自动增益控制放大器

日期：2024 年 5 月 19 日

摘　　要

本文实现了一种自动增益控制放大器的设计和实现。通过方案论证，我们选择了基于数字电位器的 AGC 电路方案，该方案在电路结构和调节方面具有明显优势。硬件设计包括运算放大器、有效值检波电路、数字电位器、ADC 采样模块、MCU 和显示屏的选型与设计等。软件设计部分详细介绍了 AD 采样、数字电位器调节以及主程序的编写等。系统测试表明，所设计的自动增益控制系统在不同输入信号条件下均能保持良好的性能，满足设计要求。在测试过程中，我们也发现了信号源准确性和电路共地问题对系统性能的影响。最终，本设计验证了其可靠性和稳定性，为自动增益控制领域提供了一种有效的解决方案。

本文完全采用 LaTeX 书写,工程细节和完整代码见 GitHub: https://github.com/wanansu/AGC_digital_potentiometer

关键词： 自动增益控制放大器 (AGC)，数字电位器，运算放大器，有效值检波电路，ADC 采样，STM32 单片机

1　方案论证

1.1　方案一: 运算放大器结合倒 T 型 DAC 电阻网络

使用运算放大器结合倒 T 型数字模拟转换器（DAC）电阻网络来实现自动增益控制（AGC）电路。该方案的核心是通过数字信号控制倒 T 型 DAC 电阻网络的等效电阻，从而调节运算放大器的增益。输入信号接入运算放大器的输入端，输出信号通过反馈电阻网络回馈到反相输入端，以实现所需增益。设计信号检测电路，通过模数转换器（ADC）和 STM32 单片机实时监测和调整输入信号的幅度。

如图1所示，为使用 multisim 进行的运算放大器结合倒 T 型 DAC 电阻网络的电路仿真图。

- 优点
  - **原理简单：**运算放大器和 DAC 电阻网络结合，原理易于理解。
  - **可调性强：**通过数字控制 DAC 电阻网络，实现精确的增益调节。
- 缺点
  - **电路复杂度高：**需要设计复杂的 DAC 电阻网络，增加了电路设计难度。
  - **控制难度大：** STM32 对电路的控制需要精确的数字控制，调试和校准较为复杂。

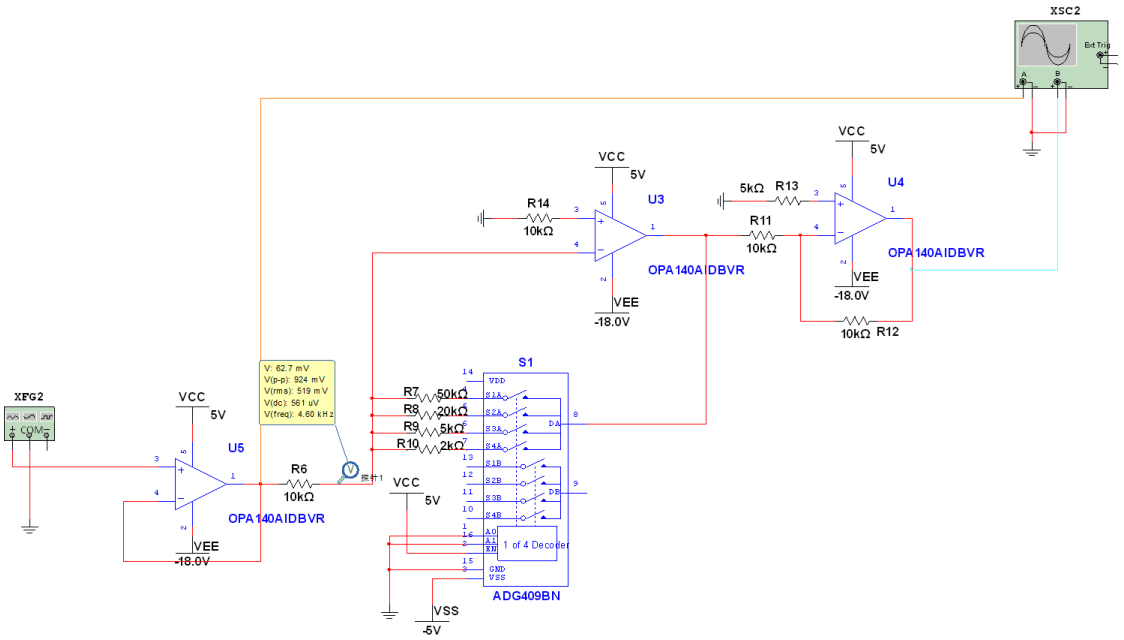


图 1: 运算放大器结合倒 T 型 DAC 电阻网络 - mustisim 仿真

1.2 方案二: 基于 VCA821 的 AGC 电路

采用基于 VCA821 的 AGC 电路。VCA821 是一款直流耦合、宽带、dB 线性的压控增益放大器。将 VCA821 与 OPA695 后级放大器组合，输出信号再经过 OPA820 积分器，连接至 VCA821 的增益控制（VG）引脚，形成闭环，从而保证输出信号的稳定性。

如图2为使用 TI 公司的电路仿真软件 Tina 进行的基于 VCA821 设计的 AGC 电路仿真图：

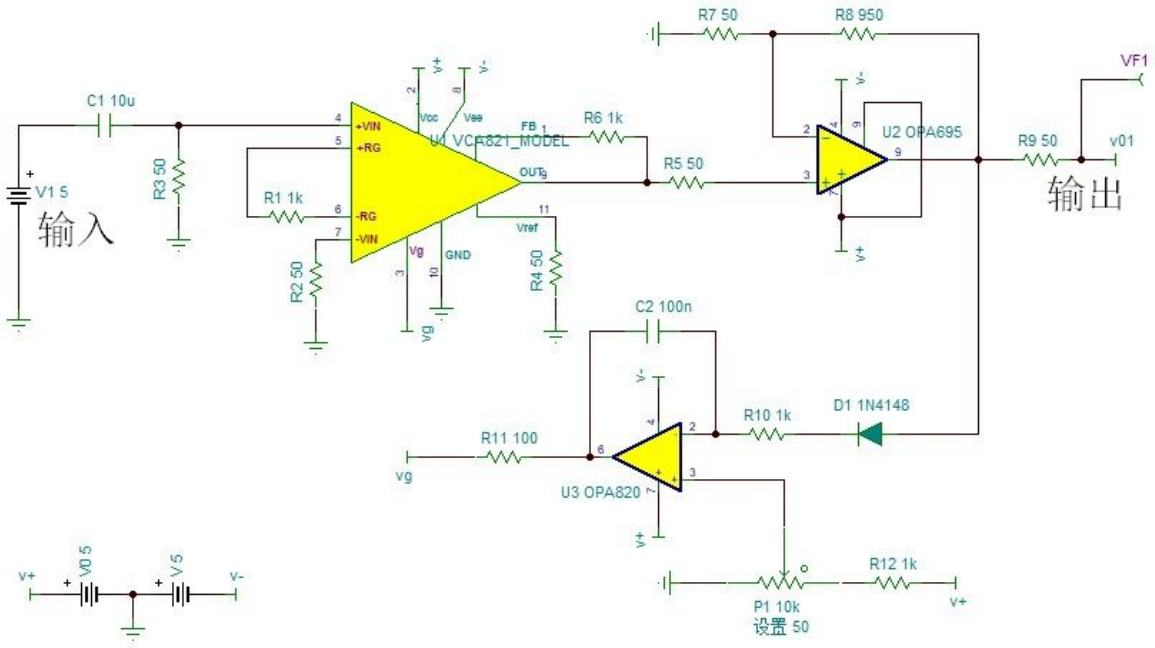


图 2: 基于 VCA821 的 AGC 电路 - Tina 仿真

- 优点

  - **原理清晰：** VCA821 的特性直观，增益可由外部控制引脚调节。
  - **频率响应好：**适用于宽带信号处理，输出信号稳定性高。
- 缺点
  - **电路复杂度高：** VCA821 和其配套电路较为复杂，需要精确匹配和调试。
  - **输入信号范围限制：**VCA821 的输入信号范围可能不满足特定应用需求，需额外考虑信号处理。

1.3 方案三: 基于数字电位器的 AGC 电路

采用基于数字电位器的 AGC 电路。设计一个两级反相放大电路，通过数字电位器调节电路的放大倍数，使用 STM32 单片机控制数字电位器，实现自动增益调节。对于交流信号，使用有效值检波电路将其转换为直流信号。利用 ADC 模块采集直流信号，通过单片机处理，实现对输入信号与输出信号的精确测量和调节。

- 优点
  - **电路结构简单：**两级反相放大电路和数字电位器组成的调节电路结构清晰简单。
  - **调节方便：**通过 STM32 单片机控制数字电位器，调节方便快捷。
- 缺点
  - **噪声和干扰：**数字电位器和其他数字元件可能引入噪声和干扰，影响输出信号质量。
  - **精确度问题：**有效值检波电路在处理非正弦波交流信号时可能会产生误差，需要精确设计和校准以确保检测精度。

