**基于200kHz正弦波的相位计算法的超声波测距系统**



**2022年08月14**

**摘要**

未来AI技术的发展需要以机器人为现实载体，而机器人离不开精准的环境测距传感器，在未来的机器作业中，会面临许多需要精准测量的场景与应用。本系统基于对200kHz正弦波的处理，并通过ADC采集到接收波形，通过定时器记录超声波传播时间。相比于普通超声波测距的只利用定时器计时，我们还利用傅里叶变换对其进行相位计算，根据dds与ADC波形的相位差计算出误差时间，从而更为精准的测出距离，可以为未来AI机器人的环境感知提供更为精准的数据。

**关键字：**

200kHz正弦波，DDS(AD9850), ADC, 傅里叶变换，计算相位差

Abstract: The development of future AI technology requires robots as the carrier of reality, and robots are inseparable from accurate environmental ranging sensors, and in future machine operations, they will face many scenes and applications that require accurate measurement. This system is based on the processing of the 200kHz sine wave. The received waveform is collected by the ADC. We record the ultrasonic propagation time though the timer. Compared to ordinary ultrasonic ranging devices，we calculated the Phase by the Fourier transform, and the error time is calculated according to the phase difference between the dds wave and the ADC wave, so as to measure the distance more accurately. It can provide more accurate data for the environmental perception of future AI robots.

Keywords:

200kHz sine wave

DDS(AD9850)

ADC

Fourier transform

Phase Calculation

目录

[Abstract: 1](#_Toc23858)

[Keywords: 1](#_Toc16994)

[一、 系统设计 1](#_Toc4648)

[（一）信号发生器的选择 1](#_Toc11850)

[（二） 放大电路的选择 2](#_Toc23347)

[二、 理论分析与计算 2](#_Toc9593)

[（一） 电压放大电路的理论计算 2](#_Toc5339)

[（二）波形初相位计算的理论分析 3](#_Toc22662)

[（三）超声波测距距离计算的理论分析 3](#_Toc8892)

[三、 软件设计 4](#_Toc25700)

[（一）整体实现流程： 4](#_Toc24253)

[（二）AD9850的驱动程序： 5](#_Toc4399)

[（三）定时器计时程序： 5](#_Toc14100)

[四、 硬件设计 6](#_Toc26791)

[五、 系统测试方案与测试结果 7](#_Toc6487)

[（一）DDS发波测试 8](#_Toc27001)

[（二）发射电路放大测试一 8](#_Toc31533)

[（三）发射电路放大测试二 8](#_Toc1465)

[（四）接收测试 9](#_Toc5459)

[（五）接收电路放大测试 9](#_Toc20025)

[（六）ADC采样测试 10](#_Toc19483)

[（七）相位计算仿真 10](#_Toc31145)

[六、 结论 11](#_Toc22669)

[七、参考文献 11](#_Toc11684)

[附录I：设计程序清单 12](#_Toc8138)

## 一、系统设计

系统设计主要思路如下：

本系统通过单片机外接DDS模块产生200KHz的正弦波，通过放大器将信号一分为二后再输入至加法器使信号变为两倍，放大后的信号通过超声波换能器并由探头接收，利用接收的超声波探头把超声波转换成需要计算的电压信号，电压信号被再次放大后输入到ADC转换成可以计算的数字信号，将数字信号与定时器计时得到的传播时间与相位计算后得出所求距离，最后计算结果距离在OLED屏上显示。

流程图如图1-1：

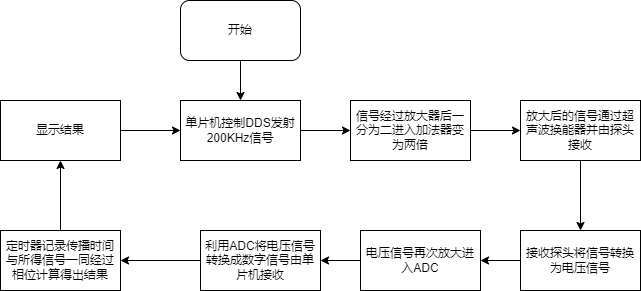


图1-1 系统设计流程图

### （一）信号发生器的选择

方案一：DAC

单片机自带的DAC功能的持续发波需要持续占用单片机资源，程序上无法精确定位发波时间点，会影响定时器计时的准确性，且频率无法达到项目要求的200kHz。

方案二：DDS\_AD9850

DDS是一种直接数字频率合成技术，内涵可编程DDS系统和和高速比较器，可实现全数字编程控制的频率合成，相对于DAC更专业应用与信号发生领域，可以做到实时调节输出频率和相位。

综上所述选择方案二。

### （二）放大电路的选择

方案一：使用三级管共射极放大电路

电流稳定，输出电压放大效果稳定，不易出现失真现象

方案二：使用运放电路放大

仿真出现放大不稳情况，易出现失真现象且出现相位移动情况

综上所述选择方案一。

## 二、理论分析与计算

### （一）电压放大电路的理论计算

发射端电阻为R29和R25控制放大倍数：……………………………………式2.1.1

取R29=56Ω，R25=240Ω，C17用于调整发射端阻抗微调放大倍数，R27、R26用于调整Re压降，………………………………………………………………………式2.1.2

