ソリオンの電気的特性についての考察と応用

千葉県立船橋高等学校理数科二年　世良　倖太郎

1. はじめに

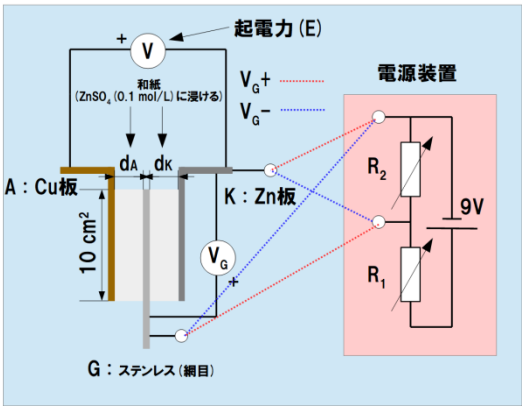
ソリオン（Solion）とは、Solution（溶液）とionを合わせた造語で、その名の通り、電解質溶液内のイオンを制御することで整流作用やスイッチング作用、トランジスタ作用などの特性を得ることができる液体電気素子である。

ソリオンはイオンの拡散を利用しているので、固体の半導体より不安定であり、極めて低周波でしか利用できない。しかし、液体で動作するという利点がある為、バイオセンサーや色素増感太陽電池で利用することができる可能性がある。

そこで本論では、この液体の電気素子であるソリオンの原理を活かしてイオンを制御することで動作するトランジスタを製作し、整流作用などの特性を得ることと、その特性を制御することを目的とする。

2. 実験方法

今回考えた液体内蔵電圧付三端子素子は以下の図のようなものである。



（図 1）

この素子のアノードとカソードを強く挟んで起電力とG電圧・電流の関係を測定する。測定にはEasy senseを用いて20ミリ秒ずつ10秒間（500サンプル）電圧を測ったものの平均をとった。

特に今回は極板間の間隔による影響を調べる為、$d_A = d_K$として、和紙の巻数（一巻0.1mm）を変えて実験する。

電源装置の回路より、Gに流れる電流は、

$$I_G = \frac{9R_2 - V_G(R_1 + R_2)}{R_1 R_2}$$

で表される。

3. 仮説

G電圧が負の時、G電極は銅板側から拡散しようとする負電荷の流れを妨げるので、

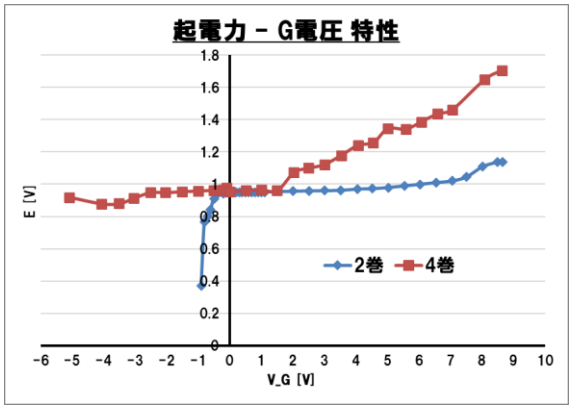
亜鉛イオンができにくくなり、起電力は下がると考えられる。

G 電圧が正の時は以上のようなことが起きず、起電力は一定のままであると考えられる。

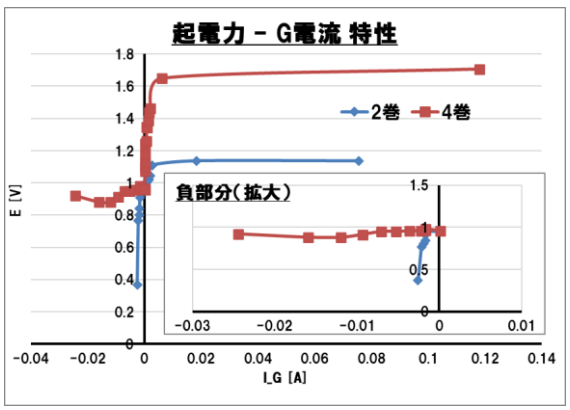
以上の二つの事から、G 電圧が正の時は電圧に対して起電力が一定で、負の時は電圧に関係して起電力が下がると考えられるので、整流作用が見られるはずである。