1.4 综合分析

综上所述，选择方案三，设计一个基于数字电位器的 AGC 电路。尽管方案一原理简单，方案二频率响应好，但方案三在电路结构和调节方面具有明显优势，且通过精确设计和校准可以有效解决噪声和干扰问题。因此，方案三更适合实际应用需求。

2 硬件设计

2.1 硬件选型

2.1.1 运算放大器：OPA2134

选择理由： OPA2134 是一个高性能、低噪声的双运算放大器，适用于音频应用。其低失真、高开环增益和宽频率响应使其在音频信号处理方面表现出色。

- **NE5532：**NE5532 是一款广泛使用的双运算放大器，具有较低的噪声和失真。然而，与 OPA2134 相比，其音质略逊色，尤其在高端音频应用中表现不佳。
- **TL072：**TL072 具有低功耗和低偏置电流，但在音频质量和噪声性能上不如 OPA2134，特别是在高增益应用中。
- **LM4562：**LM4562 是一款高性能音频运放，具有极低的噪声和失真，性能接近 OPA2134，但价格较高，且供货不如 OPA2134 稳定。

最终选择： OPA2134 由于其卓越的音频性能和合理的价格，是最佳选择。

2.1.2 有效值检波电路：LF353

选择理由： LF353 是一个 JFET 输入的双运算放大器，具有高输入阻抗和低偏置电流，非常适合构建有效值检波电路。

- **TL082**：TL082 也是一个 JFET 输入运放，但其噪声性能不如 LF353，且在高频响应上稍显不足。
- **MC1458**：MC1458 是一个通用的双运放，但其输入阻抗和偏置电流性能不如 LF353，特别是在高阻抗信号源应用中表现较差。
- **OPA2604**：OPA2604 具有优异的音频性能，但在价格上比 LF353 高出许多，不适用于成本敏感的设计。

最终选择： LF353 由于其高输入阻抗和低噪声特性，是构建有效值检波电路的最佳选择。

2.1.3 数字电位器：AD5293

选择理由： AD5293 是一款高分辨率、非易失性的数字电位器，具有良好的线性度和宽范围的电阻选择。

- **MCP41100**：MCP41100 是一个易于使用的数字电位器，但其分辨率较低，仅为 8 位，不适用于需要精确调节的应用。
- **DS1803**：DS1803 具有较低的成本，但其电阻范围和温度稳定性不如 AD5293。
- **X9C103**：X9C103 提供多种电阻选择，但其线性度和分辨率不如 AD5293。

最终选择： AD5293 由于其高分辨率和良好的线性度，是最佳选择。

2.1.4 ADC 采样模块：ADS1115

选择理由： ADS1115 是一款 16 位精度的模数转换器，具有四通道和内置增益放大器，适用于高精度信号采集。

- **MCP3424**：MCP3424 也是一款 16 位 ADC，具有类似的通道数，但其采样速率较低，且不具备内置的可编程增益放大器。
- **ADS1015**：ADS1015 是 ADS1115 的低分辨率版本，只有 12 位精度，不适用于需要高精度的应用。
- **ADS7828**：ADS7828 具有多通道，但其分辨率和精度不如 ADS1115，且缺乏内置增益放大器。

最终选择： ADS1115 由于其高分辨率、内置增益放大器和多通道特性，是最佳选择。

2.1.5 MCU：STM32F103ZET6

选择理由： STM32F103ZET6 是一款性价比高的 ARM Cortex-M3 核心单片机，具有丰富的外设接口和较高的处理性能。

- **ATmega328P**：ATmega328P 是一款常用的 8 位单片机，虽然易于使用且成本低，但在处理性能和外设数量上不及 STM32F103ZET6。
- **PIC18F45K22**:PIC18F45K22 是一款高性能的 8 位单片机,但其性能和功能扩展性与 STM32F103ZET6 相比仍有不足。
- **MSP430G2553**：MSP430G2553 是一款超低功耗的 16 位单片机，适用于低功耗应用，但在处理能力和外设接口上不及 STM32F103ZET6。

最终选择： STM32F103ZET6 由于其高性能、多功能外设和性价比，是最佳选择。

2.1.6 显示屏：OLED 显示屏 0.96 寸

选择理由： 0.96 寸的 OLED 显示屏具有高对比度、低功耗和广视角，适合用于各种小型显示应用。

- **Nokia 5110 LCD**：Nokia 5110 LCD 具有较低的分辨率和对比度，且功耗相对较高，不如 OLED 显示屏在视觉效果上出色。
- **TFT 1.44 寸显示屏**：TFT 显示屏虽然具有更大的尺寸和更丰富的颜色，但功耗较高，且在小型、低功耗应用中不如 OLED 适用。
- **0.96 寸 LCD 显示屏**：相同尺寸的 LCD 显示屏在对比度和视角上不如 OLED，且功耗相对较高。

最终选择： 0.96 寸的 OLED 显示屏由于其高对比度、低功耗和广视角，是最佳选择。

2.1.7 总结

综上所述，经过比较和权衡，我们最终选择了以下硬件：

- 运算放大器：OPA2134
- 有效值检波电路：LF353
- 数字电位器：AD5293
- ADC 采样模块：ADS1115
- MCU：STM32F103C8T6
- 显示屏：0.96 寸

这些硬件在性能、可靠性和性价比上均表现优异，适合本项目的需求。

2.2 硬件设计方案

硬件的总体设计方案如图3所示。输入信号经过可调两级反向放大电路输出，输入信号与输出信号经过有效值检波电路，转换为直流信号，通过 ADC 采样模块采集，传输至 STM32 单片机进行处理，将结果显示在 OLED 显示屏上。STM32 单片机通过数字电位器控制两级反向放大电路的放大倍数，实现对输入信号的自动增益控制。

总体的硬件仿真图见附录 1。

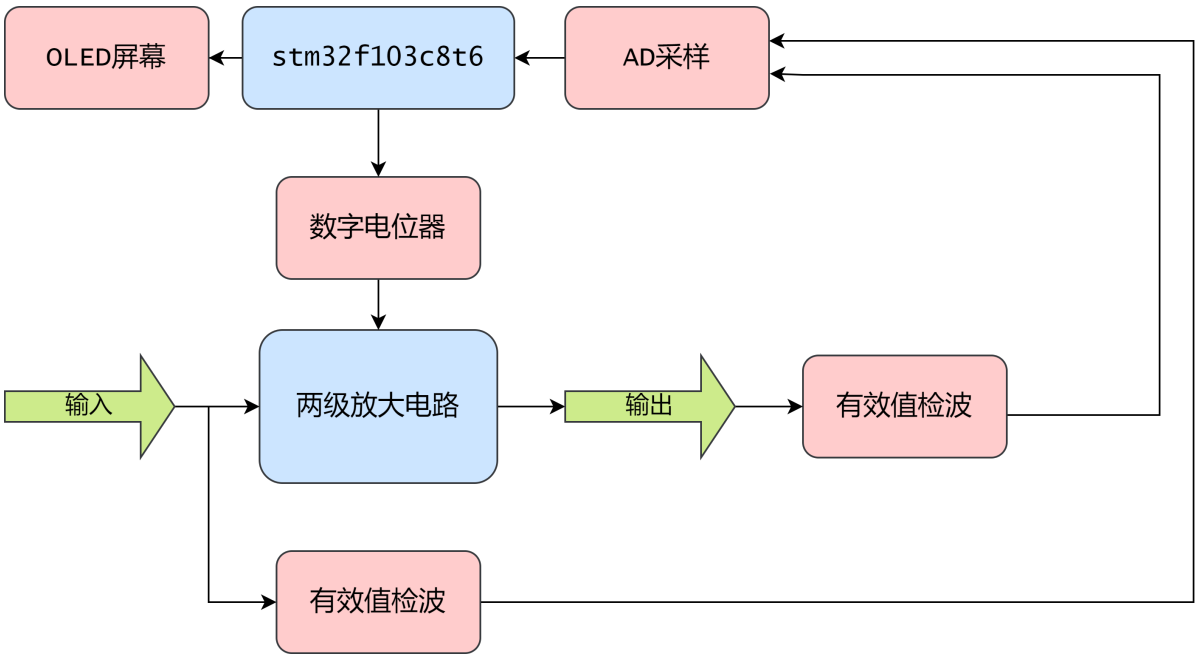


图 3: 硬件总体框图

2.3 可调两级反向放大电路

该电路由三个运算放大器和数字电位器构成。一级运算放大器作为电压跟随器，来减少有效值检波与两级反向放大电路的干扰，并提高输入阻抗。后两级运算放大器（OPA2134）构成了反向放大电路，二级运算放大器的反馈电阻由数字电位器（AD5293）提供，通过调节数字电位器实现对该两级方向放大电路的放大倍数的调节，从而实现对输入信号的放大倍数的调节。如图 4 所示，该电路原理如下：

