中国地质大学（武汉）自动化学院
智能地学虚拟仪器实习报告

课　　程：智能地学虚拟仪器

学　　号：20201000128

班　　级：231202

姓　　名：刘瑾瑾

指导老师：王亚午、孟庆鑫
王广军、谢桂辉

二①二二年十二月二十日

目录

第一章 实习概况

1.1 实习目的

- 学习并使用配置 LabVIEW 实验平台和使用硬件开发装置；
- 掌握嵌入式系统采样工作原理和工程开发装置；
- 熟练使用简单的 51 单片机程序编程；
- 学习使用 LabVIEW 实验平台构成虚拟仪器。

1.2 实习内容

- 基于 LabVIEW 的软件功能模块练习；
- 仪器功能模块设计（软件）；
- 仪器功能模块（硬件）；
- 地学虚拟仪器构建。

第二章 LabView 部分

2.1 实习目的

通过 LabVIEW 实验平台，搭建信号发送程序以及数据采集程序，并与单片机实现串口交互，实现发送信号和采集信号的功能。需重点掌握知识点有：

- LabVIEW 程序结构（循环、事件、顺序）；
- LabVIEW 串口通信；
- LabVIEW 数据类型（数值、字符串、枚举、数组、簇）及其操作函数；
- LabVIEW 图形显示（波形图、波形图表）；
- LabVIEW 文件I\O 操作；
- LabVIEW 信号生成等。

2.2 登录界面设计

2.2.1 设计要求

设计一个登录界面，当账号和密码输入正确时，进入发射和采集功能选择界面，使虚拟仪器的使用具有一定的保密性。

2.2.2 设计思想

图 1 登陆界面设计

在 while 循环中采用层叠式结构，检测到账号和密码的输入，执行比较程序部分，若账号和密码与设定一致，则打开动态界面，否则不执行任何操作。

2.2.3 登录界面程序设计

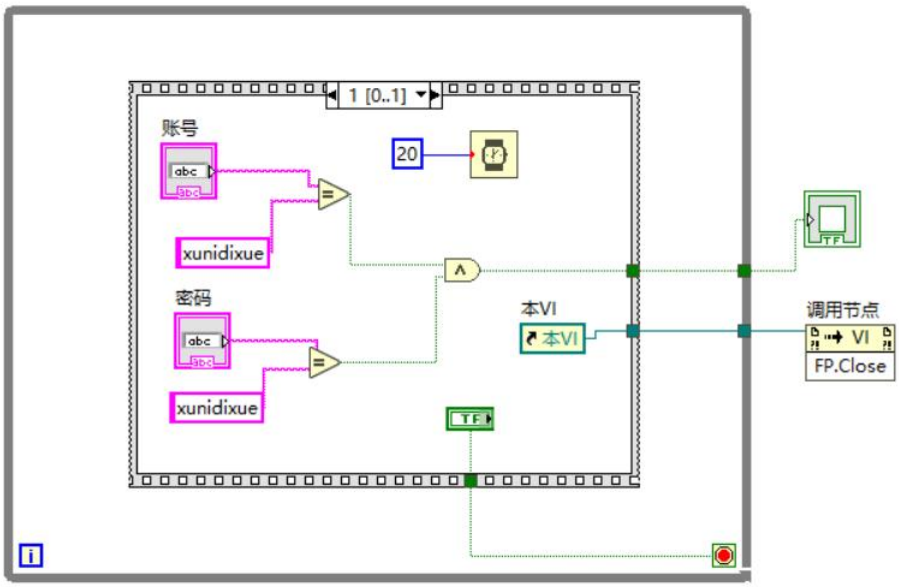


图 2 登录界面程序设计

2.3 动态界面设计

2.3.1 设计要求

在动态界面中，通过在动态界面 vi 中调用采集子 vi 和发射子 vi,可以直接选择发射部分和采集部分，方便用户切换功能，随时实现发射模块和采集模块的切换。

2.3.2 设计思想

在 while 循环中设置事件结构,通过控件选择调用相应的子 vi 或者退出界面。

图 3 动态界面前面板上设计

2.3.3 动态界面程序设计

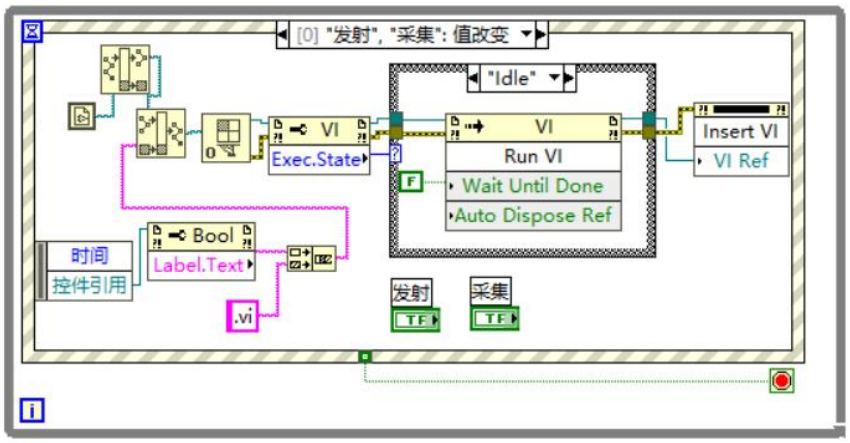


图 4 动态界面程序设计

2.4 发射部分

2.4.1 设计要求

发送部分可以通过串口向单片机下发指令，可进行波形选择：单极性方波和双极性方波，设置方波频率在 10-1000Hz 之间，可设置放大倍数、叠加次数、采样长度和采样频率，可添加校验码。

2.4.2 包括模块

- While 循环，For 循环；
- 事件结构（发射按钮值改变事件，停止按钮值改变事件）；
- 顺序程序结构；
- VISA 配置串口；
- 枚举类型数据（设置激励类型）；
- 数值输入控件（设置激励频率）。注意：信号界限；
- 激励信号显示（方波波形；波形图）；
- 将带有 ASCII 码的整数转换为相应的 ASCII 字符（重难点）；
- VISA 写入，VISA 关闭。

