Project 2 Basic Logic Circuits

Design and Implementation of SOP and POS circuits

Introduction

This project provides further experience with logic circuits. First, you are asked to **download** a predesigned FPGA configuration (a **.bit file**) to configure your Boolean with several logic circuits. Your assignment is to interact with the circuits, discover their response to all inputs, deduce their SOP and POS logic equations, and then re-implement them yourself. Next, two circuit requirements are presented as worded descriptions. After capturing the requirements in a truth table, you are asked to design a circuit and check its performance on the Boolean board. Finally, you are asked to create another circuit using the design methods discussed. Many new terms and concepts are introduced, and these are discussed in the topic documents.

Before you begin, you should:

- Have the Xilinx Vivado tool installed;
- Have a Boolean board and know how to program it;
- Know how to start a Vivado project and construct a basic logic circuit;
- Understand logic gates and basic logic circuits.

After you are done, you should:

- Be comfortable creating new designs in Vivado;
- Be able to derive logic functions from truth tables;
- Be able to define SOP and POS logic functions in Verilog;
- Be able to translate a worded problem description into Verilog.

Background

In this project, you will take some significant steps towards designing your own digital logic circuits. The design process involves many steps that are best learned through doing actual designs, but some fundamental background knowledge will help that process – read the **theory** topics!

Requirements

☑ 1. Configure your Boolean board with a downloaded .bit file, and deduce the logic circuits it contains

Download and **decompress** the pre-compiled bitstream from the given **zip** file. Open the **Hardware Manager** from Vivado's **mainIDE**, **auto-connect** to the Boolean board, select "**Program Device**", and select the **.bit** you just downloaded. The .bit file will configure your Boolean board with four different circuits that use slide switches as inputs and LEDs for outputs. After your board is programmed, you can interact with the circuits, **complete the truth tables** below to document their behavior, and then **deduce their design.** Note you do not need to create a project for this task.

Circuit 1

The first circuit uses slide switches **SW0** and **SW1** as inputs and **LD0** as an output. Use the switches to apply all four possible input patterns, and record LD0's response in the table below (note LD0 will illuminate when the circuit's output is a '1').

SW1	SW0	LD0
0	0	
0	1	
1	0	
1	1	

Circuit 2

The second circuit uses **SW1**, **SW2,** and **SW3** as inputs, and **LD1** as an output. Probe Circuit 2 and complete the truth table below.

SW3	SW2	SW1	LD1
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Circuit 3

The third circuit uses **SW4**, **SW5**, **SW6**, and **SW7** as inputs, and **LD2** as output. Probe Circuit 3 the same way you did Circuit 1 and 2, complete the truth table below.

SW7	SW6	SW5	SW4	LD2
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

☑ 2. Create your own duplicate circuits

After you have completed the truth tables in the first task, create your own circuits to implement the same behavior as the three circuits you just probed. **Create a new Vivado project**, and add a new Verilog **source** file. Define **8 slide switches as inputs, and three LEDs as outputs.** Then, **based on your truth table results**, write **three Verilog assignment statements** (using **SOP** logic) to drive the **three LEDs** exactly as they were driven in the downloaded file. Your completed Verilog module will have three assignment statements; the first one is provided, but you must write the other two. Your Verilog should be similar to the code below.

```
module top (
    input [7:0] sw,
    output [2:0] led
);

assign led[0] = (sw[0] & ~sw[1]) | (~sw[0] & sw[1]);
assign led[1] = //you must write this statement
assign led[2] = //and this one too

endmodule
```

Create a new **constraints** file, and connect the Verilog "**sw**" inputs and "**led**" outputs to the proper **pins** on the device.

Note: Instead of typing all the entries, you can **download** the complete Boolean board **constraints** file and add it to your project. Note that we selected signal names for all the peripheral devices, so you'll need to use **those samenames in your Verilog source file**, **or change the names in the .xdc** file to match whatever names you used in your source file.

When your Verilog and constraints source files are complete, **implement** the design, **program** the Boolean board, and check the **results**.

Note: In Vivado, if you execute a process that uses outputs from one or more previous processes, Vivado will automatically check to make sure all previous processes and files are up to date. If any files needed to be updated, Vivado will automatically run any required processes.

☑ 3. Circuit 4

Program your Boolean board with the **.bit** file you downloaded in requirement 1. That .bit file contains a fourth circuit that uses **SW15 – SW12** as inputs, and **LD15 - LD12** as an output. **SW11** is used as an "**enable**" signal; that is, this fourth circuit will only work if SW11 is set to a '1'.

Probe this fourth circuit, create a new Verilog **source** file that behaves identically, including the **SW11** enable signal. **Program** your Boolean board, and **verify** it performs identically.