一级运算放大器的放大倍数为 $-\frac{R_{17}}{R_{\text{数字电位器}}}$，二级运算放大器的放大倍数为 $-\frac{R_{18}}{R_{19}}$。因此，整个电路的放大倍数为：

$$\left(-\frac{R_{18}}{R_{19}}\right)\times\left(-\frac{R_{17}}{R_{\text{数字电位器}}}\right)=\frac{R_{18}}{R_{19}}\times\frac{R_{17}}{R_{\text{数字电位器}}}$$

通过调节数字电位器的电阻值，可以调节电路的放大倍数。

由于市面上没有该型号的数字电位器模块，于是我们自行设计了数字电位器模块，原理图及 PCB 设计见附录 2。

2.4 有效值检波电路

有效值检波电路的作用是将交流信号转换成直流信号，以解决题目中提高要求的 (2) 中的交流信号检测问题。

如图5所示，该电路由三级运算放大器（LF353）和 RC 二极管检波网络构成，一级运算放大器作为电压跟随器，用于防止有效值检波电路与可调两级放大电路的相互干扰。

二级和三级运算放大器及 RC 二极管检波网络构成整流电路，完成有效值检波的功能。该电路原理如下：

(1) 当输入信号 $U_i > 0$ 时，二极管 $D5$ 截止，$D6$ 导通，此时一级运算放大器的增益为 $-\frac{R_{2}2}{R_{2}1}=$

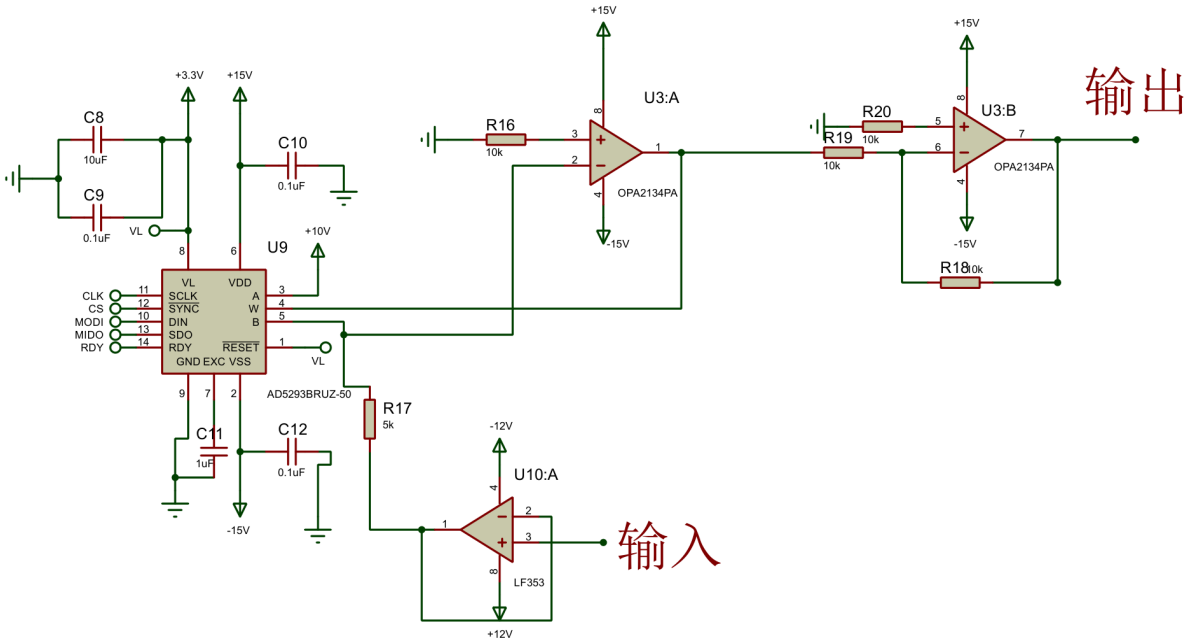


图 4: 可调两级放大电路 - proteus 仿真

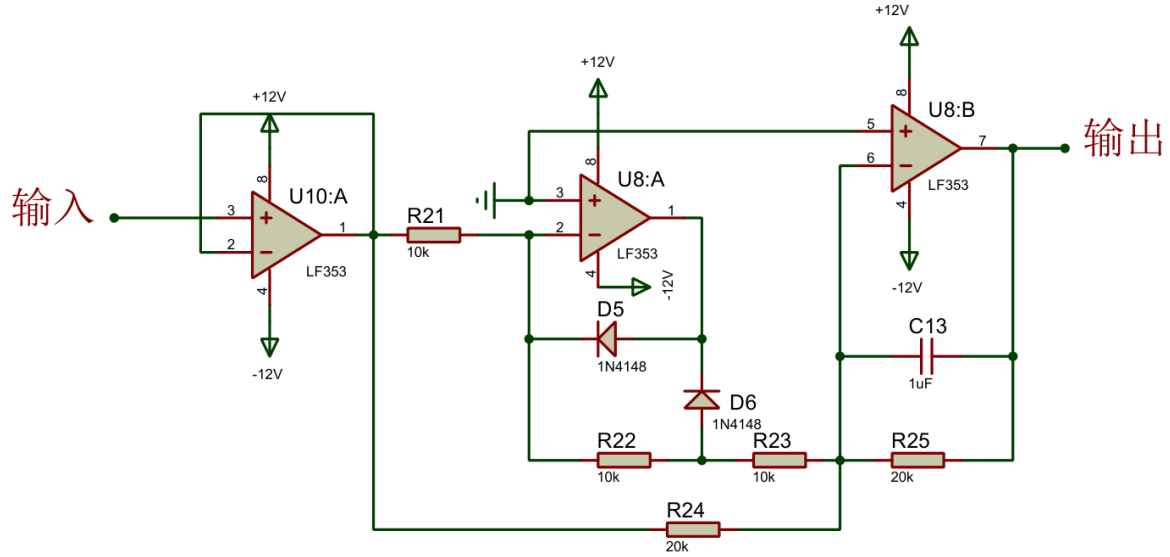


图 5: 有效值检波电路 - proteus 仿真

-1，所以 $U_{o1} = -U_i$。此时电路的输出信号 U_o 为：

$$U_o = -\frac{R_25}{R_24}U_i + \frac{R_25}{R_23}U_{o1} = U_i$$

(2) 当输入信号 $U_i < 0$ 时，二极管 $D5$ 导通，$D6$ 截止，此时一级运算放大器的增益为 0，所以 $U_{o1} = 0$。此时电路的输出信号 U_o 为：

$$U_o = -\frac{R_25}{R_24}U_i = -U_i$$

由上所述，该电路可以实现整流功能。其中，电容 $C1$ 起到滤波作用。

2.5 ADC 采样模块

由于 STM32F103C8T6 单片机的 ADC 模块只能采集 0-3.3V 的电压信号，而题目要求的最大电压为 5V，因此我们外接了一个 ADC 进行电压信号的采集。如图6 为 ADC 采样模块的展示。

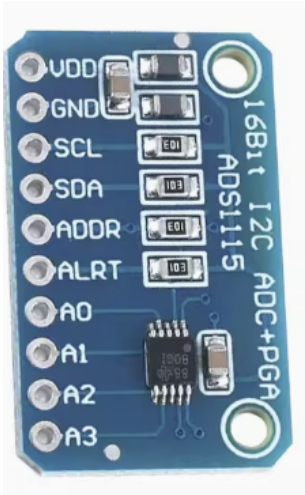


图 6: ADC 采样模块

所使用的 ADC 为 ADS1115，其是德州仪器推出的具有 IIC 接口的 16 位 ADC 转换器，超小型 X2QFN 或 VSSOP 封装，低功耗（20uA），宽电压输入 2.0V-5.5V，可编程数据转换速率 8SPS-860SPS，四个单端输入或两个差分输入。ADS1115 具有内部低漂移基准源，支持单端或差分输入模式，可通过 I2C 接口进行编程。

3 软件设计

3.1 AD 采样

3.1.1 ADS1115 简介

ADS1115 是德州仪器推出的具有 IIC 接口的 16 位 ADC 转换器，超小型 X2QFN 或 VSSOP 封装，低功耗（20uA），宽电压输入 2.0V-5.5V，可编程数据转换速率 8SPS-860SPS，四个单端输入或两个差分输入。可应用于，电池电压电流检测，低速便携式仪表以及温度测量系统中。如图7所示，为 ADS1115 芯片结构。

