基于 AVR 单片机的简易风洞 PID 稳定系统

张振尧 国机 1901 2018110071

一、系统的总体架构及功能描述

1.1 系统总体方案设计

系统以 ATMEGA328P-PU 单片机为核心模块，通过电机驱动模块带动扇叶，向圆管之内鼓风，将容器吹起。通过激光测距模块将容器在圆管之内的位置检测出来之后返回位置数据给单片机，通过单片机的判断，控制风力的大小，将容器控制在所要求的位置，系统面包板设计及初步连线如图 1 所示（使用 fritzing 绘制）。

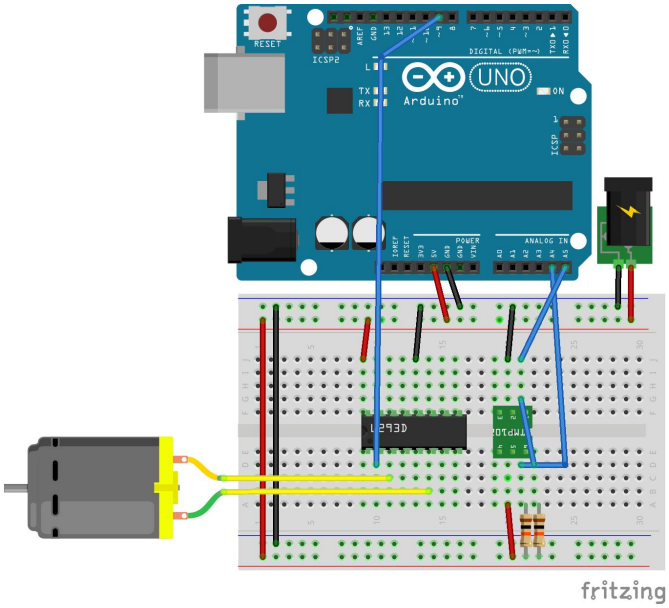


图 1 系统面包板设计

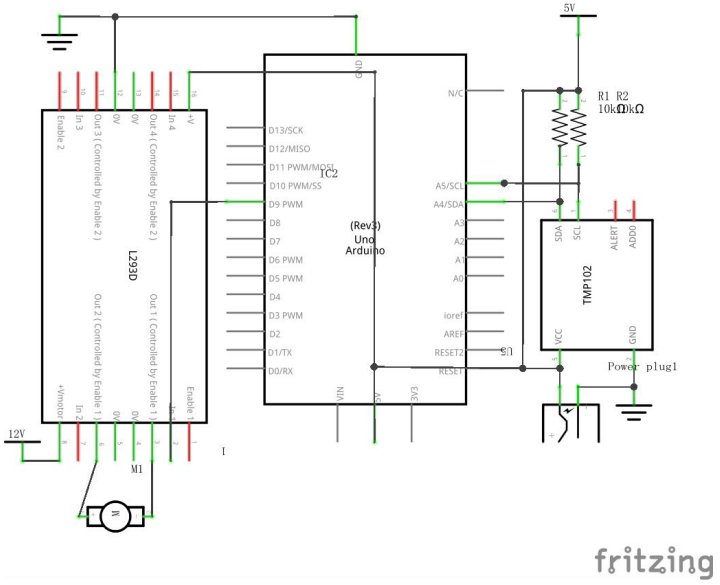


图 2 系统初步连线

1.2 设计方案论证

1.2.1 电机驱动模块

系统采用无刷电机 ESC 电子调速器驱动模块。ESC 具有三相输出，可以驱动一个直流电机。且该模块的驱动电流最大可以达到 10 A。可以满足电机的驱动要求，还可以在输入端加上 PWM 方波控制转速以及设定刹车，可以对电机转速进行精准控制。

1.2.2 定位模块

VL53L0X 激光测距模块的检测距离长，可达到 10m，而且该模块的反应速度也较快。可以根据自己所要求的时间来具体调节发射和接收时间，操作较为方便。另一方面，激光检测方式具有模块的操作简单，不足在于在垂直距离较长情况下，需要进行中心对准才能精准测距。

二、硬件设计及接口电路图

2.1 涵道风机的确定

在系统中，需要将容器在圆管之内上下吹动，这就会对风机的要求很高。普通的小风扇无法满足要求。所以选择了高速无刷电机，采用大 PVC 管和竖直圆筒，并结合鼓风机结构，将风吸入风道之后，经过一圈回流之后送入圆管。这样无刷涵道风机结构简单，稳定度高，较为可靠。图 3 为涵道风机示意图。

