```
/*******************************************************************************
 Module
   SPILeader.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 "SPILeader.h"
#include "PIC32_SPI_HAL.h"
#include "dbprintf.h"
#include "ES_CheckEvents.h"
#include <proc/p32mx170f256b.h>
#include <pic32m_builtins.h>
#include <sys/attribs.h>
#include "GameplayHSM.h"
#include "CalibrationSM.h"
#include "TapeFollowSM.h"
/*---------------------------- Module Defines ----------------------------*/


#define T4_PERIOD 25000 //10ms

//queries you can send to SPI follower - should be passed as eventParams in
//the ES_UPDATE_QUESTION
#define BEACON_QUERY 0x10
#define DISTANCE_QUERY 0x20
#define DUMMY_VALUE 0xAA


//possible Beacon answers
#define BEACON_A_COMMAND 0x1A
#define BEACON_B_COMMAND 0x1B
#define BEACON_C_COMMAND 0x1C
#define BEACON_D_COMMAND 0x1D
#define NO_BEACON_COMMAND 0x1E

//possible distance answers
#define DISTANCE_CLOSE_COMMAND 0x21
#define DISTANCE_FAR_COMMAND 0x22

//possible motor commands
#define LIGHT1_COMMAND_OFF 0x80
#define LIGHT2_COMMAND_OFF 0x90
```

```
#define LIGHT1_COMMAND_ON 0x81
#define LIGHT2_COMMAND_ON 0x91

//gameplay indicator commands
#define GAME_START_COMMAND 0x40
#define GAME_END_COMMAND 0x41




/*---------------------------- Module Functions ----------------------------*/
/* prototypes for private functions for this service.They should be functions
   relevant to the behavior of this service
*/
static void configSPI1Module(void);
static void configSPIInterrupts(void);

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

static uint8_t currQuestion = BEACON_QUERY;
static volatile uint8_t beaconData;


static volatile uint8_t currAnswer;


/*---------------------------- Module Code ----------------------------*/
/****************************************************************************
 Function
     InitSPILeader

 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 InitSPILeader(uint8_t Priority)
{
  //setup SPI1 Module
  //config SPI interrupts

}


/****************************************************************************
 Function
    RunSPILeader
```

```
 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 RunSPILeader(ES_Event_t ThisEvent)
{

      //If INIT event
            //start looping timer
      //If TIMEOUT event
            //post ask follower to self
            //reinit loop timer

      //If UPDATE QUESTIONS event
            //change question to event param
      //If NEW DATA event
          //ignore random data
            //if beacon data
                  //if no beacon: setup NO BEACON event
                  //if beacon A B C or D: setup BEACON FOUND event


                  //check if gameplay is in the right state to post beacon events and
                  if so, post said beacon event

            //if distance data
                  //if distance close: setup DISTANCE CLOSE event
                  //if distance far: setup DISTANCE FAR event
                  //check if gameplay is in the right state to post distance events
                  and if so, post said distance event

      //If ASK FOLLOWER event
            //ask follower the current question


void LightCommand(uint8_t whichLight, bool isOn){
      //if controlling light 1
            //if we want it on
                  //send light 1 on command
            //else send light 1 off command
      //if controlling light 2
            //if we want it on
                  //send light 2 on command
            //else send light 2 off command
}

void gameplayCommand(bool isGamePlaying){
      //if game is starting
            //send game start command
      //otherwise send game end command
}
```

```
/*****************************************************************************
 private functions
 *****************************************************************************/
static void configSPI1Module(void){
    //INITIALIZE SPI
       //disable SPI
       //use PBCLK
   //master mode, disable the analog function of B14, set B14 to an output // B14 = SCK

    //disable analog function of A0, set RA0 to an output, RA0 to the SS output, master
mode slave select control on
    //disable the analog function of A1, set A1 to an output, RA1 to the SDO1
    //disable the analog function of B11, set B11 to an input, RB11 to the SDI1
    //8 bit transfer
    //CKE to 1
    //CKP to 1
    //Enhanced buffer on
    //bit rate to 10kHz
    //enable SPI
    //clear initial flag
}


static void configSPIInterrupts(void){
       //disable global interrutps
       //enable multivector mode
    //set SPI1 priority
    //Clear any pending interrupt flags
    //enable receive interrupt
    //enable global interrupts
}



void __ISR(_SPI_1_VECTOR,IPL7SOFT) SPI1_data(void){
    //receive data flag set
                  //read SPI buffer;
                     //tell service we have new data to process

       //continue reading SPI1BUF until empty
    //clear interrupt flags
}


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