如表1所示，为 ADS1115 引脚定义。

3.1.2 通信方式

ADS1115 采用的是 IIC 通信。

IIC 地址的选择 ADS1115 具有一个地址引脚 ADDR，用于设置器件的 I2C 地址。该引脚可以是连接到 GND，VDD，SDA 或 SCL，因此可以通过一对 IIC 引脚选择四个不同的地址。

一般我们将地址位接 GND，1001000, 最后一位是确定 IIC 的写/读状态, 写的时候是 1，读的时候是 0. 所以 slave address 读写地址是 0x90/0x91(10010000/10010001)，如表2所示。

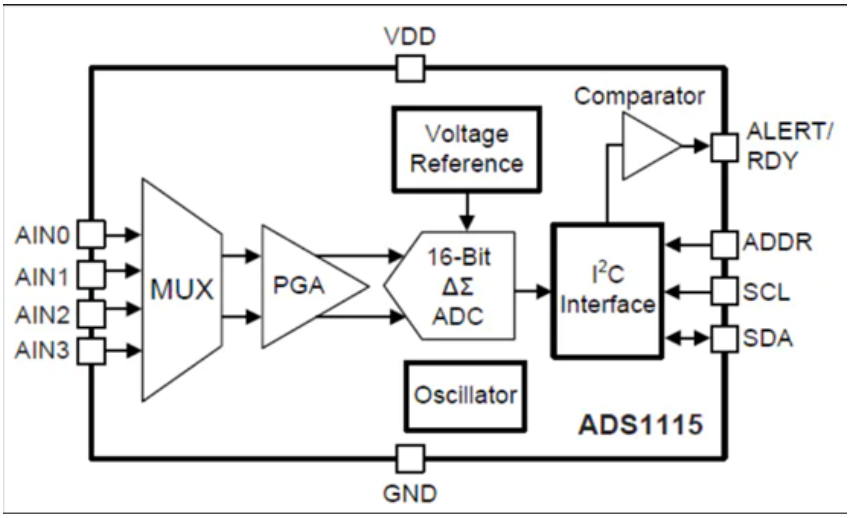


图 7: ADS1115 芯片结构

表 1: ADS1115 引脚定义

引脚名称	类型	描述
ADDR	数字量输入	I2C 从机地址选择
AIN0	模拟量输入	模拟量输入 0 通道
AIN1	模拟量输入	模拟量输入 1 通道
AIN2	模拟量输入	模拟量输入 2 通道
AIN3	模拟量输入	模拟量输入 3 通道
ALERT/RDY	数字量输出	比较器输出或转换就绪
GND	模拟量	接地
SCL	数字量输入	IIC 时钟
SDA	数字量输入/输出	IIC 数据线
VDD	模拟量	VCC（2.0V-5.5V)

表 2: ADS1115 的 IIC 地址

ADDR PIN	SLAVE ADDRESS
Ground	1001000
VDD	1001001
SDA	1001010
SCL	1001011

接收模式 ADS1115 在从机接收模式下，主机发送到从机的第一个字节由 7 位设备地址组成, 其次是低的 R / W 位。主机发送的下一个字节是，ADS1115 收到地址指针寄存器字节，接下来的两个字节被写入地址由寄存器地址指针位 P [1：0] 给出。最后 ADS1115 返回字节。数据寄存器字节为首先发送最高有效字节，然后发送最低有效字节。

发送模式 在从机发送模式下，主机发送的第一个字节是 7 位从机地址，后跟高 R / W 位。该字节将从机设置为发送模式，从机发送的字节是数据寄存器的最高有效字节，由数据寄存器地址指针位 P [1：0] 指示，然后，其余的最低有效字节由从机发送。

3.1.3 ADS1115 读写时序流程

如图8所示，为 ADS1115 读写时序流程。

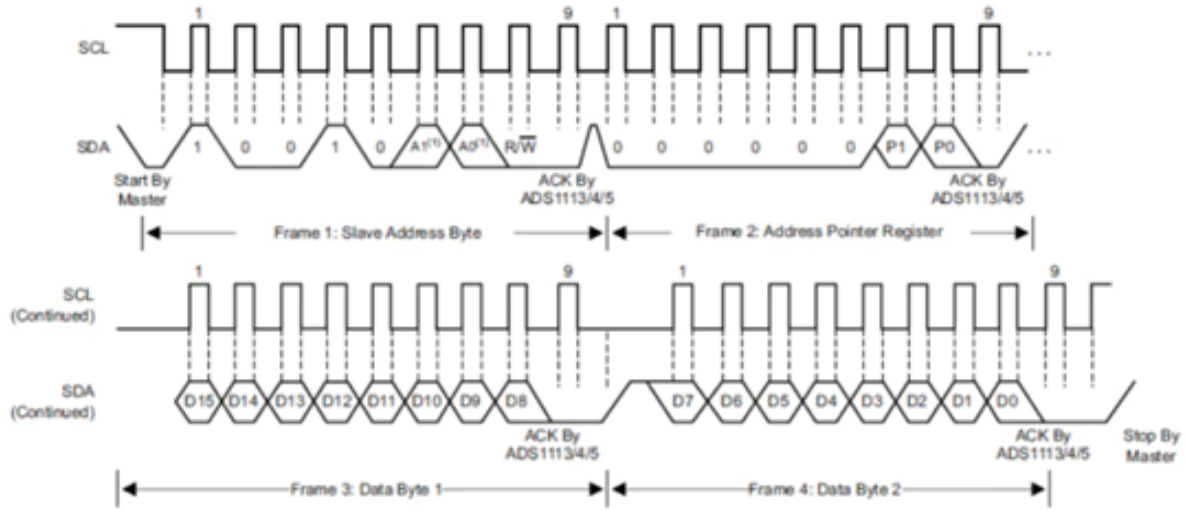


图 8: ADS1115 读写时序流

读时序操作步骤

1. 发送写地址给 ADS1115 (0x90)；
2. 向地址指针寄存器写数据，后两位有效，只能写 0x00, 0x01, 0x02, 0x03；
3. 发送读地址给 ADS1115 (0x91)；
4. 读取 ADS1115 的数据（两个字节，MSB 先行）。

写时序操作步骤

1. 发送写地址给 ADS1115 (0x90)；
2. 向地址指针寄存器写数据，后两位有效，只能写 0x00, 0x01, 0x02, 0x03；
3. 发送数据给 ADS1115（高位在前）。

3.1.4 ADS1115 输出数据格式

ADS1115 以二进制补码格式提供 16 位数据。正满量程（+ FS）输入时，输出的 AD 值代码为 7FFFh，负满量程（-FS）输入时，输出的 AD 值代码为 8000h。这些代码的输出为了提示超量程的提示。

3.1.5 程序流程图

如图9所示，为 ADS1115 程序流程图。首先初始化 IIC，然后读取范围，之后读取 ADS1115 的数据，最后将数据转换为电压值的小数显示。

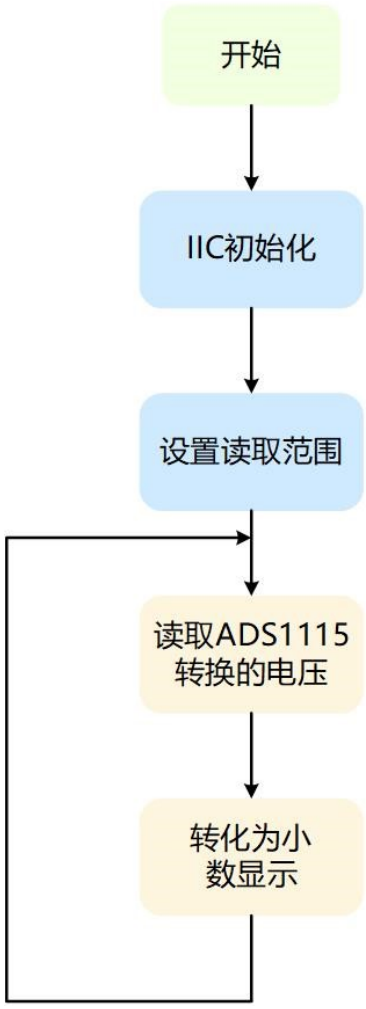


图 9: ADS1115 程序流程图

3.2 数字电位器

3.2.1 AD5293 简介

AD5293 是一款单通道、1024 位数字电位计 1 ，端到端电阻容差 <1%。该器件可实现与机械电位计相同的电子调整功能，而且具有增强的分辨率、固态可靠性和出色的低温度系数性能。它可在高电压下工作，既可采用 ±10.5 V 至 ±15 V 双电源供电，也可采用 21 V 至 33 V 单电源供电。MCU 通过 SPI 方式与其通信。如图10所示，为 AD5293 芯片结构。

3.2.2 程序流程图

3.2.3 AD5293 通信

AD5293 采用的是标准的 SPI 通信。

3.2.4 SPI 通信简介

SPI 接口主要应用在 EEPROM、FLASH、实时时钟、AD 转换器，还有数字信号处理器和数字信号解码器之间。SPI 是一种高速的，全双工，同步的通信总线，并且在芯片的管脚上只占用