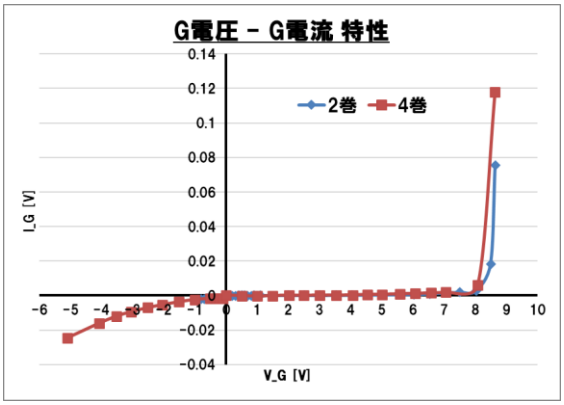
4. 実験結果

実験結果は、以下のようになった



。





5. 考察

起電力と G 電圧の関係について、和紙の巻き数が 2 回のものでは整流作用を見ることができた。巻き数が 4 回のもので見ることができなかったのは極板同士の間隔が広く、拡散速度の影響が強く出てしまった為だと考えられる。

また、起電力と G 電流、G 電流と G 電圧の関係について、G 電圧が+8.5V 付近で急激に起電力・G 電流が上昇している。これは、高電圧で電気分解が起きたためだと考えられる。

この現象について数値シミュレーションを行う。今回の現象において、G 電圧が負の時で、

$$\mathrm{G}：2\mathrm{H}^{+}+2e^{-}\rightarrow H_2$$

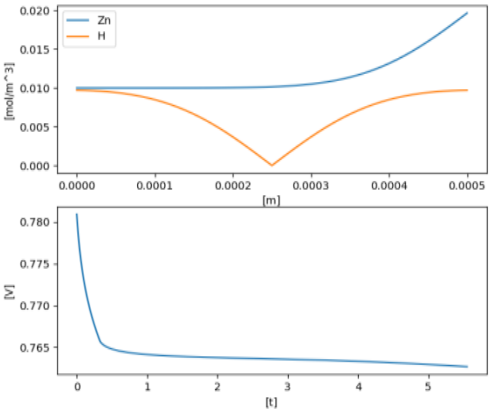
$$\mathrm{K}：\mathrm{Zn}\rightarrow \mathrm{Zn}^{2+}+2e^{-}$$

のような化学反応が起きると仮定して、この反応によってそれぞれの電極で一秒間に $b[\mathrm{mol/m^3}]$の濃度が減ると仮定して、ネルン

ストの方程式より、起電力は、亜鉛の標準電極電位を使って、

$$E = E^0_{Zn} + \frac{RT}{2F} log(\frac{2[H^+]_A}{[Zn^{2+}]_K})$$

で表さし、また、拡散方程式を数値シミュレーションするプログラムとして、初期条件・境界条件を今回の実験と同じように設定して付録に示すプログラムを書き、数値計算を行うと、以下の結果を得た。(b=0.001)(上は 5.5 秒経過時のそれぞれの濃度、下は電圧の時間変化)



この結果も示す通り、G 電圧が負の時は電圧が減少する。

6. 結論

液体内蔵電圧付三端子素子を製作し、イオンの流れを制御することで整流作用を得ることができた。また、G 電極の間隔が広いほど、拡散速度の影響を受けやすいことが分かった。

7. 参考文献

[1]押田 勇雄 . ソリオン -電気回路素子としての電解質溶液- . 日本物理学会誌 . 1959 年 14 巻 11 号 p. 619-625

[2]押田 勇雄 . ソリオン -液体エレクトロニクスへの道- . 電氣化學 . 1960 年 28 巻 9 号 p. 475-479

[3]松尾 武雄, 尾上 秀夫, 朝倉 祝治 . ソリオンダイオードの電流応答に関する研究 . 電気化学 . 1966 年 34 巻 12 号 p. 953-958

