```
#include "mbed.h"
#include "pinout.h"
#include "hardware.h"
#include "to_7seg.h"
#include "range_finder.h"
#include "switch.h"

// mux stuff
static Ticker         g_mux_tick;
static bool volatile  gb_mux_evnt;

static void mux_isr (void) {
  gb_mux_evnt = true;
}

// range_finder automatic start
static Ticker         g_meas_tick;
static bool volatile  gb_meas_evnt;

static void meas_isr (void) {
  gb_meas_evnt = true;
}

//variable en curso
bool volatile en_curso;;

int main (void) {
  //variable en curso
   en_curso = false;

  // right display on?
  bool      b_right = false;
  // auto measure?
  bool      b_auto_measure = false;
  // the 4 MSB of this variable hold the symbol to be displayed at the
  //  left display, the 4 LSB that to be displayed at the right one
  uint8_t   disp = 0;

  hw_init();

  g_seven_seg = 0;
  g_dsr = b_right;
  g_dsl = !b_right;
  g_mux_tick.attach_us(mux_isr, 4000);      // 250 Hz

  // initialize the range finder FSM
  rf_init(&g_trg, &g_ech);

  // initialize the middle switch FSM
  swm_init(&g_swm);

  //  ------------------------------------------------------------------
  for (;;) {

    // the range finder FSM
    rf_fsm();

    // the middle switch FSM
    swm_fsm();

    //codigo
    if(gb_swm_long_msg){
      gb_swm_long_msg = false;

      if(!en_curso){
           en_curso = true;
           g_meas_tick.attach_us(meas_isr, 100000);

      }else{
          en_curso = false;
            g_meas_tick.detach();
      }
    }

    if(gb_swm_msg && !en_curso){
      gb_swm_msg = false;
      en_curso = false;
        meas_isr();
    }
   //  ------------------------------------------------------------------

    // start a new range measurement every 100 ms
    if (gb_meas_evnt) {
      gb_meas_evnt = false;
```

```
85          gb_rf_start_msg = true;
86        }
87
88        // when the measurement is complete, update variable disp
89        if (gb_rf_done_msg) {
90          gb_rf_done_msg = false;
91          if (g_rf_range_cm > 99) {
92            disp = 0xBB;  // --
93          } else if (-1 == g_rf_range_cm) {
94            disp = 0xEC;  // Er
95          } else {
96            disp = (g_rf_range_cm / 10) << 4 | (g_rf_range_cm % 10);
97          }
98        }
99
100       // display multiplex
101       if (gb_mux_evnt) {
102
103         gb_mux_evnt = false;
104         b_right = !b_right;
105         if(b_right){
106             g_dsr = 1;
107             g_dsl= 0;
108           g_seven_seg = (g_rf_range_cm > 99 ) ? disp : to_7seg(disp%10);
109
110           }else{
111             g_dsr = 0;
112             g_dsl= 1;
113             g_seven_seg = (g_rf_range_cm > 99 ) ? disp : to_7seg(disp/10);
114
115           }
116         }
117
118       // sleep
119       __disable_irq();
120       if (!gb_meas_evnt && !gb_rf_done_msg && !gb_mux_evnt
121           && !gb_rf_start_msg && gb_rf_can_sleep
122           && !gb_swm_msg && !gb_swm_long_msg && gb_swm_can_sleep) {
123         __WFI();
124       }
125       __enable_irq();
126     } // forever
127   } // main()
128
129
```