2.4.3 设计思想

在 while 循环放置事件结构，以便处理多个事件：超时事件和值改变事件。超时事件默认为-1，代表永不超时；值改变事件有两个：发射和停止；在平铺式顺序结构中使用枚举类型控件进行波形频率和波形类型的选择，可以设置放大倍数、叠加次数、采样长度和采样频率，可添加校验码。发送信号时，通过波形显

示在前面板上显示发送的信号波形。

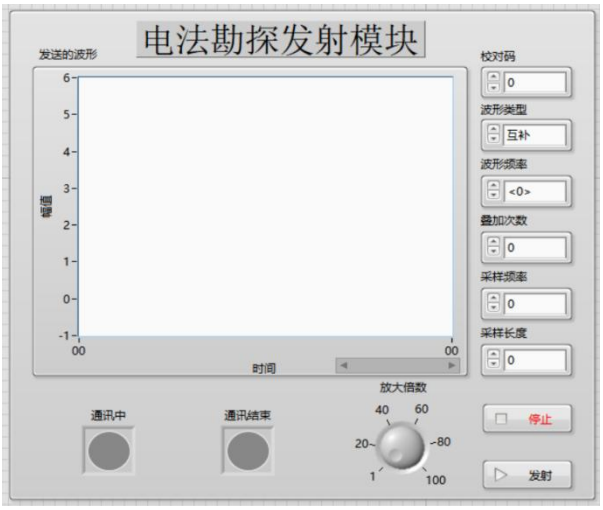


图 5 发射部分前面板设计

配置 VISA 串口通信的相关参数，如波特率、数据比特，便于与单片机进行通信。由于串口通信发送的数据类型是字符，需要将带有 ASCII 码的整数转换为相应的 ASCII 字符。要求的频率范围为 10-1000Hz 将需要发送的字符串连接起来，写入 VISA 缓冲区后，关闭 VISA。

