**电路与电子III**

**课程报告**

肌电信号的采集与分析

学 院： 未来技术学院

专业班级： 智能机器与平台2班

姓 名： 刘宇轩 张佳瑜 冒惠敏

郭伟祺 罗松寒 白 韬

学 号： 3020209107 3020233028

3020208068 3020001089

3020202153 3020234081

**未来技术学院**

**二零二二年 8月**

摘要

干电极肌电传感器是一种主动感应传感器，能提供高质量的信号搜集，且易于使用。不论是被用到静态还是动态的应用领域，仅需要一些极为简单的准备工作即可。本项目使用干电极导联，无须导电凝胶也可得到良好的信号质量，因此具有寿命长、使用简单方便等特点，更适合普通用户；而采用凝胶探头的医用电极通常为一次性，使用起来较为麻烦。干电极肌电传感器的测量具有非侵入性、无创伤、操作简单等优点，可用于人机交互等相关应用。虽然测量肌肉活动历来被用于医学研究，然而随着不断缩小但功能更强大的微控制器和集成电路的完善，肌电图电路和传感器也逐渐被应用于各种控制系统。  
 干电极肌电传感器集成了滤波、放大电路，并通过差分输入、模拟滤波电路的方式对噪音（特别是工频干扰）进行有效抑制。输出信号为模拟量形式输出，输出信号的大小取决于选定肌肉的活动量，输出信号的波形可显著指示被观察位置皮下肌肉的情况，方便做肌电信号的分析与研究，如使用Arduino作为控制器检测肌肉活动情况，如肌肉是否紧绷，强度如何，是否疲劳等。

在基于EMG肌电信号测量的肌肉强度检测装置中，用arduino连接舵机、蜂鸣器等配件，可以根据串口绘图器中肌电信号的强度来设置阈值，作为肌肉恢复训练的强度标准。若受试者的肌肉强度恢复达到预期，则装置连接的蜂鸣器会播放乐曲，同时相连的舵机会随机转动到一个角度，实现对肌肉恢复成功的受试者的奖励抽取。

在基于EMG肌电信号测量的可用于临床陪护呼叫装置中，患者在卧床治疗或静脉输液过程中，可能不方便独自移动并呼叫陪护人员时，可以使用该装置。设定较低的阈值后，患者可以通过连续握拳两次，串口监视器检测到连续的两次高电平后会通过arduino传输给护士站的监视装置，对应的lcd显示屏会显示患者的编号，同时蜂鸣器会提醒护士陪护。

**关键词：表面肌电信号；手势肌电识别；**

**目录**

[第一章 研究背景 1](#_Toc29472)

[第二章 项目规划 3](#_Toc6441)

[2.1 研究目标 3](#_Toc19119)

[2.2 实施计划 3](#_Toc31336)

[2.3 成员及分工 4](#_Toc20686)

[2.4 进度安排 4](#_Toc17814)

[第三章 硬件设计与实现 6](#_Toc17733)

[3.1硬件介绍 6](#_Toc6740)

[3.2技术规格 7](#_Toc5180)

[3.3硬件配置 7](#_Toc6906)

[3.4接线图设计 7](#_Toc19029)

[第四章 软件设计与编程 9](#_Toc20145)

[4.1软件设计 9](#_Toc12991)

[4.2程序设计 9](#_Toc31201)

[4.3仿真及测试结果 16](#_Toc13964)

[第五章 产品集成与功能测试 18](#_Toc18053)

[5.1外观及结构设计 18](#_Toc4118)

[5.2 功能测试 18](#_Toc9377)

[参考文献 19](#_Toc14906)

[附录 20](#_Toc19187)

第一章 研究背景

表面肌电信号是一种由于肌肉收缩在皮肤面产生的微弱电信号，其中包含有丰富的肢体运动信息，且该信号可以方便通过无创的方式采集，在现代社会中有着极其丰富的应用场景。由于表面肌电信号的安全性、自然性和直接性的特点，基于表面肌电信号新颖的人机交互控制方式已经渐渐取代了鼠标、键盘等外部设备的交互，并且广泛应用于康复治疗、假肢控制、机器人控制和助老助残等领域。本项目将从医疗领域和娱乐抽奖活动两个方面探讨表面肌电信号的应用。干电极肌电传感器通过检测人体的表面肌电信号（sEMG），进而反应出人体肌肉和神经的活动情况。

在现代的社会中,颈椎病、颈肩痛、背痛已是许多疾病过程中的常见症状,也严重影响着人们的生活和工作。减轻以致消除这些疾病一直是医学界长期探讨的内容。表面肌电信号分析技术在近几年来得到了日新月异的发展,也日渐的完善了颈肩背部肌肉功能评价方法。这些方法能实时地、准确地和在非损伤状态下反映肌肉活动状态和功能状态,且在评价肌肉功能状态方面具有良好的特异性、灵敏性、局部性的特点。研究肌电信号的目的主要在于通过研究时、频域特征以及肌肉活动状态和功能状态之间的关联性,用来探讨影响信号变化的可能原因及应用信号的变化可有效反映肌肉的活动和功能。