图 3 涵道风机示意图

2.2 电机驱动电路设计

在输入信号之后，ESC 电子调速器将输入的 PWM 波转换为电流电压相对较大的 PWM 信号。其信号保持一致或者取反。输出之后可以达到利用小电压小电流驱动大电压大电流的作用。驱动选择了常用的高电压大电流的 ESC 电子调速器模块，其动态响应能力较好，

还可以实现频繁的无极快速启动，制动和反转等优点，电机驱动电路及单片机系统原理图如图 4 所示（使用立创 EDA 在线绘制），在绘制过程中，参考了现有的 UNO 单片机以及无刷电机库文件。

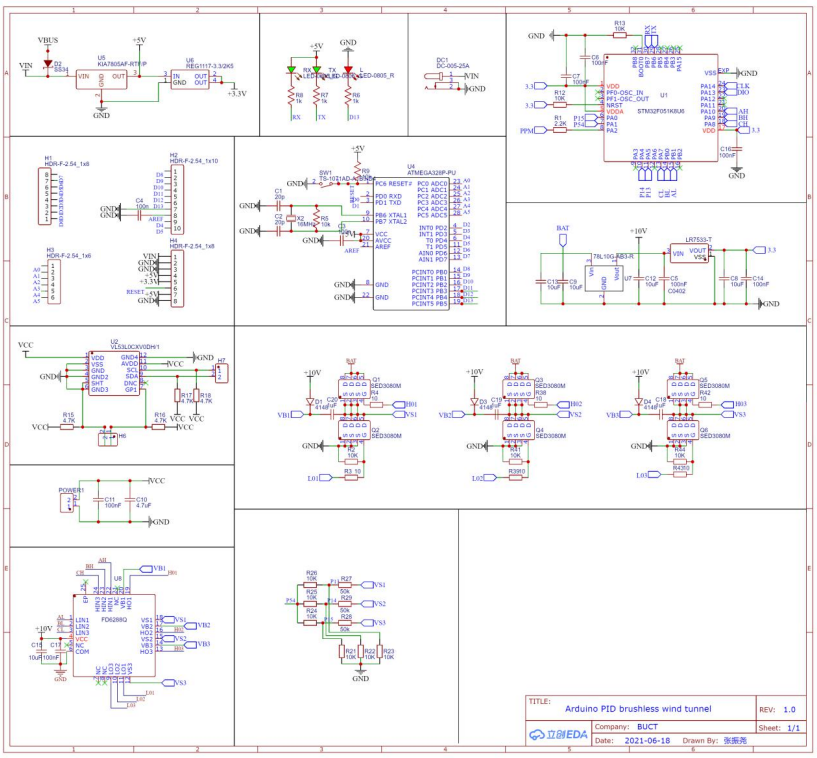


图 4 电机驱动及单片机系统整体原理图

三、软件设计思路及部分代码

3.1 激光测距检测距离计算

根据激光测距原理，激光测距在空气之中传播遇到障碍物的时候就会马上返回。假设激光测距的传播速度为 V，发射和接收的时间差为 t，就可以计算出发射点距离障碍物的距离 S，如公式(1)所示。

S=V*△t/2 (1)

根据系统之中的激光测距模块的特点，通过计算可以得出在 10 cm 到 4.5 m 之内的距离测量值。

3.2 电机控制算法的确定

对于无刷电机转速和容器的运动过程有着严格的要求。所以在控制小球的时候需要进行精确控制，通过选择适当的 PID 算法和有效的参数正定可以达到高精度控制目的。PID 公式如图)所示。

$$u(t) = Kp[e(t) + \frac{1}{Ti}\int_0^t e(t)dt + Td\frac{de(t)}{dt}]$$

其中：

Kp -> 控制器的比例系数

Ti -> 控制器的积分时间，也称积分系数

Td -> 控制器的微分时间，也称微分系数

根据上述公式，在最开始确定比例系数之后就开始确定微分器和积分器的数值。最终将三个系数写入程序之中，达到稳定。

在实际程序中，体现 PID 功能的代码如下：

```
error = distance - desired; //误差计算
pid_p = kp * error; //P 值计算
if (-8 < error && error < 8) //如果不在 8mm 误差距离范围内，不启用 I 控制
pid_i = pid_i + (ki * error);//I 值计算
pid_d = kd * ((error - previous_error) / elapsedTime); //D 值计算
previous_error = error; //保存当前误差为上一个误差
PID = pid_p + pid_i + pid_d; //计算 PID 总值
```

