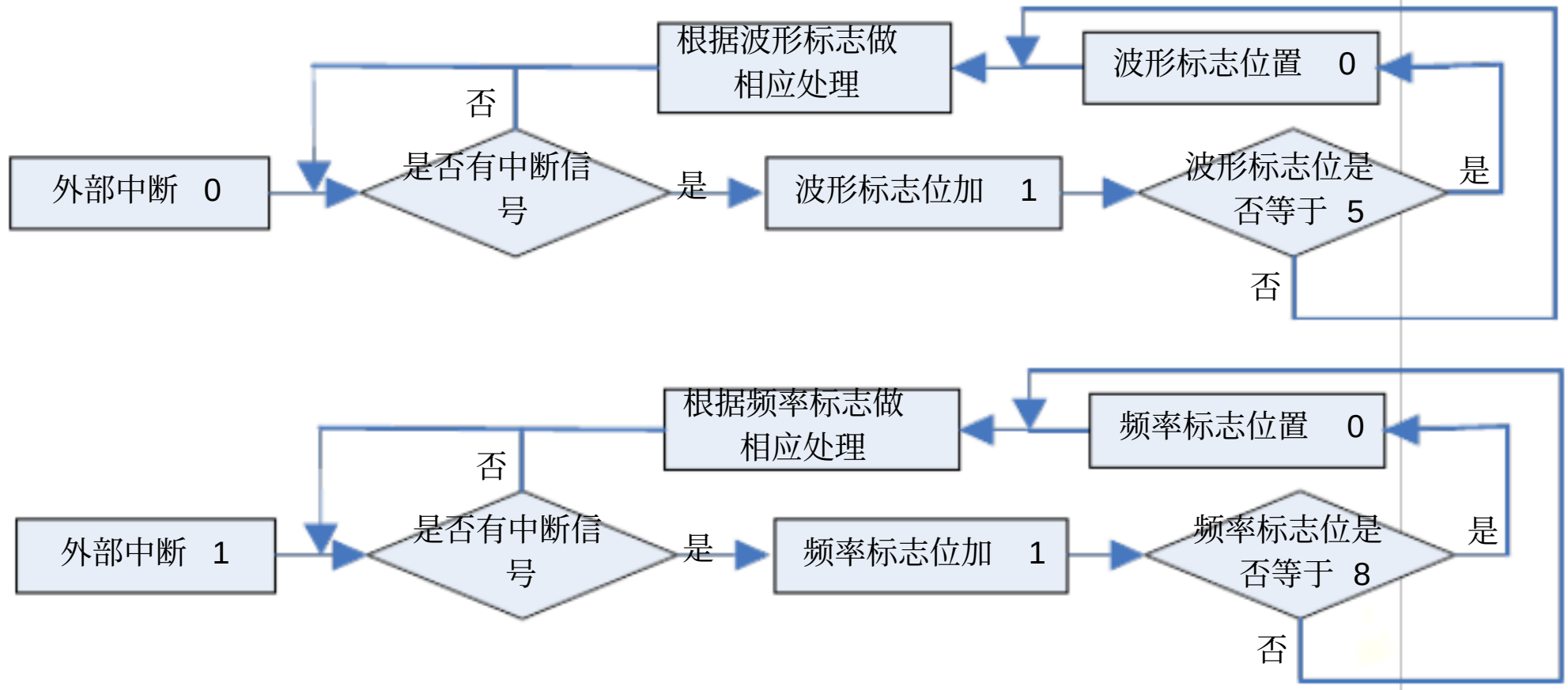

图 4.中断程序框图

四.实验结果

用 Proteus 软件绘制好原理图， 用 Keil 软件生产程序的 Hex 文件，导入到 80C51 单片机中，就可以仿真了。实现结果用一下图形表示。

1. 生成的正弦波