表面肌电信号实质上是多个运动单位因兴奋而发放出来的运动单位,这些运动单元序列沿肌纤维传播并经由皮肤、脂肪构成的容积导体滤波后在皮肤表面检

测电极处时间和空间上综合叠加的结果。肌肉收缩时能产生电,这是早就为人所知的事情。早在年加伐尼就通过一系列蛙类的肌肉收缩实验研究,发现了肌肉的收缩可引起一种生物电的产生,这种生物电就是我们所研究的肌电信号。后来到年的时候,法国的杜波依斯·雷蒙德·最先检测到人体肌肉自愿收缩时能产生电信号,也就是肌电信号发展的开端。一直到年,派帕用检流计测量到人臂肌肉可产生电势差。到年加塞和厄兰格才`用阴极射线示波器观察到了肌电。世纪中叶以来,随着电子技术和电子计算机的出现和迅猛发展和肌电信号处理技术的进步提高,以及对神经肌肉生理学的快速进展,人们对肌电信号的检测、记录技术和神经肌肉功能的状态评价进行了细致的研究。复杂的信号分析技术的发展和微处理器的出现使得肌电信号的数值分析成为可能。这些都会使肌电检测更加敏锐、

可靠,且可以实现某些程度的可复现性。肌电信号检测处理及分析在越来越多领域的的广泛应用真正的为多方面的研究奠定了良好的基础。

第二章 项目规划

2.1 研究目标

基于arduino平台或生理信号采集芯片，独立设计、焊接、调试电路板，独立开发、调试控制分析程序，实现肌电信号的采集与分析，并建立高准确率的肌电手势判断模型。

2.2 实施计划

干电极肌电传感器通过检测人体的表面肌电信号（sEMG），进而反应出人体肌肉和神经的活动情况。干电极肌电传感器集成了滤波、放大电路，将范围在±1.5mV内的微弱人体表面肌电信号进行1000倍放大，并通过差分输入、模拟滤波电路的方式对噪音（特别是工频干扰）进行有效抑制。输出信号为模拟量形式，以1.5V为基准电压，0~3.0V量程的输出。[1]输出信号的大小取决于选定肌肉的活动量，输出信号的波形可显著指示被观察位置皮下肌肉的情况，方便做肌电信号的分析与研究，如使用Arduino作为控制器检测肌肉活动情况，如肌肉是否紧绷，强度如何，是否疲劳等[2]。

干电极肌电传感器的测量具有非侵入性、无创伤、操作简单等优点，可用于人机交互等相关应用。虽然测量肌肉活动历来被用于医学研究，然而随着不断缩小但功能更强大的微控制器和集成电路的完善，肌电图电路和传感器也逐渐被应用于各种控制系统[3]。

本项目中我们使用肌电传感器的信号模拟肌肉恢复训练后的肌肉强度检测和抽奖装置，以及可用于临床的陪护呼叫装置。

在基于EMG肌电信号测量的肌肉强度检测装置中，用arduino连接舵机、蜂鸣器等配件，可以根据串口绘图器中肌电信号的强度来设置阈值，作为肌肉恢复训练的强度标准。若受试者的肌肉强度恢复达到预期，则装置连接的蜂鸣器会播放乐曲，同时相连的舵机会随机转动到一个角度，实现对肌肉恢复成功的受试者的奖励抽取。

在基于EMG肌电信号测量的可用于临床陪护呼叫装置中，患者在卧床治疗或静脉输液过程中，可能不方便独自移动并呼叫陪护人员时，可以使用该装置。设定较低的阈值后，患者可以通过连续握拳两次，串口监视器检测到连续的两次高电平后会通过arduino传输给护士站的监视装置，对应的lcd显示屏会显示患者的编号，同时蜂鸣器会提醒护士陪护。

护士接收后，即可暂停呼叫装置，在响应后可以恢复该装置的运行。

2.3 成员及分工

2.3.1 成员

项目负责人：刘宇轩

项目组成员：罗松寒、郭伟祺、张佳瑜、冒惠敏、白韬

2.3.2 分工

表2.1 分工

|  |  |
| --- | --- |
| **工作事项** | **主要负责人** |
| 项目规划 | 刘宇轩 |
| 方案制定 | 刘宇轩 |
| 进度安排 | 刘宇轩 |
| 原理图设计仿真 | 全体成员 |
| 项目设计 | 刘宇轩 |
| 硬件购买 | 白韬 |
| 代码编写 | 刘宇轩、罗松寒 |
| 硬件连接 | 张佳瑜 |
| 接口及连线设计 | 郭伟祺 |
| 外观设计 | 冒惠敏 |
| 产品集成 | 刘宇轩、罗松寒 |
| 功能测试 | 刘宇轩、郭伟祺、白韬 |
| 可靠性分析 | 张佳瑜、冒惠敏 |