3.3 程序开发平台配置

本程序开发使用 PlatformIO 综合开发平台，配置文件及具体包含库文件如下

```
[platformio]
default_envs = uno //单片机开发环境选择 uno

[env:uno]
platform = atmelavr
board = uno //单片机厂商及型号确定
framework = arduino //单片机固件确定
upload_port = COM[3] //单片机端口号设定
lib_deps =
  //包含了所用到的库文件
   arduino-libraries/Servo@^1.1.8
   adafruit/Adafruit_VL53L0X@^1.1.1
   adafruit/Adafruit GFX Library@^1.10.10
   adafruit/Adafruit VEML6070 Library@^1.0.6
   adafruit/Adafruit BusIO@^1.7.5
   marlinfirmware/U8glib-HAL@^0.4.5
  smougenot/Adafruit_VEML6070@0.0.0-alpha+sha.f56ccf3f85
```

3.3 具体代码及注释

```
#include <Arduino.h>
#include <SPI.h> //SPI 通信库，用于和激光测距模块通讯
#include <Wire.h>
#include <Servo.h> //SERVO 舵机控制程序库
#include "Adafruit_VL53L0X.h" //激光测距模块库
```

```
float desired = 200; //设定预期容器高度
int initalspeed = 1300; //设定风机初始转速，通过调整这个参数，可以使风机一开始就运行在较稳定的状态
double kp = 3.55;  //3.55
double ki = 0.005; //0.003
double kd = 2.55;  //2.05 //设定 PID 初始控制值

//设定一些变量及初始值
float elapsedTime, time, timePrev, PID, error;
float previous_error = 0;
float pid_p = 0;
float pid_i = 0;
float pid_d = 0;
int distance;
//设定 low 和 high 的值，在后面的判断程序中使 PID 输出值不超出 1000-2000 范围
int low = initalspeed - 1000;
int high = 2000 - initalspeed - 200;
设定 PWM 信号输出的引脚号为 9（UNO 可用的 pwm 输出引脚并不多，需要单独进行设定）
#define MOTOR_PIN 9
Servo motor; //利用 SERVO 库定义一个 motor 电机变量
//利用 VL53L0X 库定义一个 lox 和 measure 结构体来存储距离数据包
Adafruit_VL53L0X lox = Adafruit_VL53L0X();
VL53L0X_RangingMeasurementData_t measure;

void setup()
{
  //将用于调试的串口通讯的波特率设定为 250000
   Serial.begin(250000);
   //将 PWM 输出脚附加到 motor 变量上，设定输出范围为 1000, 2000
   motor.attach(9, 1000, 2000);
   //获取当前时间，存储到 time 变量中
   time = millis();
   //写油门最低值
motor.writeMicroseconds(1000);
   if (!lox.begin())
      while (1)
{
//如果激光测距模块没有正常启动，在串口中输出报错
         Serial.println(F("Failed to boot VL53L0X sensor"));
          delay(3000); //防抖功能
}
void loop() //循环执行函数
{
   timePrev = time; //定义上一个时间值
```