系统开机默认的是正弦波，通过按动 S1 按钮可以选择波形，按动 S2 可以调节频率。

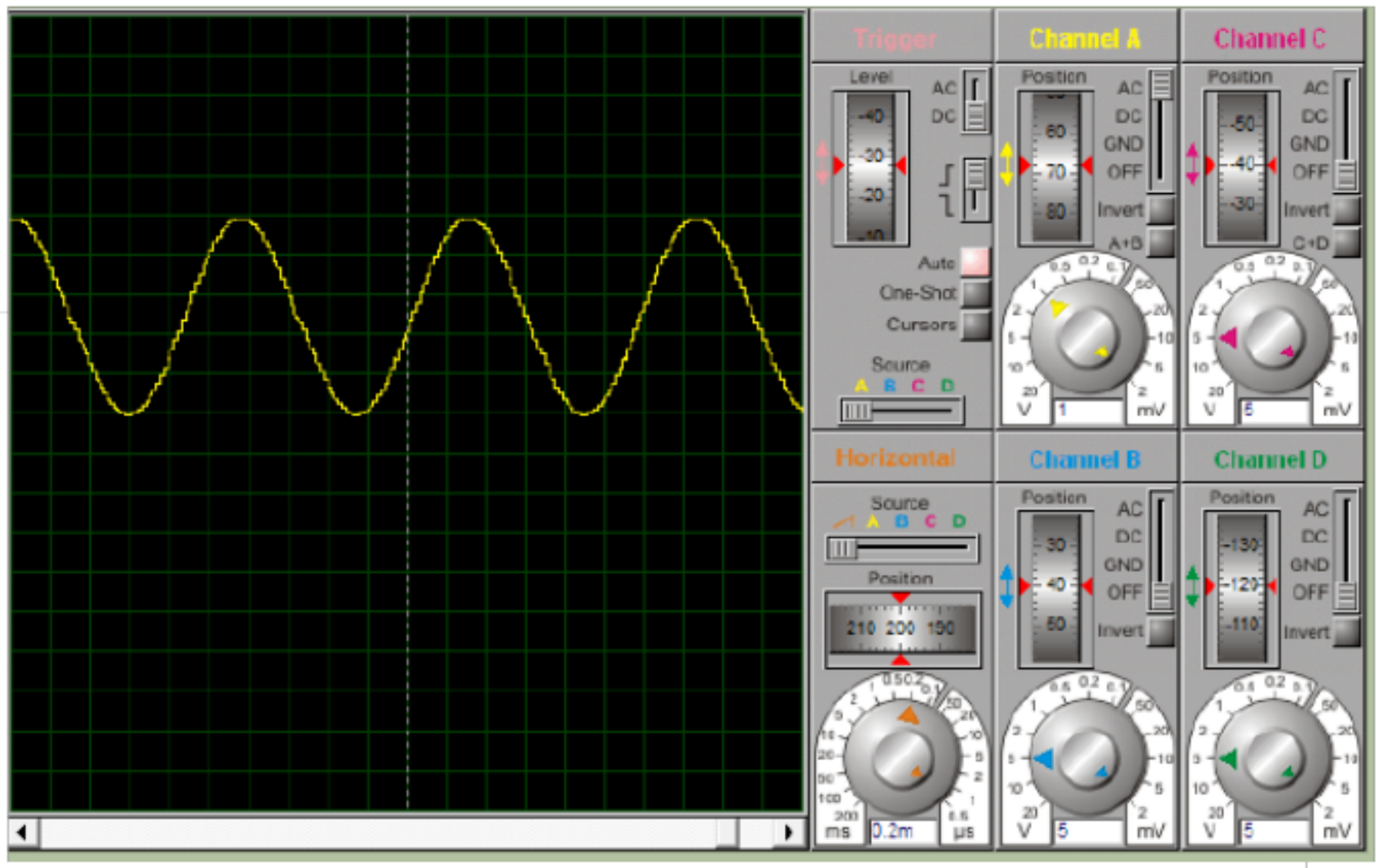

图 5 798.6Hz 频率正弦波

按 S2 按钮，频率降低三倍后的正弦波

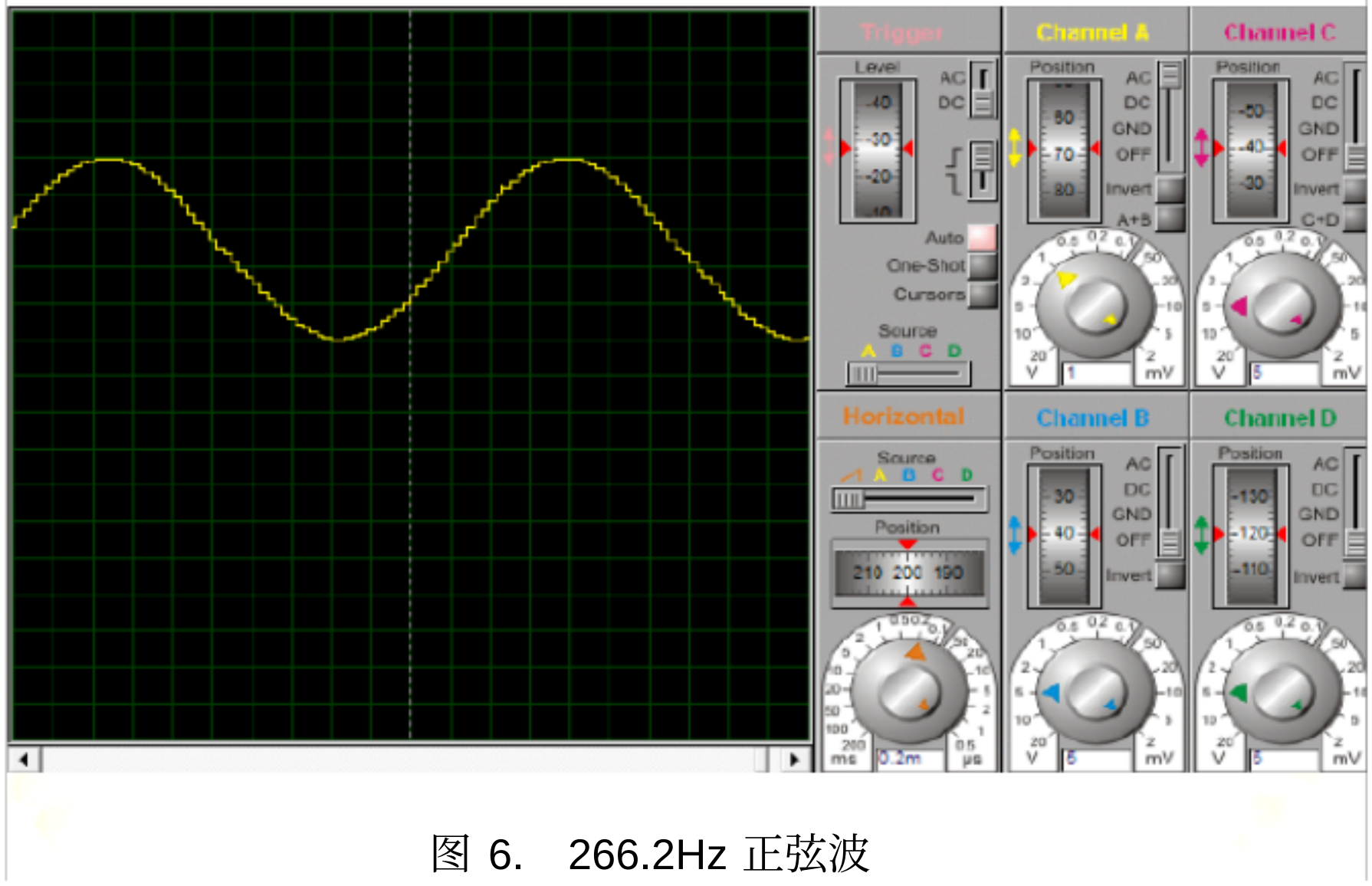