[4]山本 俊介 . 電子とイオンを操る高分子材料と電気化学トランジスタ素子 . 応用物理学会 M&BE . Vol.34, No.3, pp.116

[5]H. Letaw, J. Bardeen . The Electrolytic Analogue Transistor . J. Appl. Phys. 25 (1954) 600-606

8. 付録

拡散シミュレーションプログラム(Python)

```
import numpy as np
from matplotlib import pyplot as plt
from matplotlib import animation as an

fig = plt.figure()
ax1 = fig.add_subplot(2,1,1)
ax2 = fig.add_subplot(2,1,2)

#initial conditions
c_0 = 0.01
d_0 = 0.01
```

```
#conditions
D = 10**-9
L = 5*10**-4
I = 0.1
S = 5*10**-4
E_0 = 0.763
Tm = 300
l = 2.5*10**-4
F = 96485.3321233100184
R = 8.31446261815324

#caliculation conditions
dt = 0.01
dx = 10**-5
a = (D*dt)/(dx**2)
b = 0.001
k = (Tm*R)/(F)

t = 0
i_l = int(l/dx)
n = int((L/dx) + 1)
x = np.arange(0, L + dx, dx)
c_ = np.zeros(n)
d_ = np.zeros(n)
V = []
T = []

def cal(data):
    global c_0, c_m, c_p, dt, dx, a, t, n, x,
c_, V, T, i_l, k, b, d_0, d_

    #concentration
    ax1.cla()
    c = np.zeros(n)
    d = np.zeros(n)
    if t == 0:
        for i in range(len(c)):
            c[i] = c_0
        for i in range(len(d)):
            d[i] = d_0
    else:
        for i in range(len(c)):
            if i == 0:
                c[i] = a*(c_[i+1] -
c_[i]) + c_[i]
            elif i == (n - 1):
                c[i] = a*(c_[i-1] - c_[i])
+ c_[i] + (b/2)
            elif i == i_l:
                c[i] = a*(c_[i+1] -
2*c_[i] + c_[i-1]) + c_[i]
            else:
                c[i] = a*(c_[i+1] -
2*c_[i] + c_[i-1]) + c_[i]
            if c[i] <= 0:
                c[i] = 0
        for i in range(len(d)):
            if i == 0:
                d[i] = a*(d_[i+1] -
d_[i]) + d_[i]
            elif i == (n - 1):
                d[i] = a*(d_[i-1] -
d_[i]) + d_[i]
            elif i == i_l:
                d[i] = a*(d_[i+1] -
2*d_[i] + d_[i-1]) + d_[i] - b
            else:
                d[i] = a*(d_[i+1] -
2*d_[i] + d_[i-1]) + d_[i]
            if d[i] <= 0:
                d[i] = 0
                b = a*(d_[i+1] -
2*d_[i] + d_[i-1]) + d_[i]
    ax1.set_xlabel("[m]")
    ax1.set_ylabel("[mol/m^3]")
    ax1.plot(x, c, label="Zn")
    ax1.plot(x, d, label="H")
    ax1.legend()

    #voltage
    ax2.cla()
    T.append(t)
    V.append(E_0 + k *
(np.log((2*d[0])/c[n-1])))
    ax2.set_xlabel("[t]")
    ax2.set_ylabel("[V]")
    ax2.plot(T, V)

    c_ = c
    d_ = d
    t += dt

#program
print("Initial condition : " + str(c_0) + "
[mol/m^3]")
print("Diffusion coefficient : " + str(D) + "
[m^2/s]")
print("Device length : " + str(L) + " [m]")
print("a value : " + str(a) + " (a shoule be
less than 0.5)")
print("b value : " + str(b) + " [mol/m^3 s]")
anim = an.FuncAnimation(fig, cal,
```

```
interval=(dt*10))
plt.show()
```