信号发送流程图如下所示：

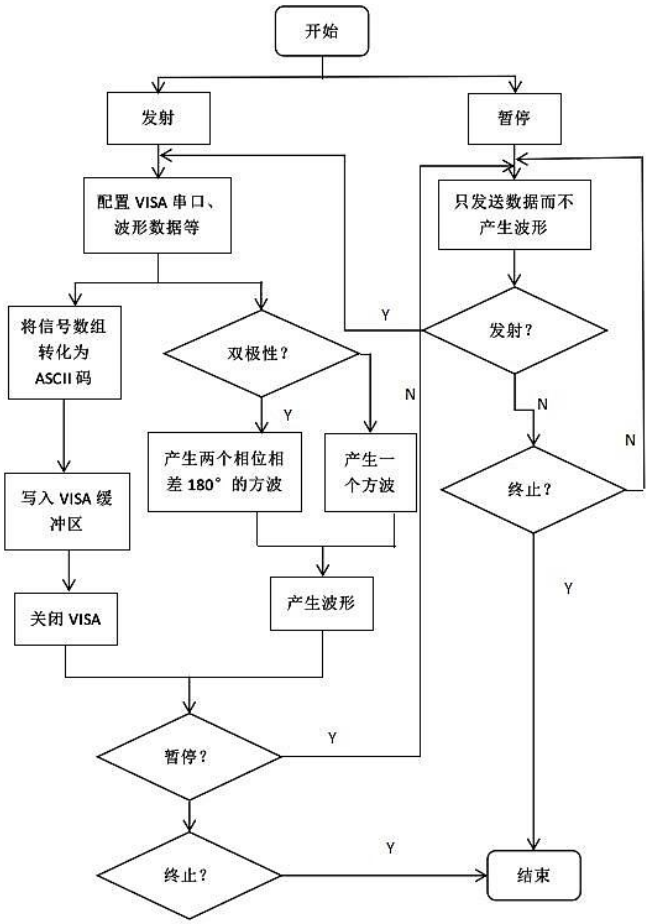


图 6 发射部分信号流程图

2.4.4 发射部分程序设计

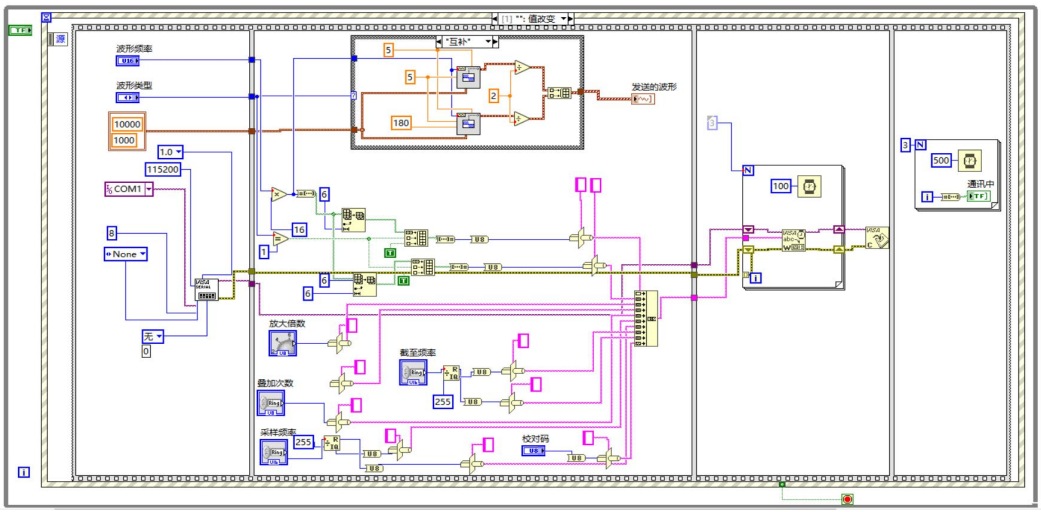


图 7 发射部分程序设计

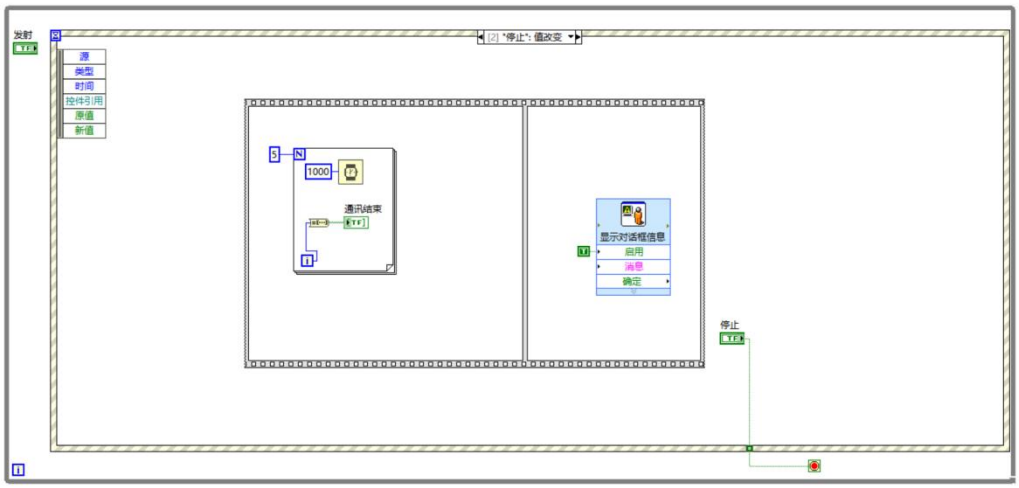


图 8 停止部分程序设计

2.5 采集部分

2.5.1 设计要求

采集部分通过串口通信接收来自单片机 ADC 采集的信息，将信息在前面板上显示并将数据记录在文件中。

2.5.2 包括模块

- 顺序程序结构；
- 事件结构（采集按钮值改变事件）；
- VISA 配置串口；
- VISA 读取；
- 信号数据转换与显示（波形图表）；

- VISA 关闭；
- 打开或创建文件；
- 写入文件；
- 关闭文件。

2.5.3 设计思想

配置 VISA 串口通信的相关参数，如波特率、数据比特，便于与单片机进行通信;通过 while 循环读取 VISA 的数据资源，将数据进行处理，在前面板上显示采集到的波形信号并记录在文件中。由于串口通信发送的数据是 8 位二进制数据，电压峰值为 5V,需要进行数据的处理：将数据除以 255 后再乘以 5，得到的便是采集到的实际数据。

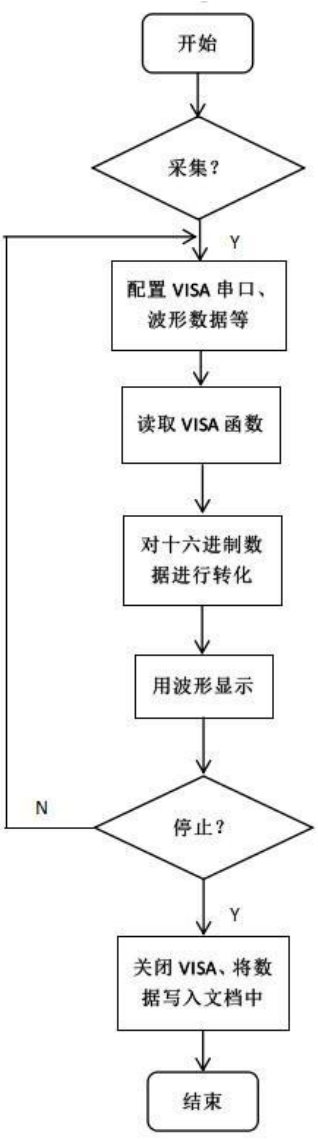


图 9 采集部分信号流程图

2.5.4 采集部分程序设计

图 10 采集部分前面板设计

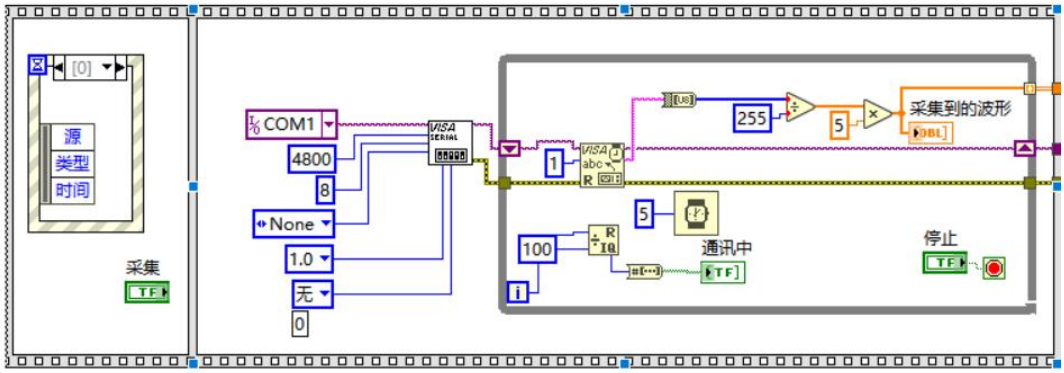


图 11 采集部分数据处理程序设计

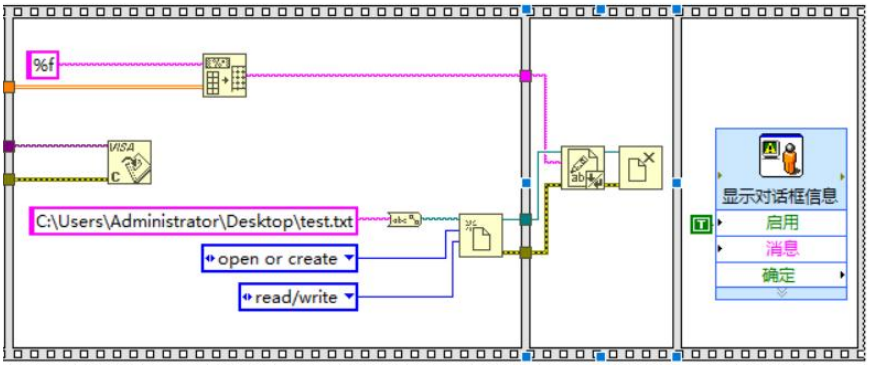


图 12 文件操作程序设计

第三章 51 单片机部分

3.1 设计要求

编写单片机程序，实现 Labview 可以通过 RS232 与单片机系统进行通信，单片机对数据进行解析并完成对输出 PWM 波的控制作用，即信号波形产生部分；进行 ADC 数据采集将其发送给上位机，即 ADC 数据上传部分