图 6.　266.2Hz 正弦波

2.生成方波

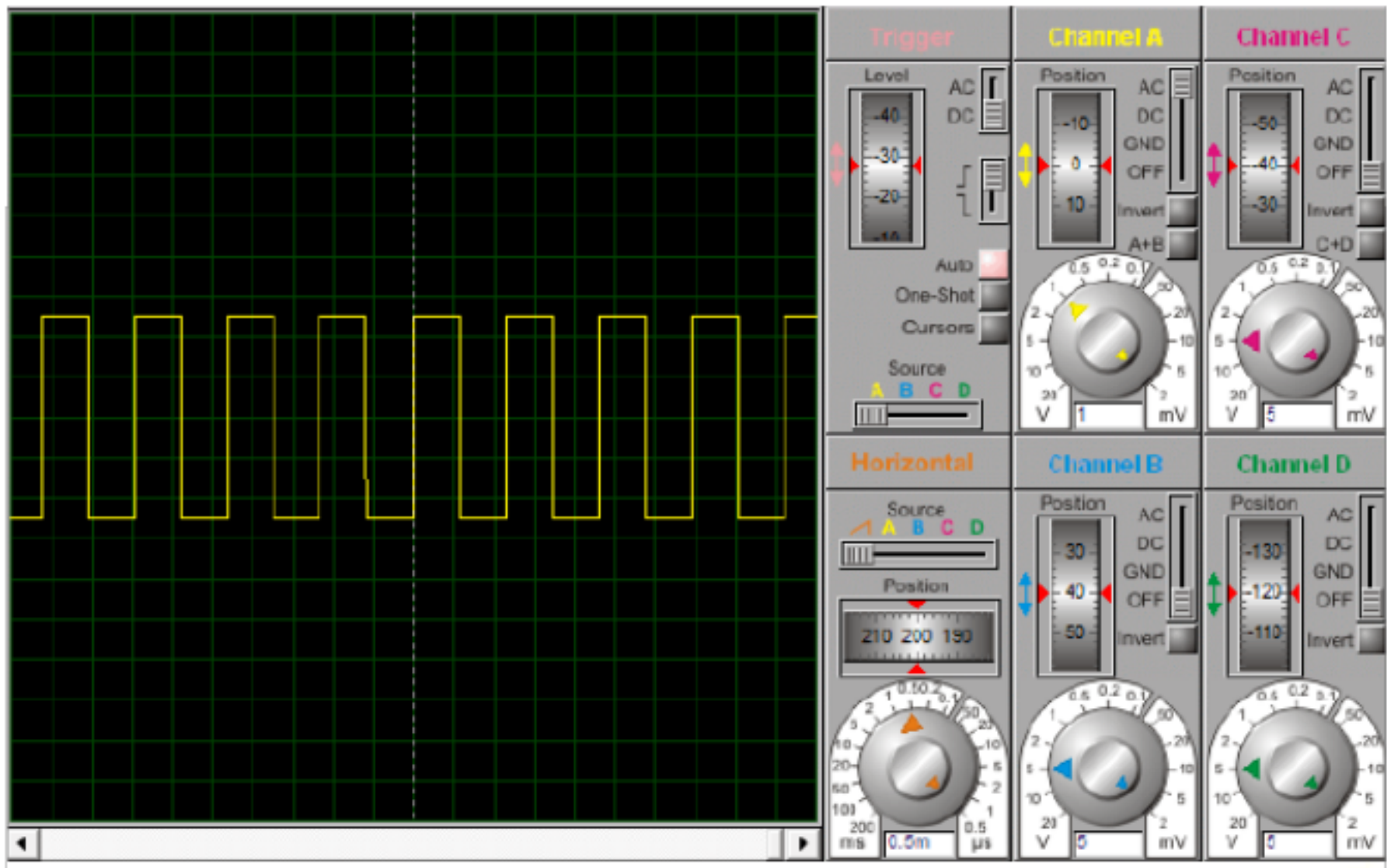

图 7 .798.6Hz 方波

调节 S2 按钮，生成 88.7Hz 的方波

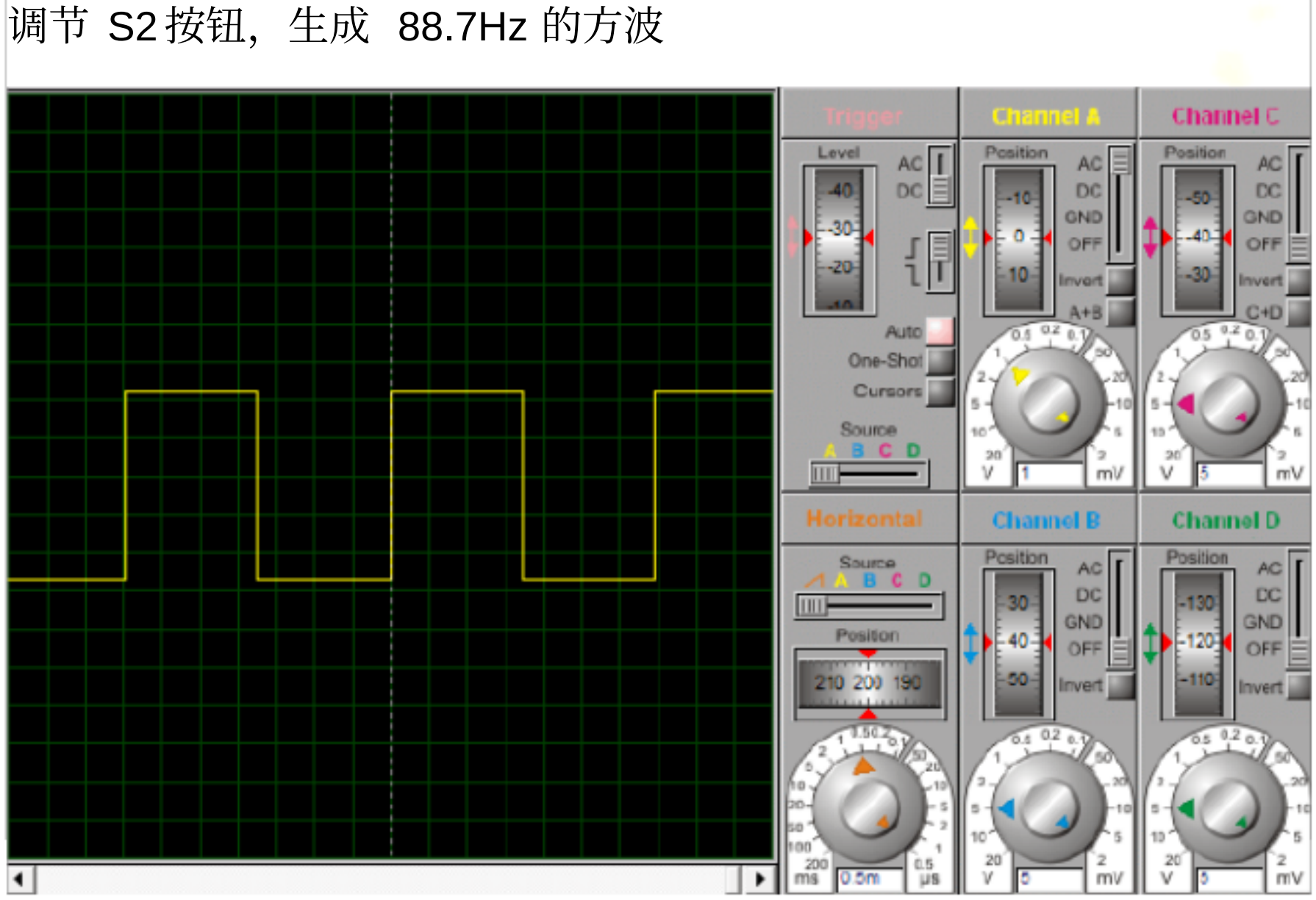

图 7 .88.7Hz 方波

