```c
#include "thePITS.h"
#include "xparameters.h"
#include "xil_io.h"

#define TICKS_IN_ONCE_SECOND 100000000 //Ticks in a second
#define CONFIG_REGISTER XPAR_PIT_0_BASEADDR + 4 //Address of config register
#define INITIAL_TICK_PER_SECOND 50 //Initial ticks per second
#define ALL_ENABLED 0xffffffff//Mask for enabling everything
#define RELOAD_BIT_INDEX 2//Bit index of reload
#define INTERRUPT_ENABLE_BIT_INDEX 1//Bit index of interrupts enable

//Init called before anything else
void thePITS_init(){
    Xil_Out32(XPAR_PIT_0_BASEADDR, TICKS_IN_ONCE_SECOND / INITIAL_TICK_PER_SECOND);//Set initial
count down from value
    Xil_Out32(CONFIG_REGISTER, ALL_ENABLED);//Set interrupts enabled, and count down, and
counting
}

//Sets count down from value
void thePITS_setTicksCountDownFrom(uint32_t tickToCountDownFrom){
    Xil_Out32(XPAR_PIT_0_BASEADDR, tickToCountDownFrom );//Write value to register
}

//Sets or clears enable bit in control register
void thePITS_setCountEnableDisable(uint8_t enabled){
    uint32_t prevRegVal = Xil_In32(CONFIG_REGISTER);//Store old value of register
    if(enabled){//If setting
        Xil_Out32(CONFIG_REGISTER, prevRegVal | 1 );//Set bit
    }else{//If clearing
        Xil_Out32(CONFIG_REGISTER, prevRegVal & ~1 );//Clear bit
    }
}

//Sets or clears interrupt enable bit in control register
void thePITS_setInterruptsEnableDisable(uint8_t enabled){
    uint32_t prevRegVal = Xil_In32(CONFIG_REGISTER);
    if(enabled){//If setting
        Xil_Out32(CONFIG_REGISTER, prevRegVal | (1 << INTERRUPT_ENABLE_BIT_INDEX) );//Set bit
    }else{//If clearing
        Xil_Out32(CONFIG_REGISTER, prevRegVal & ~(1 << INTERRUPT_ENABLE_BIT_INDEX) );//Clear bit
    }
}

//Sets or clears reload enable bit in control register
void thePITS_setReloadEnableDisable(uint8_t enabled){
    uint32_t prevRegVal = Xil_In32(CONFIG_REGISTER);
    if(enabled){//If setting
        Xil_Out32(CONFIG_REGISTER, prevRegVal | (1 << RELOAD_BIT_INDEX) );//Set bit
    }else{//If clearing
        Xil_Out32(CONFIG_REGISTER, prevRegVal & ~(1 << RELOAD_BIT_INDEX) );//Clear bit
    }
}
```