- **信号波形产生**

信号波形产生原理具体如图所示，在上位机 Labview 界面通过 RS232 将指令发送到 51 单片机系统，然后在 51 单片机中对数据进行解析，包括波形产生的启停、波形输出的类型模式以及波形的频率。解析完成后，按照指令从 51 单片机的 P1 口进行电平输出，控制高低电平时间达到调频效果。最后将输出信号波形接入放大电路进行变换。

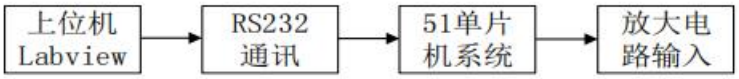


图 13 信号波形产生结构图

- **ADC 数据上传**

ADC 数据上传原理具体如图 2.2 所示，首先将外部变换得到后的信号输入到 ADC0809 模数转化芯片中，ADC0809 是美国国家半导体公司生产的 CMOS 工艺 8 通道，8 位逐次逼近式 A/D 模数转换器。其内部有一个 8 通道多路开关，它可 以根据地址码锁存译码后的信号，只选通 8 路模拟输入信号中的一个进行 A/D 转换，而且输入电压为 0-5V。ADC0809 通过逐次逼近 A/D 转化的方法进行 ADC 值采集，然后将数据存储在 51 单片机系统中，然后通过 RS232 通信将数据实时发送到上位机 Labview 中显示

图 14 ADC 数据上传结构图

3.2 信号波形产生部分

信号波形产生实验的流程图如图所示。首先先对串口、定时器 T0 进行初始化和设定初值，在 while（1）循环中对数据进行修改。当上位机下发指令时，串口接收数据产生中断，将数据存储并将接收数据完成标志位置 1，然后返回主函数。

将数据信息进行解析，包括启停、输出模式、频率（N 次定时器 T0 中断执行一次电平翻转），打开定时器 T0 后，定时器 T0 就可以每 1000us 进行一次定时器 T0 中断，在中断里循环计数，当计数 num 等于 N 时就让输出的电平翻转一次。

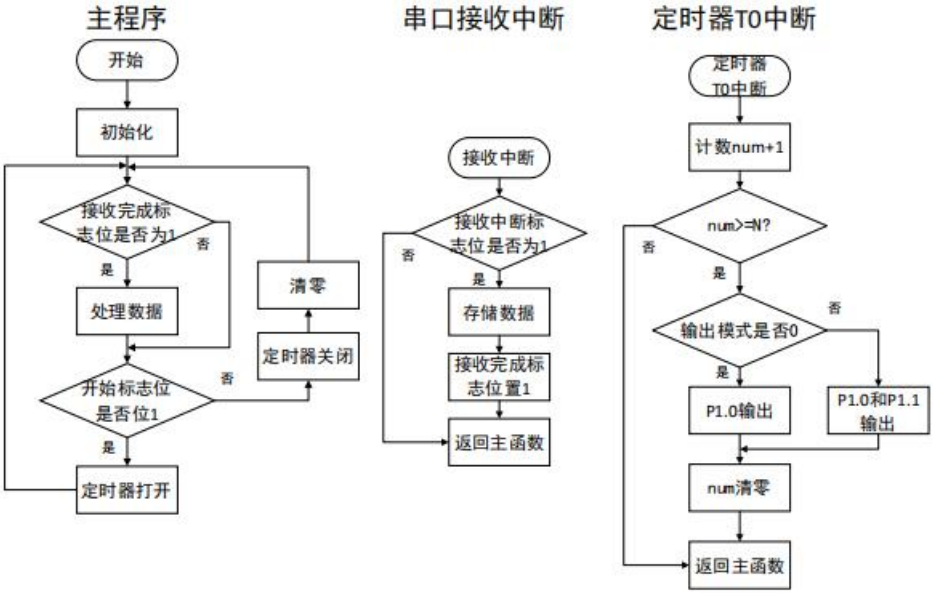
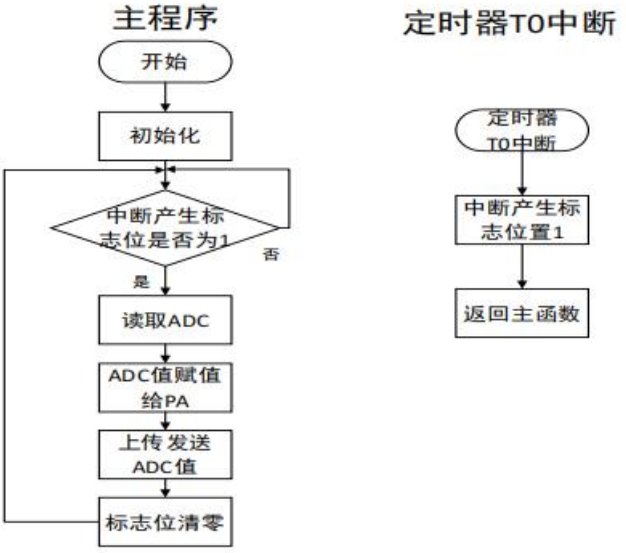


图 15 信号产生部分程序流程图

3.3 ADC 数据上传部分

ADC 数据上传实验的流程图如图所示。首先先对串口、定时器 T0 进行初始化和设定初值，在 while（1）循环中等待定时器 T0 产生中断并将标志位置 1，



当标志位置 1 时就将 ADC 数据进行采集上传。

图 16 ADC 数据上传部分程序流程图

第四章 电路部分

4.1 功率驱动电路

在地学探测仪器中，由控制电路产生的主动源信号一般没有功率驱动功能，

必须增加功率驱动模块。对于低频正弦交流信号，常用的功率驱动模块主要有甲乙类放大。但是，对于较高频率的信号，常用全桥放大电路。本次实习采用 IR2110 进行全桥电路的设计。

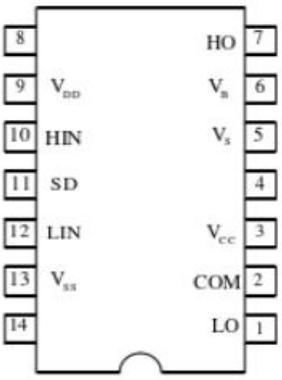


图 17 IR2110 芯片引脚图

MOSFET 的驱动电路由 IR2110 芯片构成。MOSFET 的导通与关断由栅极 g 上的电压控制，一些大功率的 MOSFET 的导通电压一般在 10V 左右。一般来说单片机引脚输出的电压为 3.3V 或者 5V，所以单片机输出的电压并不能直接驱动 MOSFET。

