

学生実験における新テーマの教材作成支援

— 筋電計回路の製作・実装および筐体製作 —

宇都宮大学 松本 恵音

機械システム工学実験

前期

	テーマ名
A	自動制御
B	薄板加工実験
C	(休止)
D	知能ロボットの制御
E	切削加工に関する実験
F	熱流体解析
G	材料の熱処理
H	(休止)
I	情報処理
J	バイオメカニクス実験

	A1班	A2班	A3班	A4班	A5班
1週目	A	B	C	D	E
2週目	自動制御	薄板加工	(休止)	知能ロボ	切削加工
3週目					
4週目	B	C	D	E	A
5週目					
6週目	C	D	E	A	B
7週目					
8週目	D	E	A	B	C
9週目					
10週目	E	A	B	C	D
11週目					
12週目					
13週目					
14週目					
15週目					

後期

	A1班	A2班	A3班	A4班	A5班
1週目	F	G	H	I	J
2週目	熱流体	熱処理	(休止)	情報処理	バイオメカ
3週目					
4週目	G	H	I	J	F
5週目					
6週目	H	I	J	F	G
7週目					
8週目	I	J	F	G	H
9週目					
10週目	J	F	G	H	I
11週目					
12週目					
13週目					
14週目					
15週目					

実施方法

- 学部3年生対象
- 全10テーマ(稼働8・休止2)
- 1テーマ3週×5テーマを15週でローテーション
- 一班およそ10人
- 前期・後期を通して全テーマ受講

バイオメカニクスと筋電計

バイオメカニクス(生体力学)

人の体の動きや力を、力学(機械工学)の考え方で調べる学問。筋肉の働きを測る道具のひとつが筋電計

筋電計とは

- 筋肉が動くときに生じる微小な電気信号(筋電位)を測る装置
- 脳からの運動指令が神経を通して筋肉に届くと、筋肉の中で電位変化が起きる
- この電位を皮膚の上の電極で拾う
- 信号は非常に小さい → 増幅(アンプ)回路

主な使い道

工学

筋電義手などのインターフェース

医療

診断・リハビリテーション

健康・スポーツ

筋肉の使われ方の評価

依頼の背景・目的

背景

- 研究室の学生は卒業で入れ替わる
- そのたびに回路作成を一から教える
必要があり、負担が大きい
- 知識・ノウハウが引き継がれにくい



目的

- 1 学ぶ**
回路作成の一連の流れを自ら学ぶ
- 2 教える**
学んだ内容を学生に教えられる
- 3 教材化**
成果を学生実験の教材づくりにつなげる

内容変更の背景・目的

背景

- 今までは市販の筋電計を使用していた
- 学生が個人で購入して自主的に学ぶには高価すぎる
- 市販品は中身が見えず、計測の仕組みを理解しにくい



目的

- 入手しやすい安価な部品で、学生自身が筋電計を製作する
- 自ら作ることで、計測の仕組みへの理解を深める
- 学生が自主的に学習を発展させられる

バイオメカニクス実験の流れ(全3週)