……………………………………………………………………式2.1.3

Ω…………………………………………………………………式2.1.4

Ω………………………………………………………………式2.1.5

发射端放大电路如图2-1：

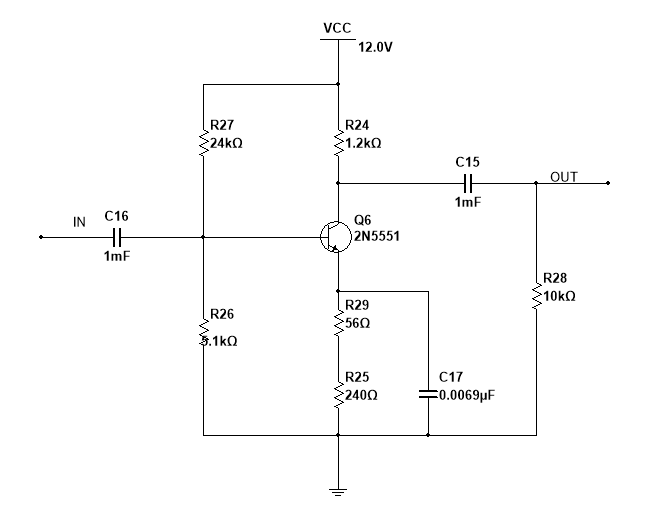


图2-1发射端放大电路图

接收部分电路通过改变C17位置电容增大值220uF,即相当于减小了Re阻抗，增加了其放大倍数使接收电压由约4mV差放大至约600mV差。

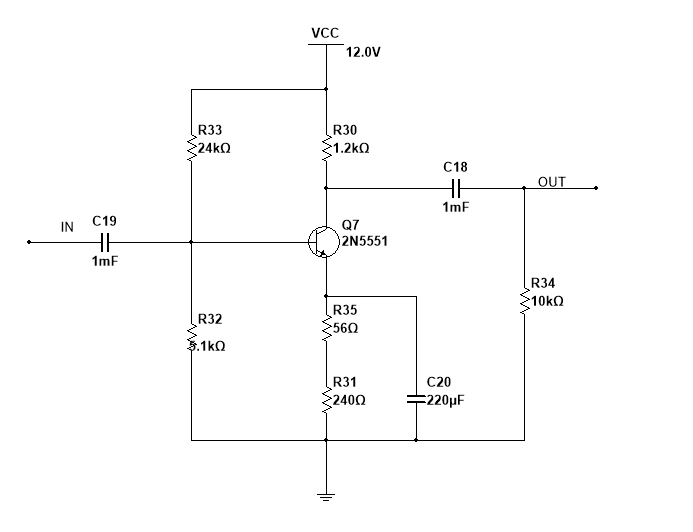


图2-2接收端放大电路图

### （二）波形初相位计算的理论分析

假设接收波为，其中；对x(t)进行傅里叶变换：；根据欧拉公式有；

因此；代入可得；

其中，因此的实部a除以虚部b可得，即接收波的初相角；

### （三）超声波测距距离计算的理论分析

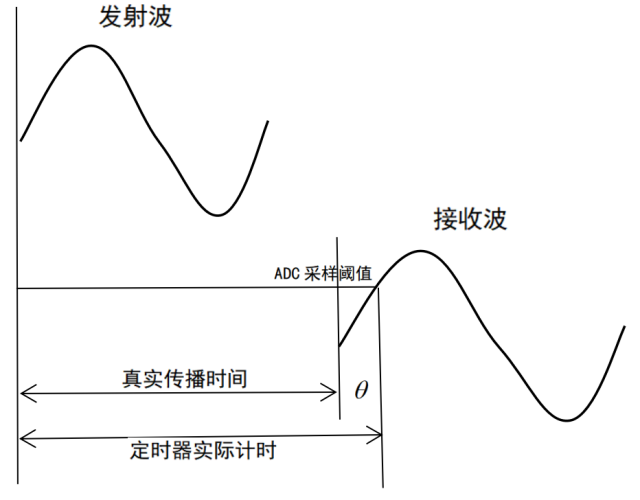


图2-3 距离计算原理图

如图2-3所示，定时器实际计时并非等于真实传播时间，期间相差的距离为ADC采集的接收波与DDS发射波的相位差，由于程序上控制DDS发出的波形相位为0，因此，若已知ADC采样的接受波的初相角，则可求出定时器实际计时与真实传播时间之差：

（单位：s）; 因此真实传播实际（单位s）

## 三、软件设计

软件设计平台：keil

开发工具：串口助手

### （一）整体实现流程：

主程序主要由控制超声波发射与停止200KHz波、ADC采样、相位计算与距离计算，以及OLED屏显示四部分组成。在初始化需要模块后，在每次循环开始的时候打开DDS模块开始发波，并且定时器开始计时，控制DDS模块发出100us的信号后关闭DDS；当ADC采集的电压数据达到设定阈值时，定时器停止计时，ADC开始把后续采集到的值存入数组，经过快速傅里叶变换fft函数计算得到波形频域函数的实部与虚部；在相位计算函数中，通过波形频域函数的实部与虚部求出ADC采样接收波的初相位，根据定时器计时与方才求出的相位差可得出距离结果，最终显示在OLED屏上。程序需延时100us等待后续超声波完全经过探头后才能结束一次主循环，进入下一次主循环。流程图如图3-1。

