

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <wiringPi.h>
4  #include <stdint.h>
5
6  #define INPUT 0
7  #define OUTPUT 1
8
9  #define goE      0
10 #define flashOnE 1
11 #define flashOffE 2
12 #define goN      3
13 #define flashOnN 4
14 #define flashOffN 5
15 #define waitButton 6
16 #define NVIC_ST_CTRL_R 0xE000E010
17 #define NVIC_ST_RELOAD_R 0xE000E014
18 #define NVIC_ST_CURRENT_R 0xE000E018
19
20 uint32_t* RELOAD;
21 uint32_t* CURRENT;
22 uint32_t* CTRL;
23 uint32_t counter = 0;
24
25 void SysTick_Init(void)
26 {
27     RELOAD = (uint32_t) NVIC_ST_RELOAD_R;
28     *RELOAD = 0x00FFFFFF;
29     CURRENT = (uint32_t*) NVIC_ST_CURRENT_R;
30     *CURRENT = 0;
31     CTRL = (uint32_t*) NVIC_ST_CTRL_R;
32     *CTRL = 0x00000005;
33 }
34
35 // The delay parameter is in units of the 80 MHz core clock(12.5 ns)
36 // void SysTick_Wait(uint32_t delay)
37 // {
38 //     *RELOAD = delay-1; // number of counts
39 //     *CURRENT = 0; // any value written to CURRENT clears
40 //     while((*CTRL&0x00010000)==0)
41 //     {
42 //         // wait for flag
43 //     }
44 // }
45 // Call this routine to wait for delay*10ms
46 // void SysTick_Wait10ms(uint32_t delay){
47 //     unsigned long i;
48 //     for(i=0; i<delay; i++){
49 //         SysTick_Wait(16000000); // wait 1000ms
50 //     }
51
52 typedef struct State
53 {
54     uint32_t out[6];
55     uint32_t time;
56     uint32_t next[4];
57 } State;
58
59 State FSM[7] = {
60     {{1, 0, 0, 0, 0, 1}, 100, {goE, goE, flashOnE, flashOnE}},
61     {{1, 0, 0, 0, 1, 0}, 200, {flashOffE, goN, flashOffE, goN}},
62     {{1, 0, 0, 0, 0, 0}, 200, {flashOnE, flashOnE, flashOnE, flashOnE}},
63     {{0, 0, 0, 1, 1, 0}, 1000, {goN, flashOnN, goN, flashOnN}},
64     {{0, 1, 0, 1, 0, 0}, 500, {flashOffN, waitButton, flashOffN, waitButton}},
65     {{0, 0, 0, 1, 0, 0}, 500, {flashOnN, flashOnN, flashOnN, flashOnN}},
66     {{1, 0, 0, 0, 0, 1}, 30000, {goE, goE, goE, goE}},

```

```
67     };
68
69     void clockWrite(int time){
70         if (time % 10 == 9)
71         {
72             digitalWrite(35, 1);
73             digitalWrite(37, 1);
74             digitalWrite(38, 1);
75             digitalWrite(32, 1);
76             digitalWrite(22, 1);
77             digitalWrite(7, 0);
78             digitalWrite(36, 1);
79             digitalWrite(18, 1);
80         }
81         else if (time % 10 == 8)
82         {
83             digitalWrite(35, 1);
84             digitalWrite(37, 1);
85             digitalWrite(38, 1);
86             digitalWrite(32, 1);
87             digitalWrite(22, 1);
88             digitalWrite(7, 1);
89             digitalWrite(36, 1);
90             digitalWrite(18, 1);
91         }
92         else if (time % 10 == 7)
93         {
94             digitalWrite(35, 0);
95             digitalWrite(37, 1);
96             digitalWrite(38, 1);
97             digitalWrite(32, 1);
98             digitalWrite(22, 0);
99             digitalWrite(7, 0);
100            digitalWrite(36, 0);
101            digitalWrite(18, 0);
102        }
103        else if (time % 10 == 6)
104        {
105            digitalWrite(35, 1);
106            digitalWrite(37, 1);
107            digitalWrite(38, 0);
108            digitalWrite(32, 1);
109            digitalWrite(22, 1);
110            digitalWrite(7, 1);
111            digitalWrite(36, 1);
112            digitalWrite(18, 1);
113        }
114        else if (time % 10 == 5)
115        {
116            digitalWrite(35, 1);
117            digitalWrite(37, 1);
118            digitalWrite(38, 0);
119            digitalWrite(32, 1);
120            digitalWrite(22, 1);
121            digitalWrite(7, 0);
122            digitalWrite(36, 1);
123            digitalWrite(18, 1);
124        }
125        else if (time % 10 == 4)
126        {
127            digitalWrite(35, 1);
128            digitalWrite(37, 0);
129            digitalWrite(38, 1);
130            digitalWrite(32, 1);
131            digitalWrite(22, 0);
132            digitalWrite(7, 0);
```

