電子制御工学実験報告書

実験題目　：電子回路設計・製作

報告者　　：4 年 32 番　平田 蓮

提出日　　：2020 年 10 月 8 日

実験日　　：2020 年 7 月 9 日, 7 月 16 日, 7 月 30 日, 8 月 6 日, 9 月 3 日, 9 月 10 日, 9 月 17 日

実験班　　：

共同実験者：

※指導教員記入欄

評価項目	配点	一次チェック ・　・	二次チェック ・　・
記載量	20		
図・表・グラフ	20		
見出し，ページ番号，その他体裁	10		
その他の減点	–		
合計	50		

コメント：

1　作品名

リアルタイム電卓

2　構造

図 1 に作品図を示す.

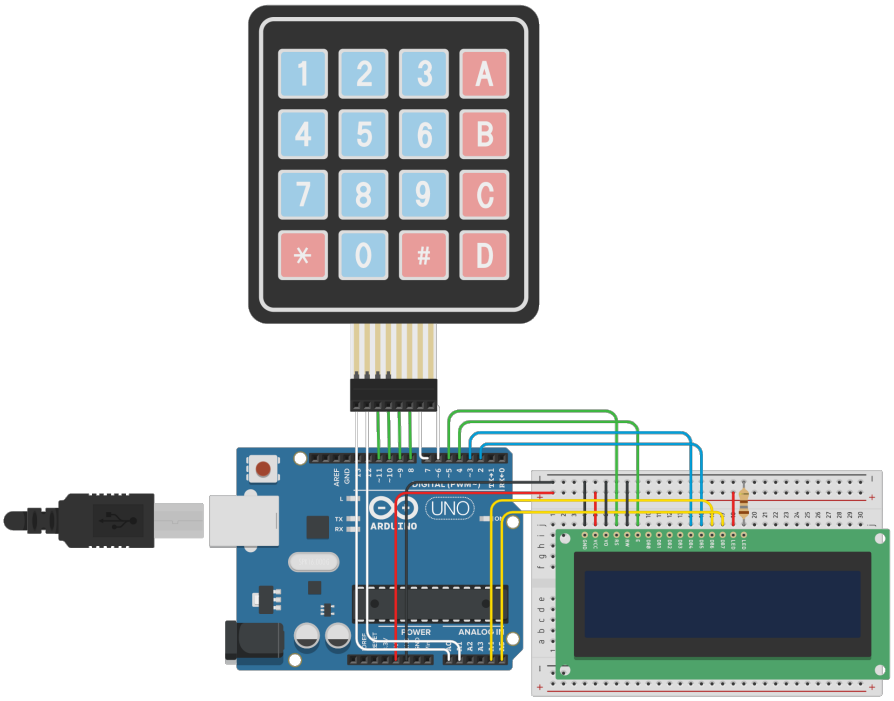

図 1　作品図

回路は Arduino, キーパッド, ブレッドボード, 16×2 液晶ディスプレイ, 180Ω 抵抗からなる.

3　動作原理

回路の動作として, まず初めにキーパッドに入力があると, キーに対応した文字が液晶ディスプレイの上段に出力される.

各キーと文字の対応は表 1 の通りである.

#キーについては, 文字が出力されるわけではなく, 液晶ディスプレイの出力が 1 文字削除される.

入力が行われるたびに, 液晶ディスプレイの上段の数式が評価される. もし数式が成り立っている場合, その数式の計算結果が液晶ディスプレイの下段に出力される.

表 1　キーと文字の対応

キー	文字
1 ～ 9	1 ～ 9
A	+
B	-
C	(
D	)
*	*

4　ハードウェア

図 2 に回路図を示す.

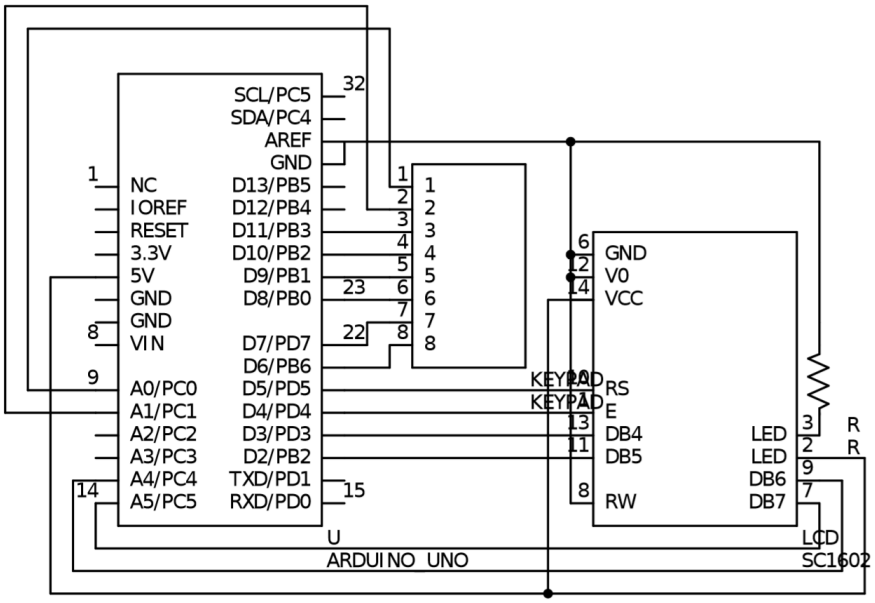

図 2　回路図

回路は文献 [1] を参考にした.