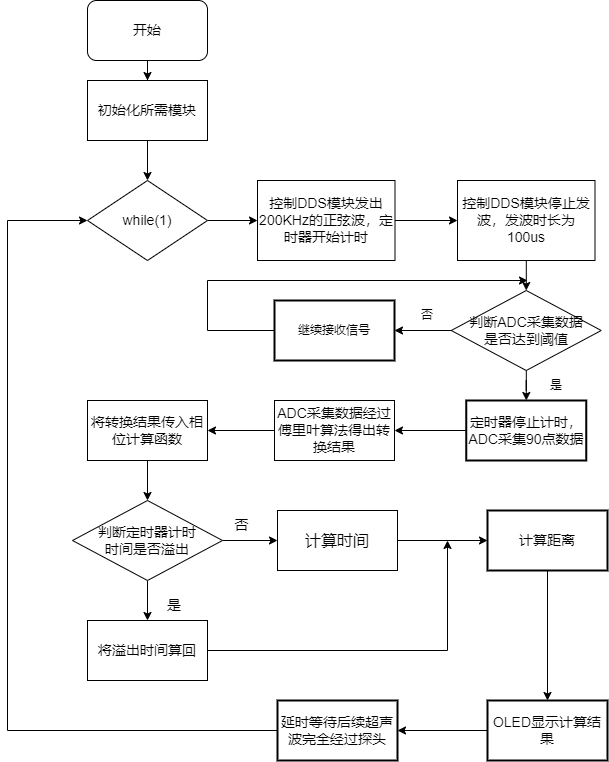


图3-1 程序设计流程图

### （二）AD9850的驱动程序：

本项目采用串行传输方式对DDS输入指令，先每次以8位数据为单位，从后往前依次往DATA引脚写入频率指令，指令总共24位，同理而后依次输入相位指令。在主程序中，本项目发波时选择了相位为0，频率为200kHz的正弦波。具体输入命令流程如图3-2：

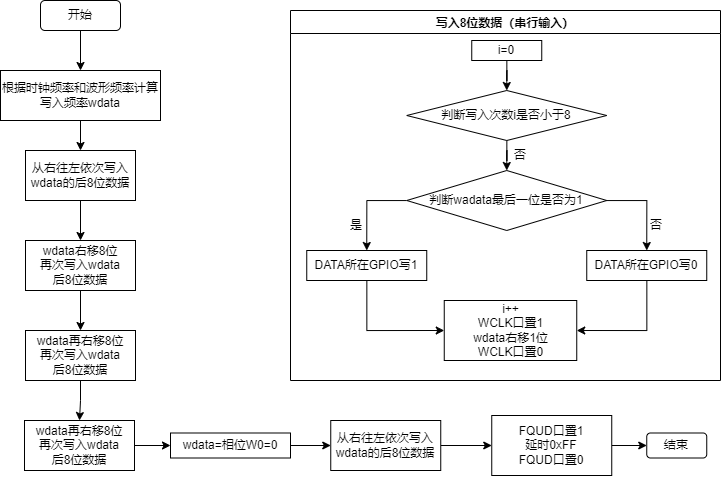


图3-2 AD9850输入指令流程图

### （三）定时器计时程序：

定时器初始化设置时使能中断，定时时间设置为10us，即每10us进入一次定时器中断函数，定时器中断函数中进行时间的递增，当时间值溢出时清零；在主程序中，当打开DDS时使能定时器，计时开始；当ADC当前采样值大于阈值时，取出当前定时器时间值，关闭定时器，流程图如图3-3：

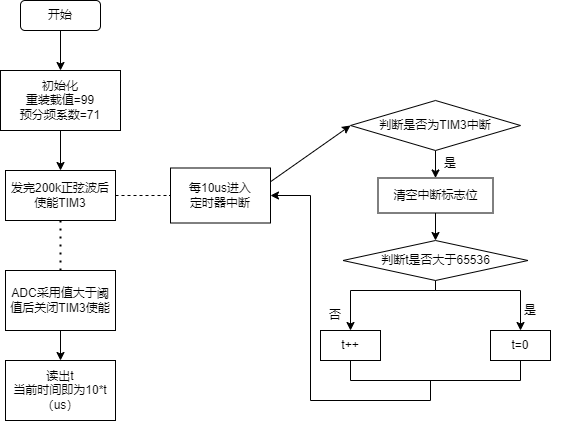


图3-3 定时器计时流程图

## 四、硬件设计

软件仿真平台：Multisim

实现方法：

由单片机控制DDS发射出200KHz的正弦信号，其幅值为0~1V，通过图2.1.1所设计的电压放大电路将电压放大约9倍，使用加法器将两个放大9倍后的信号相加实现18倍放大，将放大后的电压输入至超声波换能器探头1，使其发射对应的超声波，由超声波探头2接收信号，通过电压放大电路放大至500mV,接入ADC采样端采样。

流程图如图4-1与图4-2：

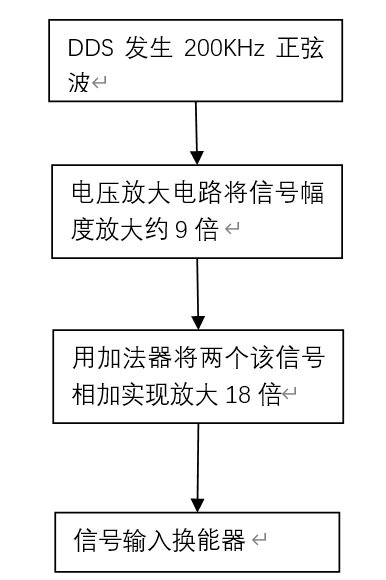
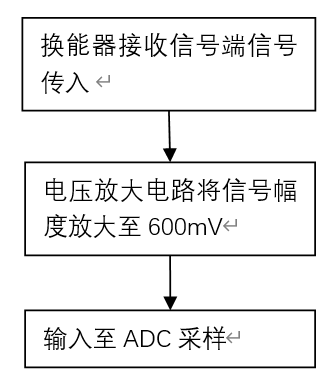
 