1週目

座学

筋電の仕組み、回路の基礎、増幅回路の原理を学ぶ

2週目

座学+配線

回路図を基にブレッドボードへ配線

3週目

計測

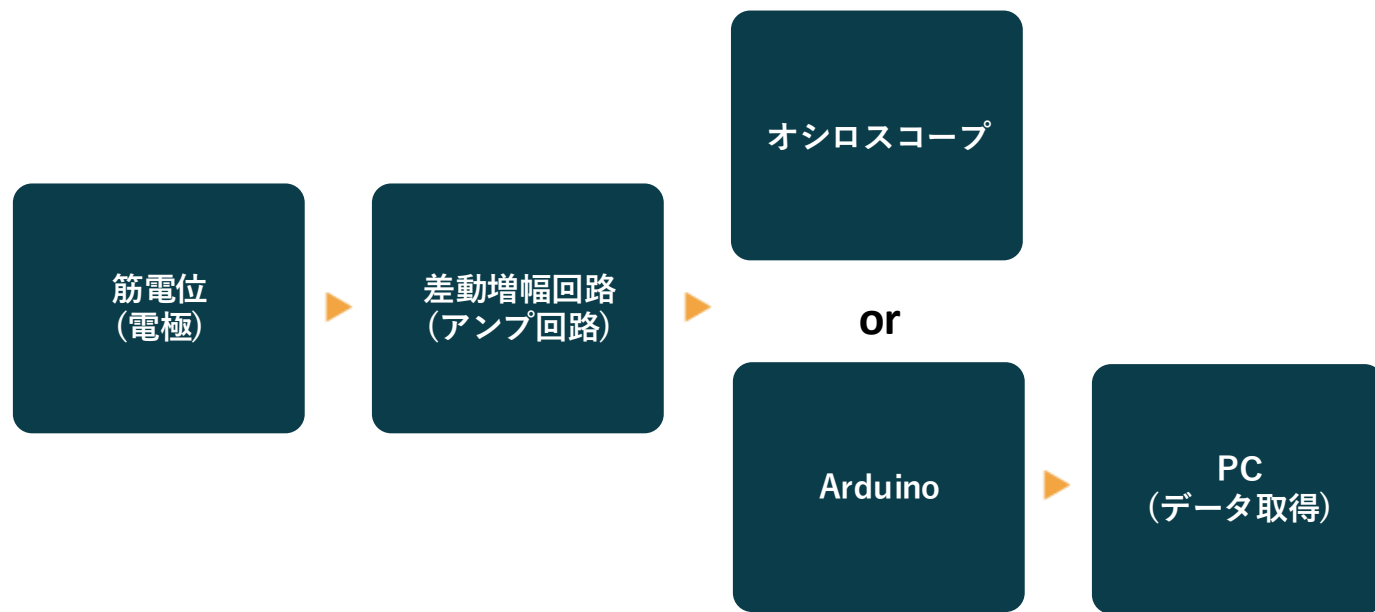
配線した回路で筋電を計測

レポート

解析

計測データを解析して提出

今回の筋電計システムの全体像



計測方法



① 使い捨て電極

ゲル付き。皮膚に貼る

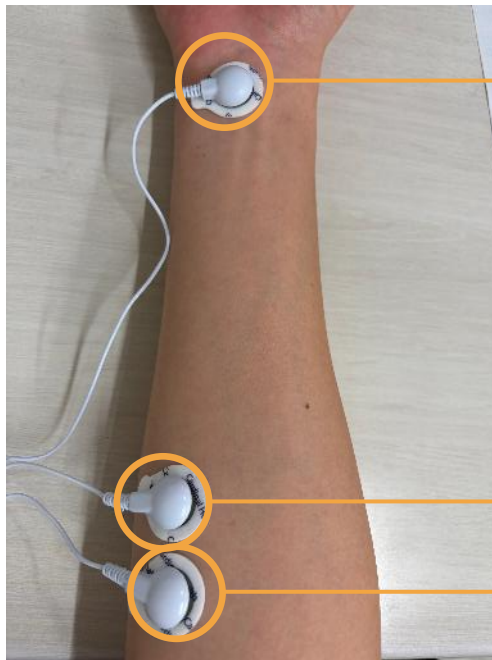


② ボタンクリップ

電極のボタンに、スナップで留める

計測条件

対象:前腕の筋(尺側手根屈筋)/ 弱・中・強
各条件5回、1回10秒



基準電極

記録電極 1

記録電極 2

③ 前腕への貼り付け位置

RIGOL

RUN

H

100.00ms

A

500.00kSa/s

1.00Mpts

D

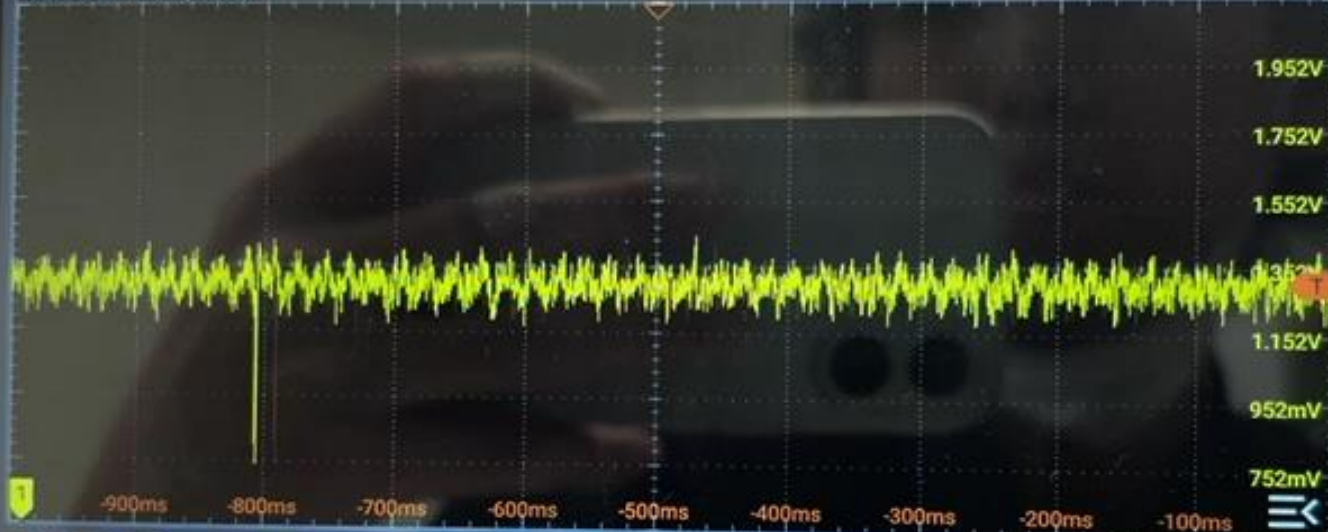
T

STOP

RUN

Measure

Waveform View(ROLL)

