Lab 5 – Combinational Logic Shift

Name: Von Kaukeano *Date*: 10/9/18

Course: Digital Circuit Design Lab – ECE 2613 *Section #*: 001

Summary/Abstract

In this lab we wired an eleven-bit data in input to an eight-bit data out output. Three of the eleven inputs are logical inputs and the rest are wired to the LED lights. We then used combinational shift logic in order to shift our bits. When LEDs are selected with switches high the inputs are then set and the LED bits are shifted based on the given logic table. We displayed right, left, arithmetic, circular, and barrel shifts within this lab.

Introduction

Background material for this lab would consist of understanding the basics of shifting and the binary system works. It is important to know how truncating of the least significant bits occur when doing an arithmetic shift and also the outcome when padding with a one. When using a circular shit that concatenations are used to perform it. These are all shifts that move the bits a specific way.

Procedure

Using the given truth table, we designed our combinational logic shifter. The switches from ten to eight were the logic switches and the rest from seven to zero selected its corresponding LED's. When selecting an LED on, then changing the logic switches to a shift, the board displays the LED moving corresponding to the logic. For example, LED light bit one is selected by the switch. Then the logic switches are changed to do a single bit shift to the right, you will see the zero-bit light lit up instead of the first which shifted.

Results

Design Code

```
//
// lab5 : version 10/01/2018
//
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////////
module comb_shifters (output logic [7:0] data_out, input logic [2:0] select,
    input logic [7:0] data_in);

    // enter your code here
always @* begin
case(select)

    3'b001: data_out = data_in << 1;
    3'b010: data_out = data_in >> 1;
    3'b011: data_out = {data_in[6:0],data_in[7]};
    3'b100: data_out = {data_in[0],data_in[7:1]};
    3'b101: data_out = $signed(data_in) >>> 1;
    3'b110: data_out = {data_in[4:0],data_in[7:5]};
    3'b111: data_out = $signed(data_in) >>> 5;
    default: data_out = data_in;
endcase
end

endmodule
```

Simulation Results

```
****** xsim v2018.2 (64-bit)
  **** SW Build 2258646 on Thu Jun 14 20:02:38 MDT 2018
  **** IP Build 2256618 on Thu Jun 14 22:10:49 MDT 2018
    ** Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.

source xsim.dir/work.tb_comb_shifters/xsim_script.tcl
# xsim {work.tb_comb_shifters} -autoloadwcfg -tclbatch {tb_comb_shifters.tcl} -onerror quit
Vivado Simulator 2018.2
Time resolution is 1 ps
source tb_comb_shifters.tcl
## run all
Simulation complete - no mismatches!!!
$finish called at time : 40960 ns : File "/home/tuh42003/2613_2018f/lab5/tb_comb_shifters.sv" Line 65
## exit
INFO: [Common 17-206] Exiting xsim at Fri Oct  5 23:18:44 2018...
Compressing vcd file to lxt2 file. ...
```

Hardware Implementation

My design was demonstrated to Catherine on the afternoon of 10/2 during the lab period.

Conclusion

This lab demonstrates the understanding of significant and sign bits along with the different Verilog designs to shift bits. The different shifts from up, down, arithmetic, circular, and barrel are all different types of shifts that have different characteristics based on what is needed.

Appendix

For 3'b010:

Input- 8’b00000010

Output-8’b00000100

It is shifted one bit to the left and then padded with a zero. Leaving the bit it was moved from 0 or off.

For 3’b011:

Input- 8’b00011000

Output- 8’b00001100

It is arithmetic shifted to the right padded with sign. Since its significant bit was one then it is padded with a one or on.

For 3’b100:

Input- 8’b00011000

Output- 8’b00110000

It is circular shifted one bit to the left. The most significant bit is concatenated and padded the opposite side.