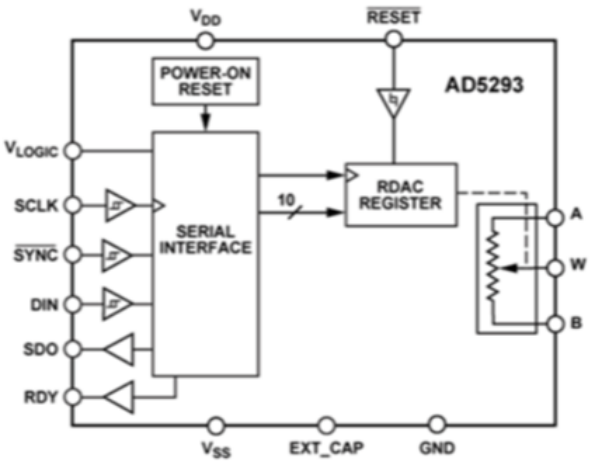


图 10: AD5293 芯片结构

四根线，节约了芯片的管脚，同时为 PCB 的布局上节省空间，提供方便，正是出于这种简单易用的特性，现在越来越多的芯片集成了这种通信协议，比如 AT91RM9200。

SPI 分为主、从两种模式，一个 SPI 通讯系统需要包含一个（且只能是一个）主设备，一个或多个从设备。SPI 接口的读写操作，都是由主设备发起。当存在多个从设备时，通过各自的片选信号进行管理。

3.2.5 引脚说明

SPI 的通信原理很简单，它以主从方式工作，这种模式通常有一个主设备和一个或多个从设备，需要至少 4 根线，事实上 3 根也可以（单向传输时）。这四根线分别是 MISO、MOSI、SCLK、CS，具体的描述见下表3：

表 3: SPI 引脚说明

名称	描述
MISO	主设备数据输出，从设备数据输入
MOSI	主设备数据输出，从设备数据输入
SCLK	时钟信号，主设备产生
CS	片选信号，主设备控制

3.2.6 程序流程图

如图11所示，为 AD5293 程序流程图。首先初始化 AD5293，之后通过 SPI 通信方式与 AD5293 通信，进行片选和读写操作。

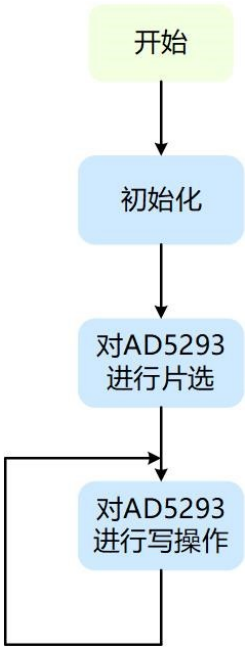


图 11: AD5293 程序流程图

3.3 主程序

3.3.1 主程序介绍

如图12主程序用来初始化各种外设，以及对整个硬件电路的逻辑编写。初始化 ADS1115，OLED 以及 AD5293 等外设之后，进入到按键检测，根据不同的按键按下来切换到不同的模式从而达到所需的要求。

3.3.2 主程序流程图

其中自动控制模式下的输出电压范围控制采用的是和 ADC 转换原理类似的逐次逼近比较，从而让他限定在所需的输出范围内。部分代码呈现如下：

```
void just_vol(float vol_ref,float vol,float output_singal)
{
    if((output_singal >vol_ref)&&((output_singal-vol_ref)>vol))
    {
        AD5293_Write(write,i--);
        if(i<=0x0001)
        i=0x0002:
      }
    else if((output_singal <vol_ref)&&((vol_ref output_singal)>vol))
    {
        AD5293_Write(write,i ++)
        if(i>=0x3FFF){
            i=0x3FFE:
         }
```

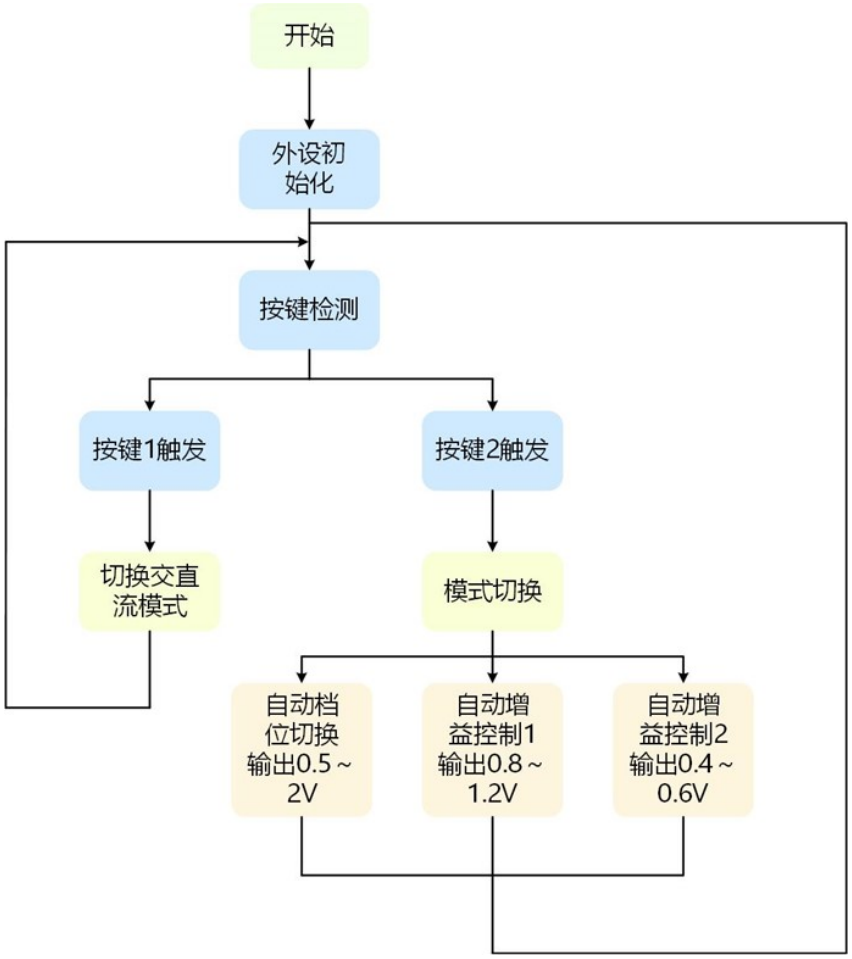


图 12: 主程序流程图

```
        }
    }
```

Listing 1: 主程序部分代码

其中 vol_ref 代表为基准电压，vol 代表为允许的最大误差，output_singal 为输出的电压。

4 系统测试

4.1 测试概述

1. 测试目的：评估该自动增益控制系统的质量、可靠性、安全性和性能。
2. 测试范围：包括所有的功能、性能和安全性测试。
3. 测试工具：

   使用以下工具进行测试，

   (a) **可调直流源和信号发生器：**用于提供测试的输入信号。

   (b) **实验箱和自制电源电路：**为系统中的芯片供电。

   (c) **电压表：**用于检测和调试电路中的信号。

4.2 基本要求 (1)

系统设置有 0.2、0.5、2、5 四挡增益，通过按键 1 可以对档位进行切换。本次进行三次测试，规定输入直流信号的大小为 0.1V、1V、2V，分别测试系统在四挡增益下的输出电压。

当输入直流信号为 0.1V 时，系统测试如表4所示。

表 4: 基本要求 (1) 测试结果 (输入为 0.1V 时)

按键按下次序	电压表测量输入信号 (V)	电压表测量输出信号 (V)	理论增益	实际增益	测试是否通过
0	0.10	0.02	0.2	0.2	通过
1	0.10	0.05	0.5	0.5	通过
2	0.11	0.21	2	2	通过
4	0.10	0.49	5	5	通过
5	0.10	0.02	0.2	0.2	通过
6	0.10	0.05	0.5	0.5	通过
7	0.10	0.20	2	2	通过
8	0.10	0.49	5	4.9	通过
9	0.10	0.02	0.2	0.2	通过
10	0.10	0.05	0.5	0.5	通过
11	0.10	0.21	2	2.1	通过
12	0.10	0.49	5	4.9	通过

当输入直流信号为 1V 时，系统测试如表5所示。

表 5: 基本要求 (1) 测试结果 (输入为 1V 时)

按键按下次序	电压表测量输入信号 (V)	电压表测量输出信号 (V)	理论增益	实际增益	测试是否通过
0	1.01	0.20	0.2	0.2	通过
1	1.01	0.51	0.5	0.5	通过
2	1.02	2.03	2	2	通过
4	1.01	5.00	5	5	通过
5	1.01	0.20	0.2	0.2	通过
6	1.01	0.51	0.5	0.5	通过
7	1.02	2.03	2	2	通过
8	1.00	5.02	5	5	通过
9	1.01	0.21	0.2	0.2	通过
10	1.00	0.51	0.5	0.5	通过
11	1.02	2.05	2	2	通过
12	1.01	5.01	5	5	通过