1 200.00mV/
-1.35V2 50.00mV/
0.00V3 50.00mV/
0.00V4 50.00mV/
0.00V

M

M1

M3

M2

M4



Intensity

POSITION

Zero

1

2

3

4



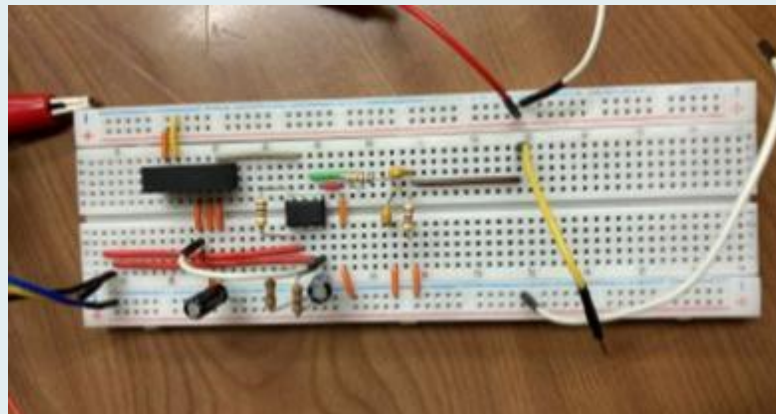
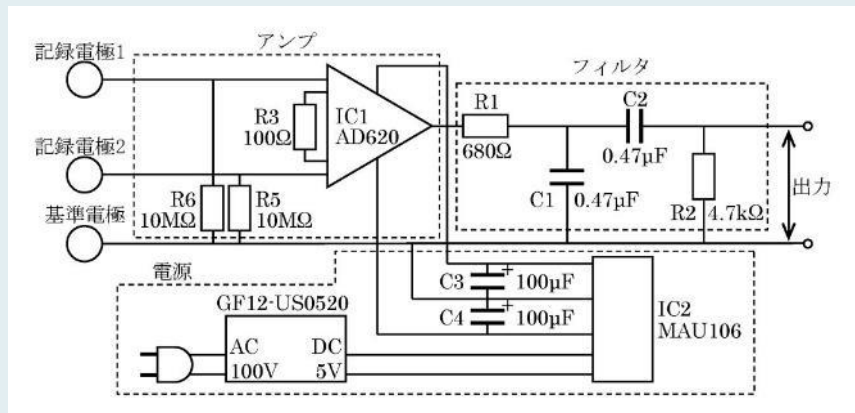
V

mV

試作と検証

- 参考文献を基に、ブレッドボードで回路の配線を行い、オシロスコープ及びArduino UNOで動作を検証

参考文献：[1] 敦賀健志, 市販の電子部品を利用した簡易的な筋電系の製作について, 日本義肢装具学会誌, Vol.35, No.1, p.24 – 30



問題点と解決策

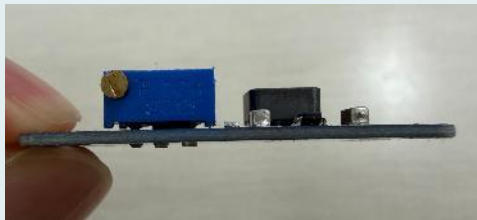
① 研究室の電源がノイズ源となり、ノイズが発生することが判明

- 電源を研究室の電源装置からリチウムイオンバッテリーに切り替え、ノイズを低減
- ※抵抗やコンデンサの足が長いとアンテナとなり、ノイズの原因となる



② 回路に必要な約5 Vに対し、バッテリーは3.7 V程度で電圧が不足

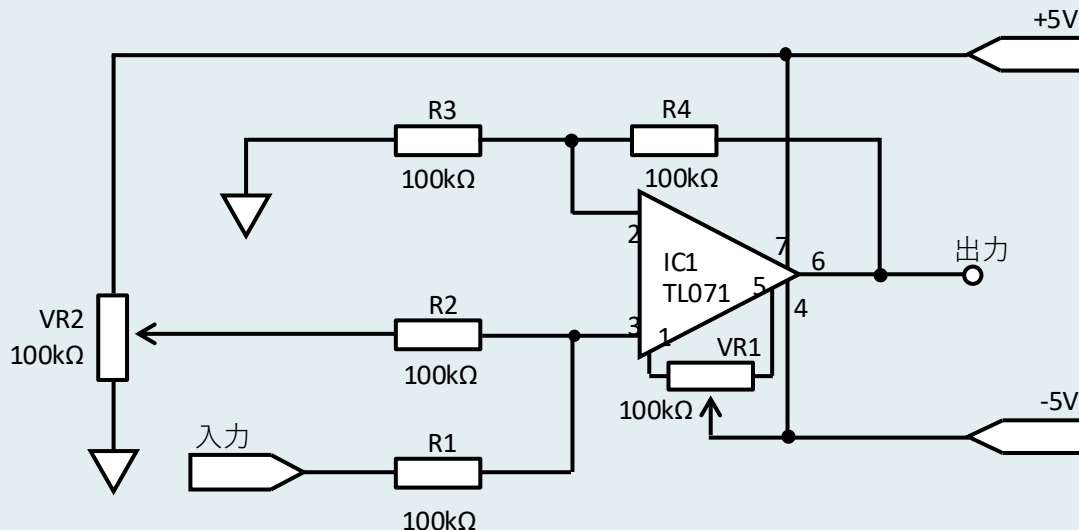
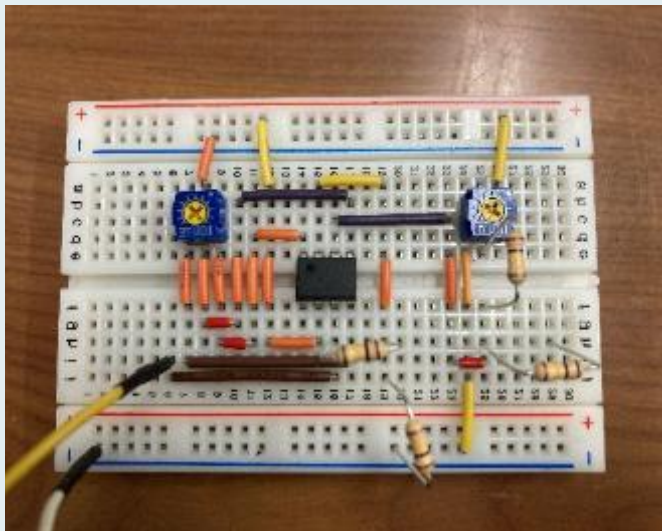
- 昇圧モジュールMT3608を使用し、5Vまで昇圧