图4-1发射电路流程图 图4-2接收电路流程图

## 五、系统测试方案与测试结果

本系统的测试分为DDS发波测试、发射电路的两次放大测试、接收端超声波换能器接收波形测试接收电路放大测试、ADC采样测试、相位计算仿真以及最终系统测距功能测试。测试装置如图5-1与5-2。

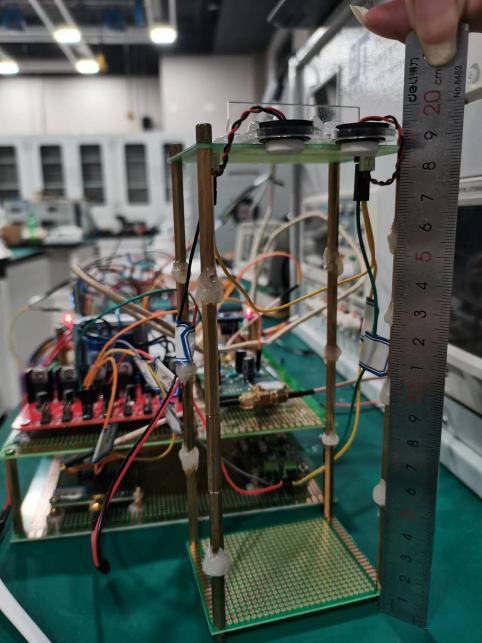
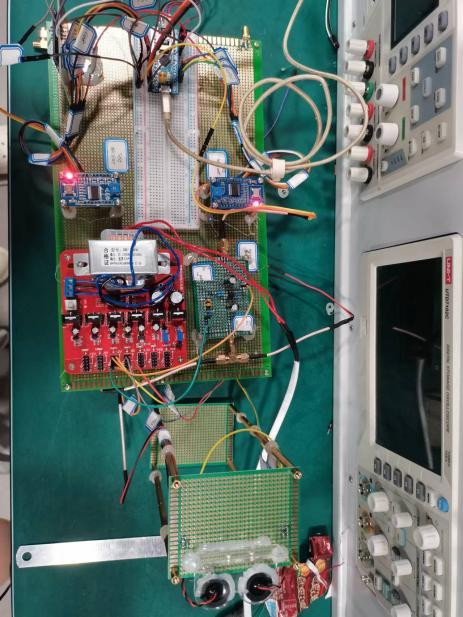
 

