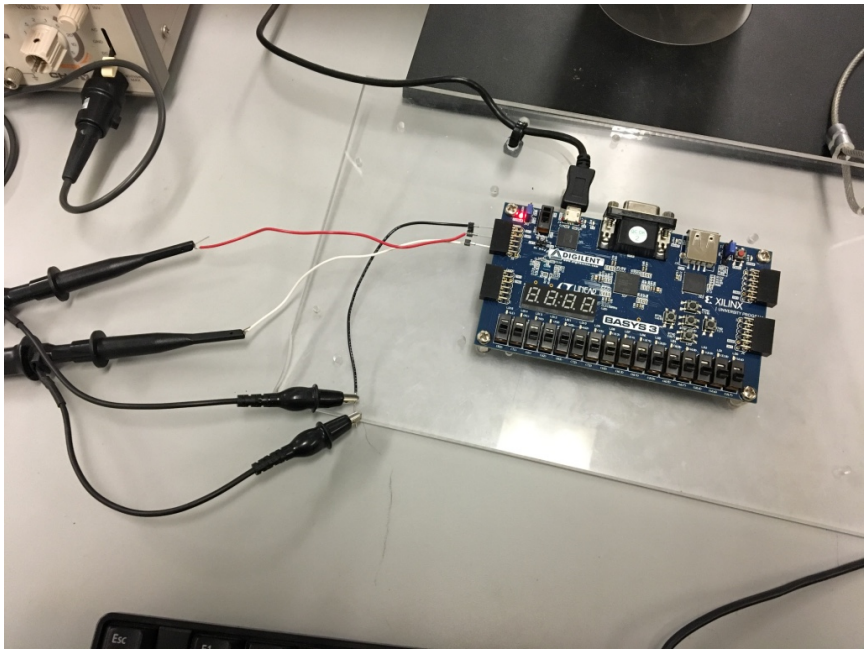
Gabriel Dimas
Lab 1
April 8, 2022
Section C

Description

In this lab, we learned how to use a Digital Oscilloscope (part 1) and the Basys3 Board (part 2). In part one, the lab introduced us to the different measurements on the O-Scope and how to accurately measure the vertical gain, the sweep rate, and the frequency of the different channels. In part two, we learned how to use a Basys board and how to program using Verilog language and the different components on the board.

Design

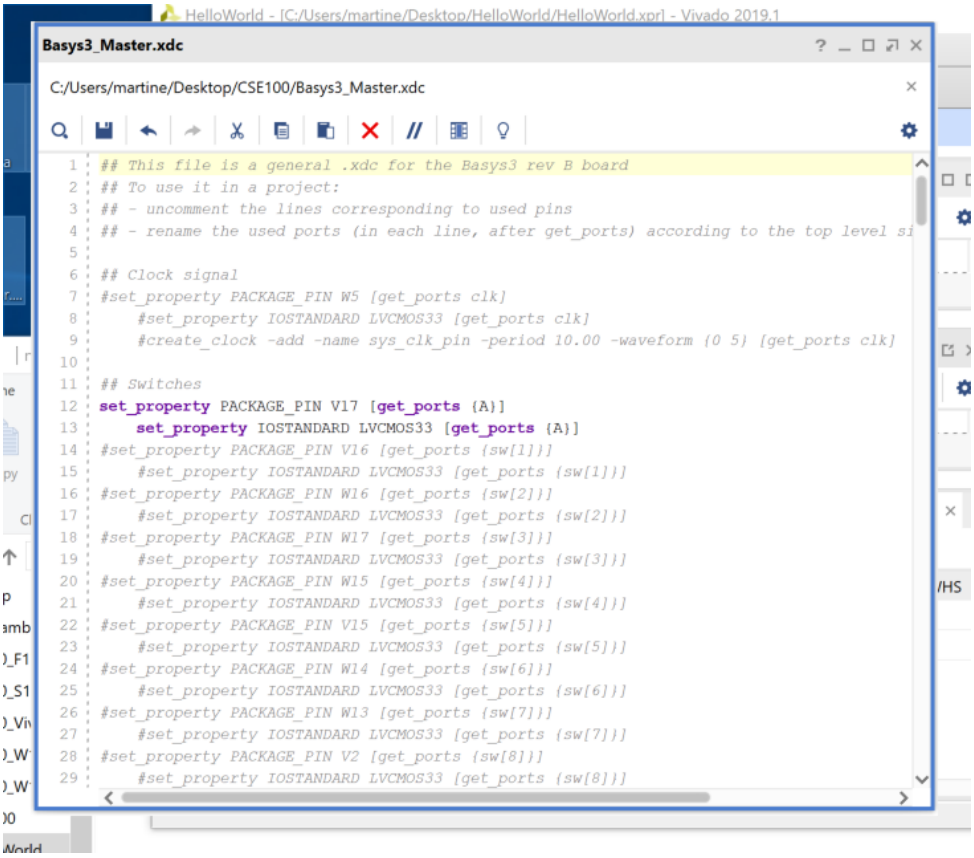
- Part One
    - For part one, we needed to read the output on the O-Scope from the Basys board. To do that, we needed to run a continuous voltage from the board and connect them to the channels of the O-Scope. This was the schematic used for the probing[1].