問題点と解決策

③ Arduinoでは、筋電位の負電圧を直接読み取れない

- アンプ (TL071) と抵抗器を用いたバイアス回路を追加
- 負電圧をArduinoが読み取り可能な0～5Vの範囲に変換し、動作することを確認



問題点と解決策



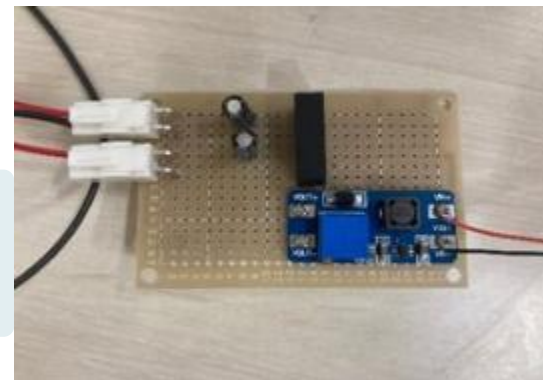
④ DC/DCコンバータ（MAU106）の接触不良を確認

- 電源回りの部品をユニバーサル基板にはんだ付けし、接続を安定化を図る

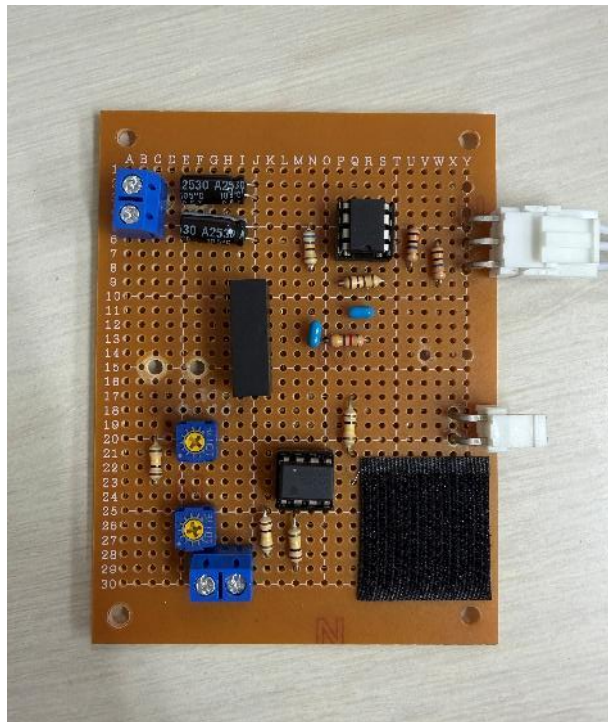
※昇圧モジュールも基盤に組み込む



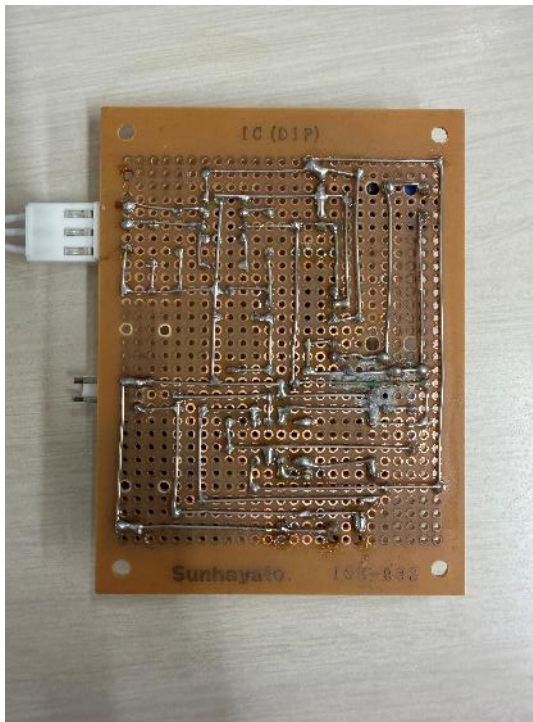
- 一時的には解決したが、可変抵抗など他の電子部品も接触不良が発生
- 回路全体をユニバーサル基盤に配線することとした



ユニバーサル基盤に配線



部品面



はんだ面



オシロスコープで正常に
動作することを確認

全体回路図

フィルタ付き筋電計測回路

記録電極1

記録電極2

基準電極

R5
10M Ω

R4
10M Ω

R3
100 Ω



R1
680 Ω

C1
0.47 μ F

C2
0.47 μ F

R2
4.7k Ω

電源回路
(ユニバーサル基板-作成済)

Li-Po 3.7V

IC1
MT3608

+Vcc

COM

-Vcc

C3
100 μ F

C4
100 μ F

IC2
MAU106

バイアス回路

R6
100k Ω

R7
100k Ω

R8
100k Ω

VR2
100k Ω

R9
100k Ω

VR1
100k Ω

IC4
TL071

v_{out}

出力

COM
(GND)