当输入直流信号为 2V 时，系统测试如表6所示。

输出信号与实际增益在误差允许的范围内，均符合理论值，测试通过。

表 6: 基本要求 (1) 测试结果 (输入为 2V 时)

按键按下次序	电压表测量输入信号 (V)	电压表测量输出信号 (V)	理论增益	实际增益	测试是否通过
0	2.00	0.42	0.2	0.2	通过
1	2.01	1.03	0.5	0.5	通过
2	2.03	4.02	2	2	通过
4	2.03	9.98	5	5	通过
5	2.01	0.41	0.2	0.2	通过
6	2.00	1.01	0.5	0.5	通过
7	2.01	4.01	2	2	通过
8	2.01	10.00	5	5	通过
9	2.01	0.41	0.2	0.2	通过
10	2.01	1.00	0.5	0.5	通过
11	2.01	4.01	2	2	通过
12	2.02	10.03	5	5	通过

4.3 基本要求 (2)

使用 0.96 寸的 OLED 显示屏，显示输出和输出信号的幅度当前放大器的增益。显示的内容为：**ADC0：X.XXXX，ADC1：X.XXXX，amp：X.XXXX**。其中 ADC0 为输入信号的幅度，ADC1 为输出信号的幅度，amp 为当前放大器的增益。显示的数值大小均为实际测量值，保留四位小数。

本次进行三次测试，规定输入直流信号的大小为 0.1V、1V、2V，观察 OLED 显示屏上的显示结果是否与电压表测量值一致，及增益是否与电路设计值一致。测试结果如表7所示。显示屏

表 7: 基本要求 (2) 测试结果

序号 / 名称	**1**	**2**	**3**
输入信号幅度 (V)	0.1	1	2
电压表测量输入信号幅度 (V)	0.12	1.01	2.03
电压表测量输出信号幅度 (V)	0.25	5.08	1.01
OLED 显示输入信号幅度 (V)	0.1217	1.0125	2.0315
OLED 显示输入信号幅度 (V)	0.2514	5.0832	1.0122
OLED 显示增益	2.0657	5.0204	0.4982
理论增益	2	5	0.5
显示是否正常	正常	正常	正常

在测试中均正常显示，显示值与测量值在误差允许的范围内，符合理论值，显示结果正确，测试通过。

4.4 基本要求 (3) 与提高要求 (1)

考虑到基本要求 (3) 与提高要求 (1) 的测试内容相似，测试指标重合，因此将两者合并进行测试。本次进行六次测试，规定输入直流信号的大小为 0.1V、1V、2V、3V、4V、5V。测试结

果如表8所示。测试结果显示，输出信号、实际增益和显示屏显示内容在误差允许的范围内，均

表 8: 基本要求 (2) 与提高要求 (1) 的测试结果

输入信号 (V)	0.10	1.2	2.4	3.3	4.5	5
ADC0(OLED)	0.1035	1.2065	2.4079	3.3721	4.5890	5.0690
ADC1(OLED)	1.1653	1.1431	1.1733	1.1711	0.9832	0.9911
Amp(OLED)	11.2589	0.9803	0.4872	0.3473	0.2142	0.1955
测量输入信号 (V)	0.10	1.20	2.40	3.37	4.58	5.06
测量输出信号 (V)	1.16	1.14	1.17	1.17	0.98	0.99
测量增益	11.25	0.98	0.48	0.34	0.21	0.19
0.52V(是否)	是	是	是	是	是	是
0.82V(是否)	是	是	是	是	是	是

符合理论值，测试通过。

值得注意的是，我们发现显示屏上显示的输入信号与信号源输入的信号有一直存在较为明显的误差，为 0.0XV，起初我们认为是 ADC 采样模块的问题，但经过多次测试，我们发现 ADC 采样模块的采样值是准确的，因此我们怀疑是 OLED 显示屏的问题。经过查阅资料，我们发现 OLED 显示屏的显示值是由其内部的 ADC 采样模块采样得到的，因此我们认为 OLED 显示屏的显示值是准确的。之后我们怀疑是电压表的问题，于是我们对电压表进行了校准，误差依然存在。我们又尝试直接对信号源输入的信号进行了测量，发现信号源本身的输出电源不准确。

4.5 提高要求 (2)

本次进行了六次测试，规定输入信号的幅度为 0.2V、2.4V、2V、4.8V、6.6V、9V、10V，输入信号的频率对应 3kHz、10kHz、30kHz、50kHz、70kHz、100kHz。测试结果如表9所示。

表 9: 提高要求 (2) 的测试结果

输入信号幅度 (Vpp)(V)	0.20	2.4	4.8	6.6	9	10
输入信号频率 (kHz)	3	10	30	50	70	100
ADC0(OLED)	0.1045	1.2030	2.4088	3.3755	4.5855	5.0678
ADC1(OLED)	1.1615	1.1455	1.1778	1.1735	0.9857	0.9954
Amp(OLED)	11.1148	0.9522	0.4889	0.3476	0.2149	0.1964
测量输入信号 (V)	0.10	1.20	2.40	3.37	4.58	5.06
测量输出信号 (V)	1.16	1.14	1.17	1.17	0.98	0.99
测量增益	11.60	0.95	0.48	0.34	0.21	0.19
0.81.2(Vpp)(是否)	是	是	是	是	是	是

由于显示屏和电压表显示的是 Vp 值，所以其值是 Vpp 的二分之一。测试结果显示，输出信号、实际增益和显示屏显示内容在误差允许的范围内，均符合理论值，测试通过。

值得注意的是，本电路在 100kHz 的高频下，仍然能保持较好的自动增益控制效果，未出现因频率过高导致的衰减现象。与此相比，网上很多 AGC 电路设计都会出现频率过高导致的衰减现象，本设计的电路在高频下仍然能保持较好的自动增益控制效果，具有较好的性能。

5 总结

本设计实现了一个自动增益控制系统，能够自动调节输入信号的放大倍数，使输出信号的幅度保持在合适的范围内。系统硬件设计合理，性能稳定，满足了设计要求。经过系统测试，系统的功能、性能和安全性均符合要求，测试结果良好，验证了系统的可靠性和稳定性。

在测试过程中，我们发现了一些问题。

输入信号的幅度与信号源的输出信号存在较大误差，经过多次测试，我们发现是信号源本身的问题。在今后的设计中，我们需要更加注意信号源的准确性，以保证测试结果的准确性。

在初次测试中我们发现电路输出异常，经过对焊接和电路设计的排查后，我们发现是电路没有“共地”，整体的电路的所有的 GND 与电源的 GND 要联通，否则会导致电路不工作或者输入出现问题，对电路影响很大。

References

[1] Juan Pablo Alegre Pérez, Santiago Celma Pueyo, Belén Calvo López. *Automatic gain control*. Springer, 2011.

[2] J. Ohlson. *Exact dynamics of automatic gain control*. *IEEE Transactions on Communications*, 22(1):72–75, 1974.

[3] Isaac Martinez. *Automatic gain control (AGC) circuits theory and design*. Term paper, University of Toronto, 2001.

[4] 卢霄. 一种延迟式自动增益控制电路的设计及仿真研究. 电子制作, 18:83–85+78, 2023.

[5] 卢霄. 声系统设备第 8 部分: 自动增益控制装置. 中华人民共和国电子工业部, SJ/Z 9140.3-1987, 1987.

[6] *stm32f103*——外部中断和事件——检测按键按下点灯. https://blog.csdn.net/qq_39577221/article/details/125304686.