    - We then gathered the sinusoidal and/or square waves and compressed them into a single result table. See **Results** below.
- Part Two
    - For part two, we learned how to use the Basys Board and how to program in Verilog. Our task was to make a NOT, OR, XOR, and OR logic gates

[1]Gathered from the class website: https://classes.soe.ucsc.edu/cse100/Spring22/lab/lab1/lab1.html

programmed onto the board. For this part, that we needed to include Basys3_Master.xdc file and uncomment the lines that we need to use on the board. Here is the snippet of directions from the lab manual that we followed[2]. It includes the necessary steps for completing the .xdc file. See **Appendix** for the full snippet of code.



- For this part, we needed the **assign** keyword to assign a variable to the boolean out of the hardware. For example, **assign NOT = !A_button;** assigns **NOT** the NOT value of the **A_button** input. We would then set:
  set_property PACKAGE_PIN U16 [get_ports {NOT}]
  set_property IOSTANDARD LVCMOS33 [get_ports {NOT}]
  to hold the value of the boolean **NOT** variable
- This would be the same for the **NOR, OR, OR** variables.

Testing & Simulation

- Part One
  - At the beginning, this testing was very difficult because I needed to set up the Oscilloscope for the very first time. We need to program the board with the **cse100-lab1.bit** file to give an output signal. There were not really any corner cases in this lab because it was introductory, but I did find myself needing an

[2]This photo is gathered from the class website: https://classes.soe.ucsc.edu/cse100/Spring22/lab/design/design.html

immense amount of help from my neighbor and TAs. I had never used one, so it was a massive learning moment. Ultimately I got help from the TAs and the lab description to ensure I got proper input. One of the problems I kept running into was we kept getting no signal from the scope. We changed the vertical and horizontal values, and we still got no change in output. After we changed the trigger values and the amplification from the scope, we were finally able to find the output signal from the Basys board and find the vertical gain, the sweep rate, and the frequency of each signal.

- Part Two
    - Learning Verilog for the first time seemed daunting. But with help, we were able to get the board going. This testing was very straightforward after we programmed the board. After some trial and error, The LEDs worked as expected with the inputs. Here is a snippet of the code used to program the OUTPUT:

```
assign NOT = !A_button;           //I: sw0, O: led ld0
assign OR = A | B;                //I: sw0 & sw1, O: LED LD1
assign XOR = (!A & B) | (A & !B); //I: sw0 &sw1, O: led ld2
assign OOR = A | B | D;
//I: sw0 & sw1 & sw2, O: led ld3
module
```