ArduinoとPythonによるデータ記録

Arduino単体ではデータを記録できないため、Pythonと接続してCSVに記録した

1 書き込み・実行

Arduinoで信号を読み取り、
PythonのCSV保存プログラムを
実行

```
1 // Arduino "Arduino_101_arduino_101"
2
3 // 1. 設定
4 // 1.1 使用するピン
5 // 1.2 使用するポート
6 // 1.3 使用するライブラリ
7 // 1.4 使用する変数
8 // 1.5 使用する関数
9 // 1.6 使用するコンストラクタ
10 // 1.7 使用するデストラクタ
11 // 1.8 使用する関数
12 // 1.9 使用する関数
13 // 1.10 使用する関数
14 // 1.11 使用する関数
15 // 1.12 使用する関数
16 // 1.13 使用する関数
17 // 1.14 使用する関数
18 // 1.15 使用する関数
19 // 1.16 使用する関数
20 // 1.17 使用する関数
21 // 1.18 使用する関数
22 // 1.19 使用する関数
23 // 1.20 使用する関数
24 // 1.21 使用する関数
25 // 1.22 使用する関数
26 // 1.23 使用する関数
27 // 1.24 使用する関数
28 // 1.25 使用する関数
29 // 1.26 使用する関数
30 // 1.27 使用する関数
31 // 1.28 使用する関数
32 // 1.29 使用する関数
33 // 1.30 使用する関数
34 // 1.31 使用する関数
35 // 1.32 使用する関数
36 // 1.33 使用する関数
37 // 1.34 使用する関数
38 // 1.35 使用する関数
39 // 1.36 使用する関数
40 // 1.37 使用する関数
41 // 1.38 使用する関数
42 // 1.39 使用する関数
43 // 1.40 使用する関数
44 // 1.41 使用する関数
45 // 1.42 使用する関数
46 // 1.43 使用する関数
47 // 1.44 使用する関数
48 // 1.45 使用する関数
49 // 1.46 使用する関数
50 // 1.47 使用する関数
51 // 1.48 使用する関数
52 // 1.49 使用する関数
53 // 1.50 使用する関数
54 // 1.51 使用する関数
55 // 1.52 使用する関数
56 // 1.53 使用する関数
57 // 1.54 使用する関数
58 // 1.55 使用する関数
59 // 1.56 使用する関数
60 // 1.57 使用する関数
61 // 1.58 使用する関数
62 // 1.59 使用する関数
63 // 1.60 使用する関数
64 // 1.61 使用する関数
65 // 1.62 使用する関数
66 // 1.63 使用する関数
67 // 1.64 使用する関数
68 // 1.65 使用する関数
69 // 1.66 使用する関数
70 // 1.67 使用する関数
71 // 1.68 使用する関数
72 // 1.69 使用する関数
73 // 1.70 使用する関数
74 // 1.71 使用する関数
75 // 1.72 使用する関数
76 // 1.73 使用する関数
77 // 1.74 使用する関数
78 // 1.75 使用する関数
79 // 1.76 使用する関数
80 // 1.77 使用する関数
81 // 1.78 使用する関数
82 // 1.79 使用する関数
83 // 1.80 使用する関数
84 // 1.81 使用する関数
85 // 1.82 使用する関数
86 // 1.83 使用する関数
87 // 1.84 使用する関数
88 // 1.85 使用する関数
89 // 1.86 使用する関数
90 // 1.87 使用する関数
91 // 1.88 使用する関数
92 // 1.89 使用する関数
93 // 1.90 使用する関数
94 // 1.91 使用する関数
95 // 1.92 使用する関数
96 // 1.93 使用する関数
97 // 1.94 使用する関数
98 // 1.95 使用する関数
99 // 1.96 使用する関数
100 // 1.97 使用する関数
101 // 1.98 使用する関数
102 // 1.99 使用する関数
103 // 2.00 使用する関数
```

