

San Jose State University  
Department of Computer Engineering

CMPE 125 Lab Report

Lab 1 Report

Title Design Entry and Functional Verification

Semester FALL 2019

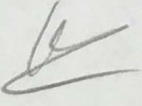
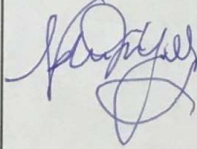
Date 09/09/2019

by

Name Phat Le  
(typed)

SID 012067666  
(typed)

Lab Checkup Record

Week	Performed By (signature)	Checked By (signature)	Tasks Successfully Completed*	Tasks Partially Completed*	Tasks Failed or Not Performed*
1			✓		

\* Detailed descriptions must be given in the report.

## Introduction

The purpose of this lab is to go through the design entry and functional verification procedure using Vivado. The simple AND-OR combinational logic will be used as a sample test run for the process as shown in Figure 1 below.

## Design Methodology

Given the schematics of a combinational circuit of 2 AND gates and 1 OR gate, the task is to understand the given design module (*figure 2,3,4,5*) and the design code. Then, these codes need to be simulated in Vivado software. The last task is to apply the given testbench code as it compared the actual output of the source design code.

Table 1: Function of design modules

Module	Function
and2	2-input AND gate
or2	2-input OR gate
and_or	Top Level Module

