11.1　简述从控制系统的 Matlab 仿真到生成单片机代码的基本流程。

解答：作为一款功能强大并被广泛应用的科学计算软件，Matlab 也经常用于对控制系统的性能进行初期的验证。其目的是在复杂系统建立之前能够预测系统的性能和参数，使设计的控制系统达到最优指标。控制系统 Matlab 仿真的基本步骤包括：

（1）建立控制系统的数学模型，包括：微分方程模型，传递函数模型等；

（2）将建立的模型转换为仿真模型；

（3）利用 Matlab/Simulink 编写控制系统仿真程序；

（4）对仿真模型进行修改校验，看与实际系统是否一致，确认模型的正确性；

（5）运行仿真程序，在不同的初始条件和参数下，对系统进行反复分析。

控制系统仿真的目的是为搭建实际系统做前期准备。在得到合理的仿真结果以后，就需要着手实物验证了。这其中最重要的工作之一是编写能够在单片机中运行的代码。Matlab 中集成了 m 代码（Simulink 代码）转 C/C++代码的工具，可以方便地将 Matlab/Simulink 编写的控制系统仿真程序转换为能够在单片机中运行的有效代码，其基本流程包括：

（1）打开 MATLAB，选中菜单栏中的 HOME，点击下面的 new 按钮，并选择 Function，在函数中添加设计好的算法，保存到相应的文件中；

（2）选中菜单栏中的 APPS，点击下面的 MATLAB Coder 按钮，出现工具箱的操作界面，点击浏览按钮，选择目标函数的 m 文件，确认无误之后点击 Next 进入 Define Input Types 界面；

（3）在 Define Input Types 界面，单击 Let me enter input or global types directly 定义函数中的变量的类型，确认无误之后点击 Next 按钮进入 Check for Run-Time Issues 界面；

（4）在 Check for Run-Time Issues 界面中可以直接点击 Next 进入代码生成页面；

（5）在代码生成页面选择好代码语言，单击 generate 按钮生成 C/C++代码；

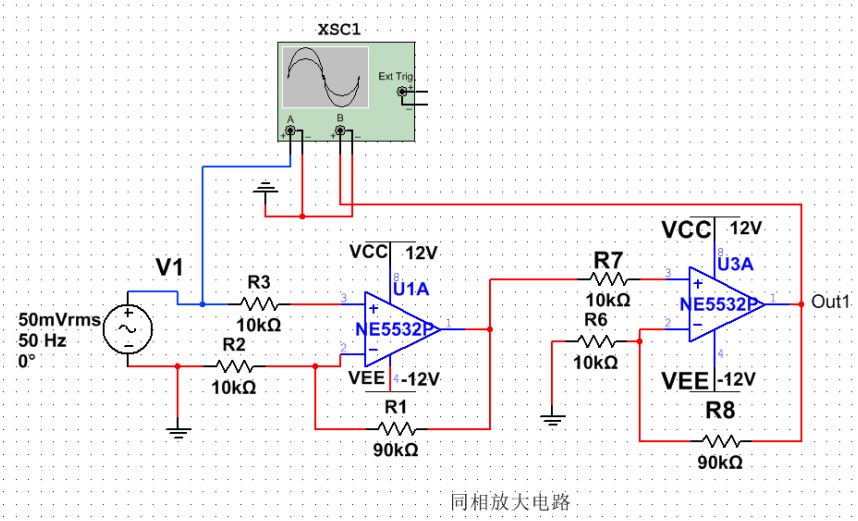
（6）由于生成的代码是一个集成的函数，所以需要把生成的 C/C++代码函数复制到单片机程序的相应部分，调用算法实现算法的功能。

11.2　结合 Keil 软件开发环境在线仿真和 Proteus 仿真特点，分析二者有何异同点。

解答：Keil 软件开发环境在线仿真可以查看程序变量在程序运行过程中的状态及取值，同时 Keil 软件开发环境可以在线仿真单片机的定时器中断功能，但不中断间隔时间与实际单片机运行的间隔时间存在较大误差，此外 Keil 软件开发环境无法仿真验证单片机外围逻辑信号输入和输出变化，如无法验证按键输入、数码管显示等。Proteus 通过加载单片机程序，通过与实际电路非常接近的仿真电路，进一步验证单片机外围信号变化情况，但无法在仿真过程中及时查看变量改变情况，二者各有优缺点。