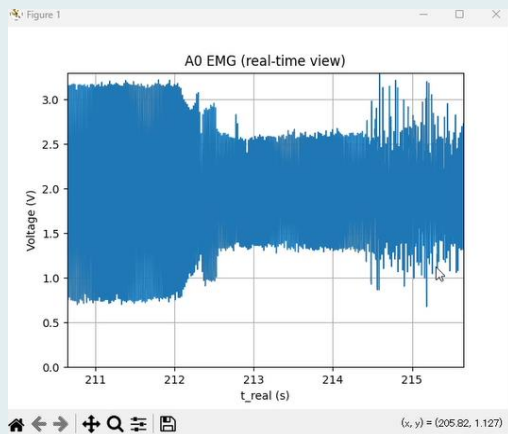
Arduino

```
1 // 1. 設定
2 // 1.1 使用するピン
3 // 1.2 使用するポート
4 // 1.3 使用するライブラリ
5 // 1.4 使用する変数
6 // 1.5 使用する関数
7 // 1.6 使用するコンストラクタ
8 // 1.7 使用するデストラクタ
9 // 1.8 使用する関数
10 // 1.9 使用する関数
11 // 1.10 使用する関数
12 // 1.11 使用する関数
13 // 1.12 使用する関数
14 // 1.13 使用する関数
15 // 1.14 使用する関数
16 // 1.15 使用する関数
17 // 1.16 使用する関数
18 // 1.17 使用する関数
19 // 1.18 使用する関数
20 // 1.19 使用する関数
21 // 1.20 使用する関数
22 // 1.21 使用する関数
23 // 1.22 使用する関数
24 // 1.23 使用する関数
25 // 1.24 使用する関数
26 // 1.25 使用する関数
27 // 1.26 使用する関数
28 // 1.27 使用する関数
29 // 1.28 使用する関数
30 // 1.29 使用する関数
31 // 1.30 使用する関数
32 // 1.31 使用する関数
33 // 1.32 使用する関数
34 // 1.33 使用する関数
35 // 1.34 使用する関数
36 // 1.35 使用する関数
37 // 1.36 使用する関数
38 // 1.37 使用する関数
39 // 1.38 使用する関数
40 // 1.39 使用する関数
41 // 1.40 使用する関数
42 // 1.41 使用する関数
43 // 1.42 使用する関数
44 // 1.43 使用する関数
45 // 1.44 使用する関数
46 // 1.45 使用する関数
47 // 1.46 使用する関数
48 // 1.47 使用する関数
49 // 1.48 使用する関数
50 // 1.49 使用する関数
51 // 1.50 使用する関数
52 // 1.51 使用する関数
53 // 1.52 使用する関数
54 // 1.53 使用する関数
55 // 1.54 使用する関数
56 // 1.55 使用する関数
57 // 1.56 使用する関数
58 // 1.57 使用する関数
59 // 1.58 使用する関数
60 // 1.59 使用する関数
61 // 1.60 使用する関数
62 // 1.61 使用する関数
63 // 1.62 使用する関数
64 // 1.63 使用する関数
65 // 1.64 使用する関数
66 // 1.65 使用する関数
67 // 1.66 使用する関数
68 // 1.67 使用する関数
69 // 1.68 使用する関数
70 // 1.69 使用する関数
71 // 1.70 使用する関数
72 // 1.71 使用する関数
73 // 1.72 使用する関数
74 // 1.73 使用する関数
75 // 1.74 使用する関数
76 // 1.75 使用する関数
77 // 1.76 使用する関数
78 // 1.77 使用する関数
79 // 1.78 使用する関数
80 // 1.79 使用する関数
81 // 1.80 使用する関数
82 // 1.81 使用する関数
83 // 1.82 使用する関数
84 // 1.83 使用する関数
85 // 1.84 使用する関数
86 // 1.85 使用する関数
87 // 1.86 使用する関数
88 // 1.87 使用する関数
89 // 1.88 使用する関数
90 // 1.89 使用する関数
91 // 1.90 使用する関数
92 // 1.91 使用する関数
93 // 1.92 使用する関数
94 // 1.93 使用する関数
95 // 1.94 使用する関数
96 // 1.95 使用する関数
97 // 1.96 使用する関数
98 // 1.97 使用する関数
99 // 1.98 使用する関数
100 // 1.99 使用する関数
101 // 2.00 使用する関数
```

Python

2 Enterで計測開始

筋電位がリアルタイムで
プロットされる



リアルタイム表示画面

3 Ctrl + Cで終了

計測データがCSVファイルで
自動保存される

	A	B	C	D	E
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	1.000000	1.000000	1.000000	1.000000	1.000000
3	1.000000	1.000000	1.000000	1.000000	1.000000
4	1.000000	1.000000	1.000000	1.000000	1.000000
5	1.000000	1.000000	1.000000	1.000000	1.000000
6	1.000000	1.000000	1.000000	1.000000	1.000000
7	1.000000	1.000000	1.000000	1.000000	1.000000
8	1.000000	1.000000	1.000000	1.000000	1.000000
9	1.000000	1.000000	1.000000	1.000000	1.000000
10	1.000000	1.000000	1.000000	1.000000	1.000000
11	1.000000	1.000000	1.000000	1.000000	1.000000
12	1.000000	1.000000	1.000000	1.000000	1.000000
13	1.000000	1.000000	1.000000	1.000000	1.000000
14	1.000000	1.000000	1.000000	1.000000	1.000000
15	1.000000	1.000000	1.000000	1.000000	1.000000
16	1.000000	1.000000	1.000000	1.000000	1.000000
17	1.000000	1.000000	1.000000	1.000000	1.000000
18	1.000000	1.000000	1.000000	1.000000	1.000000
19	1.000000	1.000000	1.000000	1.000000	1.000000
20	1.000000	1.000000	1.000000	1.000000	1.000000
21	1.000000	1.000000	1.000000	1.000000	1.000000
22	1.000000	1.000000	1.000000	1.000000	1.000000
23	1.000000	1.000000	1.000000	1.000000	1.000000
24	1.000000	1.000000	1.000000	1.000000	1.000000
25	1.000000	1.000000	1.000000	1.000000	1.000000
26	1.000000	1.000000	1.000000	1.000000	1.000000
27	1.000000	1.000000	1.000000	1.000000	1.000000
28	1.000000	1.000000	1.000000	1.000000	1.000000
29	1.000000	1.000000	1.000000	1.000000	1.000000
30	1.000000	1.000000	1.000000	1.000000	1.000000