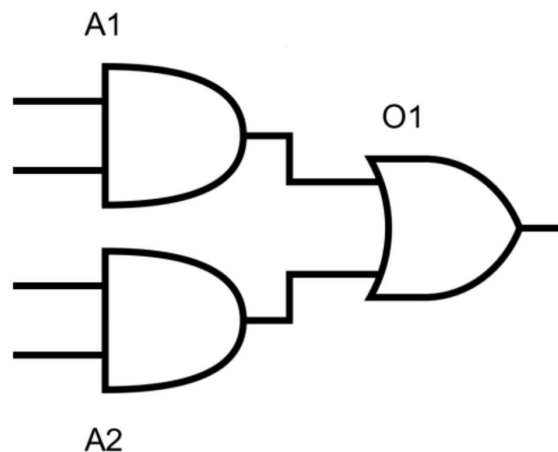


Figure 1: Schematic of a simple combinational circuit

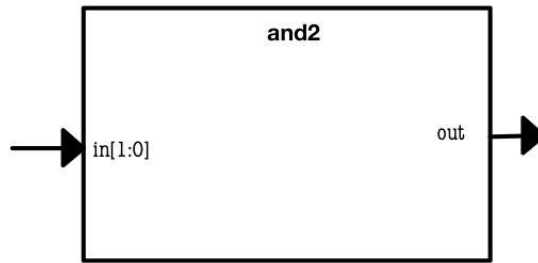


Figure 2: Block diagram of `and2.v` module

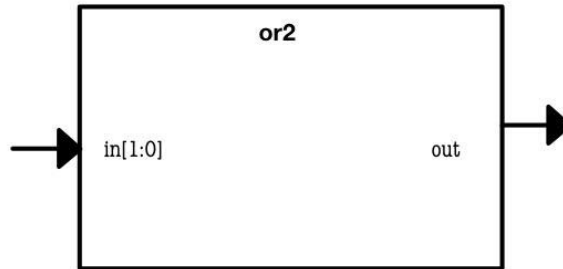


Figure 3: Block diagram of `or2.v` module

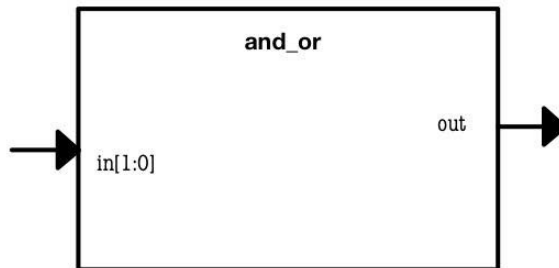


Figure 4: Block diagram of `and` and `and_or.v` module

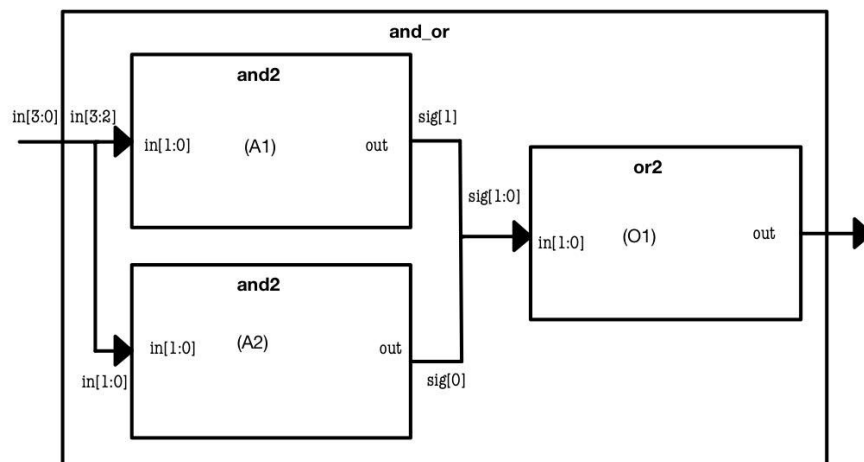


Figure 5: Block diagram of the interior of `and_or` module

## Simulation Results

The extra waveform [3], [2], [1], [0] are the 4 inputs signal. The module's output *tb\_out* matches the expected output *tb\_output\_expected* produced by the self-checking simulation file as shown in *Figure 6* below.

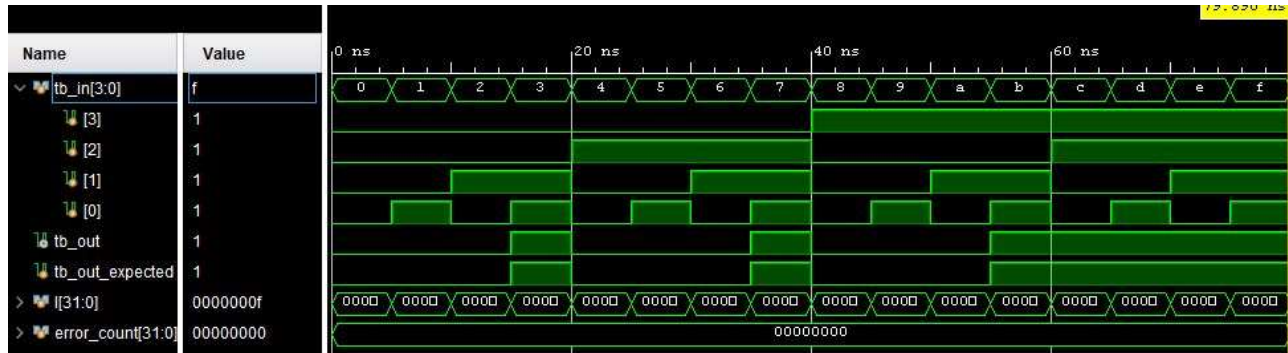


Figure 6: Simulation waveforms produced using Vivado

## FPGA Validation

This lab is not required FPGA Validation step.

## Conclusion

The program has successfully built as it produces the waveform that matches the expected waveform. Ultimately, this lab has provided important knowledge that can support future lab assignment. Specifically, it helps me understand the syntax to translate the design module into code as well as verify it using another test code.

One problem that I met during this lab is not to create the wrong type of file. For instance, constrain, simulations, and design files are different from each other.

## Appendix

### Design code

#### and2.v

```
module and2 (
    input    wire [1:0] in,
    output wire          out
);

    assign out = in[1] & in[0];

endmodule
```

#### or2.v

```
module or2 (
    input    wire [1:0] in,
    output wire          out
);

    assign out = in[1] | in[0];

endmodule
```

#### and\_or.v

```
module and_or (
    input    wire [3:0] in,
    output wire          out
);

    wire [1:0] sig;

    and2 A1 (
        .in      (in[3:2]),
        .out      (sig[1])
    );

    and2 A2 (
        .in      (in[1:0]),
        .out      (sig[0])
    );

    or2 O1 (
        .in      (sig),
        .out      (out)
    );

endmodule
```

## Simulation code

### **tb\_and\_or.v**

```
module tb_and_or;

    reg    [3:0] tb_in;
    wire    tb_out;

    and_or DUT (
        .in      (tb_in),
        .out     (tb_out)
    );

    initial begin
        tb_in = 4'b0000; #5; tb_in = 4'b0001; #5; tb_in =
        4'b0010; #5; tb_in = 4'b0011; #5; tb_in = 4'b0100;
        #5; tb_in = 4'b0101; #5; tb_in = 4'b0110; #5; tb_in
        = 4'b0111; #5; tb_in = 4'b1000; #5; tb_in =
        4'b1001; #5; tb_in = 4'b1010; #5; tb_in = 4'b1011;
        #5; tb_in = 4'b1100; #5; tb_in = 4'b1101; #5; tb_in
        = 4'b1110; #5; tb_in = 4'b1111; #5;
        $display ("Simulation Finished");
        $finish;
    end
endmodule
```