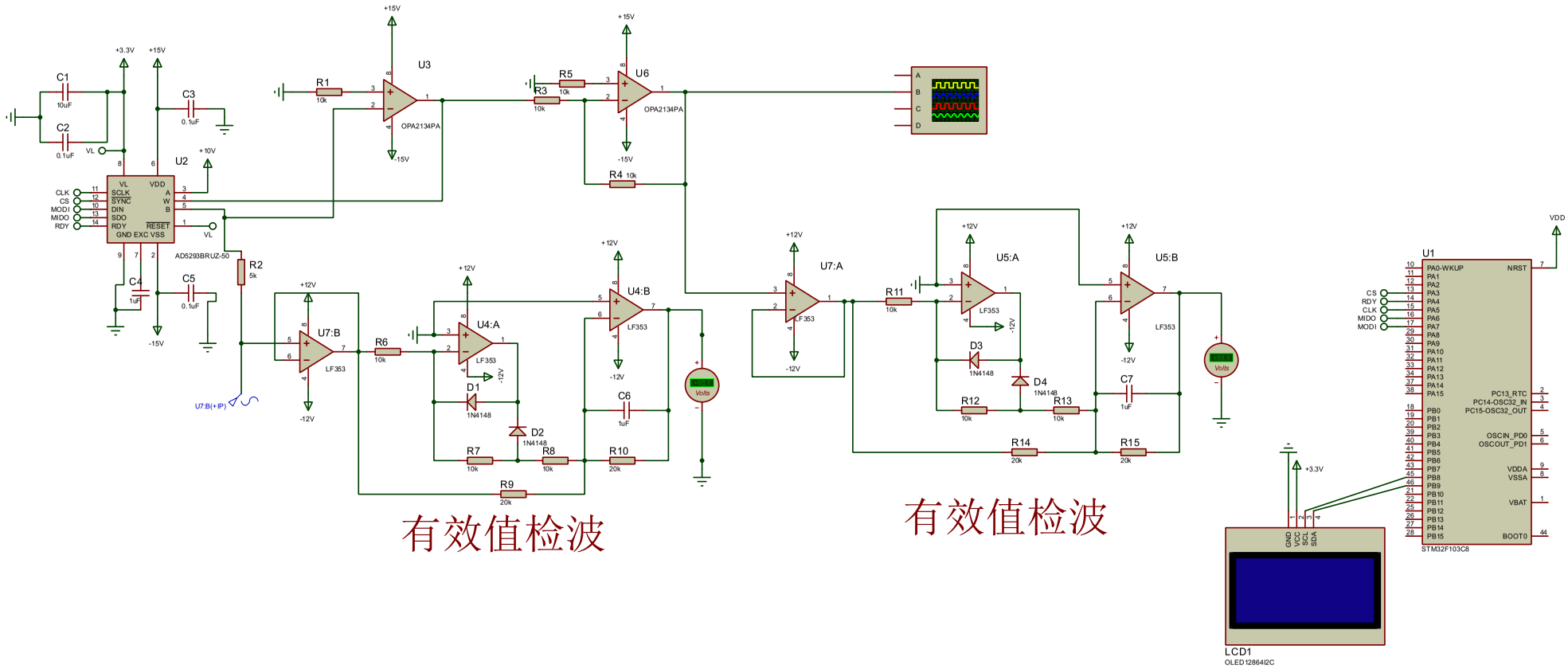
[7] 运放的参数详解及应用电路. https://blog.csdn.net/qq_21794157/article/details/124024270.

[8] 可变增益放大器电路 *VCA* 电路，自动增益控制电路 *AGC* 电路. https://www.bilibili.com/video/BV1JD4y1b7Gzvd_source=565626543e6250b5a2589e84e31771f6.

[9] 增益自动切换电压放大电路设计. https://blog.csdn.net/weixin_45012516/article/details/109273168.

[10] 基于 *VCA821* 的 *AGC*，电压自动增益控制设计. https://zhuanlan.zhihu.com/p/644879914.

A 附录 1 - 硬件总体仿真电路图 (proteus 仿真)


+3.3V
+15V
C1
10uF
C2
0.1uF
C3
0.1uF
VL
U2
AD5293BRUZ-50
CLK
CS
MOSI
MISO
RDY
SCLK
SYNC
DIN
SDO
RDY
RESET
VL
VDD
A
W
B
GND EXC VSS
+10V
C4
1uF
C5
0.1uF
R2
5k
U7:B(+IP)
U3
R1
10k
OPA2134PA
-15V
U6
R5
10k
R3
10k
R4
10k
+12V
-12V
U7:B
LF353
R6
10k
U4:A
D1
1N4148
D2
1N4148
R7
10k
R8
10k
R9
20k
U4:B
C6
1uF
R10
20k
U7:A
R11
10k
U5:A
D3
1N4148
D4
1N4148
R12
10k
R13
10k
R14
20k
U5:B
C7
1uF
R15
20k
Volts
有效值检波
有效值检波
U1
STM32F103C8
PA0-WKUP
NRST
PC13_RTC
PC14-OSC32_IN
PC15-OSC32_OUT
OSCIN_PD0
OSCOUT_PD1
VDDA
VSSA
VBAT
BOOT0
LCD1
OLED12864I2C
GND
VCC
SCL
SDA