2.4 进度安排

1. 第一阶段 2022.6.1

1.1. 项目规划、目标、方案制定

1.2. 成员分工、进度安排

2. 第二阶段 2022.6.3

2.1. 原理图设计仿真

3. 第三阶段 2022.6.5

3.1. 项目设计

4. 第四阶段 2022.6.10

4.1. 硬件购买

4.2. 代码编写

5. 第五阶段 2022.8.27

5.1. 代码编写

5.2. 接口及连线设计

5.3. 硬件连接

5.4. 外观设计

6. 第六阶段 2022.8.30

6.1. 产品集成

6.2. 功能测试

6.3. 可靠性分析

第三章 硬件设计与实现

**3.1硬件介绍**

干电极肌肉传感器：干电极肌电传感器通过检测人体的表面肌电信号（sEMG），进而反应出人体肌肉和神经的活动情况，如图1所示。

干电极肌电传感器集成了滤波、放大电路，将范围在±1.5mV内的微弱人体表面肌电信号进行1000倍放大，并通过差分输入、模拟滤波电路的方式对噪音（特别是工频干扰）进行有效抑制。输出信号为模拟量形式，以1.5V为基准电压，0～3.0V量程的输出。输出信号的大小取决于选定肌肉的活动量，输出信号的波形可显著指示被观察位置皮下肌肉的情况，方便做肌电信号的分析与研究。使用Arduino作为控制器检测肌肉活动情况，如肌肉是否紧绷，强度如何，是否疲劳等。

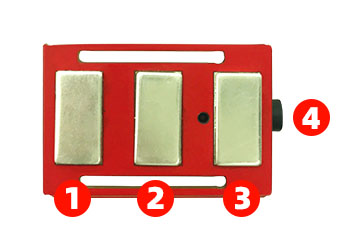
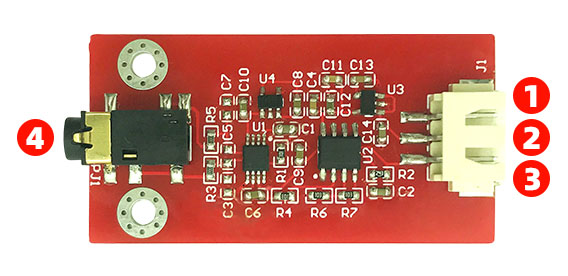


图3.1 干电极肌肉传感器

Arduino UNO控制板: Arduino生态中包括多种[开发板](https://so.csdn.net/so/search?q=%E5%BC%80%E5%8F%91%E6%9D%BF&spm=1001.2101.3001.7020)、模块、扩展板。其中UNO R3是最适合入门且功能齐全使用量最多的Arduino开发板，如图2所示。

Arduino UNO是Arduino USB接口系列的最新版本，它的处理核心是ATMEGA328P。它有14个数字输入/输出引脚(其中6个可用作PWM输出)，6个模拟输入，16MHz晶振时钟，USB连接，电源插孔，ICSP接头和复位按钮。只需要通过USB数据线连接电脑就能供电、程序下载和数据通讯。

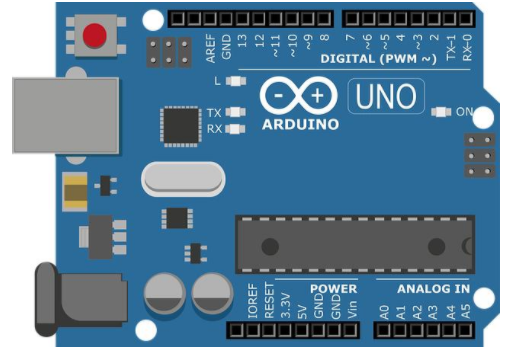
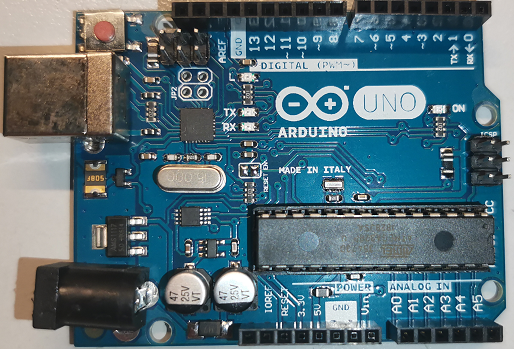


图3.2 Arduino UNO控制板

**3.2技术规格**

供电电压：+3.3V～5.5V

输出电压：0~3.0V

检测范围：+/-1.5mV

电极接口：PJ-342

模块接口：XH2.54-3P

输出范围：0～3.0V

工作温度：0～50℃

板子尺寸：25\* 48 mm

电极接口：PJ-342

电极线长：50cm

板子尺寸：23 \* 35 mm

**3.3硬件配置**

表3.1 硬件配置表

|  |  |
| --- | --- |
| 器件名称 | 数量 |
| Arduino UNO控制板 | 1 |
| 肌电传感器信号处理板 | 1 |
| 肌电传感器干电极 | 1 |
| 干电极连接线 | 1 |
| 180°舵机 | 1 |
| 扩展板 | 1 |
| LCD液晶显示屏 | 1 |
| 蜂鸣器 | 1 |
| 9V方块干电池  杜邦线 | 1  若干 |