時刻・電圧を
1行ずつ記録

筐体の製作

筐体を作るメリット

回路・部品を守る

筐体に収めることで、回路や部品を埃や水分から守れる

ノイズの影響を減らす

外部からのノイズの影響を低減できる

作成手順

① 寸法の測定・ 配置の検討

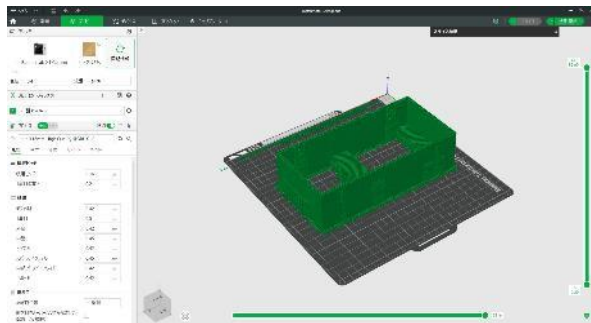
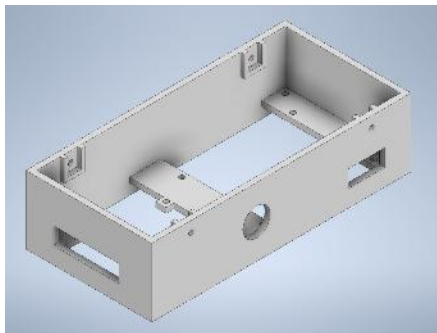
部材の寸法を測り、配置を決める

② 図面の作成

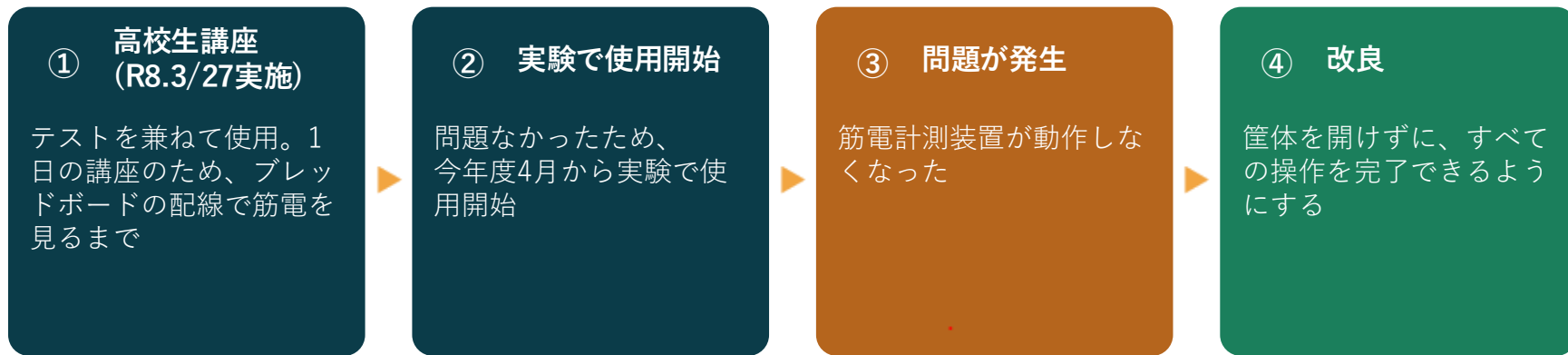
3D CADで図面を描く

③ 3Dプリンタ造形

図面を基に造形する



筋電計測装置の使用経緯と改良



問題の原因

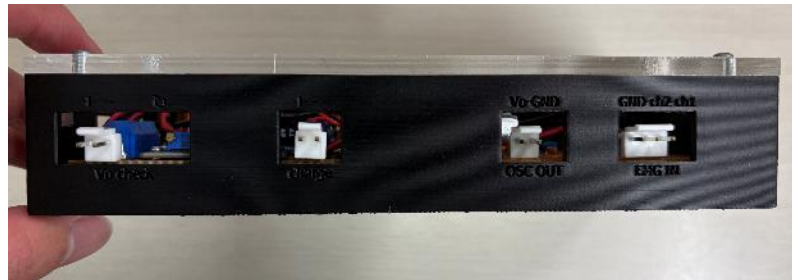
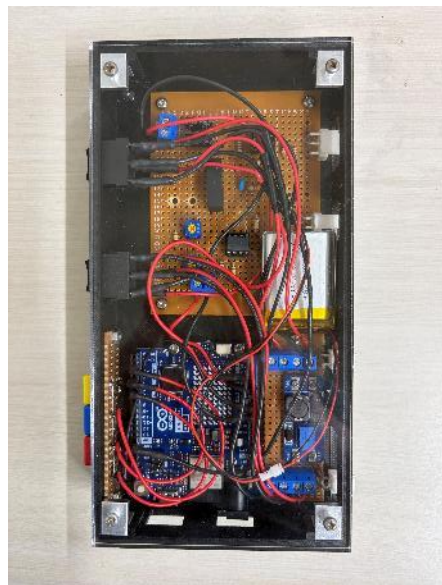
装置を使ったオシロスコープでの観察は想定していなかったため、計測時に開閉する必要があった筐体の開閉を繰り返したことで部品がずれ、他の部分と干渉して動作不良が生じたと推測

筐体の改良

条件	解決方法
充電を簡易的に、残量も確認したい	Arduino UNO R4基板上のLEDで「H,M,L」と残量表示。 充電用コネクタを設置し、外部から充電可能 3ポジションスイッチで「RUN/OFF/charge」切替
不具合時に昇圧モジュールの出力を確認したい	確認用コネクタを設置し、外部からテスターで電圧測定可能 3ポジションスイッチで「RUN/OFF/Vo check」切替
オシロスコープに分岐して外部出力したい	ユニバーサル基板上でArduinoとオシロスコープに配線を分岐