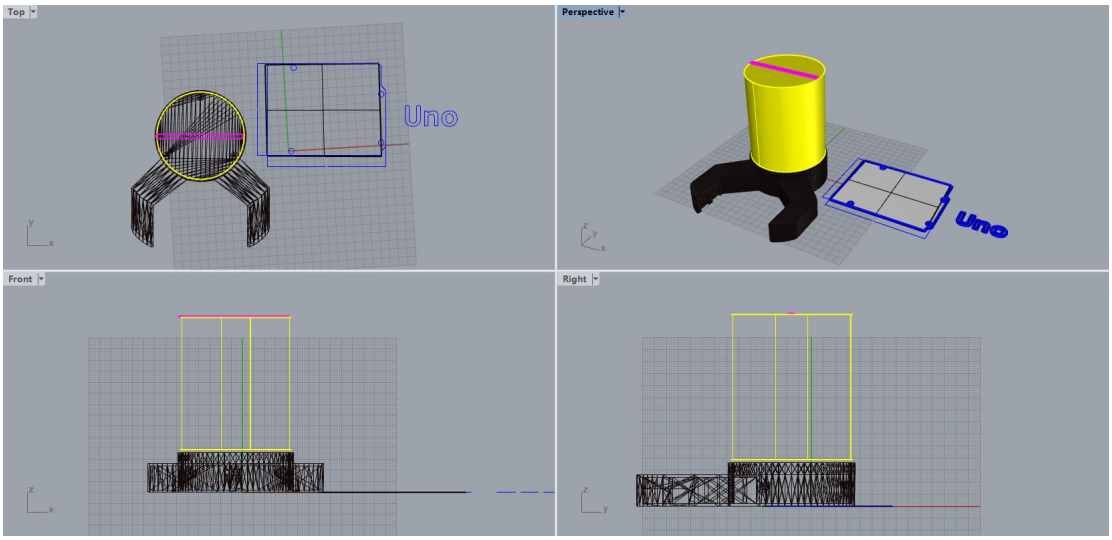
```
  time = millis(); //获取当前时间值
  elapsedTime = (time - timePrev) / 1000; //计算变换时间值
  lox.rangingTest(&measure, false); //初始化测距模块
  if (measure.RangeStatus != 4) //如果正确获得到距离数据包
    distance = measure.RangeMilliMeter; //存储距离变量到 distance 中
  else
    while (1) //如果没有正常初始化测距模块，就在串口中报错
      Serial.println(F("Failed to measure"));
Serial.print(measure.RangeMilliMeter); //输出测量到的距离值
//PID 程序
  error = distance - desired; //误差计算
  pid_p = kp * error; //P 值计算
  if (-8 < error && error < 8) //如果不在 8mm 误差距离范围内，不启用 I 控制
    pid_i = pid_i + (ki * error);//I 值计算
  pid_d = kd * ((error - previous_error) / elapsedTime); //D 值计算
  previous_error = error; //保存当前误差为上一个误差
PID = pid_p + pid_i + pid_d; //计算 PID 总值

  Serial.print(" ");
  Serial.println(PID);
  //下面这段程序，用于控制 PID 输出值 low 和 high 范围内，使最终输出值在 1000-2000
范围内
  if (PID < -low)
    PID = -low;
  if (PID > high)
    PID = high;
//加上初始设定速度就是最终输出速度
  PID = PID + initalspeed;
  //利用 SERVO 库写 PID 电机转速
  motor.writeMicroseconds(PID);
}
```

四、测试方案与测试结果

4.1 系统搭建方法及测试数据

按图 5 所示风洞结构图（使用犀牛软件进行建模渲染），进行系统搭建，然后在 PVC 管内加入容器测试。



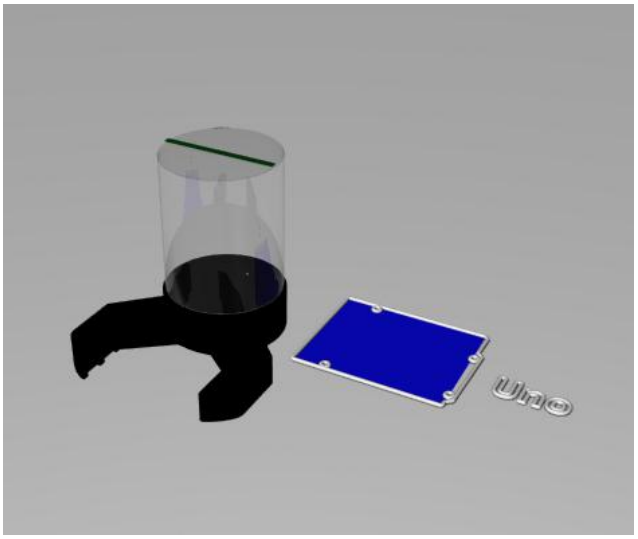


图 5 风洞系统建模结构图

4.2 测试结果

经过一系列测试表明整个系统能够完全达到要求。实际运行过程中，因为设定了涵道风机的转速区间，所以对容器内配重进行了调整。整个系统经过测试，工作稳定，精度高。系统实物如图 6 所示。

