```
/*******************************************************************************
 Module
   Debounce.c

 Revision
   1.0.1

 Description
   This is a simple service under the Gen2 Events and Services Framework.


 Notes

 History
 When           Who     What/Why
 -------------- ---     --------
 01/16/12 09:58 jec      began conversion from TemplateFSM.c
 11/11/13       whg      converted to debounce.c
*******************************************************************************/
/*----------------------------- Include Files -----------------------------*/
/* include header files for this state machine as well as any machines at the
   next lower level in the hierarchy that are sub-machines to this machine
*/
#include "ES_Configure.h"
#include "ES_Framework.h"
#include "Debounce.h"
#include "GameFSM.h"
#include "ES_PostList.h"
#include "BalanceFSM.h"

#include "ES_Timers.h"

/*----------------------------- Module Defines ----------------------------*/
 #define HI 1
/*---------------------------- Module Functions ---------------------------*/
/* prototypes for private functions for this service.They should be functions
   relevant to the behavior of this service
*/

/*---------------------------- Module Variables ---------------------------*/
// with the introduction of Gen2, we need a module level Priority variable
static uint8_t MyPriority;
static unsigned char lastButtonValue;
static unsigned char lastAccessCardValue;
static unsigned char lastBalanceValue;


/*------------------------------ Module Code ------------------------------*/
/*******************************************************************************
 Function
     InitDebounce

 Parameters
```

```
     uint8_t : the priority of this service

 Returns
     bool, False if error in initialization, True otherwise

 Description
     Saves away the priority, and does any
     other required initialization for this service
 Notes

 Author
     J. Edward Carryer, 01/16/12, 10:00
******************************************************************************/
bool InitDebounce ( uint8_t Priority )
{
  ES_Event ThisEvent;

  MyPriority = Priority;
    /********************************************
    in here you write your initialization code*/
    // initialize pins
    BUTTON_DIR = BUTTON_DIR & (~BUTTON_INPUT_PIN);

    // initialize last pin values
    lastButtonValue = BUTTON_PORT & BUTTON_INPUT_PIN;
    lastAccessCardValue = ACCESS_CARD_PORT & ACCESS_CARD_INPUT_PIN;
    lastBalanceValue = BOARD_SWITCH_PORT & BOARD_SWITCH_INPUT_PIN;

    // start debounce timer
    ES_Timer_InitTimer(DEBOUNCE_TIMER, DEBOUNCE_TIME);

    /********************************************/
  // post the initial transition event
  ThisEvent.EventType = ES_INIT;
  if (ES_PostToService( MyPriority, ThisEvent) == True)
  {
      return True;
  }else
  {
      return False;
  }
}

/******************************************************************************
 Function
     PostDebounce

 Parameters
     EF_Event ThisEvent ,the event to post to the queue

 Returns
     bool False if the Enqueue operation failed, True otherwise
```

```
 Description
     Posts an event to this state machine's queue
 Notes

 Author
     J. Edward Carryer, 10/23/11, 19:25
******************************************************************************/
bool PostDebounce( ES_Event ThisEvent )
{
  return ES_PostToService( MyPriority, ThisEvent);
}

/******************************************************************************
 Function
    RunDebounce

 Parameters
   ES_Event : the event to process

 Returns
   ES_Event, ES_NO_EVENT if no error ES_ERROR otherwise

 Description
   add your description here
 Notes

 Author
   J. Edward Carryer, 01/15/12, 15:23
******************************************************************************/
ES_Event RunDebounce( ES_Event ThisEvent )
{
    // ---- Initializations ----
    unsigned char currentButtonValue;
    unsigned char currentAccessCardValue;
    unsigned char currentBalanceValue;

    ES_Event ReturnEvent;
    ReturnEvent.EventType = ES_NO_EVENT; // assume no errors

    // ---- Read in pins ----
    currentButtonValue = BUTTON_PORT & BUTTON_INPUT_PIN;
    currentAccessCardValue = ACCESS_CARD_PORT & ACCESS_CARD_INPUT_PIN;
    currentBalanceValue = BOARD_SWITCH_PORT & BOARD_SWITCH_INPUT_PIN;

    switch ( ThisEvent.EventType ) {
        case ES_INIT:
            //do nothing
            break;

        case ES_TIMEOUT:
            // ---- Check start button ----
            if ( currentButtonValue != lastButtonValue ) {
                if ( currentButtonValue == BUTTON_INPUT_PIN ) {
```

```
            // button down
            ES_Event NewEvent;
            NewEvent.EventType = ES_ButtonDown;
            // Post Event
            PostGameFSM(NewEvent);

            puts("Button press seen!!\n\r");
        } else {
            // button up
            // do nothing
        }
    }
    // ---- Check board balance switch ----
    if ( currentBalanceValue != lastBalanceValue ) {

       //If HI
        if ( currentBalanceValue == BOARD_SWITCH_INPUT_PIN ) {
            // switch down meaning unbalanced
            ES_Event NewEvent;
            NewEvent.EventType = ES_WentOffBalance;
            // Post Event to Balance FSM
            PostBalanceFSM(NewEvent);

            puts("Board went off balance!!\n\r");

        } else {
            // switch up meaning rebalanced
            ES_Event NewEvent;
            NewEvent.EventType = ES_WentOnBalance;
            // Post Event to Balance FSM
            PostBalanceFSM(NewEvent);

            puts("Board is balanced again!!\n\r");
        }
    }

    // ---- Check access card ----
    if ( currentAccessCardValue != lastAccessCardValue ) {
        if ( currentAccessCardValue == ACCESS_CARD_INPUT_PIN ) {
            // button down
            ES_Event NewEvent;
            NewEvent.EventType = ES_AccessCardDetected;
            // Post Event
            ES_PostList00(NewEvent);

            puts("Access Card detected!!!\n\r");
        } else {
            // button up
            ES_Event NewEvent;
            NewEvent.EventType = ES_AccessCardRemoved;
            // Post Event
            ES_PostList00(NewEvent);
```

```
                    puts("Access Card removed!!!\n\r");
                }
            }

            // restart timer
            ES_Timer_InitTimer(DEBOUNCE_TIMER, DEBOUNCE_TIME);
            break;

        default:
            // do nothing or throw error
            ReturnEvent.EventType = ES_ERROR; // should never be here
            puts("Debounce received unexpected event\n\r");
            break;
    }

    // ---- set lastState to currentState ----
    lastButtonValue = currentButtonValue;
    lastAccessCardValue = currentAccessCardValue;
    lastBalanceValue = currentBalanceValue;

    return ReturnEvent;
}

/***************************************************************************
 private functions
 ***************************************************************************/

/*------------------------------ Footnotes -------------------------------*/
/*------------------------------ End of file -----------------------------*/
```