```

133         digitalWrite(36, 1);
134         digitalWrite(18, 1);
135     }
136     else if (time % 10 == 3)
137     {
138         digitalWrite(35, 0);
139         digitalWrite(37, 1);
140         digitalWrite(38, 1);
141         digitalWrite(32, 1);
142         digitalWrite(22, 1);
143         digitalWrite(7, 0);
144         digitalWrite(36, 1);
145         digitalWrite(18, 1);
146     }
147     else if (time % 10 == 2)
148     {
149         digitalWrite(35, 0);
150         digitalWrite(37, 1);
151         digitalWrite(38, 1);
152         digitalWrite(32, 0);
153         digitalWrite(22, 1);
154         digitalWrite(7, 1);
155         digitalWrite(36, 1);
156         digitalWrite(18, 1);
157     }
158     else if (time % 10 == 1)
159     {
160         digitalWrite(35, 0);
161         digitalWrite(37, 0);
162         digitalWrite(38, 1);
163         digitalWrite(32, 1);
164         digitalWrite(22, 0);
165         digitalWrite(7, 0);
166         digitalWrite(36, 0);
167         digitalWrite(18, 0);
168     }
169     else if (time % 10 == 0)
170     {
171         digitalWrite(35, 1);
172         digitalWrite(37, 1);
173         digitalWrite(38, 1);
174         digitalWrite(32, 1);
175         digitalWrite(22, 1);
176         digitalWrite(7, 1);
177         digitalWrite(36, 0);
178         digitalWrite(18, 0);
179     }
180 }
181 int main()
182 {
183     if(wiringPiSetupPhys() == -1){
184         exit(1);
185     }
186     int currState;
187     int externalButtonValue;
188     uint32_t inputs = 0;
189
190     // Initialize ports and timer
191     pinMode(16, INPUT); //Button
192     pinMode(11, OUTPUT); //Red
193     pinMode(13, OUTPUT); //Blue
194     pinMode(15, OUTPUT); //Green
195     pinMode(29, OUTPUT); //Red
196     pinMode(31, OUTPUT); //Blue
197     pinMode(33, OUTPUT); //Green
198

```

```

199 pinMode(35, OUTPUT); //Top left
200 pinMode(37, OUTPUT); //Top
201 pinMode(38, OUTPUT); //Top right
202 pinMode(32, OUTPUT); //Bottom right
203 pinMode(22, OUTPUT); //Bottom
204 pinMode(7, OUTPUT); //Bottom left
205 pinMode(36, OUTPUT); //Middle left
206 pinMode(18, OUTPUT); //Middle right
207
208 digitalWrite(11, 0);
209 digitalWrite(13, 0);
210 digitalWrite(15, 0);
211 digitalWrite(29, 0);
212 digitalWrite(31, 0);
213 digitalWrite(33, 0);
214
215 digitalWrite(35, 0);
216 digitalWrite(37, 0);
217 digitalWrite(38, 0);
218 digitalWrite(32, 0);
219 digitalWrite(22, 0);
220 digitalWrite(7, 0);
221 digitalWrite(36, 0);
222 digitalWrite(18, 0);
223
224 delay(1000);
225 // SysTick_Init();
226 // SysTick_Wait(16000000);
227 currState = goE;
228 while (1)
229 {
230     printf("\n\nPin11: %d\n", FSM[currState].out[0]);
231     printf("Pin13: %d\n", FSM[currState].out[1]);
232     printf("Pin15: %d\n", FSM[currState].out[2]);
233     printf("Pin29: %d\n", FSM[currState].out[3]);
234     printf("Pin31: %d\n", FSM[currState].out[4]);
235     printf("Pin33: %d\n", FSM[currState].out[5]);
236     digitalWrite(11, FSM[currState].out[0]);
237     digitalWrite(13, FSM[currState].out[1]);
238     digitalWrite(15, FSM[currState].out[2]);
239     digitalWrite(29, FSM[currState].out[3]);
240     digitalWrite(31, FSM[currState].out[4]);
241     digitalWrite(33, FSM[currState].out[5]);
242
243     if (currState == goN)
244     {
245         clockWrite(20 - counter);
246         counter++;
247     }
248     if (currState == flashOnN)
249     {
250         clockWrite(10 - counter);
251         counter++;
252     }
253
254     delay(FSM[currState].time);
255
256     if(currState == flashOnN && counter == 10)
257     {
258         digitalWrite(35, 0);
259         digitalWrite(37, 0);
260         digitalWrite(38, 0);
261         digitalWrite(32, 0);
262         digitalWrite(22, 0);
263         digitalWrite(7, 0);
264         digitalWrite(36, 0);

```

```
265         digitalWrite(18, 0);
266     }
267     if (currState == flashOnE)
268     {
269         counter++;
270     }
271     printf("counter: %d\n", counter);
272
273     inputs = 0;
274     externalButtonValue = digitalRead(16);
275     printf("Button: %d\n", externalButtonValue);
276     if (externalButtonValue)
277     {
278         inputs += 2;
279     }
280     if (counter == 10)
281     {
282         inputs += 1;
283         counter = 0;
284     }
285
286     printf("inputs: %d\n", inputs);
287     currState = FSM[currState].next[inputs];
288 }
289 }
```