IR2110 将 PWM 信号的幅值放大后对 MOSFET 进行驱动。驱动电路的原理图如图 所示。IR2110 驱动电路需要 5V 和 12V 两个电源进行供电，从单片机输出的 PWM 信号从 IR2110S 的 HIN 引脚和 LIN 引脚输入，SD 信号控制 IR2110S 芯片的输出，低电平时 HO 引脚与 LO 引脚有输出，高电平时 HO 引脚与 LO 引脚无输出。

经过 IR2110 后 PWM 控制信号的幅值为 12V。其中电容 C19、C28 为自举电容，快速恢复二极管 D2、D8 为自举二极管，自举二极管是为了防止 MOS 管 Q1 和 Q2 导通时，高电压进入引脚 VCC 端而损坏该芯片。电容 C21、C22、C30、C31 为功率电源的滤波电容，电容 C27、 C14 为逻辑电源的滤波电容。

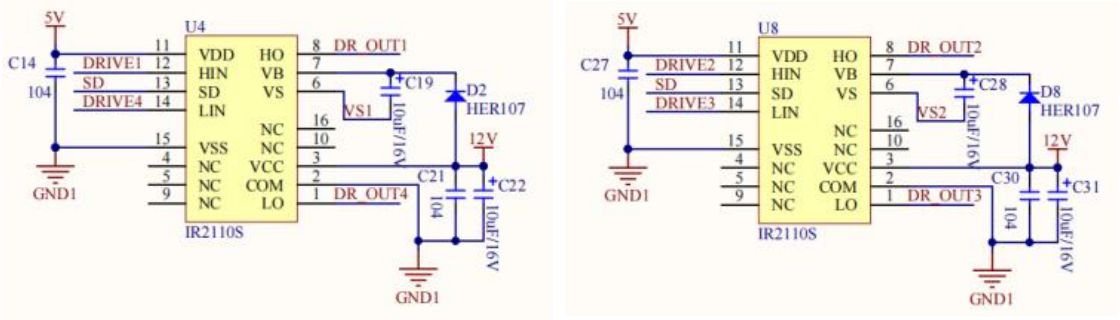


图 18 功率驱动电路电路图

4.2 全桥驱动电路

全桥驱动电路由 4 个 N 沟道 MOSFET 构成，如图所示。全桥电路需要外接电源 VCC 和 GND，全桥电路一般由 PWM 信号控制桥臂的通断，当 PWM 信号的高电平施加到 Q1 和 Q4 的栅极时，此时 Q2 与 Q3 的栅极施加的 PWM 信号为低电平，电流到的流向如图 所示，此时负载电阻上得到正向的电流。当 PWM 信号的高电平施加到 Q2 和 Q3 的栅极时，此时 Q1 与 Q4 的栅极施加的 PWM 信号为低电平，电流到的流向如图 所示，此时负载电阻上得到反向的电流。当该电路连续工作时，负载上就可以得到双极性的方波交流电。

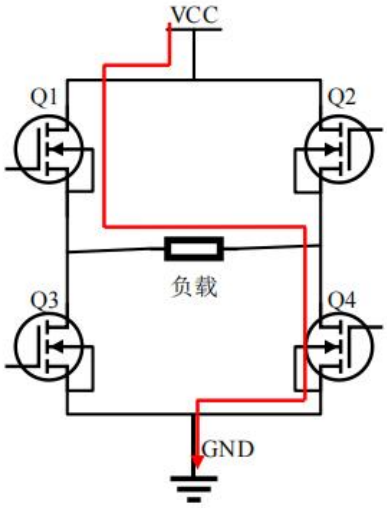


图 19 Q1 Q2 导通工作

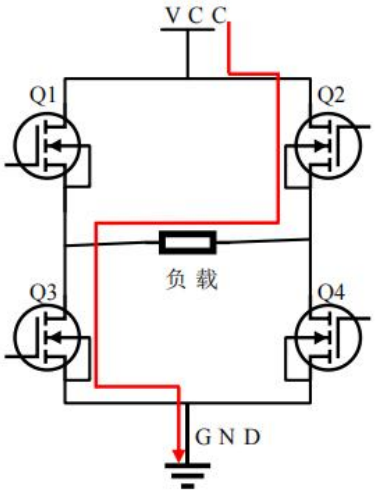


图 20 Q2 Q3 导通工作

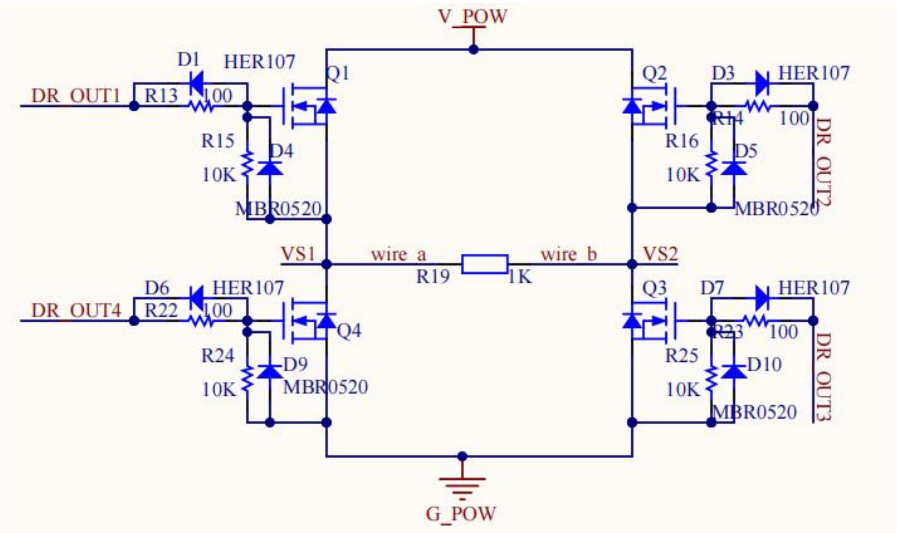


图 21 全桥驱动电路电路图

基于 IR2110 的 H 桥驱动电路的电路图如图 所示。电阻 R13、R22、R14、R23 是 IR2110 芯片的输出通道驱动 MOS 管栅极的限流电阻，取值一般为 100 欧姆，以防止栅极电流过大而损坏 MOS 管。由于弥勒效应的存在，MOS 管漏极产生的浪涌电压会通过漏极与栅极间的电容耦合到栅极上，并击穿栅极的氧化层，因此通常在栅极和源极之间加上稳压二极管来钳位栅极与源极之间的电位差。图中 D4、D9、D5、D10 为稳压二极管。