**3.4接线图设计**

如图3所示，将电源输入正极连接Arduino UNO控制板的VCC端口，将电源输入负极连接Arduino UNO控制板的GND端口，将模拟信号输出端连接Arduino UNO控制板的Analog Signal端口，将两个PJ-342干电极接口相连。之后，通过在Arduino UNO控制板中其他接口接入舵机、蜂鸣器、LCD液晶显示屏等器件实现预期功能。

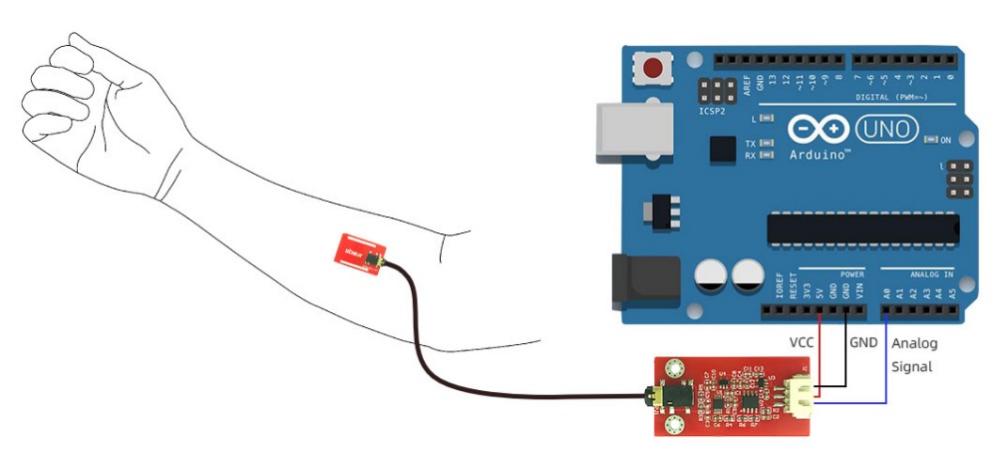


图3.3 接线图设计

**第四章 软件设计与编程**

4.1软件设计

4.1.1基于EMG肌电信号测量的可用于临床陪护呼叫装置

在临床陪护呼叫装置软件中，通过串口绘图器中肌电信号的强度来设置阈值。当接收到的肌电信号强度超过阈值两次，即串口监视器检测到连续两次的高电平后，系统对检测结果做出回应，即lcd显示屏显示对应编号，同时蜂鸣器运行进行提醒。

4.1.2基于EMG肌电信号测量的肌肉强度检测装置

在肌肉强度检测装置软件中，通过串口绘图器中肌电信号的强度来设置阈值。当接收到的肌电信号强度超过阈值，即串口监视器检测到高电平后，系统对检测结果做出回应，即舵机随机旋转某一角度，同时蜂鸣器运行进行提醒。

4.2程序设计

4.2.1 临床陪护呼叫装置的Arduino控制程序设计

代码：

#if defined(ARDUINO) && ARDUINO >= 100

#include <Servo.h>

#include "Arduino.h"

#else

#include "WProgram.h"

#endif

#include "EMGFilters.h"

#define SensorInputPin A0

#include <Wire.h>

#include "rgb\_lcd.h"

unsigned long threshold = 150;

unsigned long EMG\_num = 0;

EMGFilters myFilter;

Servo servo1;

SAMPLE\_FREQUENCY sampleRate = SAMPLE\_FREQ\_500HZ;

NOTCH\_FREQUENCY humFreq = NOTCH\_FREQ\_50HZ;

rgb\_lcd lcd;

int speakerPin = 7;

int length = 15;

int tempo = 300;

void setup()

{

myFilter.init(sampleRate, humFreq, true, true, true);

Serial.begin(115200);

pinMode(speakerPin, OUTPUT);

randomSeed(analogRead(A0));

servo1.attach(3);

pinMode(A3,INPUT);

lcd.begin(16, 2);

pinMode(speakerPin, OUTPUT);

lcd.print("No patient is");

lcd.setCursor(0,2);

lcd.print("waiting");

delay(1000);

}

void loop()

{

int data = analogRead(SensorInputPin);

int dataAfterFilter = myFilter.update(data);

int envelope = sq(dataAfterFilter);

envelope = (envelope > threshold) ? envelope : 0;

if (threshold > 0)

{

if (getEMGCount(envelope))

{ EMG\_num++;

Serial.print("EMG\_num: ");

