```c
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>
#include <stdint.h>

#define INPUT  0
#define OUTPUT 1

#define goE        0
#define flashOnE   1
#define flashOffE  2
#define goN        3
#define flashOnN   4
#define flashOffN  5
#define waitButton 6
#define NVIC_ST_CTRL_R 0xE000E010
#define NVIC_ST_RELOAD_R 0xE000E014
#define NVIC_ST_CURRENT_R 0xE000E018

uint32_t* RELOAD;
uint32_t* CURRENT;
uint32_t* CTRL;
uint32_t counter = 0;

void SysTick_Init(void)
{
 RELOAD = (uint32_t) NVIC_ST_RELOAD_R;
 *RELOAD = 0x00FFFFFF;
 CURRENT = (uint32_t*) NVIC_ST_CURRENT_R;
 *CURRENT = 0;
 CTRL = (uint32_t*) NVIC_ST_CTRL_R;
 *CTRL = 0x00000005;
}

// The delay parameter is in units of the 80 MHz core clock(12.5 ns)
// void SysTick_Wait(uint32_t delay)
// {
//   *RELOAD = delay-1;  // number of counts
//   *CURRENT = 0;       // any value written to CURRENT clears
//   while((*CTRL&0x00010000)==0)
//   {
//      // wait for flag
//   }
// }
// Call this routine to wait for delay*10ms
// void SysTick_Wait10ms(uint32_t delay){
//  unsigned long i;
//  for(i=0; i<delay; i++){
//    SysTick_Wait(16000000); // wait 1000ms
// }

typedef struct State
{
    uint32_t out[6];
    uint32_t time;
    uint32_t next[4];
} State;

State FSM[7] = {
    {{1, 0, 0, 0, 0, 1}, 100, {goE, goE, flashOnE, flashOnE}},
    {{1, 0, 0, 0, 1, 0}, 200, {flashOffE, goN, flashOffE, goN}},
    {{1, 0, 0, 0, 0, 0}, 200, {flashOnE, flashOnE, flashOnE, flashOnE}},
    {{0, 0, 1, 1, 0, 0}, 1000, {goN, flashOnN, goN, flashOnN}},
    {{0, 1, 0, 1, 0, 0}, 500, {flashOffN, waitButton, flashOffN, waitButton}},
    {{0, 0, 0, 1, 0, 0}, 500, {flashOnN, flashOnN, flashOnN, flashOnN}},
    {{1, 0, 0, 0, 0, 1}, 30000, {goE, goE, goE, goE}},
```

```
};

void clockWrite(int time){
    if (time % 10 == 9)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 1);
        digitalWrite(38, 1);
        digitalWrite(32, 1);
        digitalWrite(22, 1);
        digitalWrite(7, 0);
        digitalWrite(36, 1);
        digitalWrite(18, 1);
    }
    else if (time % 10 == 8)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 1);
        digitalWrite(38, 1);
        digitalWrite(32, 1);
        digitalWrite(22, 1);
        digitalWrite(7, 1);
        digitalWrite(36, 1);
        digitalWrite(18, 1);
    }
    else if (time % 10 == 7)
    {
        digitalWrite(35, 0);
        digitalWrite(37, 1);
        digitalWrite(38, 1);
        digitalWrite(32, 1);
        digitalWrite(22, 0);
        digitalWrite(7, 0);
        digitalWrite(36, 0);
        digitalWrite(18, 0);
    }
    else if (time % 10 == 6)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 1);
        digitalWrite(38, 0);
        digitalWrite(32, 1);
        digitalWrite(22, 1);
        digitalWrite(7, 1);
        digitalWrite(36, 1);
        digitalWrite(18, 1);
    }
    else if (time % 10 == 5)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 1);
        digitalWrite(38, 0);
        digitalWrite(32, 1);
        digitalWrite(22, 1);
        digitalWrite(7, 0);
        digitalWrite(36, 1);
        digitalWrite(18, 1);
    }
    else if (time % 10 == 4)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 0);
        digitalWrite(38, 1);
        digitalWrite(32, 1);
        digitalWrite(22, 0);
        digitalWrite(7, 0);
```

```
            digitalWrite(36, 1);
            digitalWrite(18, 1);
        }
        else if (time % 10 == 3)
        {
            digitalWrite(35, 0);
            digitalWrite(37, 1);
            digitalWrite(38, 1);
            digitalWrite(32, 1);
            digitalWrite(22, 1);
            digitalWrite(7, 0);
            digitalWrite(36, 1);
            digitalWrite(18, 1);
        }
        else if (time % 10 == 2)
        {
            digitalWrite(35, 0);
            digitalWrite(37, 1);
            digitalWrite(38, 1);
            digitalWrite(32, 0);
            digitalWrite(22, 1);
            digitalWrite(7, 1);
            digitalWrite(36, 1);
            digitalWrite(18, 1);
        }
        else if (time % 10 == 1)
        {
            digitalWrite(35, 0);
            digitalWrite(37, 0);
            digitalWrite(38, 1);
            digitalWrite(32, 1);
            digitalWrite(22, 0);
            digitalWrite(7, 0);
            digitalWrite(36, 0);
            digitalWrite(18, 0);
        }
        else if (time % 10 == 0)
        {
            digitalWrite(35, 1);
            digitalWrite(37, 1);
            digitalWrite(38, 1);
            digitalWrite(32, 1);
            digitalWrite(22, 1);
            digitalWrite(7, 1);
            digitalWrite(36, 0);
            digitalWrite(18, 0);
        }
    }
    int main()
    {
        if(wiringPiSetupPhys() == -1){
            exit(1);
        }
        int currState;
        int externalButtonValue;
        uint32_t inputs = 0;

        // Initialize ports and timer
        pinMode(16, INPUT);  //Button
        pinMode(11, OUTPUT); //Red
        pinMode(13, OUTPUT); //Blue
        pinMode(15, OUTPUT); //Green
        pinMode(29, OUTPUT); //Red
        pinMode(31, OUTPUT); //Blue
        pinMode(33, OUTPUT); //Green

```

```
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 }
```