```
#include "switch.h"

/// extended state ------------------------------------------------------
  // "basic" state
typedef enum {SW_IDLE, SW_IN, SW_OUT} swm_state_t;
static swm_state_t g_swm_state;

  // externally reachable objects
bool              gb_swm_long_msg;    // set (externally) to start a measurement
bool              gb_swm_msg;     // set when a measurement is completed

    // parameter to rf_done_msg
bool volatile     gb_swm_can_sleep;    // this FSM can sleep

  // hardware resources
static InterruptIn    *g_swm;

static Timeout         tout_4ms;        // timeout // pulsacion de un segundo
static Timer           swm_tmr;
static Timeout         tout_20ms;


  // internal objects
static bool volatile  gb_swm_initd;            // true after call to rf_init()

static bool volatile  swm_fall_evnt;
static bool volatile  swm_rise_evnt;

static bool volatile  tout_4ms_evnt;           // rebotes
static bool volatile tout_20ms_evnt;

// end of extended state  ----------------------------------------------

// ISRs ----------------------------------------------------------------
  // swhtch SWM ISR
static void swm_fall_isr(void) {
  swm_fall_evnt = true;
  gb_swm_can_sleep = false;
}

static void swm_rise_isr(void){
  swm_rise_evnt = true;
  gb_swm_can_sleep  = false;
}


//timeout ISR
static void tout_4ms_isr (void) {
  tout_4ms_evnt = true;
  gb_swm_can_sleep = false;
}

static void tout_20ms_isr (void) {
  tout_20ms_evnt = true;
  gb_swm_can_sleep = false;
}

// end of ISRs  --------------------------------------------------------

// FSM  ----------------------------------------------------------------
void swm_fsm (void) {
  if (gb_swm_initd) {  // protect against calling rf_fsm() w/o a previous call to rf_init()
    switch (g_swm_state) {

       case SW_IDLE :

        if(swm_fall_evnt){
          swm_fall_evnt = false;
          tout_4ms.attach_us(tout_4ms_isr,4000);
        }
        if(tout_4ms_evnt){
          tout_4ms_evnt = false;
          tout_20ms.attach_us(tout_20ms_isr,20000);
          g_swm_state = SW_IN;
        }
       break;

      case SW_IN :
         swm_fall_evnt = false;

        if(tout_20ms_evnt){
          tout_20ms_evnt = false;

          if(0U == *g_swm){
```

```
85              swm_tmr.reset();
86              swm_tmr.start();
87              g_swm_state = SW_OUT;
88
89            }else{
90               g_swm_state = SW_IDLE;
91            }
92          }
93         break;
94
95        case SW_OUT :
96            swm_fall_evnt = false;
97
98        if(swm_rise_evnt){
99            swm_rise_evnt = false;
100           tout_4ms.attach_us(tout_4ms_isr,4000);
101         }
102
103         if(tout_4ms_evnt){
104           tout_4ms_evnt = false;
105
106           if(swm_tmr.read_us() >= 1000000){
107               swm_tmr.stop();
108               gb_swm_long_msg = true;
109               g_swm_state = SW_IDLE;
110           }
111
112           if(swm_tmr.read_us() < 1000000){
113              swm_tmr.stop();
114              gb_swm_msg = true;
115              g_swm_state = SW_IDLE;
116           }
117         }
118
119         break;
120
121       //  ------------------------------------------------------------------
122     } // switch (rf_state)
123
124
125     __disable_irq();
126     if(!swm_fall_evnt && !tout_4ms_evnt && !tout_20ms_evnt && !swm_rise_evnt &&
    !tout_4ms_evnt ) {
127       gb_swm_can_sleep = true;
128     }
129     __enable_irq();
130   } // if (gb_rf_initd)
131 }
132 // end of FSM ------------------------------------------------------------
133
134 // initialize FSM machinery  ---------------------------------------------
135 void swm_init(InterruptIn *swm){
136   if (!gb_swm_initd) {
137     gb_swm_initd = true;    // protect against multiple calls to rf_init
138
139     // initialize state
140     g_swm_state = SW_IDLE;
141
142     // initial actions
143     g_swm = swm;
144     swm_tmr.stop();
145     swm_tmr.reset();
146
147     g_swm->fall(swm_fall_isr);
148     g_swm->rise(swm_rise_isr);
149   }
150 }
151 // end of FSM initialization  --------------------------------------------
152
```