Name: ____________________ ID: __________ Start Date: Tuesday, August 17, 2021
Due Date: Thursday, August 19, 2021

Software and Hardware Co-Design with Zybo, Summer 2021 HUST
Lab #7 Interrupt-Driven Ping-Pong Game on Zybo

This is an individual lab. Each student must perform it on his or her own 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.

1 Objectives

- *Part 1: Perform the Zynq Book Tutorial 2, Next Steps in Zynq SoC Design and submit a memo on this part.*
- *Part 2: Modify Lab #6 LED Ping Pong Game to use input change interrupt and timer interrupt. Input change interrupt service will handle four push buttons and four slide switches.*
- *Timer interrupt service will handle LED display. The main program will handle terminal display only after initialization.*
- *Whenever the slide switches change, an interrupt should occur to change the delay time of the timer right away. Push button interrupt causes game play status to change.*
- *Create an ASM chart or pseudo code to describe the behavior of your interrupt-driven ping-pong program.*

2 Available source code as a template

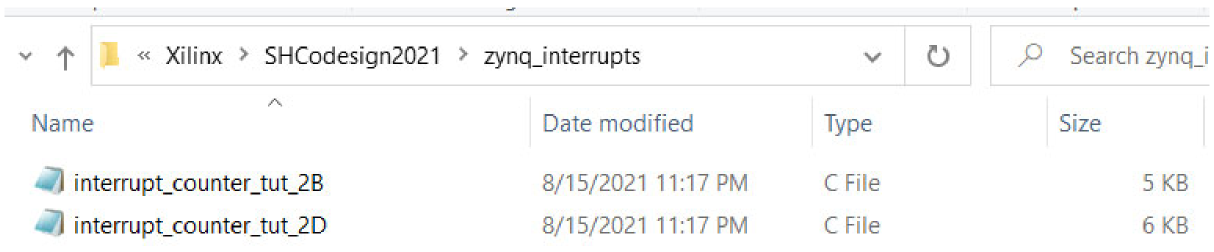
interrupt_counter_tut_2D_new.c is a template to start Part 2 of this lab. Notice that the code is more modular than the original tutorial programs in that each device is initialized by a subroutine, which also initializes local interrupt. GIC initialization is a separate subroutine for GIC and ARM core CPU interrupt configuration only.

3 Demonstration and Report by Thursday, August 19, 2021

Submit a pdf memo of some typical screen captures of the Zynq Book Tutorial 2 to show you have completed it as Part 1 of this lab. Demonstrate your board with your interrupt-driven ping-pong game and submit a pdf copy of your ASM chart or pseudo code as well as your final source code.

4 Part 1: The Zynq Book Tutorial 2: Next Steps in Zynq SoC Design

Obtain source code folder for this tutorial, The_Zynq_Book_Tutorials_Sep_14.zip from http://www.zynqbook.com/download-tuts.html. Copy The_Zynq_Book_Tutorial_Sources/sources/zynq_interrupts folder to C:\Xilinx\SHCodesign2021/.



« Xilinx > SHCodesign2021 > zynq_interrupts

Search zynq_i

Name	Date modified	Type	Size
interrupt_counter_tut_2B	8/15/2021 11:17 PM	C File	5 KB
interrupt_counter_tut_2D	8/15/2021 11:17 PM	C File	6 KB

Follow Tutorial #2 in the Zynq Book Tutorials v1.2, April 2014 [6] to complete the interrupt tutorial examples Exercise 2B and 2D. Write a memo with typical screen shots from this lab to show you have completed this tutorial.

5 Part 2: Interrupt-driven Ping-Pong Game

Implement the ping-pong game from Lab #6 with interrupt services. Your Lab #7 game should have the same behavior as that of Lab #6 but responds timer overflow and push button and slide switch changes through interrupts.

5.1 The Architecture of the Interrupt Driven LED Ping-Pong Game

This game is composed of four major subroutines: main(), push button interrupt handler, slide switch interrupt handler and timer interrupt handler. The main program will set up interrupts and initialize all ports and timer. The three interrupt handlers will deal with push buttons, slide switches and timer interrupts. The main program will then print on the terminal of the game status and results. All xil_printf() or printf() must be in the main() loop so that interrupt handlers take as little time as possible. It is not a good idea to use both xil_printf() and printf() as they are independent of each and may not display text in the right order.

Push button interrupt handler will decide which button is pressed and set game status and result accordingly. Slide switch interrupt handler will load Timer Load Register with new delay time count. Note: `XGPIO_IR_CH1_MASK` specifies the source of interrupt is from channel 1 of a gpio interface. If the source is from channel 2 of the gpio interface, `XGPIO_IR_CH2_MASK`, should be used.

Timer interrupt handler will update LED display.

Global status and result variables are used to communicate among the three major subroutines. Your code should not have any magic number or magic port.

5.2 An ASM chart or a Pseudo Code

Create either an ASM chart or a pseudo code for your interrupt-driven ping-pong code before starting to write your code. Include this chart or pseudo code as part of your deliverable.

5.3 Requirements for the Ping-Pong Game

The external behavior of the game is mainly the same as in Lab #6 except for the push buttons and slide switches, which are interrupt driven. The game will reset as soon as the reset button is pressed. The game serve will start as soon as the serve button is pressed no matter if the game is over. The slide switch change will change the delay time of the timer right away.

1) *Momentarily depressing reset button BTN2 should reset the score of the game. The score should be displayed in the SDK serial port window on your laptop PC.*

2) *Momentarily depressing serve button BTN1 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. Each player will take turn to serve.*

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 (BTN1) 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 the 4 on-board DIP 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! This speed change is instantaneous by slide switch change interrupt as soon as the slide switches change their states.*

6 The Architecture of the Interrupt Driven LED Ping-Pong Game

This game is composed of three major subroutines: main(), push button interrupt handler and timer interrupt handler. The main program will set up interrupts and initialize all ports and timer. The main program will then print on the terminal of the game status and results. The two interrupt handlers will deal with push buttons and timer interrupts.

Push button interrupt handler will decide which button is pressed and set game status and result accordingly.

Timer interrupt handler will update LED display.

Global status and result variables are used to communicate among the three major subroutines.

7 References

1. Private Timer device drivers for Cortex A9, scutimer v2_0, Xilinx.
2. General purpose I/O (XGpio) device drivers, gpio v4_0, Xilinx.
3. How to Use Interrupts on the Zynq SoC, Xcell Journal 87
4. xil_exception.c, standalone_v4_2, from C:\Xilinx\SDK\2014.4\data\embeddedsw\lib\bsp\standalone_v4_2\src\cortexa9
5. Chapter 3: Interrupt Controller, Cortex™-A9 MPCore, Revision: r2p2, Technical Reference Manual
6. The Zynq Book Tutorials, v1.2 - September 2014, University of Strathclyde, Glasgow, Scotland, UK