B 附录 2 - 数字电位器的原理图及 PCB

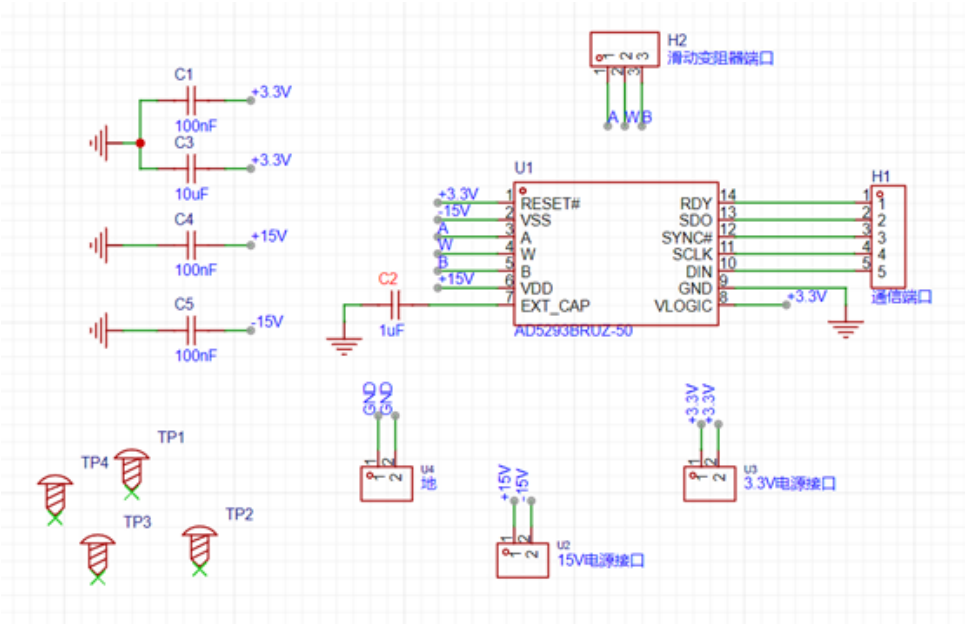


图 13: 数字电位器原理图

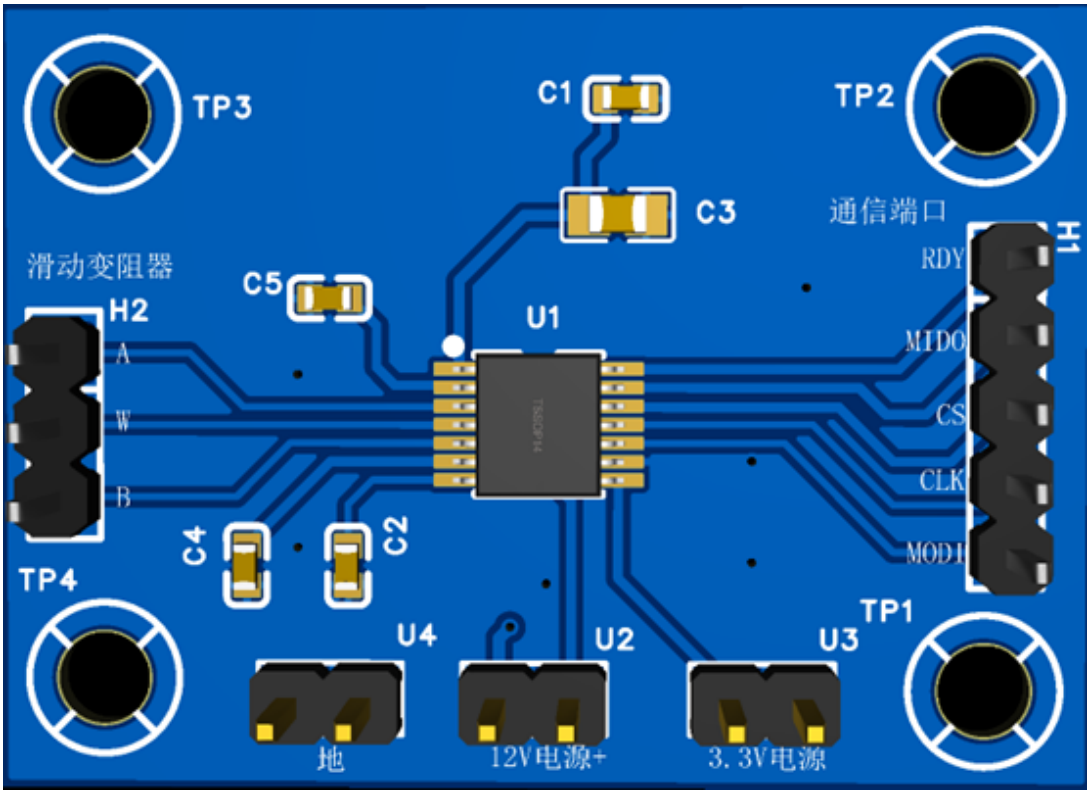


图 14: 数字电位器 PCB 正面

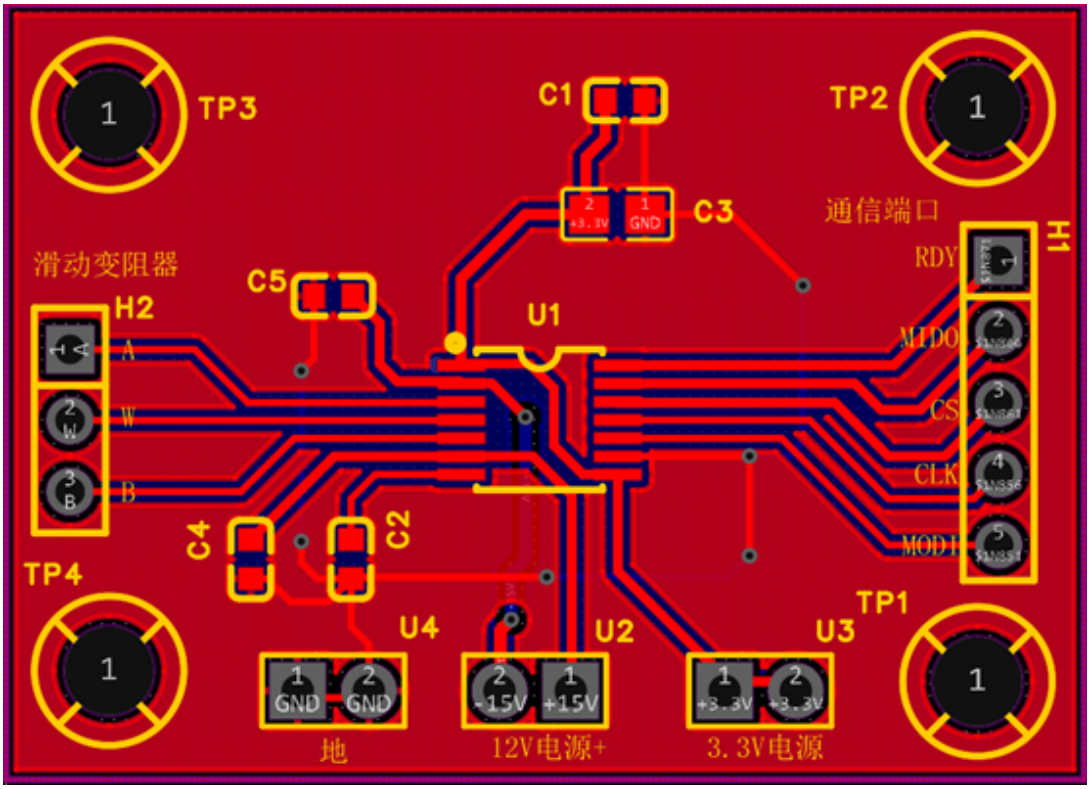


图 15: 数字电位器 PCB 背面

C 附录 3 - main.c

```c
#include "stm32f10x.h"
#include "MySPI.h"
#include "AD5293.h"
#include "delay.h"
#include "ADS1115.h"
#include "bsp_usart.h"
#include "oled.h"
#include "MyExti.h"
#include "Key.h"

int16_t ADS1115_ADC0, ADS1115_ADC1;
uint8_t ADC0_Buf[] = "ADC0:       V";
uint8_t ADC1_Buf[] = "ADC1:       V";
int i = 0x0067;
int flag = 1;
int gear = 3;
int ad_flag = 1;
float ADS1115_ADC01;
float ADS1115_ADC11;

```

```c
void just_vol(float vol_ref, float vol, float output_signal);
void gear_position(int m);
void Dis_Dat(void);
void OLED_printf(unsigned char *s, int temp_data);

void OLED_printf(unsigned char *s, int temp_data)
{
    s += 5;
    *s = (temp_data < 0) ? '-' : ' ';
    temp_data = abs(temp_data);
    *(s + 1) = temp_data / 1000 % 10 + 0x30;
    *(s + 2) = '.';
    *(s + 3) = temp_data / 100 % 10 + 0x30;
    *(s + 4) = temp_data / 10 % 10 + 0x30;
    *(s + 5) = temp_data % 10 + 0x30;
}

void Dis_Dat(void)
{
    if (ad_flag == 0)
    {
        ADS1115_ADC0 = ADS1115_ReadVoltage(0);
        OLED_printf(ADC0_Buf, ADS1115_ADC0);
        Usart_SendString(DEBUG_USARTx, ADC0_Buf);
        OLED_ShowString(0, 0, ADC0_Buf, 16);

        ADS1115_ADC1 = ADS1115_ReadVoltage(1);
        OLED_printf(ADC1_Buf, ADS1115_ADC1);
        Usart_SendString(DEBUG_USARTx, ADC1_Buf);
        OLED_ShowString(0, 2, ADC1_Buf, 16);
    }
    else
    {
        ADS1115_ADC0 = ADS1115_ReadVoltage(0);
        ADS1115_ADC01 = (float)((ADS1115_ADC0 * 3.1415) / 2000.0);
        OLED_printf(ADC0_Buf, ADS1115_ADC0);
        Usart_SendString(DEBUG_USARTx, ADC0_Buf);
        OLED_ShowString(0, 0, "ADC0:", 16);
        OLED_Float(0, 50, ADS1115_ADC01, 4);

        ADS1115_ADC1 = ADS1115_ReadVoltage(1);
        ADS1115_ADC11 = (float)((ADS1115_ADC1 * 3.1415) / 2000.0);
        OLED_printf(ADC1_Buf, ADS1115_ADC1);
```

```
64          Usart_SendString(DEBUG_USARTx, ADC1_Buf);
65          OLED_ShowString(0, 2, "ADC1:", 16);
66          OLED_Float(2, 50, ADS1115_ADC11, 4);
67      }
68  }
69
70  int main(void)
71  {
72      float input_signal, output_signal, input_original, output_original,
            amp;
73      AD5293_Init();
74      AD5293_Unlock();
75      AD5293_Write(write, 0x00200);
76
77      DelayInit();
78      OLED_Init();
79      OLED_Clear();
80      USART_Config();
81      ADS1115_IIC_Init();
82      ADS1115_SetGain(GAIN_TWOTHIRDS);
83      Key_Init();
84
85      while (1)
86      {
87          Dis_Dat();
88          DelayMs(300);
89          input_original = ADS1115_ADC0 / 1000.0;
90          output_original = ADS1115_ADC1 / 1000.0;
91
92          switch (Key_GetNum())
93          {
94              case 0: break;
95              case 1: ad_flag = ~ad_flag; break;
96              case 2:
97                  flag++;
98                  if (flag > 3) flag = 1;
99                  break;
100         }
101
102         output_signal = (ad_flag == 1) ? (output_original * 3.1415) / 2.0
                : output_original;
103         input_signal = (ad_flag == 1) ? (input_original * 3.1415) / 2.0 :
                input_original;
```

```
104
105         if (flag_auto == 0)
106         {
107             switch (flag)
108             {
109                 case 1:
110                     if (output_signal > 2.0) gear_position(--gear);
111                     else if (output_signal < 0.5) gear_position(++gear);
112                     break;
113                 case 2:
114                     just_vol(1.0, 0.2, output_signal);
115                     break;
116                 case 3:
117                     just_vol(0.5, 0.1, output_signal);
118                     break;
119             }
120         }
121         else if (Key_GetNum() == 3)
122         {
123             gear_position(k++);
124         }
125 
126         amp = output_original / input_original;
127         OLED_ShowString(0, 4, "amp:", 16);
128         OLED_Float(4, 30, amp, 4);
129     }
130 }
131 
132 void just_vol(float vol_ref, float vol, float output_signal)
133 {
134     if ((output_signal > vol_ref) && ((output_signal - vol_ref) > vol))
135     {
136         AD5293_Write(write, i--);
137         if (i <= 0x0001) i = 0x0002;
138     }
139     else if ((output_signal < vol_ref) && ((vol_ref - output_signal) >
        vol))
140     {
141         AD5293_Write(write, i++);
142         if (i >= 0x3FFF) i = 0x3FFE;
143     }
144 }
145 
```

```
146 void gear_position(int m)
147 {
148     switch (m)
149     {
150         case 0: AD5293_Write(write, 0x0014); break;
151         case 1: AD5293_Write(write, 0x0033); break;
152         case 2: AD5293_Write(write, 0x00CD); break;
153         case 3: AD5293_Write(write, 0x0200); break;
154         default: AD5293_Write(write, 0x00CD); break;
155     }
156 }
157 \end{lstlisting}
```

Listing 2: main.c