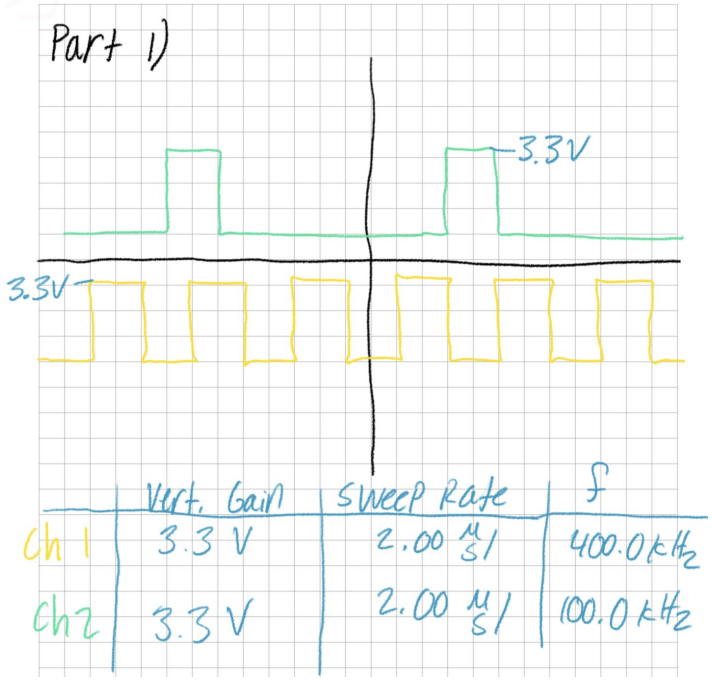
        - A snippet of my code[3]
    - One corner cases I needed to consider was the fact that we had to use the **assign** keyword and not the = key by itself. There were many errors because I didn't use this keyword, so I know for the future that I would need to use the **assign** keyword when I want to assign a boolean value. I also ran into a problem when using the BE104 lab computer because it was not updated to the most current version, so I was not able to upload the bit file to the correct board. It took me, the TAs, and the instructor to finally figure out that this was the issue. I finally moved onto my personal laptop which had the updated version of the software to upload the bit file.

Results

- Part One
    - We were able to finally get the correct results from the Oscilloscope. From the Basys Board, we had to set the output of the board to ensure that it was producing an output. We found that both channels of the output are different. We find that

[3] See **Appendix** for the full version of my code. This is a snippet of the entire code file.

channel one produced a square wave with a vertical gain of about 3.3V. Here is a snapshot of what I recorded during lab time[4].



- We can also see how the channels show different periods over the same time. This can mean that the frequency of each of the channels are different. We also see that when channel two is high, then channel one is low. They have different outputs.

- Part Two
  - After several hours, we were able to finally gather the correct Verilog code. We were able to program the correct switches to ensure we have the correct output for the Basys Board. While testing, when the BTNC pushbutton is pressed, the LED LD0 light would stop illuminating, and would then illuminate again after the button was depressed. The LED LD1 light would illuminate if the SW0 or SW1 switches were in the on position. Else, the light would turn off. LED LD2 light would illuminate if and only if SW0 or SW1 were switched on. Else the light would be low. Lastly, the LED LD3 light would illuminate if SW0, SW1, or SW2 were switched on. Else, the light would be low. This is the correct output.

Conclusion

- Part One
  - Overall, I learned the correct way to use an Oscilloscope and what it is capable of. This tool is used to determine the frequencies of a wave form, wave form (AC/DC) itself, and many other important pieces of information so that we can better understand and input signal. We also were able to better understand the

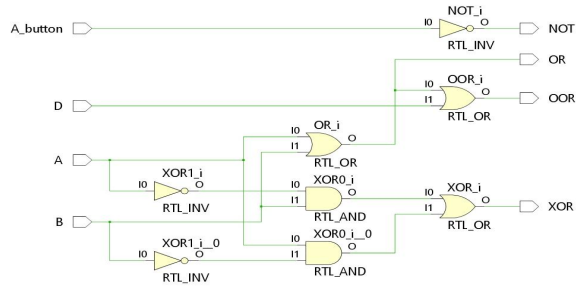
[4] See **Appendix** for the entire snapshot of the lab for the design. I took this section during the lab.

capabilities of the oscilloscope and how to operate many of the different buttons associated with it. On top of that, we learned that if we have a Basys board connected to an o-scope, then we can learn what the board is capable of doing in real-life applications.