液晶ディスプレイには 180Ω の抵抗を接続してある. これは, 電源電圧を 5V, LED の順方向電圧を 1.7V, LED の順方向電流を 20mA としたときに次のように計算できる.

$$\frac{5 - 1.7\ \mathrm{V}}{20 \times 10^3\ \mathrm{A}} = 165\ \Omega \tag{1}$$

理論上は 165Ω の抵抗を使えばいいことになるが, 今回は最も近い値を持つ E24 系列の抵抗として 180Ω のものを使った.

5　ソフトウェア

図 3 に動作のフローチャートを示す.

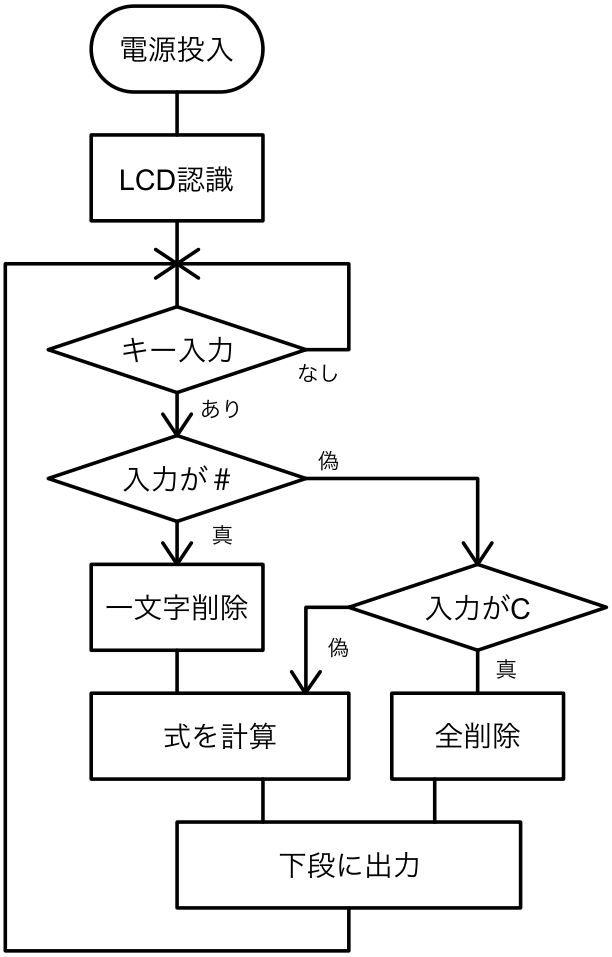

図 3　フローチャート

このフローチャートを元に作成したソースコードをリスト 1 に示す.