图5-1 系统侧视图 图5-2 系统俯视图

### （一）DDS发波测试

第一个测试是对dds发波的频率、电压的测试。测试结果如图5-3,5-4所示。

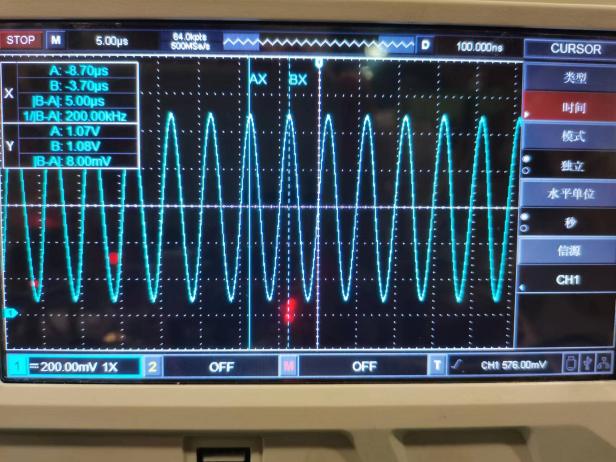
 

图5-3 DDS频率测试 图5-4 DDS电压测试

由图5-3和图5-4可知，DDS发出的波形频率为200kHz，电压为从56mv到1.01V，幅值为476mv。

### （二）发射电路放大测试一

第二次测试时对发射电路第一次放大效果的测试。输入为DDS发出的从56mv到1.01V，幅值为476mv的正弦波，输出如图5-5，对输出的频率验证如图5-6。

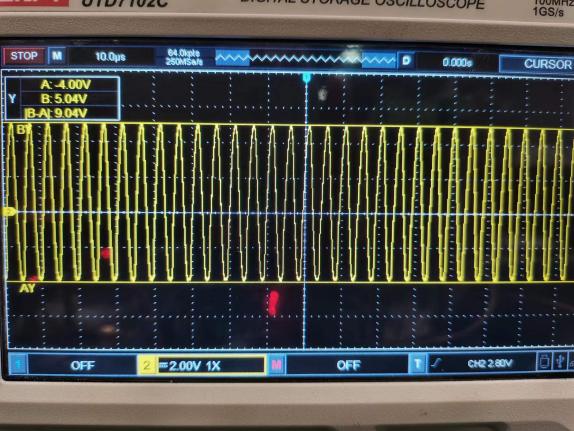
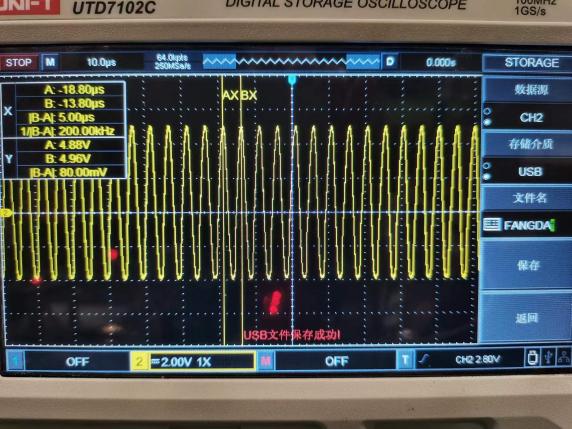
 

图5-5 对DDS发波的初次放大结果 图5-6 初次放大结果频率验证

由图5-5与图5-6可知，输出结果为从-4V到5.04V，赋值为4.52V的正弦波，即放大了约10倍，频率保持在200kHz。

### （三）发射电路放大测试二

第三个实验是将第二个实验放大的结果一分为二接入加法器模块，需要验证加法器输出结果是否在第二次实验的基础上又放大了2倍，输出结果与频率验证如图5-7与5-8。

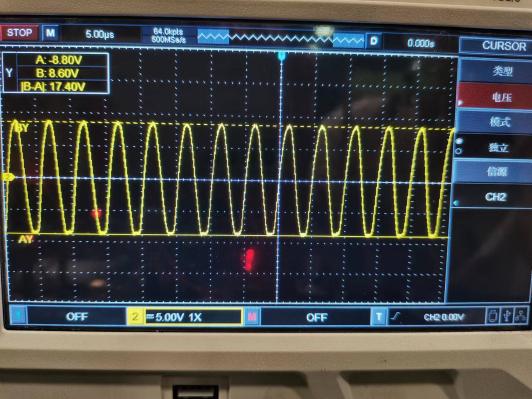
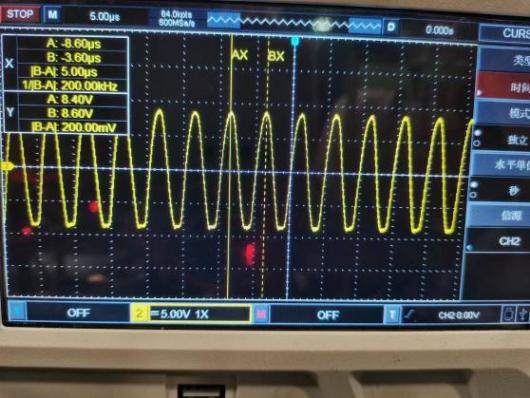
 

图5-7 加法器输出结果 图5-8 加法器输出结果的频率验证

如5-7与5-8所示，电压从-4V到5.04V，赋值为4.52V放大至从-8.8V到8.6V，赋值变为8.7V，即放大了约2倍，频率保持200kHz。

### （四）接收测试

第四个实验是将放大两次的结果输入超声波换能器的发射端，并测量接收端电压，结果如图5-9与5-10所示。

图5-9 接收端波形频率 图5-10 接收端波形电压

由5-9与5-10可知，频率保持200kHz，接收电压变为从-2.24mv到1.68mv，赋值为1.96mv。

### （五）接收电路放大测试

第五个实验为对接收端波形进行放大，放大结果如图5-11与5-12。

图5-11 接收波放大结果频率 图5-12 接受波放大结果

由图5-11与5-12可知放大结果为从-236mv到292mv，赋值为264mv，即放大了约135倍。

### （六）ADC采样测试

第六个实验是对放大后的接收波进行ADC采样，测试采样串口实时打ADC采样值，并把数据导入excel进行绘图，ADC采样范围为0-3.3V，采样结果如图5-13，而当关闭DDS时ADC采样如图5-14。

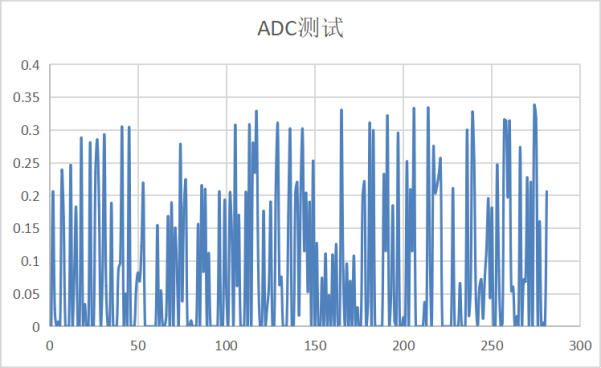
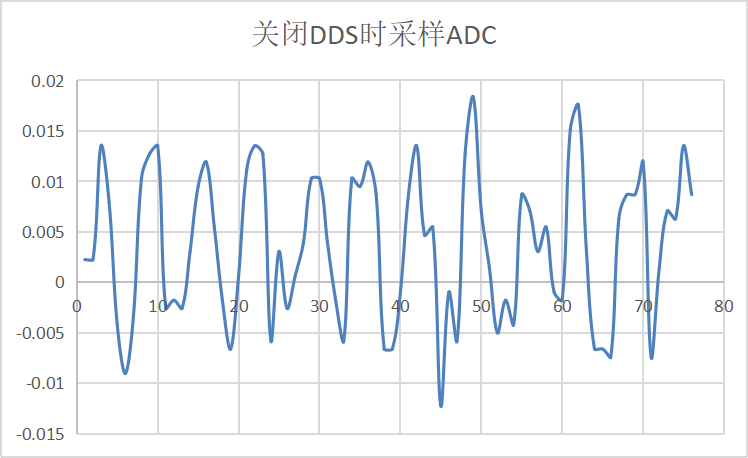
 

