```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 goNdigitR  3
#define goNdigitL  4
#define flashOnN   5
#define flashOffN  6
#define waitButton 7

double counter = 0.00; // Used for flashing the LED's and segment display the appropriate amount of times

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

State FSM[8] = {
    {{1, 0, 0, 0, 0, 1}, 100, {goE, goE, flashOnE, flashOnE}},
    {{1, 0, 0, 0, 1, 0}, 200, {flashOffE, goNdigitR, flashOffE, goNdigitR}},
    {{1, 0, 0, 0, 0, 0}, 200, {flashOnE, flashOnE, flashOnE, flashOnE}},
    {{0, 0, 1, 1, 0, 0}, 5, {goNdigitL, flashOnN, goNdigitL, flashOnN}},
    {{0, 0, 1, 1, 0, 0}, 5, {goNdigitR, goNdigitR, goNdigitR, goNdigitR}},
    {{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 rightDigit(double time) // Prints the right digit on the dual 14-segment display
{
    digitalWrite(40, 1); // Adjust the ground pins appropriately
    digitalWrite(12, 0);

    if (time >= 9 && time < 10) // Print the number
    {
        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 >= 8 && time < 9)
    {
        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 >= 7 && time < 8)
```

```
66        {
67            digitalWrite(35, 0);
68            digitalWrite(37, 1);
69            digitalWrite(38, 1);
70            digitalWrite(32, 1);
71            digitalWrite(22, 0);
72            digitalWrite(7, 0);
73            digitalWrite(36, 0);
74            digitalWrite(18, 0);
75        }
76        else if (time >= 6 && time < 7)
77        {
78            digitalWrite(35, 1);
79            digitalWrite(37, 1);
80            digitalWrite(38, 0);
81            digitalWrite(32, 1);
82            digitalWrite(22, 1);
83            digitalWrite(7, 1);
84            digitalWrite(36, 1);
85            digitalWrite(18, 1);
86        }
87        else if (time >= 5 && time < 6)
88        {
89            digitalWrite(35, 1);
90            digitalWrite(37, 1);
91            digitalWrite(38, 0);
92            digitalWrite(32, 1);
93            digitalWrite(22, 1);
94            digitalWrite(7, 0);
95            digitalWrite(36, 1);
96            digitalWrite(18, 1);
97        }
98        else if (time >= 4 && time < 5)
99        {
100           digitalWrite(35, 1);
101           digitalWrite(37, 0);
102           digitalWrite(38, 1);
103           digitalWrite(32, 1);
104           digitalWrite(22, 0);
105           digitalWrite(7, 0);
106           digitalWrite(36, 1);
107           digitalWrite(18, 1);
108       }
109       else if (time >= 3 && time < 4)
110       {
111           digitalWrite(35, 0);
112           digitalWrite(37, 1);
113           digitalWrite(38, 1);
114           digitalWrite(32, 1);
115           digitalWrite(22, 1);
116           digitalWrite(7, 0);
117           digitalWrite(36, 1);
118           digitalWrite(18, 1);
119       }
120       else if (time >= 2 && time < 3)
121       {
122           digitalWrite(35, 0);
123           digitalWrite(37, 1);
124           digitalWrite(38, 1);
125           digitalWrite(32, 0);
126           digitalWrite(22, 1);
127           digitalWrite(7, 1);
128           digitalWrite(36, 1);
129           digitalWrite(18, 1);
130       }
131       else if (time >= 1 && time < 2)
```

```
    {
        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 >= 0 && time < 1)
    {
        digitalWrite(35, 1);
        digitalWrite(37, 1);
        digitalWrite(38, 1);
        digitalWrite(32, 1);
        digitalWrite(22, 1);
        digitalWrite(7, 1);
        digitalWrite(36, 0);
        digitalWrite(18, 0);
    }
}

void leftDigit(double time) // Prints the left digit on the dual 14-segment display
{
    if (time >= 10 && time < 20)
    {
        digitalWrite(12, 1);
        digitalWrite(40, 0);

        // Left side digit 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)
    {
        // No digit on left side
        digitalWrite(12, 0);
        digitalWrite(40, 1);
    }
}

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

```

```
198       pinMode(35, OUTPUT); //Top left
199       pinMode(37, OUTPUT); //Top
200       pinMode(38, OUTPUT); //Top right
201       pinMode(32, OUTPUT); //Bottom right
202       pinMode(22, OUTPUT); //Bottom
203       pinMode(7, OUTPUT);  //Bottom left
204       pinMode(36, OUTPUT); //Middle left
205       pinMode(18, OUTPUT); //Middle right
206
207       pinMode(40, OUTPUT); //Ground for Left Digit
208       pinMode(12, OUTPUT); //Ground for Right Digit
209
210       digitalWrite(11, 0);
211       digitalWrite(13, 0);
212       digitalWrite(15, 0);
213       digitalWrite(29, 0);
214       digitalWrite(31, 0);
215       digitalWrite(33, 0);
216
217       digitalWrite(35, 0);
218       digitalWrite(37, 0);
219       digitalWrite(38, 0);
220       digitalWrite(32, 0);
221       digitalWrite(22, 0);
222       digitalWrite(7, 0);
223       digitalWrite(36, 0);
224       digitalWrite(18, 0);
225
226       delay(1000); // Shut both pins off for a second to recognize program reset
227       currState = goE; // Initial state
228       while (1)
229       {
230           // Light up the LED's correspondent to the current state
231           digitalWrite(11, FSM[currState].out[0]);
232           digitalWrite(13, FSM[currState].out[1]);
233           digitalWrite(15, FSM[currState].out[2]);
234           digitalWrite(29, FSM[currState].out[3]);
235           digitalWrite(31, FSM[currState].out[4]);
236           digitalWrite(33, FSM[currState].out[5]);
237
238           // Print the remaining time on the 14-segment display if necessary
239           if (currState == goNdigitR)
240           {
241               rightDigit(10 - counter);
242               counter += 0.01;
243           }
244           else if (currState == goNdigitL)
245           {
246               leftDigit(20 - counter);
247           }
248           else if (currState == flashOnN)
249           {
250               rightDigit(9 - counter);
251               counter += 1;
252           }
253
254           // Wait for how long the state is supposed to delay for
255           delay(FSM[currState].time);
256
257           // Turn off segment display when the time is up
258           if(currState == flashOnN && counter == 10)
259           {
260               digitalWrite(35, 0);
261               digitalWrite(37, 0);
262               digitalWrite(38, 0);
263               digitalWrite(32, 0);
```

```
264             digitalWrite(22, 0);
265             digitalWrite(7, 0);
266             digitalWrite(36, 0);
267             digitalWrite(18, 0);
268         }
269 
270         // Increment the flash count when the traffic light is flashing
271         if (currState == flashOnE)
272         {
273             counter++;
274         }
275 
276         // Use the inputs to determine which state to go to next
277         inputs = 0;
278         externalButtonValue = digitalRead(16);
279         if (externalButtonValue)
280         {
281             inputs += 2;
282         }
283         if (counter >= 10) // The counter will always be at 10 when it's ready to go to
        a new state
284         {
285             inputs += 1;
286             counter = 0;
287         }
288 
289         // Go to the next state (it might loop back to the same state)
290         currState = FSM[currState].next[inputs];
291     }
292 }
```