模拟电路实验 diy 项目结题报告——双路语音同传系统的制作

By 李梓源 李沅朔 徐焓茁 潘耿炜

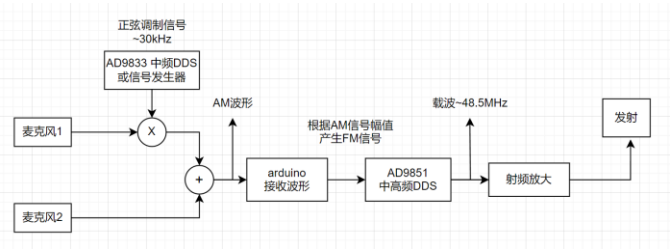
1.项目背景

“双路语音同传”系统，顾名思义，是一种能够同时传输两个语音信号的系统。这种系统的设计背景主要体现在国际会议、双语教学和跨国商业合作中。同时传输两路语音信号的设计能够同时给不同语言的参与者发送不同语言的讲话内容，从而实现无障碍沟通。 这一个系统的核心技术包括 FM 调制，频分复用技术，无线收发技术等。通过这个项目，我们可以将课堂上学到的电子理论知识，如电路分析、信号处理、单片机编程等应用于实际设计中，加深理解，并掌握多种电子技术在真实世界中的应用方法。总体的设计与组装过程涵盖了电路设计、PCB 布局布线、元器件选择与采购、焊接与装配、调试与测试等多个环节，有助于我们掌握电子工程领域必备的工程技术与技能。这个系统的搭建还需要团队协作，能够锻炼了我们的的沟通与合作能力，让我们学习到如何在团队中分配任务、协调进度，以及有效沟通设计理念和解决方案。本项目不仅是对技术能力的全面考验，更是对我们成为具备实战能力、创新思维和良好职业素养的电子工程师的良好锻炼。

2.项目组成（基础部分）

2.1 发射端

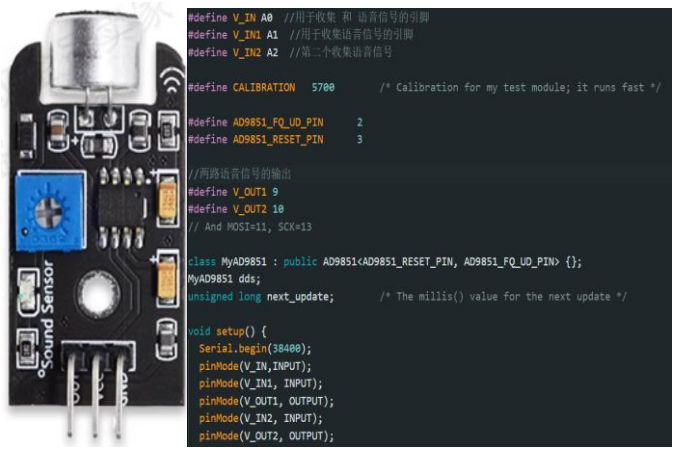
2.1.1 电路拓扑



在这个系统中，两个麦克风分别接受语音信号，并将振动信号转换为模拟的电信号。一路信号在之后通过乘法器与 AD9833 发出的中低频载波信号（约为 30kHz）相乘，进行 AM 调制，将这一路信号（下称 A 路信号）调制到相对较高的频段，与另一路信号（B 路信号）形成基于 AM 调制的频分复用。A 路信号之后与 B 路信号通过加法器相加。之后，arduino 会接受这一个 AM 波形，读取其幅值。之后，根据幅值的波动，arduino 会动态调整 AD9851 产生信号的频率，使其在载波附近根据信号幅值波动，完成 FM 调制。之后，FM 调制的信号通过射频放大模块和天线发射到无线信道中。

2.1.2 硬件原理

1. 驻极体麦克风



驻极体麦克风模块是一个将语音信号转换为模拟电信号的语音接收模块。利用 Arduino UNO 可直接获得输入语音信号。在本项目中，由于为双路语音同传，因此使用两个驻极体麦克模块。在左侧的调试代码中，定义接受引脚为模拟输入 A0 和 A1。