图5-13 ADC采样结果 图5-14 关闭DDS时采样结果

由图5-14可知，当关闭DDS时ADC采样值最大不超过0.02V，而当打开DDS时采集到的波形电压普遍超过0.02V，保险起见，我们采取0.1V作为阈值，即当ADC电压超过0.1V时关闭定时器，将后续ADC采集的电压值存入数组中进行快速傅里叶变换。

### （七）相位计算仿真

第七个实验是对相位测试进行仿真，我们采用matlab的fft函数对设定好的200kHz波形离散点进行初相位计算，测试结果如图5-16，5-17，5-18。由此验证相位计算正确，相关代码见附件。

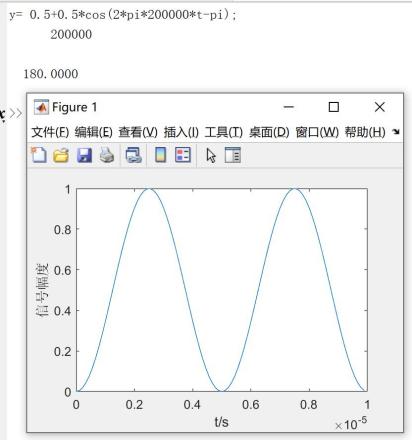
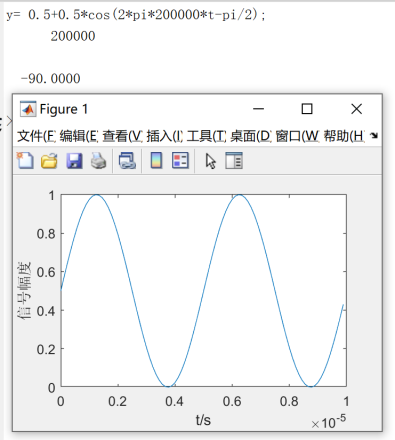
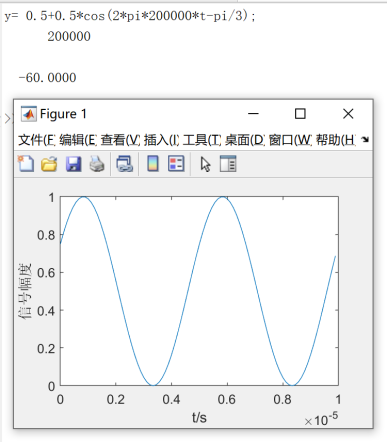
  

图5-16 相位为-π时 图5-17 相位为-π/2时 图5-18 相位为-π/3时

由于时间原因，最终测距效果仍未调试完成，此处不做调试结果展示。

## 六、结论

本系统暂时完成了DDS发出200kHz正弦波、发射信号放大20倍、超声波换能器接收信号放大135倍、ADC采样与相位计算仿真功能。

## 七、参考文献

1. 袁梅,汪鹏程,张学军.超声波测距的双相位检测方法[J].电子测量技术,2021,44(17):112-117.DOI:10.19651/j.cnki.emt.2107536.
2. 沈梦婷. 基于相位检测的高精度异体收发式超声波测距系统[D].天津大学,2014.

# 附录I：设计程序清单

**main.c文件**

#include "stm32f10x.h" //STM32头文件

#include "sys.h"

#include "delay.h"

#include "oled0561.h"

#include "tim.h"

#include "bsp\_adc.h"

#include "usart.h"

#include "AD9850.h"

#include "AD9850\_2.h"

#include <math.h>

#include "table\_fft.h"

#include "stm32\_dsp.h"

#define PI 3.14159265

#define V 344

#define NPT 90 //采点次数=90

float fashe[90];//发射

float jieshou[90];//接受

int juli;//距离【mm】

int i;

u16 second1;//发射秒

u16 second2;//接收秒

u16 times;

extern vu16 ADC\_DMA\_IN[2]; //声明外部变量

extern uint32\_t t; //TIM3的计时秒钟

extern u8 Capture\_State;

extern u8 flag;

char p[10];

extern uint16\_t TIM5\_GetTime(void);

int32\_t input[90];

int32\_t output[90];

int32\_t OUTPUT\_THETA[90];

// ADC1转换的电压值通过MDA方式传到SRAM

extern \_\_IO uint16\_t ADC\_ConvertedValue[NOFCHANEL];

// 局部变量，用于保存转换计算后的电压值

float ADC\_ConvertedValueLocal[NOFCHANEL];

void theta\_calculate(int32\_t \*IBUFOUT) //拿官方计算幅值的函数改的fft计算相角函数

{

s16 lx,ly;

u32 i;

for(i=0;i<NPT/2;i++)//由于FFT的频域结果是关于奈奎斯特频率对称的，所以只计算一半

{

lx = (IBUFOUT[i] << 16)>>16; //取低十六位，虚部

ly = (IBUFOUT[i] >> 16); //取高十六位，实部

float X = NPT\*((float)lx)/32768;

float Y = NPT\*((float)ly)/32768;

float theta = atan(X/Y);