4.3 电路测试

功率放大电路和全桥驱动电路实物图如下：

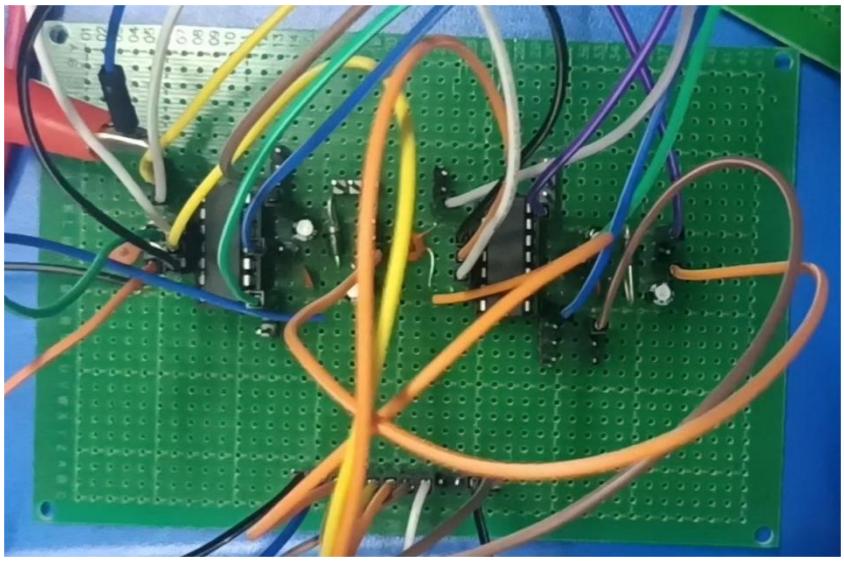


图 22 功率放大电路实物图

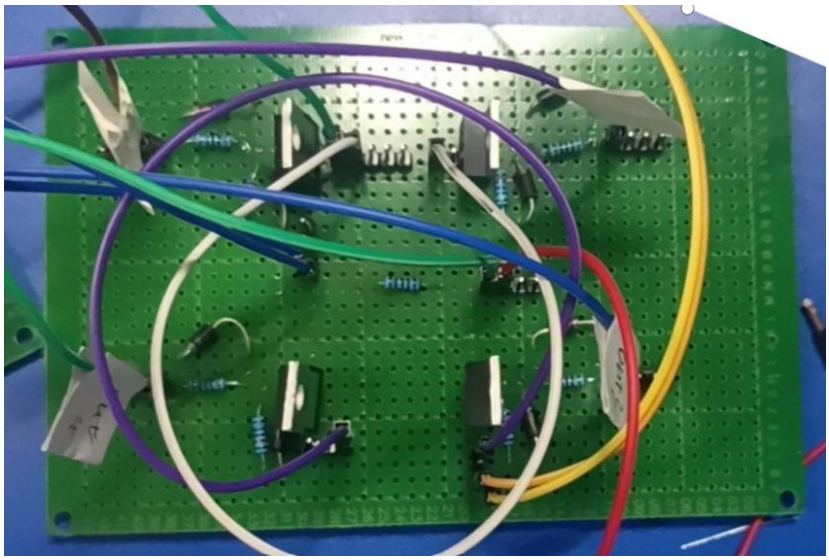


图 23 全桥驱动实物图

信号发生器输出两路互补 PWM 波，一路连接 IR2110 芯片的 DRIVE1 和 DRIVE3 引脚，一路连接 IR2110 芯片 DRIVE2 和 DRIVE4，将其输出的 PWM 波的正电压设置为 5V，负电压设置为 0V。

在 100HZ，占空比为 50%的方波下，波形图如下：

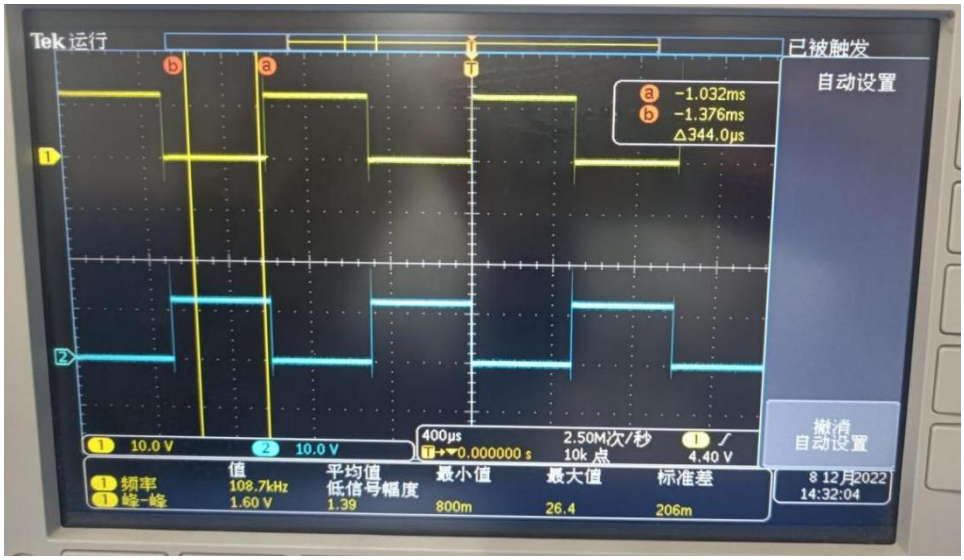


图 24　电路测试波形图

第五章 地学虚拟仪器构建

5.1 电法虚拟仪器构建

利用地下介质的电学性质差异为基础，使用专用的仪器设备，观测和研究与这些电性差异有关的（天然或人工）电场或电磁场的变化和分布规律，进而查明地下地质构造及有用矿产的一类地球物理勘探方法。