3.生成锯齿波

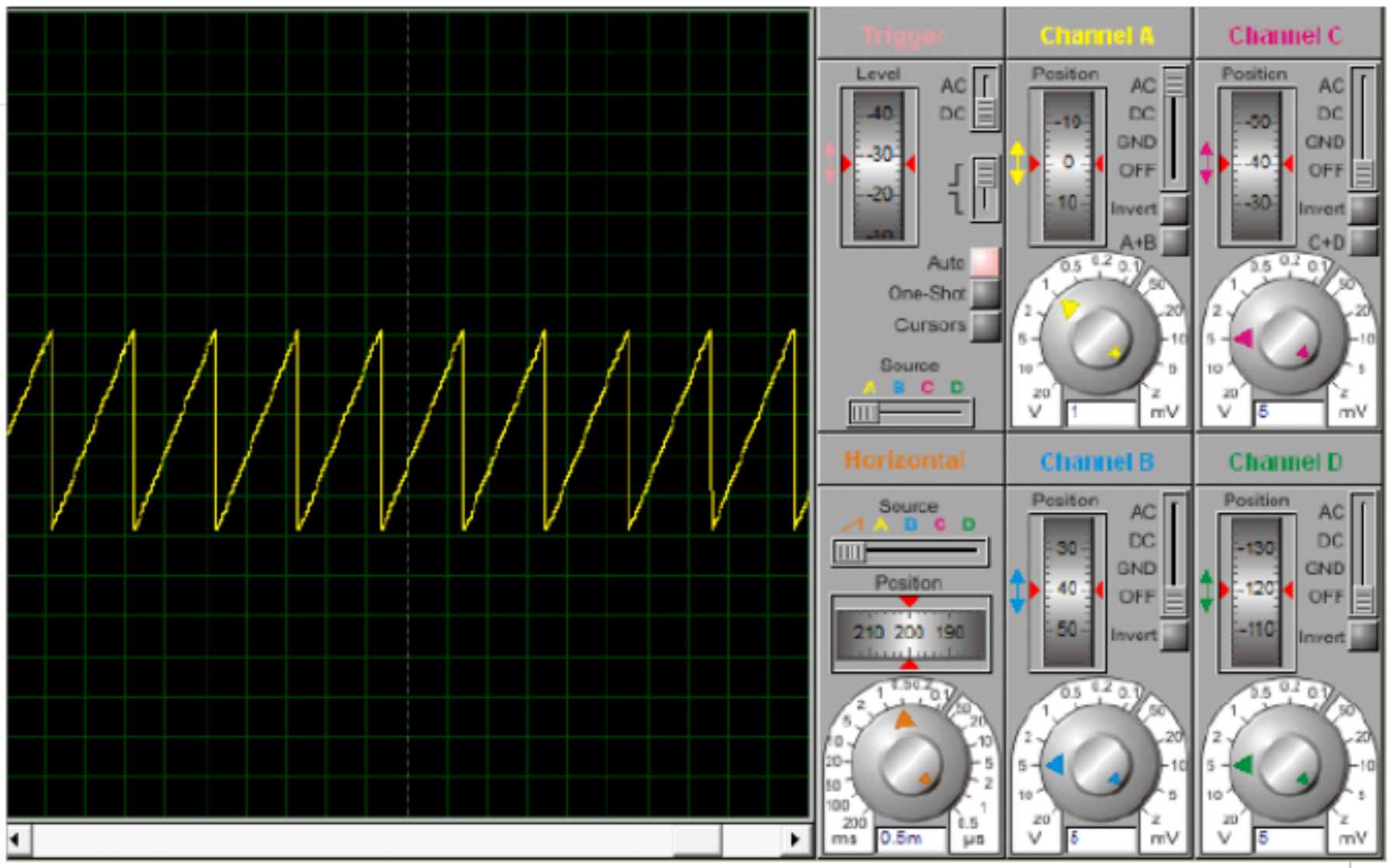

图 8 .798.6Hz 锯齿波

4. 生成梯形波

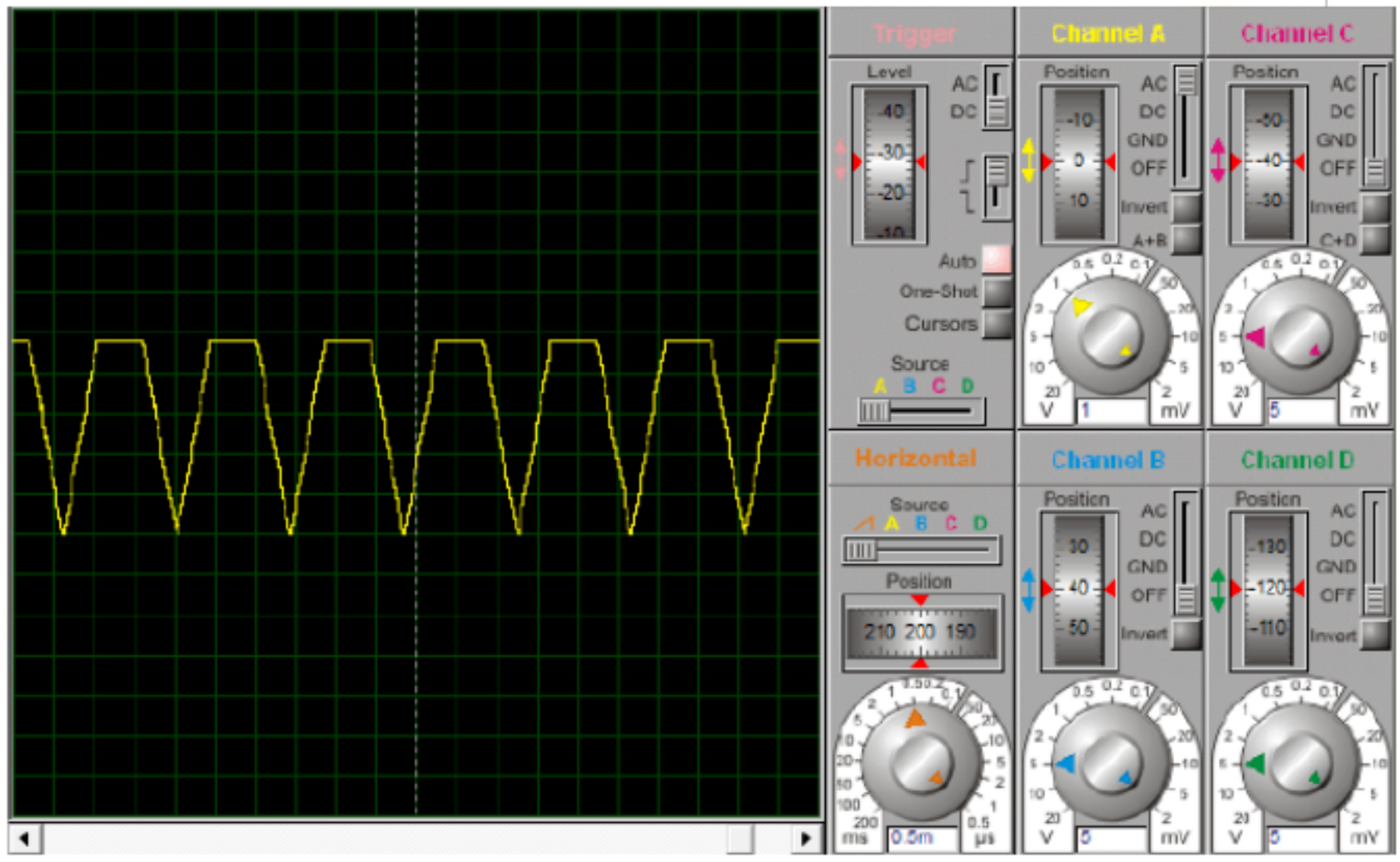

图 9.798.6Hz 梯形波

5.生成三角波