11.3　采用 Multisim 搭建一个两级放大电路，输入信号为 0~50mV，每级放大 10 倍，通过示波器查看放大后波形。

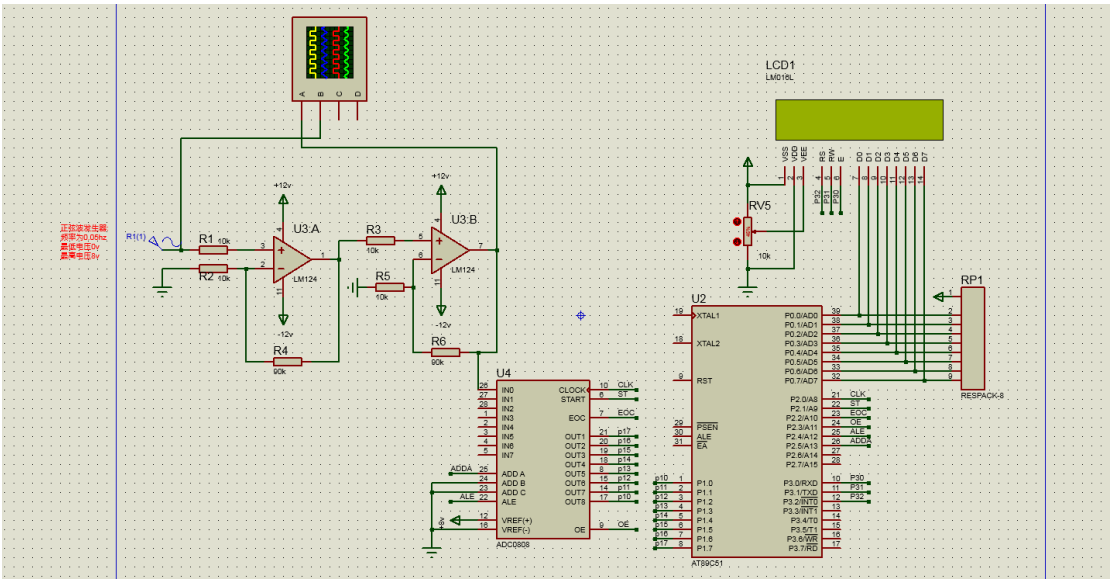
解答：硬件仿真电路如下



同相放大电路

11.4　结合 Proteus 仿真环境，选择合适的单片机和 A/D 转换芯片，将习题 11.3 的放大后信号进行采集，采集之后的结果在 LCD 或多位数码管进行显示，通过编写程序验证单片机采集结果。

解答：仿真电路图如下。




```
#include<reg52.h>
#define uint unsigned int
#define uchar unsigned char

sbit clock=P2^0;
sbit start=P2^1;
sbit eoc=P2^2;
sbit oe=P2^3;
sbit ale=P2^4;
sbit adda=P2^5;
sbit key=P2^7;
```

```
sbit rs=P3^2;//1602 的数据/指令选择控制线
sbit rw=P3^1;//1602 的读写控制线
sbit en=P3^0;//1602 的使能控制线

uchar qian,bai,shi,ge;
uint temp;
uint n=0;

uchar lcd_number[]={0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39};
//LCD1602 显示数字 0-9
void delay(uint z) /*****延时函数*****/
{
    uint x,y;
    for(x=z;x>0;x--)
    for(y=1;y>0;y--);
}

void lcd_write_command(uchar com) /*****1602 写命令函数*****/
{
    rs=0;//选择指令寄存器
    rw=0;//选择写
    P0=com; //把命令字送入 P0
    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_write_data(uchar dat) /*****1602 写数据函数*****/
{
    rs=1;//选择数据寄存器
    rw=0;//选择写
    P0=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_write_command(0x38);//8 位数据，双列，5*7 字形
    lcd_write_command(0x0c);//开启显示屏，关光标，光标不闪烁
    lcd_write_command(0x06);//显示地址递增，即写一个数据后，显示位置右移一位
    lcd_write_command(0x01);//清屏
}
```

```
void timer0_init()/*****定时器 0 初始化设置*****/
{
    TMOD=0X21;//定时器 0，工作方式 1
    TH0=(65536-50000)/256;
    TL0=(65536-50000)%256;//50ms 中断一次
    TR0=1;//中断启动位
    ET0=1;//中断允许位,1 允许中断
    EA=1;
}

void timer1_init()/*****定时器 1 初始化设置*****/
{
    TMOD=0x21; //设置定时器 1 为工作方式 2
    TH1=(65536-200)/256; //0.2ms
    TL1=(65536-200)%256;
    EA=1;          //开总中断
    ET1=1;         //开 t1 中断
    TR1=1;
}

/********主函数**********/
void main()
{
    lcd_init();        //液晶初始化
    timer0_init();  //定时器 0 初始化
    timer1_init();  //定时器 1 初始化

    start=0; //复位
    oe=0; //输出
    adda=0;
    ale=0; //关闭地址选择

    while(1)
    {
      if(n%20==0)
      {
        start=0;
        start=1; //  复位
        ale=1; //  打开地址选择
        adda=0;
        start=0; //  开始转换
        ale=0; //  关地址
        while(eoc==0); //  等待 eoc 变为 1
```

```
            oe=1; //  打开输出
            temp=P1; //  取 p1 到 p3
            oe=0; //  关输出

            temp=temp*1.0/255*8*100;  //转换数据，8 位，基准电压 5v，扩大了 100 倍
            bai=temp/100;             //取十位百位个位
            shi=temp%100/10;
            ge=temp%10;

            lcd_write_command(0x80);              //LCD1602 写地址
            lcd_write_data(lcd_number[bai]);   //LCD1602 写数据
            lcd_write_command(0x81);
            lcd_write_data('.');                  //写小数点
            lcd_write_command(0x82);
            lcd_write_data(lcd_number[shi]);   //LCD1602 写数据
            lcd_write_command(0x83);
            lcd_write_data(lcd_number[ge]);       //LCD1602 写数据
            lcd_write_command(0x84);
            lcd_write_data('V');                  //写单位
          }
        }
}

void timer0() interrupt 1    /*****定时器中断 0*****/
{
        TH0=(65536-50000)/256;
        TL0=(65536-50000)%256;//50ms 中断一次
        n++;
        if(n>=65500)
            n=0;
}

void timer1() interrupt 3      /*****定时器中断 1*****/
{
        TH1=(65536-200)/256; //0.2ms
        TL1=(65536-200)%256;
        clock=!clock;
}
```