5.2 电法测量的基本结构

电法测量的基本结构如图所示，即由上位机 LabVIEW 发送指令使得单片机发送 100Hz 的 PWM 方波信号，经功率放大后传给发射电机，电极传感器采集到信号经差分信号放大后通过单片机采集并由上位机 LabVIEW 采集信号。

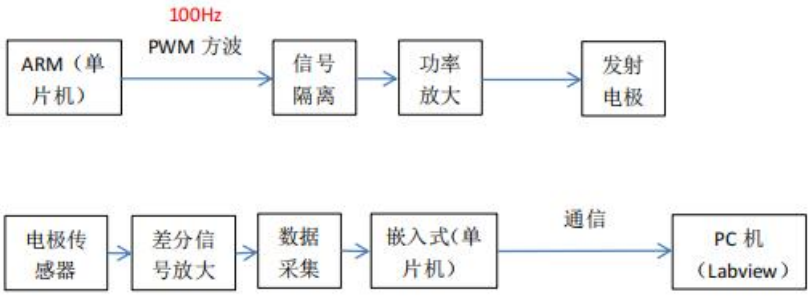


图 25 电法测量基本结构

5.3 电法勘探原理

如图所示，在 AB 两点插入信号源（一定频率或直流）产生激励信号，假设电流为 I，给电流在 AB 两点之间会产生电场，并在地下产生磁场。地下介质在磁场作用下会发生极化现象。如果地下介质属于高电阻率，极化不明显，AB

两点间的电场几乎是半圆形的，相应的磁场垂直与电场均匀分布。当介质中出现低阻时，低阻介质在磁场作用下就会发生极化产生涡流，涡流反过来影响磁场的分布。在表明插入电极 MN，MN 两点之间的电源随着介质电阻率的不同而发生变化。通过反演探测的数据，就可以获取介质的电阻率。

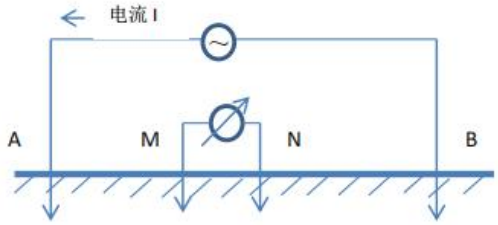


图 26 电法原理图

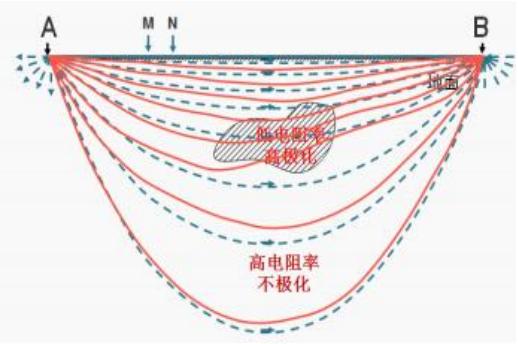


图 27 电流分布示意图

5.4 操作步骤

使用两台计算机，其中一台作为发射装置连接驱动电路，另外一台作为采集装置连接差分放大电路。在操作过程中移动测量极的位置，观测采集到的波形是否稳定。

在均匀介质、单低电阻率和双电阻率介质重复下面实验：

①按图 16 方法接入电极 AB，AB 点距离约为 150cm；

②在 AB 电极之间接入信号源（频率 10K 或直流），电压约为 5V；

③靠近 A 插入电极 MN，MN 之间的距离为 10cm；

④采集 MN 两点的信号电压 Vi ；

⑤移动 MN，重复④，直到 MN 接近 B 点。

第六章 总结与分析

本次智能地学虚拟仪器分为四个部分：基于 LabVIEW 的软件功能模块练习、仪器功能模块设计（软件）、仪器功能模块（硬件）和地学虚拟仪器构建。其中地学虚拟仪器构建内容更加综合，是在前三个部分的基础上组建而成的，要求每一部分系统都可以正常工作。这次实习的涉及面非常广，涵盖了 LabVIEW 虚拟仪器设计、单片机程序设计、硬件模块电路设计和电法勘探，设计过程很考验综合能力和团队合作能力，

在本次虚拟仪器实习中，我主要负责的是 LabVIEW 部分。这次实习综合运用了虚拟仪器 LabVIEW 的相关知识，如：事件结构、循环结构、条件结构和文件操作等，设计了发射部分和采集部分的图形化程序，让我明白了如何使用 LabVIEW 构建具有实际使用功能的虚拟仪器，了解了虚拟仪器在工程项目中的应用。在构建地学虚拟仪器的过程中，发射端上位机通过串口通信向单片机下发指令，单片机发出 PWM 波通过全桥驱动电路产生方波信号，接收端通过差分放大电路连接单片机进行 ADC 采集，单片机采用定时器中断将数据发往接收端上位机。通过对构建地学虚拟仪器的实际操作，我对虚拟仪器的实际应用有了更深刻的理解，在实际操作中运用所学知识，提高了我的实践能力。

通过本次实习，我熟悉了 LabVIEW 的使用、电路焊接、电法勘探的原理及应用，在和小组同学的合作下完成了本次试验。在实习过程中，我们也遇到了一些问题，但最终通过请教同学和老师得到了解决。经过本次实习，我对 LabVIEW 软件的使用更加熟练，可以自己独立的设计一些简单的实际应用程序，但仅仅掌握这些远远不够，未来的路任重而道远，需要我学习更多 LabVIEW 的相关知识，相信有本次实习的经验，后面的学习会更加顺利。