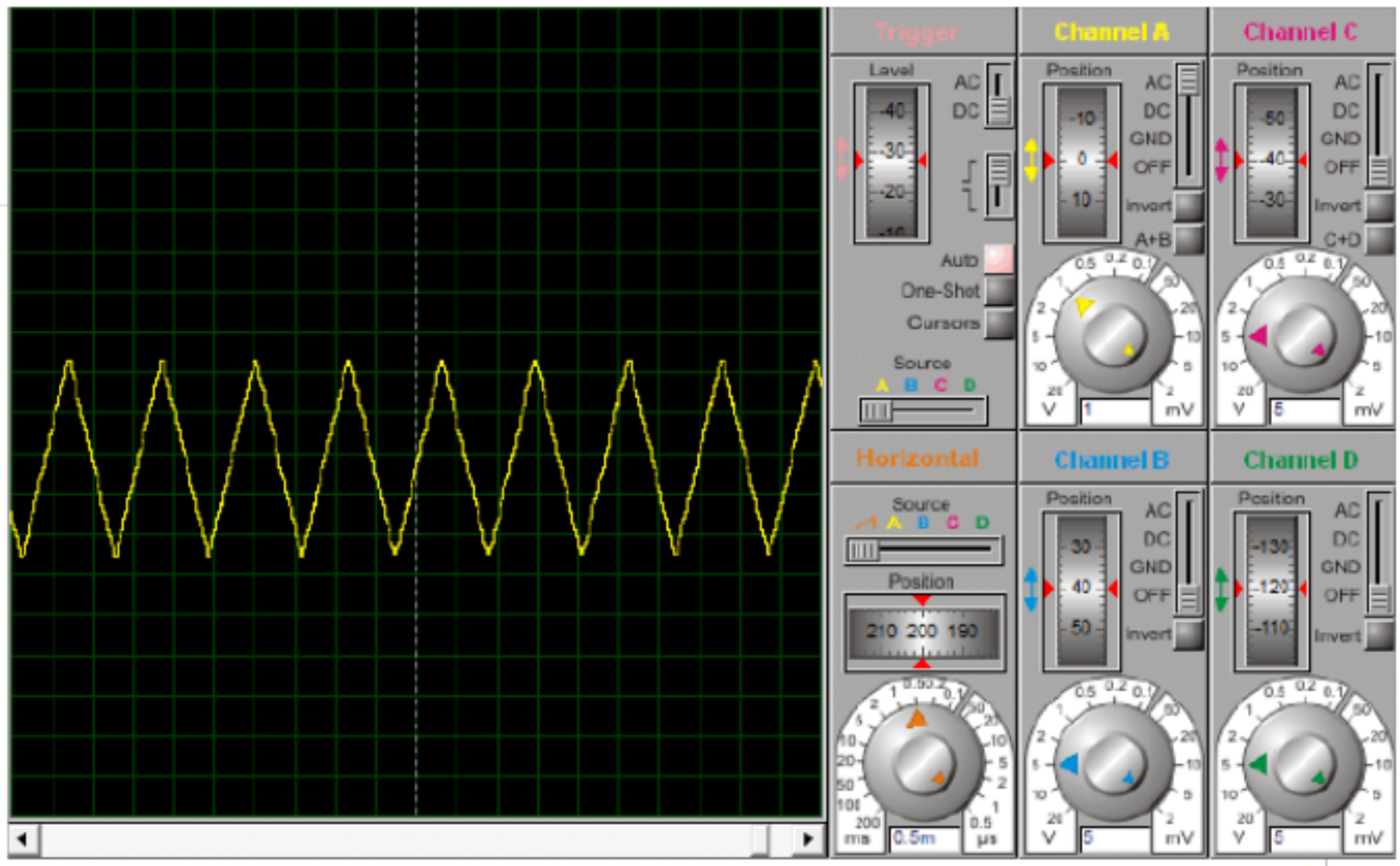

图 10.798.6Hz 梯形波

按动 S 按钮，生成 266.2Hz 三角波

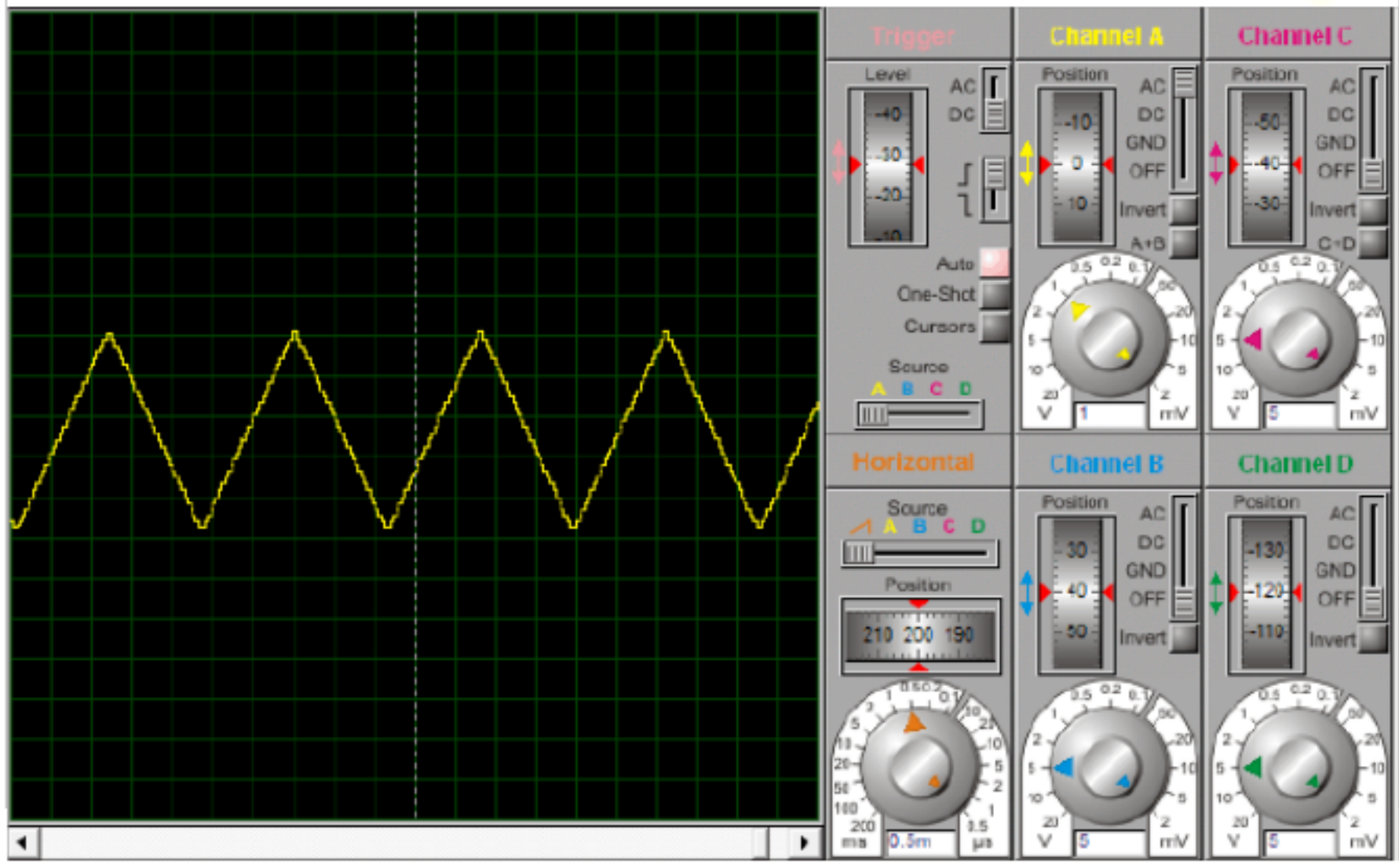

图 11.266.2Hz 三角波

五．实验总结

第一次使用 Proteus 软件，感觉这个软件使用非常方便，上手很快，仿真的时候还可以看到引脚上的电平变化，对于故障排除与实验分析非常方便。

本次实验产生的波形能够满足一般的要求，但是频率还是比较低，最大仅有 798.6Hz，而且分辨率还不够高，把时间调小后还可以看到阶梯线，这个把函数表做的大一点，可以提高分辨率，但是那样又会影响频率。

