Lab #6 LED Ping Pong Game on Zybo with Polling

Name: Cunyang Liu ID:U201811446 Date:2021/08/18

Code

- pingpongGame.c

```c
//pingpong template file for Lab #6
//Revised by Cunyang Liu to add pressing early penalty
//August 2021
#include "xparameters.h"
#include "xgpio.h"
#include "led_ip.h"
// Include scutimer header file
#include "XScuTimer.h"
//=====================================================
XScuTimer Timer; /* Cortex A9 SCU Private Timer Instance */
void delay(void);
void MoveBallRight(void);
void MoveBallLeft(void);
void Game(int);
void switchSpeed(void);

#define ONE_TENTH 32500000 // half of the CPU clock speed/10
#define START 1
#define STOP  0
#define LEFT  0
#define RIGHT 1
#define RESETBUTTON 0b0100
#define STARTBUTTON 0b0010
#define LEFTPADDLE  0b1000
#define RIGHTPADDLE 0b0001

int psb_check, dip_check, dip_check_prev, LedState, Status;
XGpio dip, push;
// PS Timer related definitions
XScuTimer_Config *ConfigPtr;
//XScuTimer_Config *TimerConfigPtr;
XScuTimer *TimerInstancePtr = &Timer;

int LED_PATTERNS[4]={0b1000, 0b0100, 0b0010,0b0001};
int scoreright, scoreleft;
char GameOver, StartDirection;

int main(void) {
    //initialize variables,ports
    xil_printf("-- Start of the Ping Pong Program --\r\n");

    XGpio_Initialize(&dip, XPAR_SWITCHES_DEVICE_ID);
    XGpio_SetDataDirection(&dip, 1, 0xffffffff);
    XGpio_Initialize(&push, XPAR_BUTTONS_DEVICE_ID);
```

```
    XGpio_SetDataDirection(&push, 1, 0xffffffff);

    //initialize timer
    ConfigPtr = XScuTimer_LookupConfig(XPAR_PS7_SCUTIMER_0_DEVICE_ID);
    Status = XScuTimer_CfgInitialize(TimerInstancePtr, ConfigPtr,
ConfigPtr->BaseAddr);

    if (Status != XST_SUCCESS) {
        xil_printf("Timer init() failed\r\n");
        return XST_FAILURE;
    }
    switchSpeed();

    //set autoload mode
    XScuTimer_EnableAutoReload(TimerInstancePtr);

    //start timer
    XScuTimer_Start(TimerInstancePtr);

    xil_printf("-- Start of the Ping Pong Program --\r\n");
    GameOver = STOP;
    scoreright = 0;
    scoreleft = 0;
    xil_printf("Score Left = %d Score Right = %d\r\n", scoreleft,
scoreright);

    int resetOneShot = 0, startPlayer = LEFT;
    while (1) {
        switchSpeed();

        //read reset score and push buttons if Button 2 is pressed
        psb_check = XGpio_DiscreteRead(&push, 1);

        //reset the game
        if (psb_check == RESETBUTTON && resetOneShot == 0) {
            resetOneShot = 1;
            GameOver = STOP;
            scoreright = 0;
            scoreleft = 0;
            LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0, 0b0000);
            xil_printf("\n\rNew Game - Scores Reset\r\n");
        }

        if (psb_check == STARTBUTTON) {
            //start game
            GameOver = START;
            xil_printf("\n\rGame Start\r\n");

            //start the game and follow startdirection
            startPlayer = !startPlayer;
            Game(startPlayer);

            //game end
            resetOneShot = 0;
            xil_printf("\n\rGame End\r\n");
            xil_printf("Score Left = %d Score Right = %d\r\n",
scoreleft,scoreright);
        }
```

```
    }
}

void Game(int startPlayer) {
    if (startPlayer == LEFT) {
        LedState = 0;
        StartDirection = RIGHT;
    } else {
        LedState = 3;
        StartDirection = LEFT;
    }

    //clear time counter
    delay();

    LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0,
LED_PATTERNS[LedState]);
    delay();

    while (GameOver == START) {
        if (StartDirection == LEFT) {
            MoveBallLeft();
        } else {
            MoveBallRight();
        }
    }
}

void MoveBallRight(void) {
    char EarlyPress;
    EarlyPress = 0;

    //move LED to right
    LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0,
LED_PATTERNS[++LedState]);
    delay();

    //check for button pushing
    if (psb_check == RIGHTPADDLE) {
        EarlyPress = 1;
    }

    //set startDirection
    if (LedState == 3 && EarlyPress == 1) {
        StartDirection = LEFT;
    }

    //set gameover and display scores
    else if (LedState == 3) {
        GameOver = STOP;
        scoreleft++;
    }
}


void MoveBallLeft(void) {
    char EarlyPress;
    EarlyPress = 0;
```

```
156
157     //move LED to left
158     LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0, LED_PATTERNS[--
    LedState]);
159     delay();
160
161     //check for button pushing
162     if (psb_check == LEFTPADDLE) {
163         EarlyPress = 1;
164     }
165
166     //set StartDirection
167     if (LedState == 0 && EarlyPress == 1) {
168         StartDirection = RIGHT;
169     }
170
171     //set gameover and display scores
172     else if (LedState == 0) {
173         GameOver = STOP;
174         scoreright++;
175     }
176 }
177
178 void delay(void) {
179
180     //Load timer with delaying in multiple of ONE_T
181     int psb = 0;
182     while (!XScuTimer_IsExpired(TimerInstancePtr)) {
183         psb_check = XGpio_DiscreteRead(&push, 1);
184         if (psb == 0) {
185             psb = psb_check;
186         }
187     }
188
189     //clear status bit
190     XScuTimer_ClearInterruptStatus(TimerInstancePtr);
191     psb_check = psb;
192 }
193
194 void switchSpeed(void) {
195     dip_check = XGpio_DiscreteRead(&dip, 1);
196     if (dip_check != dip_check_prev) {
197         xil_printf("Switch Game Speed: %d\r\n", dip_check);
198         dip_check_prev = dip_check;
199
200         //load tiemr with new switch settings
201         XScuTimer_LoadTimer(TimerInstancePtr, ONE_TENTH * dip_check);
202     }
203 }
```