

```

1  /*****
2  *
3  *   _____   _____   _____   _____   _____   _____
4  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
5  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
6  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
7  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
8  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
9  * *****/
10 *
11 * File       : app.c
12 * Version    : 1.0
13 *
14 *****/
15 *
16 * Description : Managing global state machine
17 *
18 *****/
19 *
20 * Author      : Miguel Santos
21 * Date       : 25.09.2023
22 *
23 *****/
24 *
25 * MPLAB X     : 5.45
26 * XC32        : 2.50
27 * Harmony     : 2.06
28 *
29 *****/
30
31 #include "app.h"
32 #include "modules/TLC5973.h"
33 #include "driver/tmr/drv_tmr_mapping.h"
34
35 #define LED_WIFI TLC_DRV_ID_0
36
37 /*****/
38
39 /* Timeout for turning off output (in milliseconds) */
40 #define APP_TIME_OUT_MS 7200000
41
42 /* Wait time to allow user to react (in milliseconds) */
43 #define APP_TIME_WAIT_MS 60000
44
45 /*****/
46
47 /* Declaration of global application data */
48
49 APP_DATA appData;
50
51 /*****/
52
53 /**
54  * @brief APP_Initialize
55  *
56  * Initialize APP state machine
57  *
58  * @param void
59  * @return void
60  */
61 void APP_Initialize ( void )
62 {
63     /* Place the App state machine in its initial state. */
64     appData.state = APP_STATE_INIT;
65
66     /* Initialize TLC5973 interface */
67     TLC_Initialize();
68
69     /* Start application's timer */
70     DRV_TMR0_Start();
71
72     /* Initialize timeout counter */
73     CNT_Initialize(&appData.timeOut, 50);

```

```

74 }
75
76 /*****
77
78 /**
79  * @brief APP_Tasks
80  *
81  * Execute APP state machine, should be called cyclically
82  *
83  * @param void
84  * @return void
85  */
86 void APP_Tasks ( void )
87 {
88     /* Check the application's current state. */
89     switch ( appData.state )
90     {
91         /* Application's initial state. */
92         case APP_STATE_INIT:
93         {
94             AC_SETOn();
95             appData.state = APP_STATE_IDLE;
96             break;
97         }
98
99         case APP_STATE_IDLE:
100         {
101             if(CNT_Check(&appData.timeOut))
102             {
103                 AC_SETOff();
104                 appData.state = APP_STATE_SETUP_WIFI;
105             }
106             break;
107         }
108
109         case APP_STATE_SETUP_WIFI:
110         {
111             break;
112         }
113
114         case APP_STATE_SETUP_RFID:
115         {
116             break;
117         }
118
119         case APP_STATE Asking:
120         {
121             break;
122         }
123
124         case APP_STATE_OFF:
125         {
126             break;
127         }
128
129         case APP_STATE_ON:
130         {
131             break;
132         }
133
134         /* The default state should never be executed. */
135         default:
136         {
137             /* TODO: Handle error in application's state machine. */
138             break;
139         }
140     }
141 }
142
143 /*****
144
145 /* End of File *****/
146

```