```
//#define STANDALONE
/****************************************************************************
 Description
   This module is a timer service to emit 10 mS HI and 30 mS LO pulses to
   retrieve balls

 Notes


 Author
   Jina Wang 02/27/2013

****************************************************************************/

/*----------------------------- 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 <stdio.h>
#include <stdlib.h>
#include <mc9s12e128.h>
#include <S12e128bits.h>
#include <Bin_Const.h>
#include <termio.h>
#include <hidef.h>
#include "S12eVec.h"

#include "E128_PWM.h"        //has all prescale definitions
#include "E128_SPI.h"
#include "E128_Servo.h"
#include "FAC_FSM.h"
#include "NavigationFSM.h"
#include "AlignPPService.h"
#include "DriveTrainService.h"
#include "ArtilleryFSM.h"
#include "ResupplyService.h"

/*----------------------------- Module Defines ----------------------------*/
#define PERIOD 7500 //10 mS time

/*---------------------------- Module Variables ---------------------------*/
//Static Variables


/*---------------------------- Module Functions ---------------------------*/
//Private functions
void interrupt _Vec_tim1ch4 IREmitResponse(void);


/*------------------------------ TIMER CODE ------------------------------*/

void InitIREmitHardware(void)
//Set up IC5 to detect alignment with enemy's power plant
{
    //Enable timer system 1 and prescale values
    TIM1_TSCR1 = _S12_TEN;     //enable timer system
    TIM1_TSCR2 = PSCALE32;     //prescale to /32, 1 tick = 1.33 uS

    TIM1_TIOS |= _S12_IOS4; //set output compare
    TIM1_TC4 = TIM1_TCNT + PERIOD; // schedule first rise
    TIM1_TFLG1 = _S12_C4F;     //clear any existing flag coming out of reset

    //DO NOT ENABLE ON INITIALIZATION B/C BEAM WILL BE EMITTING ALWAYS
    //TIM1_TIE |= _S12_C4I;      //enable local interrupt
    EnableInterrupts;          //globally enable interrupts

    DDRT |= IREMIT_DIR; //set PTT_PTT4 to output
    IREMIT_PORT = LO; //initialize LO

    DDRT &= ~BACKUP_DIR; //set PTT_PTT3 input

}

//Interrupt Response
void interrupt _Vec_tim1ch4 IREmitResponse(void)
{
    static int PulseCount = 0;

    TIM1_TFLG1 = _S12_C4F; //clear flag

    //Pulse IR 10 times
    if (IREMIT_PORT == LO)
       {
          IREMIT_PORT = HI; //set HIGH
          TIM1_TC4 += PERIOD; //hold hi for 10 mS
          PulseCount++;
       }
    else if ( IREMIT_PORT == HI)
       {
          IREMIT_PORT = LO; //set LOW
          TIM1_TC4 += (3 * PERIOD); //hold lo for 30 mS
       }
}


/****************************PUBLIC FUNCTION*******************/
void EmitResupplySignal(boolean input_)
{
    if (input_ == True)
       {
          //enable interrupts on this channel
          TIM1_TIE |= _S12_C4I;
       }
    else
       {
          TIM1_TIE &= BIT4LO;
       }
}

/*****************************************************************/

#ifdef STANDALONE

void main ( void )
{
    InitIREmitHardware();
}
#endif
```