```
/*
 * File:   Hello_World_main.c
 * Author: AutonomousSystemsLab
 *
 * Created on July 5, 2019, 1:32 PM
 */

//Standard C libraries:
#include <stdio.h>
#include <stdlib.h>

//Microchip's specialized library
#include <xc.h>

//A library to use the uc32 board
#include "BOARD.h"
#include "roach.h"
#include "timers.h"




char FL_pressed_EventFlag = 0;
char FL_released_EventFlag = 0;
char timer0_Expired_EventFlag = 0;
char light_EventFlag = 0;
char dark_EventFlag = 0;

int FLBumperPressed()
{
    static char previous_bumper_state = BUMPER_NOT_TRIPPED;
    char current_bumper_state = Roach_ReadFrontLeftBumper();

    if (previous_bumper_state == BUMPER_TRIPPED &&
        current_bumper_state == BUMPER_NOT_TRIPPED) {
        //then an event occurred!
        FL_pressed_EventFlag = 1;
    }
    previous_bumper_state = current_bumper_state;
}

int FLBumperReleased()
{
    static char previous_bumper_state = BUMPER_NOT_TRIPPED ;
    char current_bumper_state = Roach_ReadFrontLeftBumper();

    if (previous_bumper_state == BUMPER_NOT_TRIPPED &&
        current_bumper_state == BUMPER_TRIPPED) {
        //then an event occurred!
        FL_released_EventFlag = 1;
    }
```

```
    previous_bumper_state = current_bumper_state;
}

int CheckForTimerEvent()
{
    static char previous_timer_state = TIMER_NOT_ACTIVE;
    char current_timer_state = TIMERS_IsTimerActive(1);

    if (previous_timer_state == TIMER_ACTIVE &&
        current_timer_state == TIMER_NOT_ACTIVE) {
        //then an event occurred!
        timer0_Expired_EventFlag = 1;
    }
    previous_timer_state = current_timer_state;
}

//int darkToLight()
//{
//    static char previous_bumper_state = BUMPER_NOT_TRIPPED;
//    char current_bumper_state = Roach_ReadFrontLeftBumper();
//
//    if (previous_bumper_state == BUMPER_TRIPPED &&
//            current_bumper_state == BUMPER_NOT_TRIPPED) {
//        //then an event occurred!
//            FL_pressed_EventFlag = 1;
//    }
//    previous_bumper_state = current_bumper_state;
//}
//int lightToDark()
//{
//    static char previous_bumper_state = BUMPER_NOT_TRIPPED;
//    char current_bumper_state = Roach_ReadFrontLeftBumper();
//
//    if (previous_bumper_state == BUMPER_TRIPPED &&
//            current_bumper_state == BUMPER_NOT_TRIPPED) {
//        //then an event occurred!
//            FL_pressed_EventFlag = 1;
//    }
//    previous_bumper_state = current_bumper_state;
//}



//void ToggleLED_Service()
//{
//    Roach_LEDSSet(~Roach_LEDSGet());
//}

void pressed(){

    printf("FL_bumper_released\r\n");
```

```
}

void released(){

    printf("FL_bumper_pressed\r\n");


}
//void threeSeconds(){
//
//        printf("-B--\r\r\n");
//
//}
//void fiveSeconds(){
//
//        printf("--C--\r\r\n");
//
//
//}

//void threeSeconds(){
//
//        printf("-B--\r\r\n");
//
//}
//void fiveSeconds(){
//
//        printf("--C--\r\r\n");
//
//
//}

int main(void)
{
    // <editor-fold defaultstate="collapsed" desc="//Initialization code
     (BOARD_Init(), etc)">


    //These calls configure the Pic32 so that it can interact with the Roach
     hardware
    BOARD_Init();
    Roach_Init();
    TIMERS_Init();

    //Initialization code here:
    printf("Welcome to Events and Services lab, compiled on %s %s\r\r\n",
     __TIME__, __DATE__);
    // </editor-fold>
    TIMERS_Init();
```

```
    BOARD_Init();
    Roach_Init();
    //first, setup the framework:
    TIMERS_InitTimer(0,500);

    while (1) {
        printf("%d \n", Roach_LightLevel());
        //continuous services (event checkers):
        FLBumperPressed();
        FLBumperReleased();
        //poll for events:
        if (timer0_Expired_EventFlag = 1){
            if ( FL_pressed_EventFlag == 1) {


                //clear event flag
                FL_pressed_EventFlag = 0;

                //run service
                // ToggleLED_Service();
                pressed();
            }

            if ( FL_released_EventFlag == 1) {


                //clear event flag
                FL_released_EventFlag = 0;

                //run service
                // ToggleLED_Service();
                released();
            }


        }


    }

    return (EXIT_SUCCESS);
}
```