Serial.println(EMG\_num);

lcd.setCursor(0, 1);

if(EMG\_num>=2){

lcd.clear();

lcd.setRGB(227,23,13);

lcd.print("No.3 patient");

lcd.setCursor(0,2);

lcd.print("call");

if((millis()/1000)>10){

digitalWrite(speakerPin, HIGH);

}

}

}

}

else {

Serial.println(envelope);

}

delayMicroseconds(500);

}

int getEMGCount(int gforce\_envelope)

{

static long integralData = 0;

static long integralDataEve = 0;

static bool remainFlag = false;

static unsigned long timeMillis = 0;

static unsigned long timeBeginzero = 0;

static long fistNum = 0;

static int TimeStandard = 200;

integralDataEve = integralData;

integralData += gforce\_envelope;

if ((integralDataEve == integralData) && (integralDataEve != 0))

{

timeMillis = millis();

if (remainFlag)

{

timeBeginzero = timeMillis;

remainFlag = false;

return 0;

}

if ((timeMillis - timeBeginzero) > TimeStandard)

{

integralDataEve = integralData = 0;

return 1;

}

return 0;

}

else {

remainFlag = true;

return 0;

}

}

4.2.2 肌肉强度检测装置的Arduino控制程序设计

代码：

#if defined(ARDUINO) && ARDUINO >= 100

#include <Servo.h>

#include "Arduino.h"

#else

#include "WProgram.h"

#endif

#include "EMGFilters.h"

#define SensorInputPin A0

unsigned long threshold = 50;

unsigned long EMG\_num = 0;

EMGFilters myFilter;

Servo servo1;

SAMPLE\_FREQUENCY sampleRate = SAMPLE\_FREQ\_500HZ;

NOTCH\_FREQUENCY humFreq = NOTCH\_FREQ\_50HZ;

int speakerPin = 7;

int length = 15;

char notes[] = "ccggaagffeeddc ";

int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 };

int tempo = 300;

void playTone(int tone, int duration) {

for (long i = 0; i < duration \* 1000L; i += tone \* 2) {

digitalWrite(speakerPin, HIGH);

delayMicroseconds(tone);

digitalWrite(speakerPin, LOW);

delayMicroseconds(tone);

}

}

void playNote(char note, int duration) {

char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };

int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

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

if (names[i] == note) {

playTone(tones[i], duration);

}

}

}

void setup()

{

myFilter.init(sampleRate, humFreq, true, true, true);

Serial.begin(115200);

pinMode(speakerPin, OUTPUT);

randomSeed(analogRead(A0));

servo1.attach(3);

pinMode(A3,INPUT);

}

void loop()

{

int data = analogRead(SensorInputPin);

int dataAfterFilter = myFilter.update(data);

int envelope = sq(dataAfterFilter);

envelope = (envelope > threshold) ? envelope : 0;

if (threshold > 0)

{

if (getEMGCount(envelope))

{

EMG\_num++;

Serial.print("EMG\_num: ");

Serial.println(EMG\_num);

int a;

a = random(512,1024);

a = map(a,0,1024,0,180);

servo1.write(a);

for (int i = 0; i < length; i++)

{

if (notes[i] == ' ')

{

delay(beats[i] \* tempo); // rest

}

else

{

playNote(notes[i], beats[i] \* tempo);

}

delay(tempo / 2);

}

}

}

else {

Serial.println(envelope);

}

delayMicroseconds(500);

}

int getEMGCount(int gforce\_envelope)

{

static long integralData = 0;

static long integralDataEve = 0;

static bool remainFlag = false;

static unsigned long timeMillis = 0;

static unsigned long timeBeginzero = 0;

static long fistNum = 0;

static int TimeStandard = 200;

integralDataEve = integralData;

integralData += gforce\_envelope;

if ((integralDataEve == integralData) && (integralDataEve != 0))

{

timeMillis = millis();

if (remainFlag)

{

timeBeginzero = timeMillis;

remainFlag = false;

return 0;

}

if ((timeMillis - timeBeginzero) > TimeStandard)

{

integralDataEve = integralData = 0;

return 1;

}

return 0;

}

else {

remainFlag = true;

return 0;

}

}

4.3仿真及测试结果

4.3.1肌电波形图

通过握拳动作与松拳动作来检测肌电信号的采集效果。在实验过程中，实验者做松拳动作以达到肌肉放松的状态，如图4-1，测量肌肉放松状态下的肌电信号波形图，如图4-2；实验者做握拳动作以达到肌肉紧绷的状态，如图4-3，测量肌肉紧绷状态下的肌电信号波形图，如图4-4。