附录

信号波形产生程序代码：

```
#include <reg51.h>

#define mode 0x82

xdata unsigned char CTL _at_ 0x9003;//地址

unsigned char rcv_flag=0; //接收到数据标志 1：接收完成

unsigned char start=0; //启动标志 1：开启 0：停止

unsigned char out_mode=0; //输出模式 1：P1.0 和 P1.1 输出互补波形 0：

//P1.0 输出方波

unsigned char arry[3]=0; //存储接收的数据

unsigned int num=0; //定时器 T0 计数

unsigned char N=0;

unsigned char f=0;

int timeflag=0;

int hubuflag=0;

sbit Output0 = P1^0; //输出变量

sbit Output1 = P1^1; //输出变量

sbit p2=P2;

int i=0;

//中断服务程序 interrupt 0 指明是外部中断 0 的中断函数

/*

interrupt 0 指明是外部中断 0；

interrupt 1 指明是定时器中断 0；

interrupt 2 指明是外部中断 1；

interrupt 3 指明是定时器中断 1；

interrupt 4 指明是串行口中断；

*/
```

```
void sci_rcv(void) interrupt 4 using 1 //串口接收中断
{
   if(RI==1) //接收中断标志位置  1
  {
     RI = 0; //接收中断标志位清 0
arry[2]=arry[1];
arry[1]=arry[0];
     arry[0]= SBUF; //接收数据
     if(arry[0]==120)rcv_flag=1; //接收完数据
   }
  }
}
void time0(void) interrupt 1 //定时器  T0  中断
{
        TH0=0xff; //1000us  定时
        TL0=0xdb;
       if(timeflag==0)
       {
            TH0=0xfe;
            TL0=0x1a;
       }
     num++; //计数
     if(num>=N) //经过  N  执行一次
   {
       if(out_mode==0) //输出模式  0  时
        {
            if(Output0==1)
```

```
        {
          Output0=0; //电平翻转
        }
          else
              Output0=1; //电平翻转
        }
      if(out_mode==1) //输出模式  0  时
      {
          unsigned char i=0; //延时计数
        if(Output0==0)
        {
              Output1=0;
              for (i=0; i<5; i++); //延时
              Output0=1; //电平翻转
        }
        else
      {
              Output0=0; //电平翻转
              for (i=0; i<5; i++); //延时
              Output1=1;
        }
      }
      num=0; //计数清零
    }
}
void init_T0() //定时器  T0  初始化
{
```

```
    TMOD|=0x01; //工作方式 1
    TH0=0xff; //1000us 定时 (65536-1000)/256
    TL0=0xdb; // (65536-1000)%256
    TR0=0; //开启 T0 定时器
    ET0=0; //允许 T0 定时器中断
    EA=1; //开启总中断允许
}
void init_sci()
{//串?
    SCON=0x50; //串口工作方式 1
    TMOD|=0x20; //定时器 T1 工作方式 2
    PCON=0x80; //电源寄存器 SMOD=1
    TL1=243; //bps=(2^SMOD/32)*fosc/[12*(256-TH1)]
    TH1=243; //bps=4800 fosc=12M
    TR1=1; //启动定时器 1
    EA=1; //打开总中断
    ES=1; //打开串口中断
}

void main()
{
  CTL = mode; //0x82
  init_sci(); //串口初始化
  init_T0(); //定时器 T0 初始化
  Output0=1; //P1.0 初始电平高
  Output1=0; //P1.1 初始电平低
  while(1)
```

```
{
  if(rcv_flag==1) //接收到数据进行解析
 {
  timeflag=1;
  if(arry[1]&0x80&&arry[2]&0x80) //最高位为 1 开启
     start=1;
  else
     start=0; //最高位为 0 关闭
  if(arry[1]&0x40&&arry[2]&0x40) //第 7 为 1 输出模式 1
        {
             out_mode=0;
             hubuflag=0;
        }
  else
        {
             out_mode=1; //第 7 为 0 输出模式 0
             hubuflag=1;
        }
  arry[1]=arry[1]<<2; //
  arry[1]=arry[1]>>2; //后 6 为数据提取

  arry[2]=arry[2]<<2; //
  arry[2]=arry[2]>>2; //后 6 为数据提取
  if((arry[1]*64+arry[2])<45)
        {
             timeflag=0;;
             N=1000/(arry[1]*64+arry[2]);
```

```
            }
            else{
            N=10000/(arry[1]*64+arry[2]);
            }
      if(N==0) //N 等于 0 时为 1
          N=1;
      if(N>2000) //大于 200 时为 200 最大
          N=2000;
       rcv_flag=0; //数据解析完成清零
      }

    if(start==1) //开启时
    {
       TR0=1; //开启 T0 定时器
       ET0=1; //允许 T0 定时器中断
      }
    if(start==0)
    {
      TR0=0; //关闭 T0 定时器
      ET0=0; //关闭 T0 定时器中断
      rcv_flag=0; //标志位清零
      N=0; //计数清零
    }
   }
}
```

2.信号波形接收程序代码：

```
#include "reg51.h"
```

```
#define mode 0x82
xdata unsigned char CTL        _at_ 0x9003;
xdata unsigned char PA         _at_ 0x9000;
xdata unsigned char CS0809 _at_ 0x8000;
unsigned char date=0;
unsigned char rcv_flag=0;    //接收数据
unsigned int N=0;
unsigned char arry[1]=0;
unsigned char it_flag=0;
unsigned char Read0809()
{
   unsigned char i;
   CS0809 = 0;                          //    A/D
   for (i=0; i<0x20; i++) ;      //  延时  > 100us
   return(CS0809);                    //
}

void ser_int (void) interrupt 4 using 1
{
      if(RI==1)                              //标志位检验
      {
            RI = 0;                              //接收标志位置 1
            arry[0]= SBUF;
                  rcv_flag=1;
                }
}
void time0(void) interrupt 1
```

```
{     it_flag=1;
          TH0=0xD8; //150us 定时     (65536-1000)/256
          TL0=0xEF;       //     (65536-1000)%256

}
void init_T0()
{
        TMOD|=0x01;
    TH0=0xD8; //1000us 定时     (65536-1000)/256
.TL0=0xEF;       //     (65536-1000)%256
 TR0=1;              //开启 T0 定时器
    ET0=1;               //允许 T0 定时器中断
    EA=1;                 //开启总中断允许
}

void init_sci()
{
 CTL = mode;
 SCON=0x50;
 TMOD|=0x20;
 PCON=0x80;
 TL1=243;
 TH1=243;
 TR1=1;
 EA=1;
 ES=1;
}
```

```
void main()
{
      init_sci();
      init_T0();
      rcv_flag=1;
while(1)
{
if(rcv_flag==1)
{
      if(it_flag==1)
      {
         date= Read0809();
            TI=0;
         SBUF=date;
      it_flag=0;
   }
}
}
}
```