Name: _____ ID: _____ Start Date: Friday, June 7, 2019
Due Date: Monday, June 10, 2019

Software and Hardware Co-Design with Zybo, Spring 2019 HUST
Lab #7 LED Ping Pong Game on Zybo with Polling

This is an individual lab. Each student must perform it and demonstrate this lab to obtain credit for it. Late lab submission will be accepted with a grade reduction of 10% for each day that it is late.

A template is provided to demonstrate a well-structured program and well defined variables. You are welcome to use this template to implement your program. You are also encouraged to write your own program from scratch.

1 Demonstration and Code Submission by Monday, June 10, 2019

1. Demonstrate your LED ping-pong game on your Zybo board.
2. Submit a pdf copy of your program to the instructor. Your program must have a header to include your name, date and brief title. All variables of your program must be declared with meaningful English phrases. It must be well commented and It should not have any magic number or magic port.

2 Objectives

Using your hardware implementation from Labs #5 or #6, write a program in C that makes your Zybo board play "LED ping-pong". Two paddles are two pushbuttons and one LED is the ball. The score will be displayed on the SDK terminal display using xil_printf( ) statements. Your program will read switches continually in polling mode. Your program must make use of the XScuTimer to control the speed of the ball movement. Note that no new hardware is needed for this assignment!

3 Specification

1) The score should be displayed in the SDK serial port window on your laptop PC. You are free to design your scorer display.

2) Momentarily depressing BTN2 should serve a "ball"; that is, a single lit LED should appear at the left end of the line of four LEDs (LD3) and the ball (the lit LED) should begin moving from the left end (LD3) of the line of LEDs toward the right end (LD0) at a variable speed (that is selected by the binary number entered on the slide switches).

3) When the ball (the lit LED) reaches the rightmost position (LD0), the right pushbutton (BTN0), which may be regarded as the paddle of the right player, should be depressed (it may NOT be pushed ahead of this time!) in order to hit the ball back in the other direction. If the right pushbutton is not pressed during this time, or if it happens to have been pressed too early before the ball arrives at LD0, or too late after the ball has traveled past LD0, the ball should "fall off" of the end of the line of the 4 LEDs (the ball vanishes… no LEDs remain lit), and the serve terminates. If this happens, play terminates, and one point should be added to the left player's score.

4) Assuming the ball was correctly hit back toward the left, then the left push button (BTN3) must be pressed when the ball eventually arrives back at the leftmost LED position (LD3), in order to send it back the other way. If the left push button is not pressed at this crucial time while the ball is at

LD3, then the right player's score is incremented.

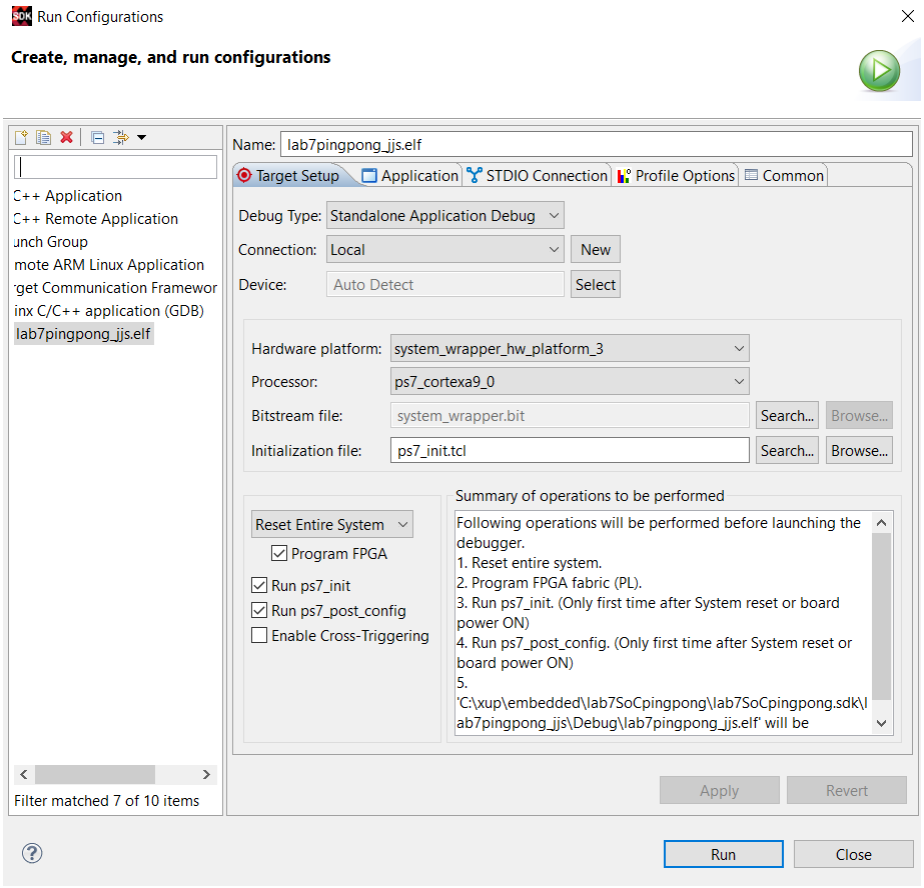
5) *Once the serve has terminated, and a point has been awarded to either the right or left player, the players must wait for the serve button (BTN2) to be momentarily depressed once again. Then the ball must be served in the other direction (the direction of the serve should alternate with each press of the center button).*
6) *The speed of play (difficulty of the game) should be adjustable in 16 steps using four on-board slide switches. A setting of 15 should be very slow, and a setting of 0 should be quite fast, though still be (barely) playable by an expert!*

4 References

1. Xilinx University Program: Embedded System Design Flow on Zynq using Vivado.
2. Private Timer device drivers for Cortex A9, scutimer v2_0, Xilinx.
3. General purpose I/O (XGpio) device drivers, gpio v4_0, Xilinx.

5 Ron Configuration to Reset Entire System

It is always a good idea to enable Run Configuration to "Reset Entire system".



6 Appendix A template program

```
/pingpong template file for Lab #7
//Revised by Jianjian Song to add pressing early penalty
//June 2019
#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);

#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, LedState, Status;
XGpio dip, push;
// PS Timer related definitions
XScuTimer_Config *TimerConfigPtr;
XScuTimer *TimerInstancePtr = &Timer;

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

int main (void)
{
unsigned int i;

  //initialize variables, timers, ports

  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", scoreright, scoreleft);
  StartDirection=LEFT;
```

```
  while (1)
  {
                // Read push buttons and reset score if Button 2 is pressed
                    psb_check = XGpio_DiscreteRead(&push, 1);
                if(psb_check== RESETBUTTON)    //reset game
                  {
                    xil_printf("\n\rNew Game - Scores Reset\r\n");
                          }
                if(psb_check == STARTBUTTON)   GameOver=START;      //start game
                        //start the game and follow StartDirection}
  }             //while(1)
} //main()

void MoveBallRight(void) {
char i, EarlyPress;
                    EarlyPress=0;
//move LED to the right
// check for button pushes
//set StartDirection
//set GameOver; display scores
}

void MoveBallLeft(void) {
char i, EarlyPress;
                EarlyPress=0;
//move LED to the left
// check for button pushes
//set StartDirection
//set GameOver, update and display scores
}


void delay(void){
                // Read dip switch values
                 dip_check = XGpio_DiscreteRead(&dip, 1);
                 // Load timer with delay in multiple of ONE_T, Timer is a count-down timer
}
```