```
/*---------------------------- Include Files ----------------------------*/
#include "ES_Configure.h"
#include "ES_Framework.h"
#include "ES_Timers.h"
#include "ES_Queue.h"
#include "TransmitXbeeSM.h"
#include "ES_Port.h"
#include <stdio.h>
#include "CheckValidSM.h"
#include "InterpretSM.h"

/*---------------------------- Module Defines ----------------------------*/
#define QUEUE_SIZE 4
#define MAX_TRMT_MSG_LENGTH 15
/*---------------------------- Module Functions ----------------------------*/
/* prototypes for private functions for this machine.They should be functions
   relevant to the behavior of this state machine
*/

/*---------------------------- Module Variables ----------------------------*/
// everybody needs a state variable, you may need others as well.
// type of state variable should match that of enum in header file
static TransmitXbeeState_t CurrentState;
static unsigned char TX_MsgArray[MAX_TRMT_MSG_LENGTH+1];
static unsigned char TX_MsgLength = 0;
static unsigned char index = 2;

// everybody needs a local store for the service's priority too!
static uint8_t MyPriority;


/*---------------------------- Module Code ----------------------------*/
/****************************************************************************
 Function
     InitTransmitXbeeSM

****************************************************************************/
boolean InitTransmitXbeeSM ( uint8_t Priority )
{
  ES_Event ThisEvent;
  MyPriority = Priority;
  // put us into the Initial PseudoState
  CurrentState = InitTransmitPState;
  // post the initial transition event
  ThisEvent.EventType = ES_INIT;
  if (ES_PostToService( MyPriority, ThisEvent) == True)
  {
      return True;
  }else
  {
      return False;
  }
```

```
}

/*****************************************************************************
 Function
     PostTransmitXbeeSM
*****************************************************************************/
boolean PostTransmitXbeeSM( ES_Event ThisEvent )
{
  return ES_PostToService( MyPriority, ThisEvent);

}

/*****************************************************************************
 Function
    RunTransmitXbeeSM
*****************************************************************************/
ES_Event RunTransmitXbeeSM( ES_Event CurrentEvent )
{
  ES_Event ReturnEvent;

  ReturnEvent.EventType = ES_NO_EVENT; // assume no problems will occur

  switch ( CurrentState )
  {
      case InitTransmitPState :       // If current state is initial Psedudo State
          if ( CurrentEvent.EventType == ES_INIT )// only respond to EF_Init
          {
              CurrentState = ComposingMsg;
           }
           break;

      case ComposingMsg :
          switch ( CurrentEvent.EventType )
          {
              case ES_CONTROLLER_FOUND :
              {
                unsigned char SUM = 0;
                unsigned int SourceAddByte = ReturnSourceAdd();
                TX_MsgArray[0] = 0x7E;
                TX_MsgArray[1] = 0x00;
                TX_MsgArray[2] = 0x07;
                TX_MsgArray[3] = 0x01;
                TX_MsgArray[4] = 0x01; //Frame ID
                TX_MsgArray[5] = (unsigned char)(SourceAddByte >> 8);//high byte source address
                TX_MsgArray[6] = (unsigned char)(SourceAddByte & 0xFF);//low byte source address
                TX_MsgArray[7] = 0x00; //Options
                TX_MsgArray[8] = 0x03;
                TX_MsgArray[9] = 0xCD;
                for (char i = 3; i <= 9; i++)
                {
                    SUM+= TX_MsgArray[i];
                }
                TX_MsgArray[10] = 0xFF - SUM;
```

```
            TX_MsgLength = 11;
            TXREG = TX_MsgArray[0];
            TX_MsgLength--;
            TXREG = TX_MsgArray[1];
            TX_MsgLength--;
            CurrentState = SendingMsg;
           }
           break;       // end switch on CurrentEvent

        case ES_COMMAND_REC :
          {
            unsigned char SUM = 0;
            unsigned int SourceAddByte = ReturnSourceAdd();
            TX_MsgArray[0] = 0x7E;
            TX_MsgArray[1] = 0x00;
            TX_MsgArray[2] = 0x06;
            TX_MsgArray[3] = 0x01;
            TX_MsgArray[4] = 0x02; //Frame ID
            TX_MsgArray[5] = (unsigned char)(SourceAddByte >> 8);//high byte source address
            TX_MsgArray[6] = (unsigned char)(SourceAddByte & 0xFF);//low byte source address
            TX_MsgArray[7] = 0x00; //Options
            TX_MsgArray[8] = 0x04;
            for (char i = 3; i <= 8; i++)
            {
                SUM+= TX_MsgArray[i];
            }
            TX_MsgArray[9] = 0xFF - SUM;
            TX_MsgLength = 10;
            TXREG = TX_MsgArray[0];
            TX_MsgLength--;
            TXREG = TX_MsgArray[1];
            TX_MsgLength--;
            CurrentState = SendingMsg;
           }
           break;       // end switch on CurrentEvent
      }
      break;            // end switch on CurrentState

  case SendingMsg :
      switch ( CurrentEvent.EventType )
      {
          case ES_TRMT_REG_EMPTY :
          {
             TXREG = TX_MsgArray[index++];
             TX_MsgLength--;
             if(TX_MsgLength == 0)
            {
                CurrentState = ComposingMsg;
                index = 2;
            }
           }
           break;       // end switch on CurrentEvent
```

```
            case ES_TIMEOUT :
              {
                 CurrentState = ComposingMsg;
                 index = 2;
               }
               break;       // end switch on CurrentEvent
          }
          break;            // end switch on CurrentState


  }                                 // end switch on Current State
  return ReturnEvent;
}


/*****************************************************************************
 Function
     QueryTransmitXbeeSM
*****************************************************************************/
TransmitXbeeState_t QueryTransmitXbeeSM ( void )
{
   return(CurrentState);
}

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