//计算相角的公式为theta=atan(虚部/实部);

OUTPUT\_THETA[i] = (u32)(theta);

//对于theta来说没必要乘回来32768，反正都约掉了

}

//之所以先除以32768后乘以65536是为了符合浮点数的计算规律

}

int main (void){//主程序

int n=0;//用于for循环存储ADC数值

int i=0;

delay\_ms(500); //上电时等待其他器件就绪

RCC\_Configuration(); //系统时钟初始化

I2C\_Configuration();//I2C初始化

OLED0561\_Init(); //OLED初始化

OLED\_DISPLAY\_8x16\_BUFFER(0," TEST "); //显示字符串

TIM3\_Init(99,71);//定时器初始化，定时10us

OLED\_DISPLAY\_8x16\_BUFFER(2,"juli/um time/us"); //显示字符串

USART3\_Init(115200); //串口初始化

Init\_AD9850();

Init\_AD9850\_2();

ADCx\_Init();

while(1){

AD9850\_2\_Write\_Serial(0,200000); //200k

TIM\_Cmd(TIM3,ENABLE); //使能TIM3

second1=t;

//100us即发20个200k和2个20k

delay\_us(100); //10cm的距离需要约400us才会传回来

//关闭发波

AD9850\_2\_Reset\_Sreial();

while((float)ADC\_ConvertedValue[1]/4096\*3.3<=0.1){

}

second2=t; //提取时间

TIM\_Cmd(TIM3,DISABLE); //关闭使能TIM3

for(n=0;n<=89;n++){

ADC\_ConvertedValueLocal[1] =(float) ADC\_ConvertedValue[1]/4096\*3.3;

input[n]= ADC\_ConvertedValueLocal[1];

}

if(second2>=second1){

times=second2-second1; //times单位为us

}

else{

times=0xffff+second2-second1; //溢出了，需要加回来0xffff

}

cr4\_fft\_256\_stm32(output,input,NPT);//进行fft计算

theta\_calculate(output);//计算相位

juli= V \*(times-OUTPUT\_THETA[0]/(2\*PI\*0.2))/2;//计算出juli,单位um

//这里时间是times-theta/2pi\*（1/200k）,但因为times单位是us，juli单位是um，所以200k变成了0.2

OLED\_DISPLAY\_8x16\_BUFFER(4," time = "); //显示字符串

OLED\_DISPLAY\_8x16(4,9\*8,times/10000+0x30);

OLED\_DISPLAY\_8x16(4,9\*8,times%10000/1000+0x30);

OLED\_DISPLAY\_8x16(4,10\*8,times%1000/100+0x30);

OLED\_DISPLAY\_8x16(4,11\*8,times%100/10+0x30);

OLED\_DISPLAY\_8x16(4,12\*8,times%10+0x30);

//距离为int，单位为um

OLED\_DISPLAY\_8x16(6,8\*8,juli/1000000+0x30);

OLED\_DISPLAY\_8x16(6,9\*8,juli%1000000/100000+0x30);

OLED\_DISPLAY\_8x16(6,10\*8,juli%100000/10000+0x30);

OLED\_DISPLAY\_8x16(6,11\*8,juli%10000/1000+0x30);

OLED\_DISPLAY\_8x16(6,12\*8,juli%1000/100+0x30);

OLED\_DISPLAY\_8x16(6,13\*8,juli%100/10+0x30);

OLED\_DISPLAY\_8x16(6,14\*8,juli%10+0x30);

OLED\_DISPLAY\_8x16\_BUFFER(6," juli=");

delay\_us(100);//等待发射的后几个波路过接收端

}

}

**AD9850.c文件**

#include "stm32f10x.h"

#include "AD9850.h"

void AD9850\_Delay(unsigned int z)

{

for(;z>0;z--)

{;}

}

void AD9850\_Reset\_Sreial(void)

{

AD9850\_WCLK\_CLR ;

AD9850\_FQUD\_CLR ;

//RST??

AD9850\_RST\_CLR ;

AD9850\_RST\_SET ;

AD9850\_Delay(0xFFFF) ;

AD9850\_RST\_CLR ;

//WCLK??

AD9850\_WCLK\_CLR ;

AD9850\_WCLK\_SET ;

AD9850\_Delay(0xFFFF) ;

AD9850\_WCLK\_CLR ;

//FQUD??

AD9850\_FQUD\_CLR ;

AD9850\_FQUD\_SET ;

AD9850\_Delay(0xFFFF) ;

AD9850\_FQUD\_CLR ;

}

void AD9850\_Write\_Serial(unsigned char W0,unsigned long freq)

{

unsigned char i,wdata ;

unsigned long y ;

y = (unsigned long )268435456.0/AD9850\_SYSTEM\_COLCK\*freq ;

//268435456.0=2^28

//y = (unsigned long )AD9850\_SYSTEM\_COLCK\*freq ;

wdata = y>>0 ; //?w4

for(i=0 ;i<8 ;i++)

{

if(wdata & 0x01)

AD9850\_DATA\_Write\_1 ;

else

AD9850\_DATA\_Write\_0 ;

AD9850\_WCLK\_SET ;

wdata >>= 1 ;