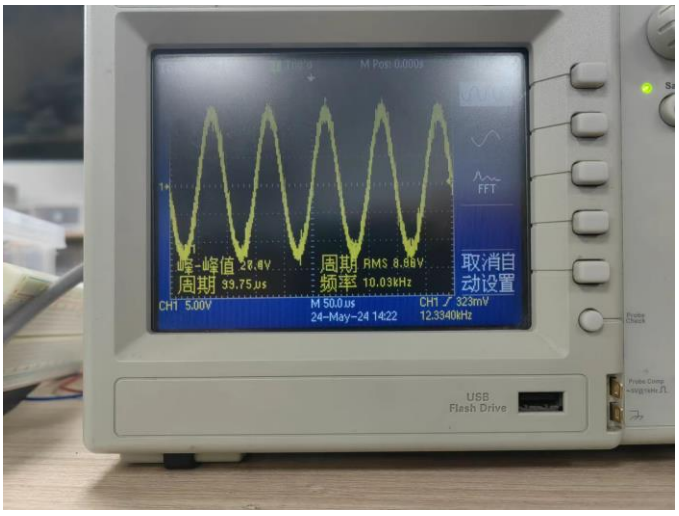
2. DDS（信号发生器）

• 本实验中使用到的 AD9833 和 AD9851 是 ADI 公司生产的两款常用的直接数字频率合成器（DDS, Direct Digital Synthesizer）。它们都可以用于生成精确的正弦波信号，完成我们系统中两个调制环节。AD9833 的输出范围在 0MHz 到 12.5MHz，而 AD9851 的输出范围为 0MHz 到 55MHz。考虑到我们使用的 AM 载波信号频率较低，且 FM 信号需要使用 48.5MHz 的载波，这两个信号发生器都可以达成相应的功能。实验中，我们使用 arduino 来同时控制这些 DDS，可以取得较好的效果。

2.1 AD9833 模块

```
void setup(void)
{
  Serial.begin(115200); // 初始化串行通信，波特率为115200
  pinMode(V_IN, INPUT);
  pinMode(V_OUT, OUTPUT);

  AD.begin();
  AD.setMode(MD_AD9833::MODE_SINE);  // 选择正弦波模式
  AD.setFrequency(MD_AD9833::CHAN_0, 20000);  // 定义信号输出频率为10kHz
}

void loop() {
  int value_IN = analogRead(V_IN); // 读取模拟输入信号
  Serial.println(value_IN); // 打印输入信号到串行监视器

  // 将读取到的模拟信号值（0-1023）映射到PWM范围（0-255），并输出到V_OUT
  analogWrite(V_OUT, value_IN / 4);
}
```

在 setup 函数中设置频率为 20kHZ（第一步 AM 调制），波形为正弦波。后在串行监视器测试输出波形。下图为测试所得波形

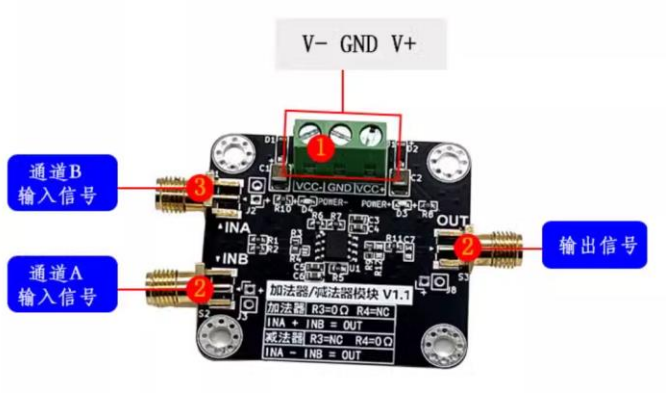


2.2AD9851 模块

```
void loop() {
  // 调制频率
  for (int i = 0; i <= 1000000; i += 1000) {
    setFrequency(48500000 + i); // 在10MHz基础上调频，频偏从0到1MHz
    delay(10); // 每个频率保持10ms
  }
}

void setFrequency(double frequency) {
  // 计算频率字
  uint32_t freqWord = (frequency * pow(2, 32)) / refFrequency;

  // 发送频率字到AD9851
  for (int i = 0; i < 32; i++) {
    digitalWrite(DATA, (freqWord >> i) & 0x01);
    digitalWrite(W_CLK, HIGH);
    delayMicroseconds(1);
    digitalWrite(W_CLK, LOW);
  }

  // 发送控制字
  // 控制字：0000 0000 0000 0000
  for (int i = 0; i < 8; i++) {
    digitalWrite(DATA, LOW);
    digitalWrite(W_CLK, HIGH);
    delayMicroseconds(1);
    digitalWrite(W_CLK, LOW);
  }

  // 更新频率
  digitalWrite(FQ_UD, HIGH);
  delayMicroseconds(1);
  digitalWrite(FQ_UD, LOW);
}
```