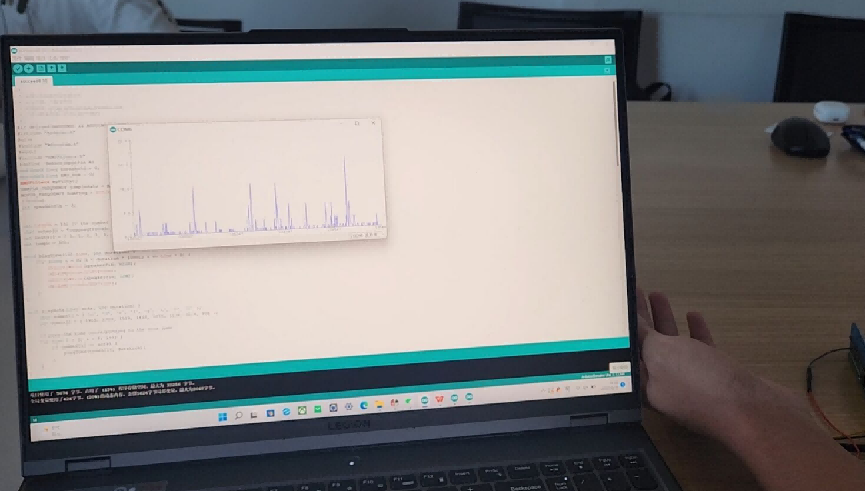


图4.1 实验者肌肉放松状态

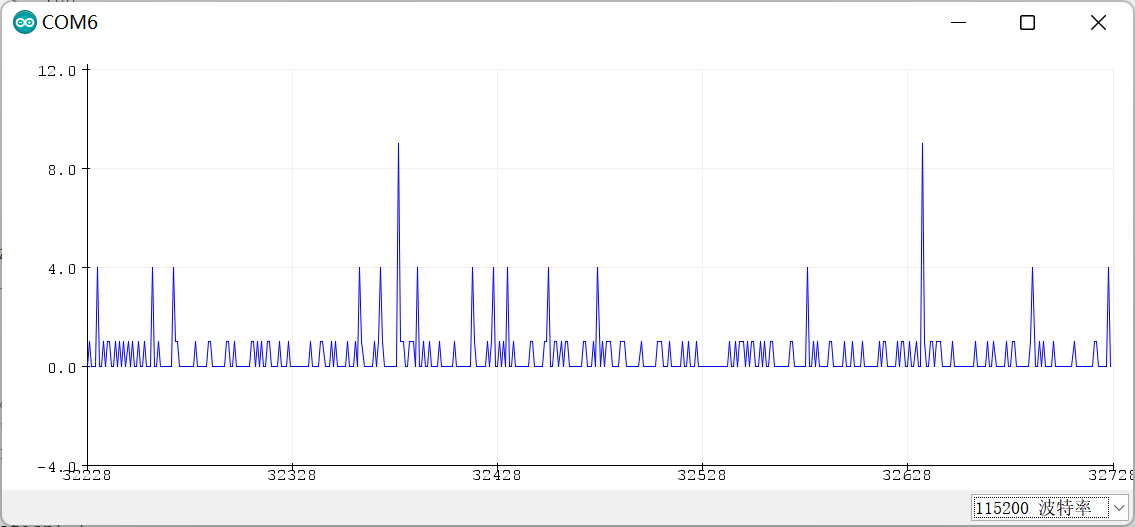


图4.2 肌肉放松状态下的波形图

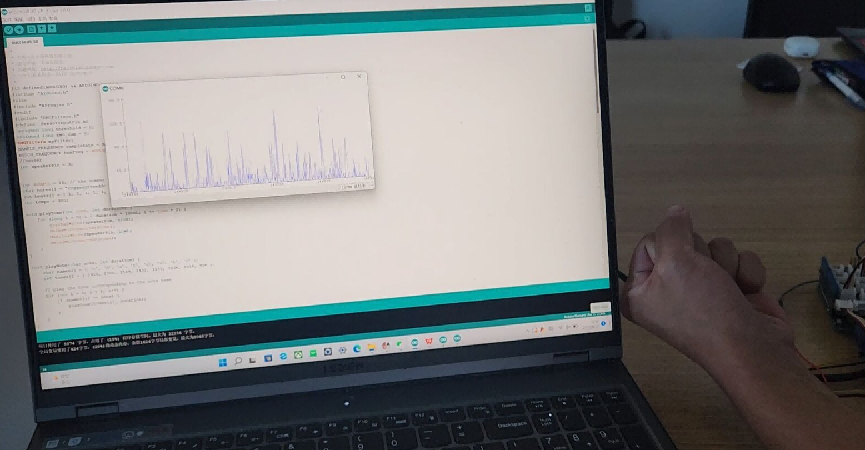


图4.3 实验者肌肉紧绷状态

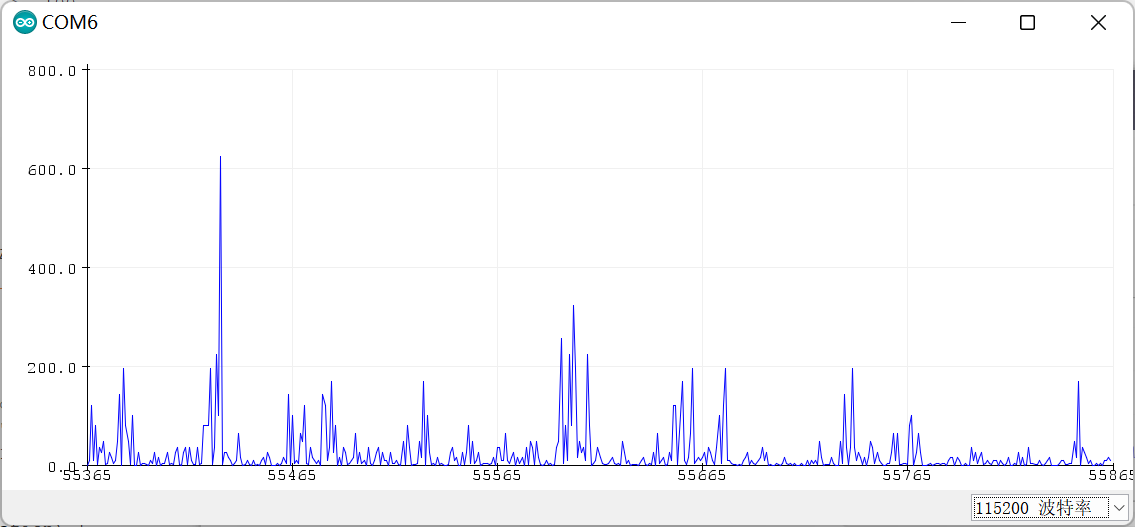
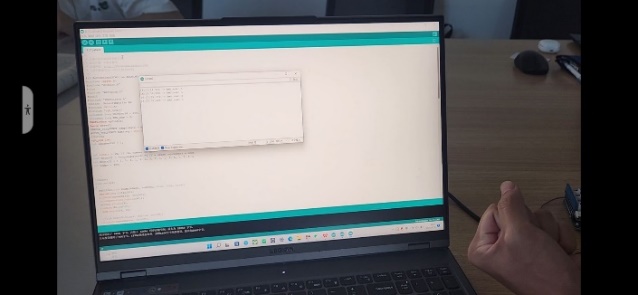


图4.4 肌肉紧绷状态下的波形图

4.3.2对握拳次数的检测

检测时，用力握拳并松开一次，可计为握拳一次。在实验过程中，实验者多次握拳并松开，通过串口监视器观察肌电信号接收次数，以测量实验对象握拳次数，如图4-5。



（a） （b）

图4.5 握拳次数的测量

**第五章 产品集成与功能测试**

**5.1外观及结构设计**

**5.1.1基于EMG肌电信号测量的可用于临床陪护呼叫装置**

在临床陪护呼叫装置中，本组设计传输、显示、报警三个模块，分别实现了肌肉电信号感知与信号传入、显示当前状态（未呼叫时为“No patient’s waiting”、已呼叫时为“NO.X patient call”）、呼叫状态下的报警功能，结构轻巧，外形灵动，功能齐全，成本低廉，适合在医院进行大规模推广。

**5.1.2基于EMG肌电信号测量的肌肉强度检测装置**

在肌肉强度检测装置中，本组设计出传输、抽奖转盘、奏乐、旋转指针等模块，分别实现了肌肉电信号感知与信号传入、随机抽奖、奏乐祝贺等功能，结构清晰，功能多样，外观实用耐看，适用于健身房中肌肉强度的测量，如图1所示。