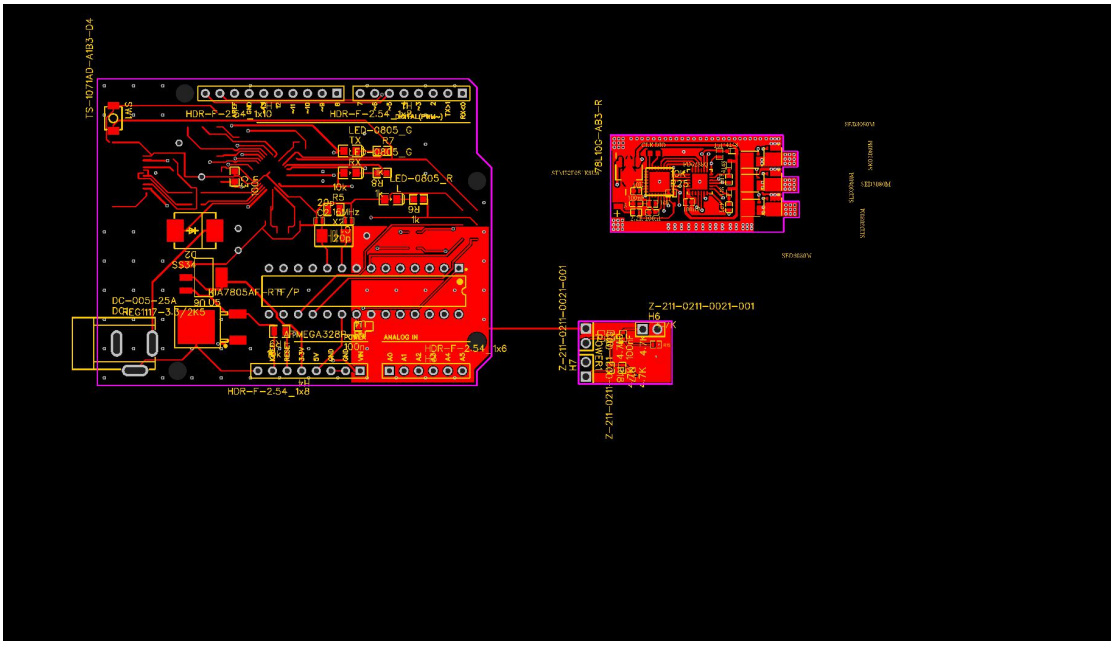
图 6 系统实物图

五、课程小结

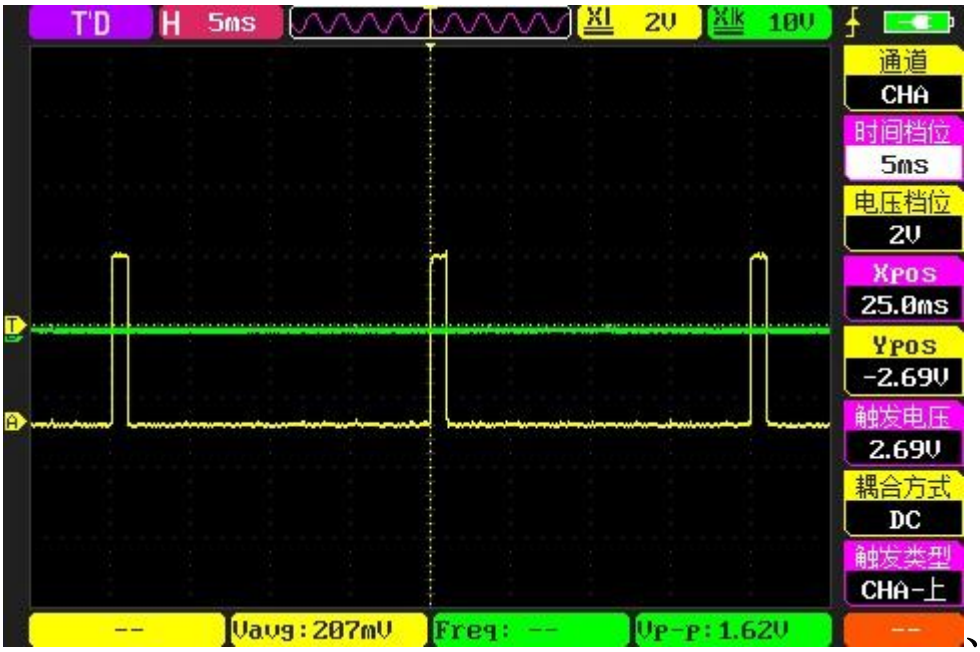
通过单片机入门课程的学习，我了解和掌握了 TIVA 和 51 系列单片机的基本功能和使用，对于自动控制原理的理解和运用有极大意义。我通过单片机实验的课程学习到了先进行系统构思，再进行原理图设计，然后在计算机上进行仿真，最后再实物测试这样的一个；流程，其中电脑仿真的过程调试速度快，反应精准，大大加快了开发的进度。我认为，只要精通了一种单片机，掌握了这样一套设计流程，再学习其他单片机是非常快的。