AD9851 模块是用于 FM 调制的主要硬件，通过调用 AD9851.h 库实现发射端 FM 调制。具体而言，首先通过加法器将双路语音信号叠加，整体输入 Arduino 中，获取目标信号的频率及幅度。后将整体语音信号映射到（-128,128）的范围并计算频率字。通过 Arduino 将频率字通过 D7 引脚输入 AD9851 模块，并采用定时中断设定时钟信号，使得 AD9851 模块输出 FM 调制后的语音信号。其中基带频率为预设 48.5MHZ，下图为测试所得波形：

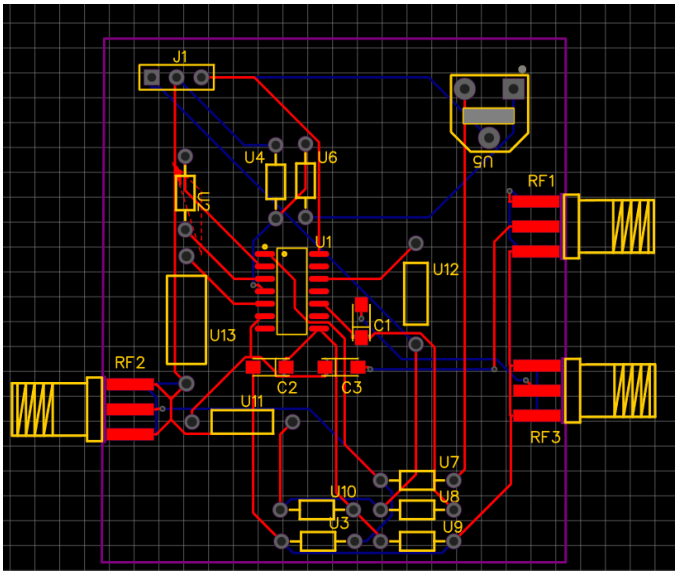
3. 加法器



• 本实验中使用到的加法器是信号同相相加的 1：1 加法器模块，用于将两路语音信号叠加并共同输入 Arduino 进行进一步 FM 调制。

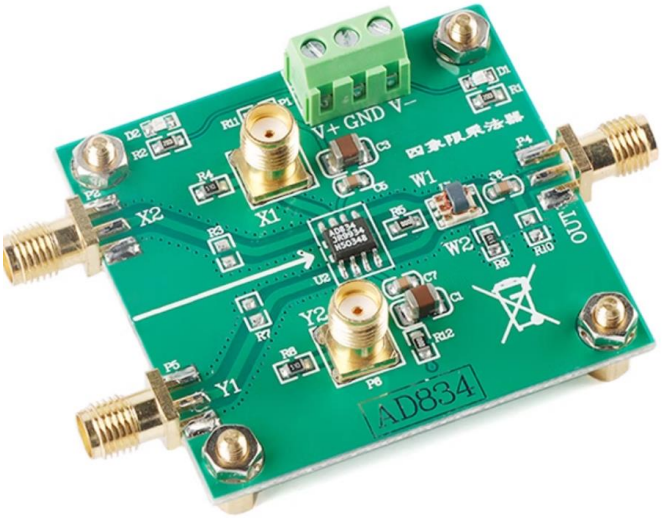
4. 乘法器

• 实验中使用的乘法器进行了两次迭代。第一次，我们设计了基于 MC1496 乘法器的电路。



如图，即一个 MC1496 芯片外加了一些外围电路。但是在打板与测试之后，发现该芯片并不能正常工作，其输出电平极小而且会对传入的正弦波信号进行削波。我们初步预计是因没有铺铜以及没有对信号线做特殊处理，导致该板子会受到很大干扰，无法正常工作。