这种单片机信号源，提高信号频率的最好方法是提高单片机的主频。

六．附录

1.实验硬件总体电路图

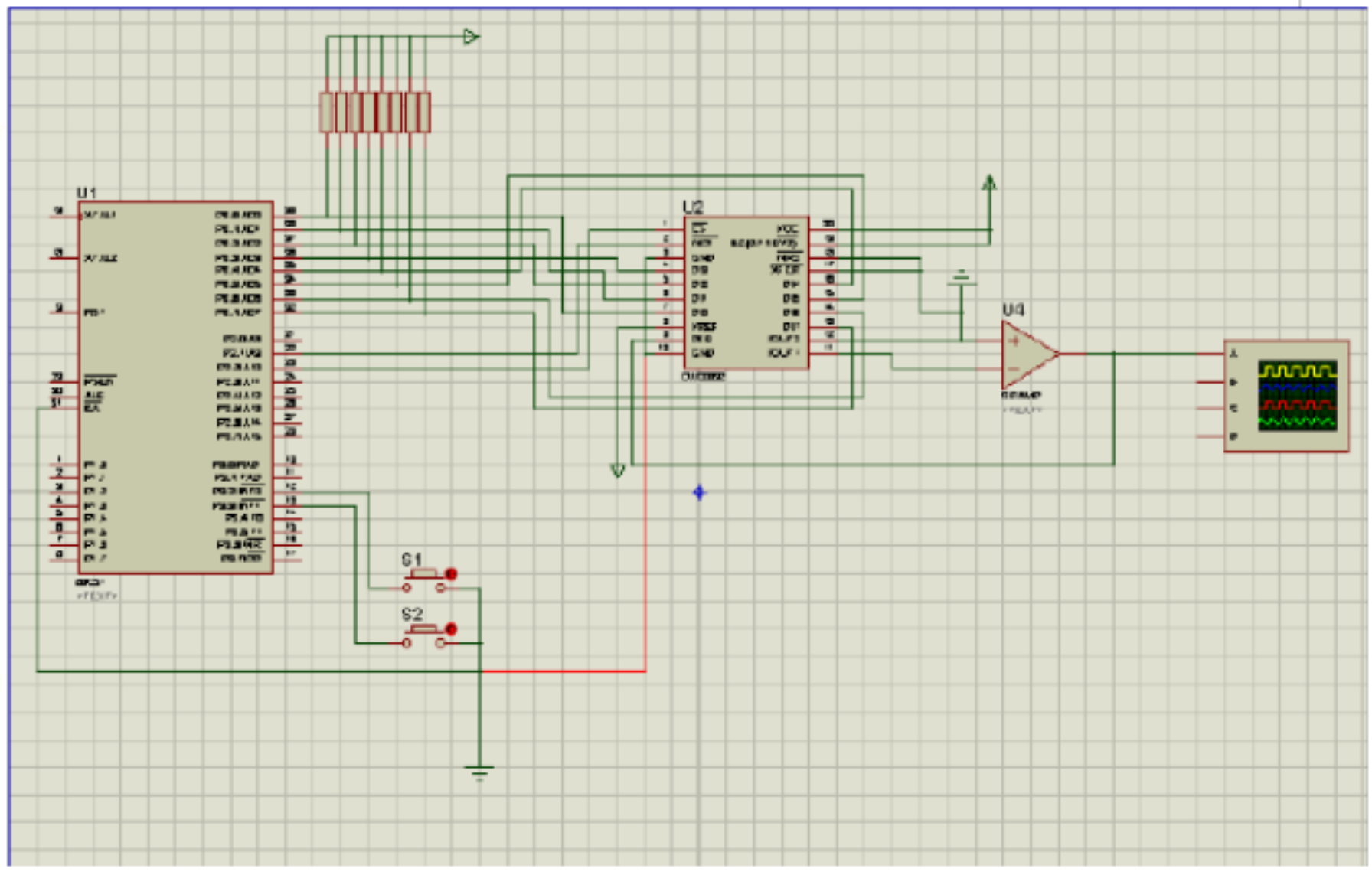

图 12.实验总电路图

2. 实验程序

```
#include<reg51.h>
#define uchar unsigned char
#define uint unsigned int
sbit csda=P2^2;                  //p2.2 口作为 0832 的片选端
sbit wr=P2^1;            //p2.1 口作为 0832 的写信号控制端
sbit s1=P3^2;            //  按键 1 的接口，选择波形
sbit s2=P3^3;            //  按键 2 的接口，选择频率
uchar k=0,p=0,delay=0;       //k 是数组下标     p 是频率标志   delay 是延时时间
uchar bxxz=0;pinlv=0;          //bxxz  波形标志，pinlv 是频率对应的延时
uchar a=1,b=0,c=0,d=0,e=0;     //  a，b，c，d，e 分别对应正弦波，方波，锯齿波，梯形波，三角波

uchar code sin[64]={
135,145,158,167,176,188,199,209,218,226,234,240
,245,249,252,254,254,253,251,247,243,237,230,22
2,213,204,193,182,170,158,
```

146,133,121,108,96,84,72,61,50,41,32,24,17,11,7,3
,1,0,0,2,5,9,14,20,28,36,45,55,66,78,90,102,114,12
8
};　　　　//正弦波函数表

uchar code juxing[64]={
255,255,255,255,255,255,255,255,255,255,255,255
,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,
255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
};　　　　//方波函数表

uchar code juchi[64]={
0,4,8,12,16,20,24,28,32,36,40,45,49,53,57,61,65,69
,73,77,81,85,89,93,97,101,105,109,113,117,121,12
5,130,134,138,142,
146,150,154,158,162,166,170,174,178,182,186,190
,194,198,202,206,210,215,219,223,227,231,235,23
9,243,247,251,255
};　　　　//锯齿波函数表

uchar code tixing[64]={
0,13,26,39,52,65,78,91,104,117,130,143,156,169,1
82,195,208,221,234,247,247,247,247,247,247,247,

```
247,247,247,247,247,247,247,247,247,247,247,247
,
247,247,247,247,247,247,247,242,229,216,203,190
,177,164,151,138,125,112,99,86,73,60,47,34,21,8
};                                  //梯形波函数表
uchar code sanjiao[64]={
0,8,16,24,32,40,48,56,64,72,80,88,96,104,112,120,
128,136,144,152,160,168,176,184,192,200,208,216
,224,232,240,248,
248,240,232,224,216,208,200,192,184,176,168,160
,152,144,136,128,120,112,104,96,88,80,72,64,56,4
8,40,32,24,16,8,0
};                          //三角波波函数表
void delay1()                       //延时时间函数,  延时一
毫秒
{
    int a,b;
    for(a=1;a>0;a--)
      for(b=122;b>0;b--);
}
void int0() interrupt 0             ///中断 1,  选择波形
{
```

```
    EX0=0;                          //关中断
    delay1();
    if(s1==0)
    {
      bxxz++;
      if(bxxz==5)                          //波形标志为 5 后,
重新置零
        bxxz=0;
      switch(bxxz)                 //每按动一次  S2, 改变
波形
        {
      case 0 :
        {a=1,b=0,c=0,d=0,e=0;}
        break;