リスト 1　ソースコード

```
1  #include <Keypad.h>
2  #include <LiquidCrystal.h>
3
4  LiquidCrystal lcd(5, 4, 3, 2, A4, A5);
5
6  const byte ROWS = 4;
7  const byte COLS = 4;
8  char keys[ROWS][COLS] = {
9      {'1', '2', '3', '+'},
10     {'4', '5', '6', '-'},
11     {'7', '8', '9', '('},
```

```
12      {'*', '0', '<', ')'}
13  };
14  byte row_pins[ROWS] = {A0, A1, 11, 10};
15  byte col_pins[COLS] = {9, 8, 7, 6};
16  Keypad keypad = Keypad(
17      makeKeymap(keys),
18      row_pins,
19      col_pins,
20      ROWS,
21      COLS
22  );
23
24  int lcd_frame = 0;
25  char content[20] = {};
26
27  void setup(){
28      lcd.begin(16, 2);
29  }
30
31
32  int calc(char* content) {
33      int a = 0, b, j = 0;
34      int sanitized[20] = {};
35
36      for (int i = 0; i < lcd_frame; i++) {
37          if (content[i] >= '0' && content[i] <= '9') {
38              a *= 10;
39              a += content[i] - '0';
40          } else {
41              if (a) {
42                  sanitized[j] = a;
43                  j++;
44                  a = 0;
45              }
46
47              if (content[i] == '+') {
48                  sanitized[j] = -1;
49              } else if (content[i] == '-') {
50                  sanitized[j] = -2;
51              } else if (content[i] == '*') {
52                  sanitized[j] = -3;
53              } else if (content[i] == '(') {
54                  sanitized[j] = -4;
55              } else {
56                  sanitized[j] = -5;
57              }
58              j++;
```

```
        }
    }

    if (a) {
        sanitized[j] = a;
        j++;
    }

    int s[20] = {}, q[20] = {}, s_ptr = 0, q_ptr = 0;
    for (int i = 0; i < j; i++) {
        if (sanitized[i] >= 0) {
            q[q_ptr] = sanitized[i];
            q_ptr++;
        } else if (sanitized[i] > -4) {
            s_ptr--;
            while (
                s_ptr >= 0 &&
                s[s_ptr] < sanitized[i] &&
                s[s_ptr] > -4
            ) {
                q[q_ptr] = s[s_ptr];
                q_ptr++;

                s[s_ptr] = 0;
                s_ptr--;
            }
            s[++s_ptr] = sanitized[i];
            s_ptr++;
        } else if (sanitized[i] == -4) {
            s[s_ptr] = sanitized[i];
            s_ptr++;
        } else {
            s_ptr--;
            while (
                s_ptr >= 0 &&
                s[s_ptr] != -4
            ) {
                q[q_ptr] = s[s_ptr];
                q_ptr++;

                s[s_ptr] = 0;
                s_ptr--;
            }

            if (s[s_ptr] == -4) {
                s[s_ptr] = 0;
                s_ptr--;
```

```
106             }
107         }
108     }
109 
110     while (s_ptr >= 0) {
111         if (s[s_ptr] != 0) {
112             q[q_ptr] = s[s_ptr];
113             q_ptr++;
114         }
115 
116         s[s_ptr] = 0;
117         s_ptr--;
118     }
119 
120     s_ptr = 0;
121     for (int i = 0; i < q_ptr; i++) {
122         if (q[i] >= 0) {
123             s[s_ptr] = q[i];
124             s_ptr++;
125         } else {
126             s_ptr--;
127             b = s[s_ptr];
128             s[s_ptr] = 0;
129 
130             s_ptr--;
131             a = s[s_ptr];
132             s[s_ptr] = 0;
133 
134             if (q[i] == -1) {
135                 s[s_ptr] = a + b;
136             } else if (q[i] == -2) {
137                 s[s_ptr] = a - b;
138             } else {
139                 s[s_ptr] = a * b;
140             }
141             s_ptr++;
142         }
143     }
144     return s[0];
145 }
146 
147 void loop(){
148     const char key = keypad.getKey();
149 
150     if (key == '<') {
151     lcd.clear();
152         lcd.setCursor(0, 0);
```

```
153         lcd_frame--;
154         content[lcd_frame] = '\0';
155 
156         for (int i = 0; i < lcd_frame; i++) {
157             lcd.print(content[i]);
158         }
159     } else if (key) {
160         if (lcd_frame < 16) {
161             lcd.setCursor(lcd_frame, 0);
162             content[lcd_frame++] = key;
163             lcd.print(key);
164         }
165     }
166 
167     lcd.setCursor(0, 1);
168     lcd.print(calc(content));
169 }
```

`calc()` は与えられた文字列を数式としてみたときにその結果を計算する関数である．式を処理する際に操車場アルゴリズム [2] というものを利用した．

実際のアルゴリズムではキューとスタックを使用するが, Arduino 内部のライブラリでキューやスタックは実装されていないようなので, 通常の配列とポインタで代用した．

操車場アルゴリズムを適用した文字列は後置記法となる．110 行目から 143 行目では再びスタックを使用して後置記法となった式の計算をしている．

6　結果・今後の課題

結果として, 満足する動作をさせることができたと思う．細かい動作にバグなどがあるかもしれないので, 今後をそれらを調べていきたい．

また, 今回は除算は表示桁数が長くなると予想したため実装しなかったが, 実装してみるのも面白いと思う．その場合は 0 での除算を防がないといけないのですこし手間がかかるかもしれない．

7　感想

今回は電子回路設計ということであったが, どちらかというと回路のハードウェアというより Arduino 内部のコードに時間をかけた．今年は実際の回路を組むわけではなくオンライン上で作業をしたということもあり, 使える部品が豊富にあった．もし次に Tinkercad Circuits を使うようなことがあれば今度はハードウェアに寄せた回路を作ってみたい．

参考文献

[1] Invincible Academy, Keypad Interfacing in Tinkercad https://www.youtube.com/watch?v=pyBRSLCXmoQ

[2] @HMMNRST, 操車場アルゴリズムで四則計算の数式をパース
https://qiita.com/HMMNRST/items/e42a2062eb3bb6f46d37