• 最后我们决定更换 AD834 四象限乘法器模块

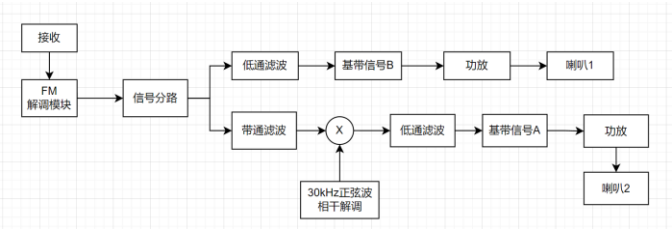


5. 射频放大模块

因为 DDS 直接发出的信号功率较小，难以满足射频发射的需要，我们在发射端的末尾加入了射频放大模块，之后再接入天线。实测可以满足发射需求。

3.接收端

3.1 电路拓扑



通过天线接收 48.5MHz 的 FM 信号，将信号传递给 FM 解调模块，在模块中进行 FM 解调之后得到的信号包括：原信号 B 和经过 AM 调制的信号 A。将此信号使用分路线一分为二，一份连接低通滤波器，进行滤波后得到单一信号 B，此时 B 信号连接功放已经可以从喇叭输出；另一路需要进行 AM 解调，此路信号首先进行带通滤波，得到 A 的 AM 调制信号，再和由信号发生器产生的 30KHz 信号（AM 解调使用的 30kHz）一起输入乘法器进行 AM 相干解调，之后产生信号输出到滤波器进行低通滤波，

得到单一原 A 信号，此信号同样输出功放并连接喇叭输出人声。

6. 总体控制

在分别对 AD9833、AD9851 和驻极体麦克风模块进行测试后，我们将三者的代码进行合并，通过何理规划 Arduino 引脚，最终实现在 setup 函数通过寄存器控制 AD9833 模块，在 loop 函数接受麦克风语音信号并控制 AD9851 模块。

```
#include <AD9851.h>
#include <SPI.h>
#include <MD_AD9833.h>
MD_AD9833 AD(/*DATA*/5, /*CLK*/4, /*FSYNC*/3);
/* Sweep details */
#define CENTRE        48500000UL  /* Hz */ // 载波频率 48.5MHz
#define DEVIATION     25000      /* Deviation in Hz (above and below) */ // 偏移量 25KHz
#define STEP          0      /* Step in Hz. Set to 0 for constant frequency */
#define RATE          100       /* ms between steps */ // 两次更新之间的时间间隔
#define V_IN A0  //用于收集 和 语音信号的引脚
#define V_IN1 A1  //用于收集语音信号的引脚
#define V_IN2 A2  //第二个收集语音信号

#define CALIBRATION   5700        /* Calibration for my test module; it runs fast */

#define AD9851_FQ_UD_PIN      2
#define AD9851_RESET_PIN      3

//两路语音信号的输出
#define V_OUT1 9
#define V_OUT2 10
// And MOSI=11, SCK=13

class MyAD9851 : public AD9851<AD9851_RESET_PIN, AD9851_FQ_UD_PIN> {};
MyAD9851 dds;
unsigned long next_update;        /* The millis() value for the next update */

void setup() {
  Serial.begin(38400);
  pinMode(V_IN,INPUT);
  pinMode(V_IN1, INPUT);
  pinMode(V_OUT1, OUTPUT);
  pinMode(V_IN2, INPUT);
  pinMode(V_OUT2, OUTPUT);
  AD.begin();
  AD.setMode(MD_AD9833::MODE_SINE);  // 选择正弦波模式
  AD.setFrequency(MD_AD9833::CHAN_0, 10000);  // 定义信号输出频率为10kH
  while (!Serial && (millis() <= 1000));
  next_update = millis();
}
void loop() {
  int X_IN1;
  int X_IN2;
  X_IN1 = analogRead(V_IN1);
  X_IN2 = analogRead(V_IN2);
  Serial.print("第一路: ");
  Serial.println(X_IN1);
  Serial.print("第二路: ");
  Serial.println(X_IN2);

  // 将读取到的模拟信号值（0-1023）映射到PWM范围（0-255），并输出到V_OUT
  analogWrite(V_OUT1, X_IN1 / 4);  //输出语音信号
  analogWrite(V_OUT2, X_IN2 / 4);

  static  unsigned long freq = CENTRE;
  float X_IN;    //记录声音信号
  X_IN = analogRead(V_IN);
  Serial.println(X_IN);
  dds.setClock(CALIBRATION);

  // Apply FM modulation
  set_modulated_frequency(X_IN);

  if ((freq += STEP) > CENTRE+DEVIATION)
    freq = CENTRE-DEVIATION;    // Start again at the bottom of the sweep
  next_update += RATE;
  unsigned long wait = next_update-millis();
  if (wait < 1000)            // Wait until RATE ms have elapsed since the last update
    delay(wait);                // Unless the update took longer than RATE
  else
    next_update = millis()+RATE;  // Resynchronise if possible, things exploded
}
```