      case 1 :
        {a=0,b=1,c=0,d=0,e=0;}
        break;
      case 2 :
        {a=0,b=0,c=1,d=0,e=0;}
        break;
     case 3 :
        {a=0,b=0,c=0,d=1,e=0;}
```

```
        break;
    case 4 :
        {a=0,b=0,c=0,d=0,e=1;}
        break;
      }
    delay1();
    while(!s1);
    }
    while(!s1);
    EX0=1;
}
void int1() interrupt 2                    //中断 1，频率选择
{
    EX1=0;
    delay1();
    if(s2==0)
    {
    p++;
    if(p==8)                               //频率标志是 8 后，重新
置零
        p=0;
    switch(p)                              //每按动一次，频率减为
```

三分之一

```
        {
        case 1 :
            pinlv=3;
            break;
        case 2 :
            pinlv=6;
            break;
        case 3 :
            pinlv=9;
            break;
        case 4 :
            pinlv=12;
            break;
        case 5 :
            pinlv=15;
            break;
        case 6 :
            pinlv=18;
            break;
       case 7 :
            pinlv=21;
```

```
        break;
    default :
        pinlv=0;
        break;
        }
        delay1();
        while(!s2);                    //按键没松开，不再
重新执行中断，防止抖动
        }
    while(!s2);
    EX1=1;              //开中断
}
void main()
{
        csda=0;
        wr=0;                          //片选和  WR  端置零,
0832 直通模式
        EA=1;                                  //开中断
    IT0=1;
    EX0=1;
    IT1=1;
    EX1=1;                             //中断 0 和 1 开中断
```

```
        while(1)
        {
          while(a)                    //根据五种波形的标志位
执行相应的循环
          {
          delay=pinlv;
          P0=sin[k];                        //把数组中的数值付
给 p0 口
          k++;
          if(k==64)                   //查表下标为  64 后重新
置零
              k=0;
          while(delay)
                delay--;                  //根据频率不同延时相
应的时间
          }
          while(b)
          {
          delay=pinlv;
          P0=juxing[k];
          k++;
          if(k==64)
```

```
    k=0;
while(delay)
    delay--;
}
while(c)
{
delay=pinlv;
P0=juchi[k];
    k--;
if(k==0)
    k=64;

while(delay)
        delay--;
}
while(d)
{
delay=pinlv;
P0=(247-tixing[k]);
    k++;
if(k==64)
    k=0;
```

```
        while(delay)
                delay--;
        }
        while(e)
        {
        delay=pinlv;
        P0=sanjiao[k];
        k++;
        if(k==64)
            k=0;
        while(delay)
                delay--;
          }
        }
}
```