```c
#include "zboss_api.h"
#include "zigbee.h"
//#include "zboss_api_core.h"
//#include "zb_mem_config_med.h"
#include "zb_zcl_reporting.h"
//#include "zb_error_handler.h"
//#include "zigbee_helpers.h"
//#include "app_timer.h"
#include "bsp.h"
//#include "boards.h"
//#include "sensorsim.h"
#include "ms_sensorsim.h"
#include "zigbee.h"
#include "button_handler.h"
#include "nrf_power.h"


#include "nrf_temp.h"

#include "nrf_log.h"
#include "nrf_log_ctrl.h"
#include "nrf_log_default_backends.h"

#include "zb_multi_sensor.h"

#include "nrf_delay.h"

#define LED_BLINK                          ZB_MILLISECONDS_TO_BEACON_INTERVAL(180) /**< Led on-off timeout. */

/**@brief Function for initializing the nrf log module.
 */
static void log_init(void)
{
    ret_code_t err_code = NRF_LOG_INIT(NULL);
    APP_ERROR_CHECK(err_code);

    NRF_LOG_DEFAULT_BACKENDS_INIT();
}

/**@brief Function for initializing LEDs.
 */
static zb_void_t leds_init(void)
{
    //ret_code_t error_code;

    /* Initialize LEDs and buttons - use BSP to control them. */
    //error_code = bsp_init(BSP_INIT_LEDS, NULL);
    //APP_ERROR_CHECK(error_code);

    //bsp_board_leds_off();

    nrf_gpio_cfg_output(ZIGBEE_NETWORK_STATE_LED);
    nrf_gpio_cfg_output(USER_LED);
    nrf_gpio_pin_set(USER_LED);

    nrf_gpio_pin_set(ZIGBEE_NETWORK_STATE_LED); // LED off
    //nrf_gpio_pin_clear(USER_LED); // LED on
    //nrf_gpio_pin_clear(ZIGBEE_NETWORK_STATE_LED); // LED off

    nrf_gpio_cfg_output(LED2_B);
    //nrf_gpio_pin_clear(LED2_B); // LED on     nrf_gpio_pin_toggle(LED2_B)
    nrf_gpio_pin_set(LED2_B);

}

/*zb_void_t led_blink(zb_uint8_t count) {
  zb_ret_t zb_err_code;
  if (count != 0) {
    nrf_gpio_pin_toggle(LED2_B);
    if (nrf_gpio_pin_out_read(LED2_B) != 0) {
      count--;
    }
    zb_err_code = ZB_SCHEDULE_APP_ALARM(led_blink, count, LED_BLINK);
  }
}
*/

/**@brief Function for application main entry.
 */
int main(void)
{
    ret_code_t     err_code;
    NRF_CLOCK->TASKS_LFCLKSTART = 1;
    while (NRF_CLOCK->EVENTS_LFCLKSTARTED == 0);

    timers_init();

    log_init();
    //sensor_simulator_init();

    //button_handler_init();
    //nrf_gpio_cfg_input(BUTTON_1,NRF_GPIO_PIN_NOPULL);

    zigbee_init();

    //led_blink(5);
    // Clear events that might be generated during pin configuration
    NRF_GPIOTE->EVENTS_PORT = 0;
    NVIC_EnableIRQ(GPIOTE_IRQn);

    //configure_attribute_reporting();

    // Do commissioning
    //comm_status = bdb_start_top_level_commissioning(ZB_BDB_NETWORK_STEERING);
    //ZB_COMM_STATUS_CHECK(comm_status);

    // Do factory reset (leave network)
    //zb_bdb_reset_via_local_action(0);

    NRF_LOG_INFO("Sensor model ID: %s)", SENSOR_INIT_BASIC_MODEL_ID);



    while(1)
    {
//       nrf_pwr_mgmt_run();
        zboss_main_loop_iteration();
        UNUSED_RETURN_VALUE(NRF_LOG_PROCESS());
 //NRF_LOG_INFO("Main loop");
    }
}
```