筐体の外観



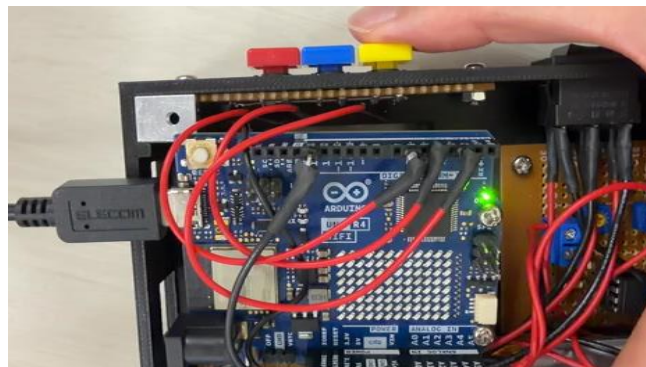
中が見える、透明な上下面

上下面をアクリル板にして、中の部品や配線を外から確認しやすくした

文字を彫り込んだ操作部

造形時にスイッチやコネクタの上下に文字を彫り込み、役割が一目で分かるようにした

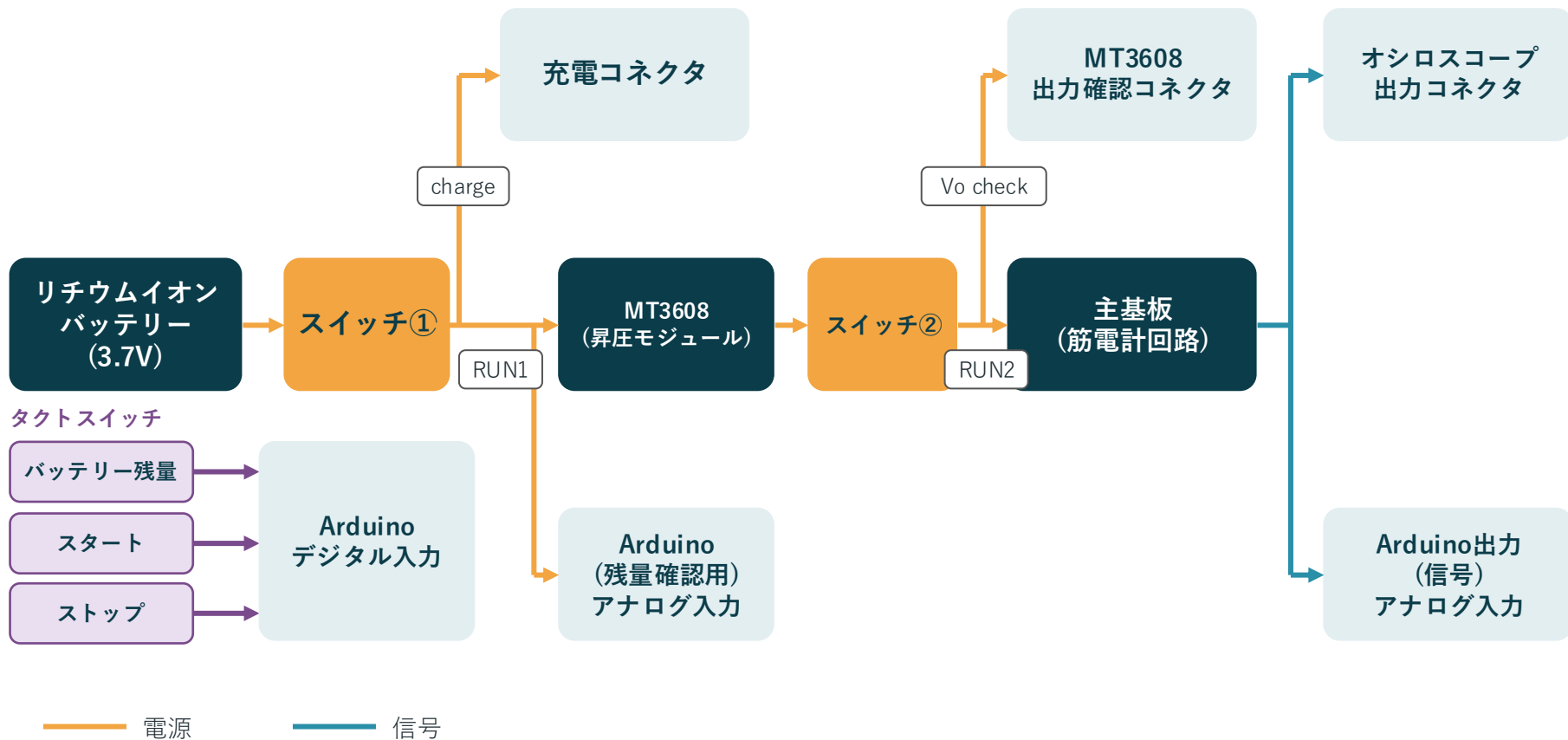
タクトスイッチによる操作



- **バッテリー残量確認**
電圧値を H(high) ・ M(mid) ・ L(low)
の3段階で表示
- **筋電計測開始**
- **停止**

3つのタクトスイッチにそれぞれ役割を持たせ、その動作をArduinoのプログラムに追加した。
筐体のボタンだけで、残量確認と計測の開始・停止ができ、操作性を向上させた。

筋電計の配線図(簡易)



結果

- 動作することは確認済み
- まだ、学生実験で使用していないのでこれから問題が発生する可能性有

今後の予定

- 筋電計測回路をもう1台追加(1台では実験への対応が困難)
- プリント基板を設計 → 外注
- 2台目の製作を通じて、学生に教えられる状態を目指す