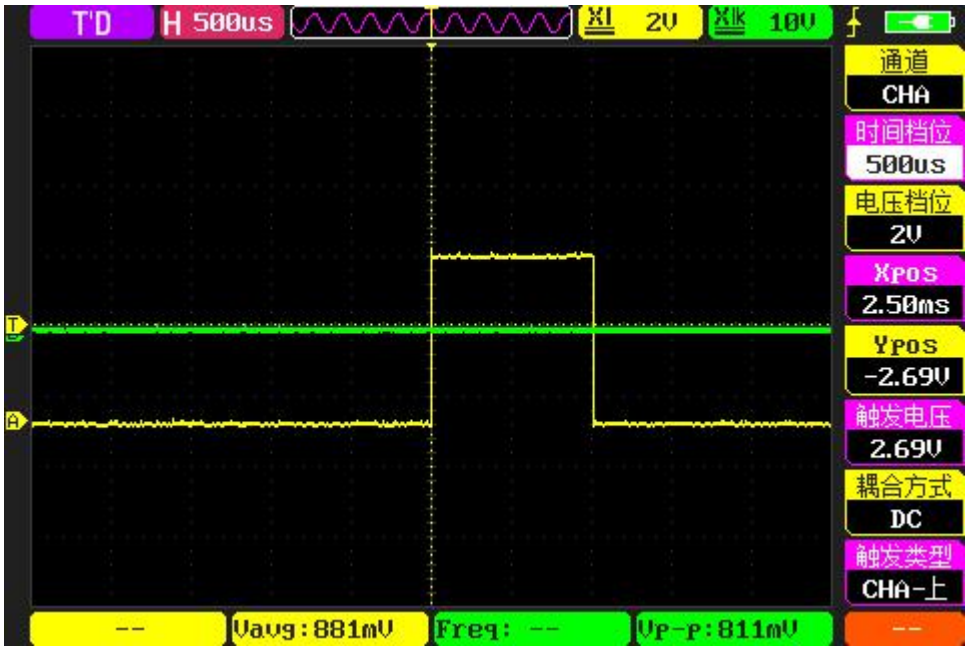
附录

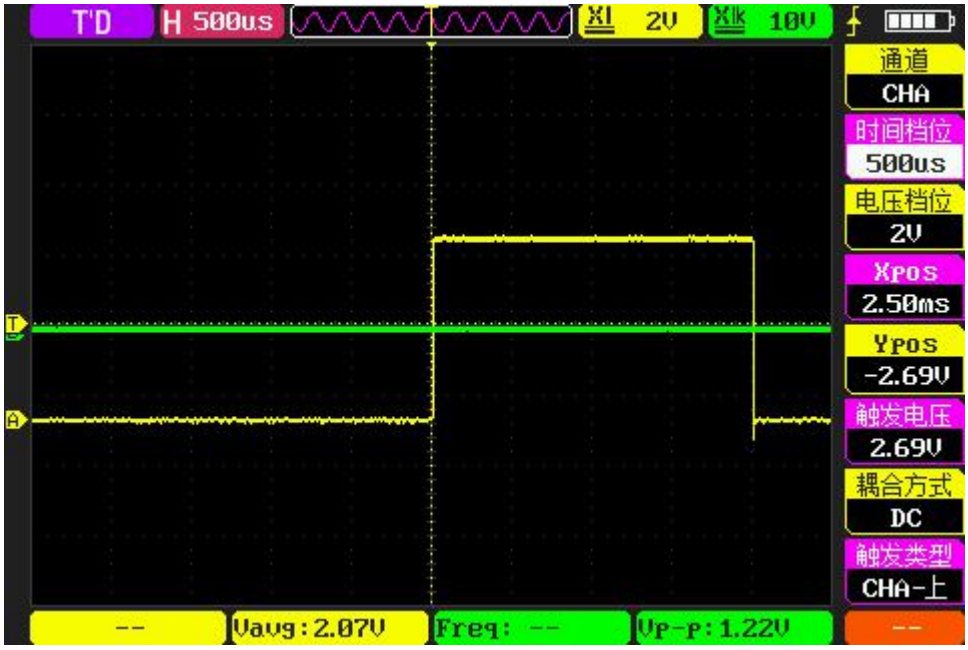
附录 1 电路 PCB 布局



附录 2 利用示波器测量输出 PWM 脉冲，最小脉冲及最大脉冲输出值







附录 3 电路芯片 BOM 表

ID	Name	Designator	Footprint	Quantity	Manufacturer Part
1	20p	C1,C2	C0402	2	
2	100n	C3,C4	C0603	2	
3	100nF	C5	C0402	1	CT41G-0603-2X1-50V-0.01uF-
4	100nF	C6,C7,C14	C0402	3	0603F104M500
5	10uF	C8,C9,C12,C13,C15	C1206	5	CL31A106KBHNNNE
6	4.7uF	C10	C0402	1	JMK105BBJ475MV-F
7	100nF	C11	C0402	1	CL05B104KO5NNNC
8	100nF	C16,C17	C0402	2	0402B104K500

9	1uF	C18,C19,C20	C0402	3	0402B104K500
10	4148	D1,D3,D4	SOD-523_L1.2-W0.8-LS1.6-RD	3	MMBD4148TS_R1_00001
11	SS34	D2	SMC_L7.1-W6.2-LS8.1-RD	1	SS34
12	DC-005-25A	DC1	DC-IN-TH_DC005	1	DC-005-25A
13	HDR-F-2.54_1x8	H1,H4	HDR-F-2.54_1X8	2	
14	HDR-F-2.54_1x10	H2	HDR-F-2.54_1X10	1	
15	HDR-F-2.54_1x6	H3	HDR-F-2.54_1X6	1	
16	Z-211-0211-0021-001	H6,H7,POWER1	HDR-TH_2P-P2.54-V	3	Z-211-0211-0021-001
17	LED-0805_R	L	LED0805_RED	1	17-21SURC/S530-A3/TR8
18	SED3080M	Q1,Q2,Q3,Q4,Q5,Q6	DFN-8_L3.0-W3.0-P0.65-BL	6	SED3080M
19	2.2K	R1	R0402	1	RTT0210R7FTH
20	10K	R2,R41,R44	R0402	3	AC0402JR-0710KL
