

```

1  /*
2  所属:3EP1-07
3  学籍番号:1614266
4  名前:大谷直也
5  作成日:2018/11/17
6  */
7
8  #include <avr/io.h>
9  #include <avr/interrupt.h>
10 #include <avr/wdt.h>
11 #include <stdbool.h>
12
13 #define user CTOP 5000UL
14 #define CTOP 1000UL
15
16 void update_led();
17 /*回路全体図*/
18 volatile unsigned char map[8] =
19 {
20     // 0b10111111,
21     0b10010000,
22     0b10011111,
23     0b11000001,
24     0b11111101,
25     0b10000001,
26     0b10111111,
27     0b10000001,
28     0b11111101
29 };
30 /*switch操作用*/
31 volatile unsigned char stat;
32 volatile unsigned char sw;
33 volatile unsigned char sw_flag;
34
35 /*ブザーの時間*/
36 volatile unsigned int period;
37
38 /*位置などの記録*/
39 static unsigned char scan = 0;
40 unsigned char my_state = 0;
41
42 /*プレイヤー操作変数*/
43 unsigned char x = 0x40;
44 /*プレイヤー点減用*/
45 unsigned char x_sub = 0;
46 /*迷路の見える範囲*/
47 unsigned char smog_b = 0xE0;
48 /*switch用*/
49 bool sw_flag2 = false;
50 /*点減中の値の変更防止*/
51 bool x_flag = false;
52
53 ISR(PCINT1_vect) /*スイッチ*/
54 {
55     stat = 1;
56 }
57
58 ISR(TIMER0_COMPA_vect) /*led アップデート*/
59 {
60     update_led();
61 }
62
63 ISR(TIMER2_COMPA_vect) /*ブザー*/
64 {
65     PORTD ^= _BV(PORTD3);
66     if(sw_flag2 == true){
67         if(period != 0){

```

```

68             period--;
69             if(period == 0){
70                 sw_flag2 = false;
71                 DDRD = 0xF6;
72             }
73         }
74     }
75     else{
76         OCR2A = 0;
77     }
78 }
79 }
80
81 void update_sw() /*switch フラグ管理*/
82 {
83     static unsigned long cnt;
84     switch (stat) {
85     case 0:
86         return;
87     case 1:
88         cnt = CTOP;
89         stat = 2;
90         return;
91     case 2:
92         cnt--;
93         if (cnt == 0) {
94             sw = ~(PINC >> 4) & 3;
95             sw_flag = 1;
96             stat = 0;
97         }
98         return;
99     }
100 }
101
102 void update_led() /*マトリクスLEDのアップデート*/
103 {
104     static unsigned char sc = 0xFE;
105
106     PORTB = 0;
107     sc = (sc << 1) | (sc >> 7);
108     PORTD = (PORTD & 0x0F) | (sc & 0xF0);
109     PORTC = (PORTC & 0xF0) | (sc & 0x0F);
110     scan = (scan + 1) & 7;
111
112     /*霧の発生*/
113     if(my_state != 0){
114         if(scan == my_state ){
115             PORTB = map[scan] & smog_b;
116             PORTB |= x;
117         }else if(scan == (my_state + 1)){
118             PORTB = map[scan] & smog_b;
119         }else if(scan == (my_state - 1)){
120             PORTB = map[scan] & smog_b;
121         }
122     }
123 }
124
125 else{
126     if(scan == my_state ){
127         PORTB = map[scan] & smog_b;
128         PORTB |= x;
129     }else if(scan == (my_state + 1)){
130         PORTB = map[scan] & smog_b;
131     }
132     /*霧の発生終了*/
133 }
134
135 int main()

```

```

136 {
137     unsigned long user_cnt = 0;
138     bool user_b = false;
139
140     DDRB = 0xFF;
141     DDRC = 0x0F;
142     DDRD = 0xF6;
143
144     PORTB = 0x00;
145     PORTC = 0x30;
146     PORTD = 0x00;
147     PCICR = _BV(PCIE1); //ピン変化割り込み
148     PCMSK1 = 0x30;
149
150     TCNT0 = 0;
151     OCR0A = 249;
152     TCCR0A = 2;
153     TCCR0B = 3;
154     TIMSK0 |= _BV(OCIE0A); //2 ミリ秒ごとに点灯行の切り替え
155
156     TCCR2A = 02;
157     TCCR2B = 0x04;
158     TIMSK2 = _BV(OCIE2A); //ブザー用割り込み
159     OCR2A = 0; //割り込み時間は音階によって変化
160     OCR2B = 0;
161
162     sei();
163     for (;;) {
164         wdt_reset();
165         update_sw();
166         user_cnt++;
167
168         /*プレイヤーを点滅*/
169         if(user_cnt >= user_CTOP){
170             user_cnt = 0;
171             if(user_b == false){
172                 user_b = true;
173                 x_sub = x;
174                 x = x & 0;
175                 x_flag = false;
176             }else{
177                 x_flag = true;
178                 user_b = false;
179                 x = x_sub;
180             }
181         }
182
183         /*switchが押されたときの処理*/
184         if (sw_flag) {
185             sw_flag = 0;
186             switch (sw) {
187                 case 0:
188                     break;
189                 case 1: /*左*/
190                     if(x_flag == true){
191                         sw_flag2 = true;
192                         DDRD = 0xFE;
193                         x = (x >> 7) | (x << 1);
194                         smog_b = (smog_b >> 7) | (smog_b << 1);
195                         if((map[my_state] & x) == 0){
196                             period = 1000;
197                             OCR2A = 62;
198                         }
199                     }else{
200                         x = (x << 7) | (x >> 1);
201                         smog_b = (smog_b << 7) | (smog_b >> 1);
202                         period = 1000;
203

```

```

204         OCR2A = 238;
205     }
206 }
207 break;
208
209 case 2: /*右*/
210     if(x_flag == true){
211         sw_flag2 = true;
212         DDRD = 0xFE;
213         x = (x << 7) | (x >> 1);
214         smog_b = (smog_b << 7) | (smog_b >> 1);
215         if((map[my_state] & x) == 0)
216         {
217             period = 1000;
218             OCR2A = 62;
219         }
220     }else{
221         x = (x >> 7) | (x << 1);
222         smog_b = (smog_b >> 7) | (smog_b << 1);
223         period = 1000;
224         OCR2A = 238;
225     }
226 }
227 break;
228
229 case 3: /*下*/
230     if(x_flag == true){
231         sw_flag2 = true;
232         DDRD = 0xFE;
233         my_state = (my_state + 1) & 7;
234         if((map[my_state] & x) == 0){
235             period = 1000;
236             OCR2A = 62;
237         }
238     }else{
239         my_state = (my_state - 1) & 7;
240         period = 1000;
241         OCR2A = 238;
242     }
243 }
244 break;
245 }
246 }
247 }
248 return 0;
249 }
250

```