

```

1  #include "mbed.h"
2  #include "pinout.h"
3  #include "to_7seg.h"
4
5  // seven segment display anodes
6  // when in a int8_t, they are 0b-GFEDCBA
7  BusOut      g_seven_seg(SGA_PIN, SGB_PIN, SGC_PIN, SGD_PIN,
8                        SGE_PIN, SGF_PIN, SGG_PIN);
9
10 // display cathodes
11 DigitalOut   g_dsr(DSR_PIN);
12 DigitalOut   g_dsl(DSL_PIN);
13
14 // leds
15 BusOut      g_leds(LDR_PIN, LDM_PIN, LDL_PIN);
16
17 //Interrupciones
18 static InterruptIn swr(SWR_PIN);
19 static InterruptIn swl(SWL_PIN);
20
21 static bool volatile swr_fall_evnt;
22 static bool volatile swl_fall_evnt;
23
24 static void swr_fall_isr (void){
25     swr_fall_evnt = true;
26 }
27
28 static void swl_fall_isr(void){
29     swl_fall_evnt = true;
30 }
31
32 // MULTIPLEXACION
33 static Ticker tick_4ms;
34 static bool volatile tick_4ms_evnt;
35 static void tick_4ms_isr (void){
36     tick_4ms_evnt = true;
37 }
38
39 //REBOTES
40 static Timeout tout_4ms_swr;
41 static bool volatile tout_4ms_swr_evnt;
42 static void tout_4ms_swr_isr (void){
43     tout_4ms_swr_evnt = true;
44 }
45
46 static Timeout tout_4ms_swl;
47 static bool volatile tout_4ms_swl_evnt;
48 static void tout_4ms_swl_isr (void){
49     tout_4ms_swl_evnt = true;
50 }
51
52 //tiempo de refresco leds
53 static Ticker tick_10ms;
54 static bool volatile tick_10ms_evnt;
55 static void tick_10ms_isr (void){
56     tick_10ms_evnt = true;
57 }
58
59 static Timeout tout_led_off;
60 static bool volatile tout_led_off_evnt;
61 static void tout_led_off_isr (void){
62     tout_led_off_evnt = true;
63 }
64
65 int main (void) {
66     uint8_t pulsaciones_m = 50;
67     bool mux = false;
68
69     g_dsl = 1;
70     g_dsr = 1;
71
72     g_seven_seg = to_7seg(pulsaciones_m);
73
74     swr.mode(PullUp);
75     swr.fall(swr_fall_isr);
76
77     swl.mode(PullUp);
78     swl.fall(swl_fall_isr);
79
80     tick_4ms.attach_us(tick_4ms_isr, 4000);
81     tick_10ms.attach_us(tick_10ms_isr, 10000);
82
83     for (;;) {
84

```

```

85     if(tick_4ms_evnt){
86         tick_4ms_evnt = false;
87         mux = !mux;
88
89         if(mux){
90             g_dsl = 0;
91             g_dsr = 1;
92             g_seven_seg = to_7seg(pulsaciones_m%10);
93
94         }else{
95             g_dsl = 1;
96             g_dsr = 0;
97             g_seven_seg = to_7seg(pulsaciones_m/10);
98         }
99     }
100
101     if(swr_fall_evnt){
102         swr_fall_evnt = false;
103         tout_4ms_swr.attach_us(tout_4ms_swr_isr, 4000);
104     }
105
106     if(swl_fall_evnt){
107         swl_fall_evnt = false;
108         tout_4ms_swl.attach_us(tout_4ms_swl_isr, 4000);
109     }
110
111     if(tout_4ms_swr_evnt){
112         tout_4ms_swr_evnt = false;
113
114         if(swr == 0){
115             pulsaciones_m = (pulsaciones_m == 0) ? 0 : (pulsaciones_m-1);
116         }
117     }
118
119     if(tout_4ms_swl_evnt){
120         tout_4ms_swl_evnt = false;
121
122         if(swl == 0){
123             pulsaciones_m = (pulsaciones_m == 99) ? 99 : (pulsaciones_m+1);
124         }
125     }
126
127     if(tick_10ms_evnt){
128         tick_10ms_evnt = false;
129
130         if(pulsaciones_m > 0 and pulsaciones_m < 51){
131             g_leds = 4;
132             tout_led_off.attach_us(tout_led_off_isr,
(200*pulsaciones_m-3.97*(50-pulsaciones_m)));
133
134         }else if (pulsaciones_m > 50 and pulsaciones_m <= 99){
135             g_leds = 1;
136             tout_led_off.attach_us(tout_led_off_isr,
(101*pulsaciones_m-107.2*(99-pulsaciones_m)));
137
138         }else if(pulsaciones_m == 0){
139             g_leds = 0;
140         }
141     }
142
143     if(tout_led_off_evnt){
144         tout_led_off_evnt = false;
145         g_leds = 0;
146     }
147
148
149     __disable_irq();
150     if(!tick_4ms_evnt && !swr_fall_evnt && !swl_fall_evnt && !tout_4ms_swr_evnt &&
!tout_4ms_swl_evnt && !tick_10ms_evnt && !tout_led_off_evnt){
151         __WFI();
152     }
153     __enable_irq();
154
155 } // for (;;)
156
157 } // main()
158
159

```