21	10	R3,R4,R38,R39,R42,R43	R0402	6	AC0402JR-0710KL
22	10k	R5,R9	R0805	2	
23	1k	R6,R7,R8	R0603	3	
24	10K	R12,R13	R0402	2	RTT0210R7FTH
25	4.7K	R15,R16,R17,R18	R0402	4	0402WGF4701TCE
26	10K	R21,R22,R23,R24,R25,R26	R0402	6	RS-05K470JT
27	50k	R27,R28,R29	R0402	3	AC0402JR-0710KL
28	LED-0805_G	RX,TX	LED0805_GREEN	2	NCD0805G1
29	TS-1071AD-A1B3-D4	SW1	SW-SMD_L4.0-W3.0-LS5.0	1	TS-1071AD-A1B3-D4
30	STM32F051K8U6	U1	UFQFPN-32_L5.0-W5.0-P0.50-TL	1	STM32F051K8U6
31	VL53L0CXV0DH/1	U2	OPTO-SMD_12P_VL53L0CXV0DH/1	1	VL53L0CXV0DH/1
32	LR7533-T	U3	SOT-23-3_L2.9-W1.6-P1.90-LS2.8-BR	1	LR7533-T
33	ATMEGA328P-PU	U4	DIP-28_L34.6-W7.3-P2.54-LS10.2-BL	1	ATMEGA328P-PU
34	KIA7805AF-RTF/P	U5	TO-252-2_L6.6-W6.1-P4.57-LS9.9-BR-CW	1	KIA7805AF-RTF/P

35	REG1117-3.3/2K5	U6	SOT-223_L6.5-W3.5-P2.30-LS7.0-BR	1	REG1117-3.3/2K5
36	78L10G-AB3-R	U7	SOT-89-3_L4.5-W2.5-P1.50-LS4.2-BR	1	78L10G-AB3-R
37	FD6288Q	U8	QFN-24_L4.0-W4.0-P0.50-BL-EP2.6	1	FD6288Q
38	16MHz	X2	OSC-SMD_L5.0-W3.2	1	X503216MSB2GI

附录 4 项目文件地址

本项目文件开发过程中的建模，代码，原理图设计均已开源上传至 github
地址：https://github.com/00make/wind-tunnel