```
/*******************************************************************************
 Module
   UltrasonicService.c

 Revision
   1.0.1

 Description
   This is a template file for implementing 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
*******************************************************************************/
/*---------------------------- 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 <sys/attribs.h>
#include "dbprintf.h"
#include "UltrasonicService.h"
#include "GameplayHSM.h"
#include "SwitchBranchSM.h"
#include "MotorService.h"


#define T3_PERIOD 0xFFFF
#define CLOSE_THRESHOLD 240
#define FAR_THRESHOLD 350

/*---------------------------- Module Defines ----------------------------*/

/*---------------------------- 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 void configUltrasonicInterrupts(void);
static void configTimer3(void);
static void configInputCapture2(void);
static void configInputCapture3(void);


typedef union{
    uint32_t FullCount;
    uint16_t Counters[2];
    //1 is the rollover counter
    //0 is the current count;
```

```
}PulseCount_t;

volatile static PulseCount_t counter; //current time
volatile static uint16_t Period;
volatile static uint16_t PeriodLeft;
volatile static uint16_t PeriodRight;
volatile static uint16_t rollovers;

static Distance_t currentLeftDistance;
static Distance_t currentRightDistance;
static uint16_t currentLeftDistancePeriod;
static uint16_t currentRightDistancePeriod;

/*------------------------------ Module Code ------------------------------*/
/****************************************************************************
 Function
     InitUltrasonicService

 Parameters
     uint8_t : the priorty 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 InitUltrasonicService(uint8_t Priority)
{
       //configure input captures
       //setup timer for IC
       //configure IC interrupts
}


/****************************************************************************
 Function
    RunUltrasonicService

 Parameters
   ES_Event_t : 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_t RunUltrasonicService(ES_Event_t ThisEvent)
{
```

```
      //if event is NEW LEFT PULSE
              //ignore extraneous pulses
              //save the left period

          //determine if left distance is close/far/perfect

          //if we are in the rotate 90 state within the switching branches state
                //if last rotation was CCW
                //if distance is close or perfect
                      //Post ROTATE_DONE to motor service

      //if event is NEW RIGHT PULSE
              //ignore extraneous pulses
              //save the right period

          //determine if right distance is close/far/perfect

          //if we are in the rotate 90 state within the switching branches state
                //if last rotation was CW
                //if distance is close or perfect
                      //Post ROTATE_DONE to motor service


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

static void configUltrasonicInterrupts(void){
    __builtin_disable_interrupts();

    //set IC2 priority
    //set IC3 priority
    //set Timer3Priority

    //Clear any pending interrupt flags

    //Enable IC2, IC3 and T3 interrupts
      //enable global interrupts
}

static void configTimer3(void){
    //disable timer 3
    //CONFIG TIMER 3
    //disable gated time accumulation mode
    //select PBCLK as source
    //prescale of 32
    //Clear timer register
    //set timer period

    //enable timer 3
}


static void configInputCapture2(void){
    //disable IC2
    //operate in idle mode
    //16 bit mode
    //select timer 3
```

```
    //capture every event
    //every edge, specified edge first
    //start with first edge
    //map IC2 to pin B10
    //enable IC2
}

static void configInputCapture3(void){
       //disable IC3
       //operate in idle mode
       //16 bit mode
       //select timer 3

       //capture every event
       //every edge, specified edge first
       //start with first edge
       //map IC3 to pin B8

       //enable IC3
}

void __ISR(_INPUT_CAPTURE_2_VECTOR,IPL7SOFT) LeftUltrasonicCapture(void){

        //read captured time
             //check if rollover flag has been set
                    //increment roll-over counter
                    //clear rollover flag

              //calculate the time difference

               //save the last time
       //keep reading from IC buffer until empty

       //tell ultrasonic service a new pulse is here
        //clear IC flag
}

void __ISR(_INPUT_CAPTURE_3_VECTOR,IPL7SOFT) RightUltrasonicCapture(void){
            //read captured time
             //check if rollover flag has been set
                    //increment roll-over counter
                    //clear rollover flag

              //calculate the time difference

               //save the last time
       //keep reading from IC buffer until empty

       //tell ultrasonic service a new pulse is here
        //clear IC flag
}

void __ISR(_TIMER_3_VECTOR,IPL6SOFT) Timer3Rollover(void){
    //disable global interrupts
      //increment roll-over counter
      //clear flag
    //enable global interrupts

}
```

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