通过示波器测试，可以看出接收端成功实现语音信号的接受及 FM 调制：

3.2 硬件部分

3.2.1FM 解调模块

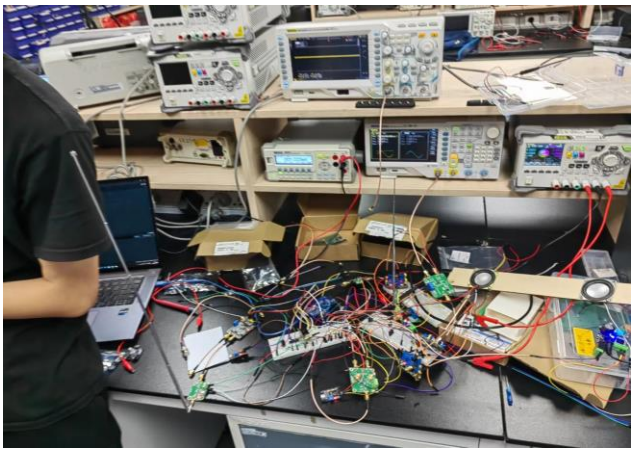
我们使用 NE5654FM 解调模块

为了解调 48.5MHz 的 FM 信号，需要使用较为特殊的解调模块，市面上的解调模块一般解调频率大约为 90MHZ 以上，我们一开始使用 TEA5767 模块进行接调发现无法正常接收便是这个原因，之后我们发现了这个问题并购买了 NE5653FM 解调模块进行 FM 的解调 。

3.2.2UAF42 滤波器

接收模块对滤波的需求较大，总共使用了三个滤波器，我们选择 UAF42 滤波器作为滤波模块，此滤波器可以通过调节跳线帽进行低通，带通，高通滤波，并且此滤波器无法通过 arduino 进行设置，需要人为调制。调试方法为通过改变滤波器上的电阻阻值测试滤波效果，以得到合适的滤波区间，这一步我们使用了信号发生器和滤波器进行，通过改变信号发生器的频率，首先找到原滤波区间，然后不断调节滤波器电阻阻值和信号频率观察示波器，最终达到合适的滤波功能。在这一步中受到了不小的阻力，此滤波器给的参考资料不够完善导致很难调节滤波范围，如：原准备将低通的截止频率设置为 2kHz（人声音频率的范围为 0~2kHz），太难实现，退而求其次只能将 AM 调制的频率升高至 30KHz，并将滤波器的截止频率设置为 30KHz，这样才完成滤波器的设置任务（此滤波器不建议购买使用，较难调且给的教程不完善）。

3.3. 完整的双路语音同传系统展示



3 主要工作内容及收获

3.1 项目设计主要难点

项目的一个难点在于指定的 FM 调制频率不在传统的民用传输频率当中（通常为 76-108MHz），需要我们自行设计调制电路，同时也需要用到特殊的解调电路才可以正常进行信号收发。

在进行解调时，需要对各个滤波器，乘法器等模块进行调试配置，才可以清晰解调。

3.2 pcb 绘制

因为是第一次绘制 PCB，使用了嘉立创的自动布线。实际测试下来，乘法器的异常很可能是因为自动布线缺乏了对信号保护的考虑。

应该在嘉立创专业版中进行绘制，做好铺铜与信号保护。

同时，发现立创商城对穿孔元件的支持较差，了解到了在自主设计 PCB 时最好都使用标准的贴片元件（如 0603 封装）。

3.3. 项目整体布线

因为我们使用的模块较多，在装配后期各种杜邦线已经较为凌乱，这很大程度上阻碍了检查，同时给信号带来了较大的干扰。以后可以使用螺丝钉+木板的方案固定各个模块，同时给板子设计统一的供电，尽量少用杜邦

线进行连接，来确保信号的完整性，提高项目的整洁性，方便检修。

同时，在信号线较多时，应该采用带有屏蔽的借口，例用 SMA 借口进行调试，还可以直接将 SMA 接口接入示波器，方便观察波形。

3.4 软件及硬件调试

由于项目发射端的 FM 调制要求的载波频率较为特殊，因此不能使用传统的硬件模块进行信号发射。在芯片的控制上，需要分别完成信号的采集、信号处理以及定时控制。这一过程涉及 setup 阶段和 loop 中同时控制四个不同模块。这不仅需要代码的逻辑清晰，还要求对单片机引脚的使用进行规划，否则可能导致电路接线复杂。

在硬件的调试上，我们采取的基本策略是先分别对单个元件进行功能检测，再完成单一功能模块的合并，最后整体测试。这样的方式虽然准确性较高，但耗时较长，下次进行相关项目时可以采用并行测试，即直接对整个对功能模块测试，再通过示波器或电压表排查出的问题对单一模块的代码逻辑进行修改。