AD9850\_WCLK\_CLR ;

}

wdata = y>>8 ; //?w3

for(i=0 ;i<8 ;i++)

{

if(wdata & 0x01)

AD9850\_DATA\_Write\_1 ;

else

AD9850\_DATA\_Write\_0 ;

AD9850\_WCLK\_SET ;

wdata >>= 1 ;

AD9850\_WCLK\_CLR ;

}

wdata = y>>16 ; //?w2

for(i=0 ;i<8 ;i++)

{

if(wdata & 0x01)

AD9850\_DATA\_Write\_1 ;

else

AD9850\_DATA\_Write\_0 ;

AD9850\_WCLK\_SET ;

wdata >>= 1 ;

AD9850\_WCLK\_CLR ;

}

wdata = y>>24 ; //?w1

for(i=0 ;i<8 ;i++)

{

if(wdata & 0x01)

AD9850\_DATA\_Write\_1 ;

else

AD9850\_DATA\_Write\_0 ;

AD9850\_WCLK\_SET ;

wdata >>= 1 ;

AD9850\_WCLK\_CLR ;

}

wdata = W0 ; //?w0

for(i=0 ;i<8 ;i++)

{

if(wdata & 0x01)

AD9850\_DATA\_Write\_1 ;

else

AD9850\_DATA\_Write\_0 ;

AD9850\_WCLK\_SET ;

wdata >>= 1 ;

AD9850\_WCLK\_CLR ;

}

AD9850\_FQUD\_SET ; //????

AD9850\_Delay(0xFFFF) ;

AD9850\_FQUD\_CLR ;

}

void Init\_AD9850(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure ;

GPIO\_InitStructure.GPIO\_Pin = AD9850\_WCLK | AD9850\_FQUD | AD9850\_RST | AD9850\_DATA ;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz ;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP;

GPIO\_Init(AD9850\_CONTROL\_PORT ,&GPIO\_InitStructure) ;

AD9850\_Reset\_Sreial() ;

}

**tim.c文件**

#include "tim.h"

void TIM3\_Init(u16 arr,u16 psc){ //TIM3 初始化 arr重装载值 psc预分频系数

TIM\_TimeBaseInitTypeDef TIM\_TimeBaseInitStrue;

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM3,ENABLE);//使能TIM3

TIM3\_NVIC\_Init (); //开启TIM3中断向量

TIM\_TimeBaseInitStrue.TIM\_Period=arr; //设置自动重装载值

TIM\_TimeBaseInitStrue.TIM\_Prescaler=psc; //预分频系数

TIM\_TimeBaseInitStrue.TIM\_CounterMode=TIM\_CounterMode\_Up; //计数器向上溢出

TIM\_TimeBaseInitStrue.TIM\_ClockDivision=TIM\_CKD\_DIV1; //时钟的分频因子，起到了一点点的延时作用，一般设为TIM\_CKD\_DIV1

TIM\_TimeBaseInit(TIM3,&TIM\_TimeBaseInitStrue); //TIM3初始化设置

TIM\_ITConfig(TIM3, TIM\_IT\_Update, ENABLE);//使能TIM3中断

TIM\_Cmd(TIM3,DISABLE); //使能TIM3

}

void TIM3\_NVIC\_Init (void){ //开启TIM3中断向量

NVIC\_InitTypeDef NVIC\_InitStructure;

NVIC\_InitStructure.NVIC\_IRQChannel = TIM3\_IRQn;

NVIC\_InitStructure.NVIC\_IRQChannelPreemptionPriority = 0x3; //设置抢占和子优先级

NVIC\_InitStructure.NVIC\_IRQChannelSubPriority = 0x3;

NVIC\_InitStructure.NVIC\_IRQChannelCmd = ENABLE;

NVIC\_Init(&NVIC\_InitStructure);

}

uint32\_t t=0;

void TIM3\_IRQHandler(void){ //TIM3中断处理函数

if (TIM\_GetITStatus(TIM3, TIM\_IT\_Update) != RESET){ //判断是否是TIM3中断

TIM\_ClearITPendingBit(TIM3, TIM\_IT\_Update);

if(t==65536){

t=0;

}

t++;

}

TIM\_ClearITPendingBit(TIM3, TIM\_IT\_Update);

}

**matlab计算相位程序**

%% 计算相位

clear

clc

close all

fs=9000000; %采样频率

N = 90;

t=0:1/fs:(N-1)/fs;

y= 0.5+0.5\*cos(2\*pi\*200000\*t-pi/3);

figure;plot(t,y);xlabel('t/s');ylabel('信号幅度')

% fft单边谱

df=fs/N; %分辨率

f=(0:N-1)\*df; %其中每点的频率

Y=fft(y)/N\*2; %真实的幅值

figure(2)

subplot(211)

plot(f(1:N/2),abs(Y(1:N/2)));xlabel('f/Hz');ylabel('单边频谱')

title('单边频谱')

% 计算信号相位

tol = 1e-6;

Y(abs(Y) < tol) = 0;

theta = angle(Y(1:N/2))/pi;

figure;

plot(f(1:N/2),theta)

ylabel('相位/pi')

disp("y= 0.5+0.5\*cos(2\*pi\*200000\*t-pi/3);");

for i=1:N/2

if abs(theta(i))~=0

disp(f(i))

disp(theta(i)\*180)

end

end