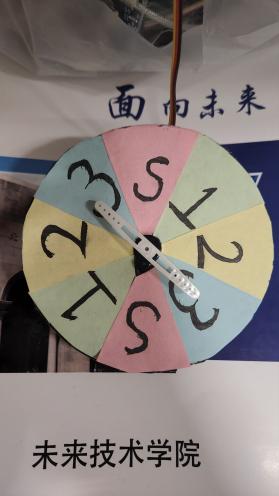


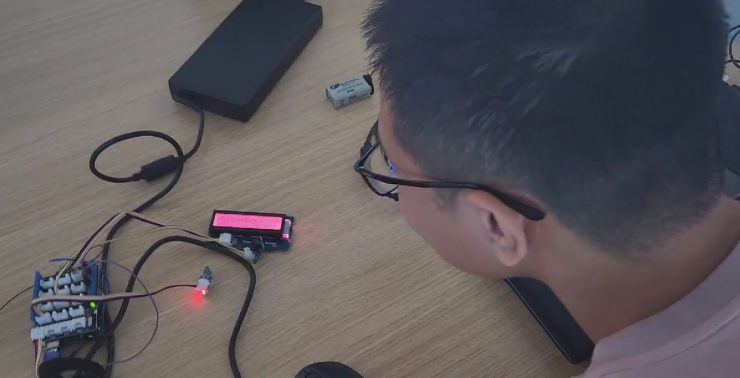
图5.1 肌肉强度检测装置抽奖转盘

**5.2 功能测试**

**5.2.1 临床陪护呼叫装置功能测试**

（1）报警功能

在患者通过肌肉紧绷进行呼叫后，临床陪护呼叫装置中蜂鸣器发出响声，提示护士前往显示器处查看患者呼叫床位，如图2所示。

图5.2 报警功能示意图

（2）提示功能

在患者未呼叫的时段内，显示当前状态，背景设置为蓝色。在患者呼叫的时段内，显示患者所在床位，背景设置为红色，旨在体现出呼叫的紧急性，如图3所示。

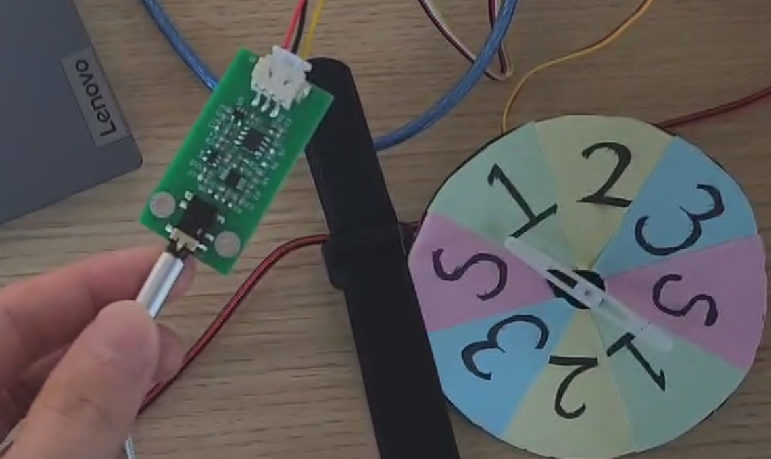


图5.3 显示屏状态显示示意图

**5.2.2 肌肉强度检测装置功能测试**

（1）抽奖功能

受试者肌肉绷紧，若肌肉强度达到预期设定的阈值，则转盘指针迅速转动，随机生成一项奖项，实现抽奖功能，如图4所示。

图5.4 抽奖结果示意图

（2）奏乐功能

在肌肉强度测试过程中，为增强其娱乐性与趣味性，本测试装置增设奏乐模块，能够在抽奖过程中进行演奏音乐，恭喜受试者肌肉强度达到设定标准。

**参考文献**

1. 陈思佳,罗志增.基于长短时记忆和卷积神经网络的手势肌电识别研究[J/OL].仪器仪表学报:1-10[2021-04-03].
2. 罗利梦,许芷毓,谢晓辉,李磊.基于卷积神经网络的表面肌电信号手势识别[J].电脑编程技巧与维护,2021(01):137-138+163.
3. 赵章琰.表面肌电信号检测和处理中若干关键技术研究［D］.合肥：中国科学技术大学，2010.

**附录**

测试代码1：舵机部分测试代码

#include <Servo.h>

Servo servo1;

void setup(){

Serial.begin(115200);

randomSeed(analogRead(A0));

servo1.attach(3);

pinMode(A3,INPUT);

}

void loop(){

int a;

a = random(1024);

a = map(a,0,1024,0,180);

Serial.println(a);

servo1.write(a);

delay(10000);

}

测试代码2：肌电部分测试代码

#if defined(ARDUINO) && ARDUINO >= 100

#include "Arduino.h"

#else

#include "WProgram.h"

#endif

#include "EMGFilters.h"

#define SensorInputPin A0

unsigned long threshold = 0;

unsigned long EMG\_num = 0;

EMGFilters myFilter;

SAMPLE\_FREQUENCY sampleRate = SAMPLE\_FREQ\_500HZ;

NOTCH\_FREQUENCY humFreq = NOTCH\_FREQ\_50HZ;

void setup()

{

myFilter.init(sampleRate, humFreq, true, true, true);

Serial.begin(115200);

}

void loop()

{

int data = analogRead(SensorInputPin);

int dataAfterFilter = myFilter.update(data);

int envelope = sq(dataAfterFilter);

envelope = (envelope > threshold) ? envelope : 0;

if (threshold > 0)

{

if (getEMGCount(envelope))

{ EMG\_num++;

Serial.print("EMG\_num: ");

Serial.println(EMG\_num);

}

}

else {

Serial.println(envelope);

}

delayMicroseconds(500);

}

int getEMGCount(int gforce\_envelope)

{

static long integralData = 0;

static long integralDataEve = 0;

static bool remainFlag = false;

static unsigned long timeMillis = 0;

static unsigned long timeBeginzero = 0;

static long fistNum = 0;

static int TimeStandard = 200;

integralDataEve = integralData;

integralData += gforce\_envelope;

if ((integralDataEve == integralData) && (integralDataEve != 0))

{

timeMillis = millis();

if (remainFlag)

{

timeBeginzero = timeMillis;

remainFlag = false;

return 0;

}

if ((timeMillis - timeBeginzero) > TimeStandard)

{

integralDataEve = integralData = 0;

return 1;

}

return 0;

}

else {

remainFlag = true;

return 0;

}

}