- Part Two
    - This was an interesting introduction to logic design. We learned how the Basys3 Board is programmed and how we are able to manipulate code in order to get the board to do what we intend. We also get a better understanding of how we should program the board in the future. I learned that I need to enable the pins we are using on the board (similar to that of Arduino) and that I need to assign values to certain boolean variables in order to get input or give output to the board. I also learned the importance of Synthesis and how the software needs to "compile" all the different pieces together in order to convert it into a **.bit** file to send to the board. The "Implementation" of the program is one of the most important parts of the design process because it parses together all of the modules and code that we need to create our bit file. In the future, we can use the hierarchy feature of the software to create many modules and compile them into one "main" top module. As we learned, the switches and lights are intended for I/O programming, and they provide us a pathway to learning logic design and the path for the rest of this course.

Appendix

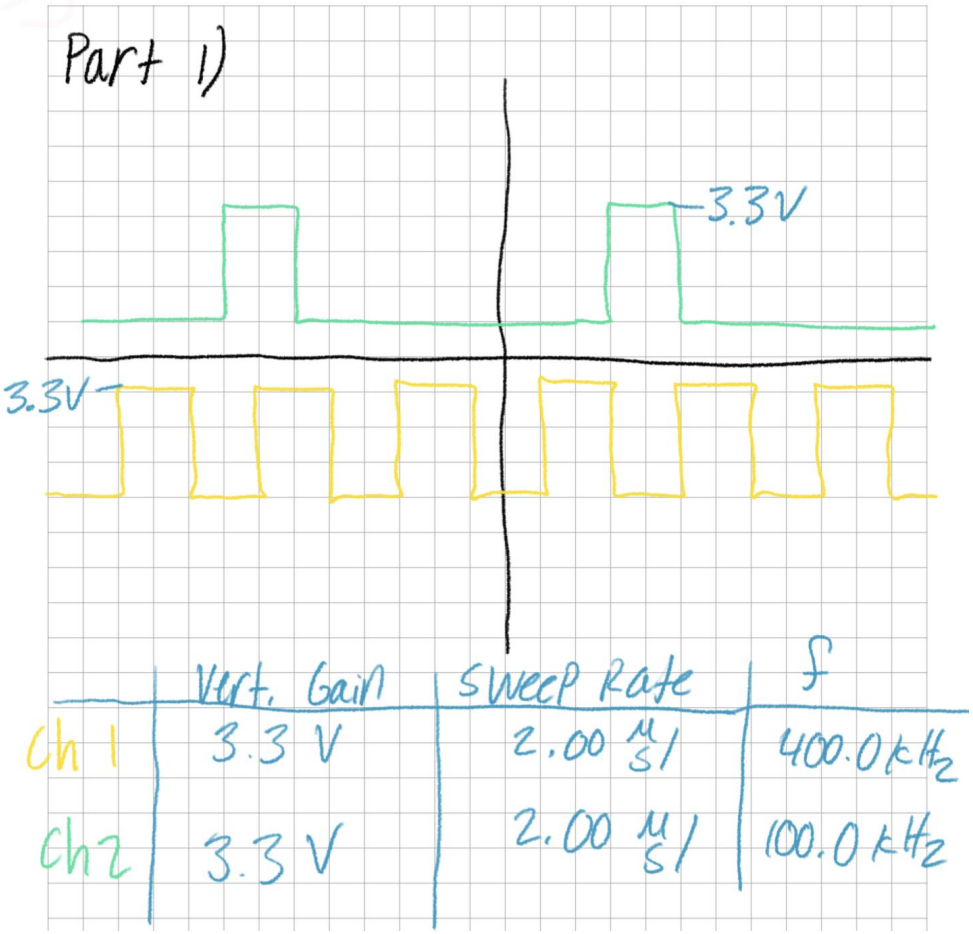
- Schematic of the Elaborated Design

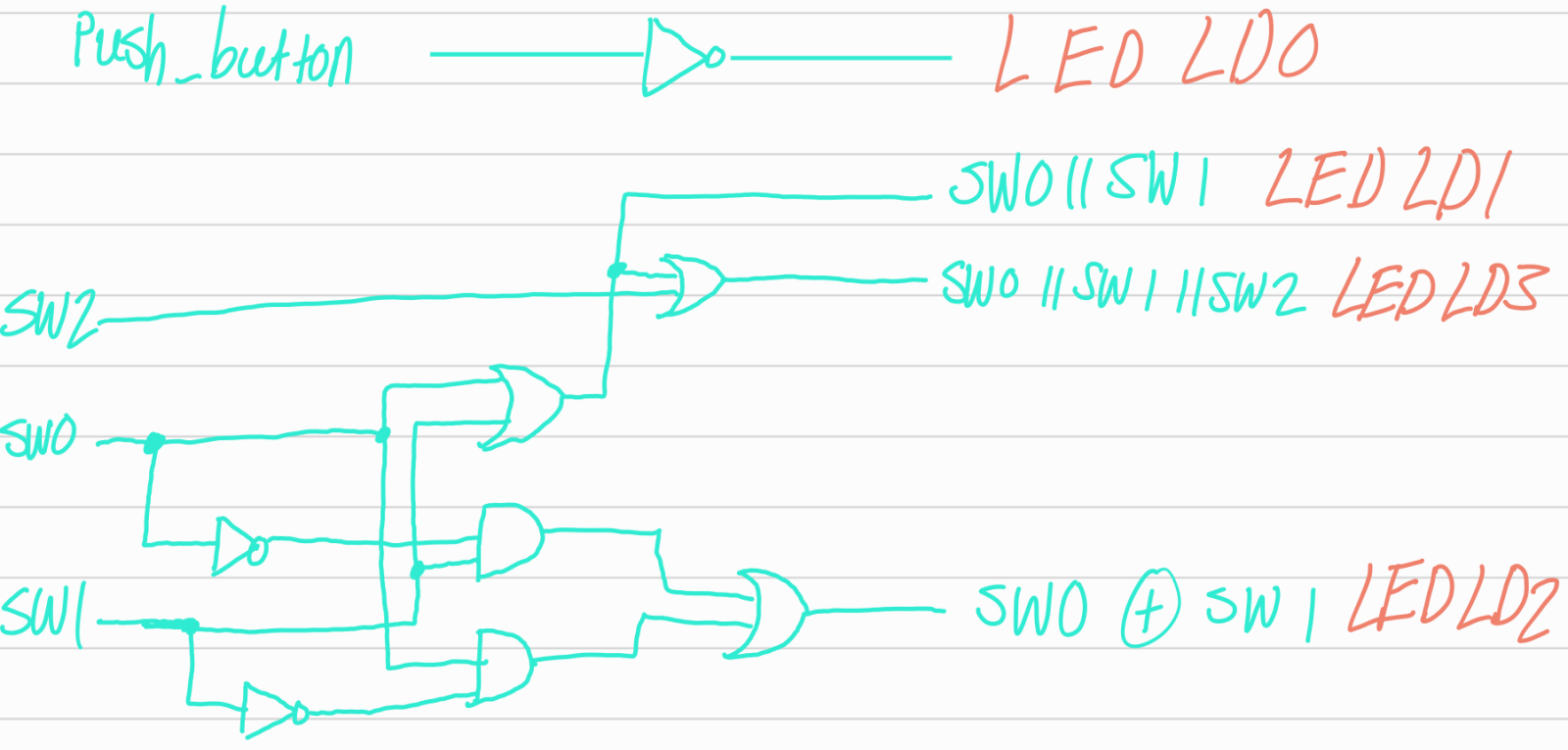


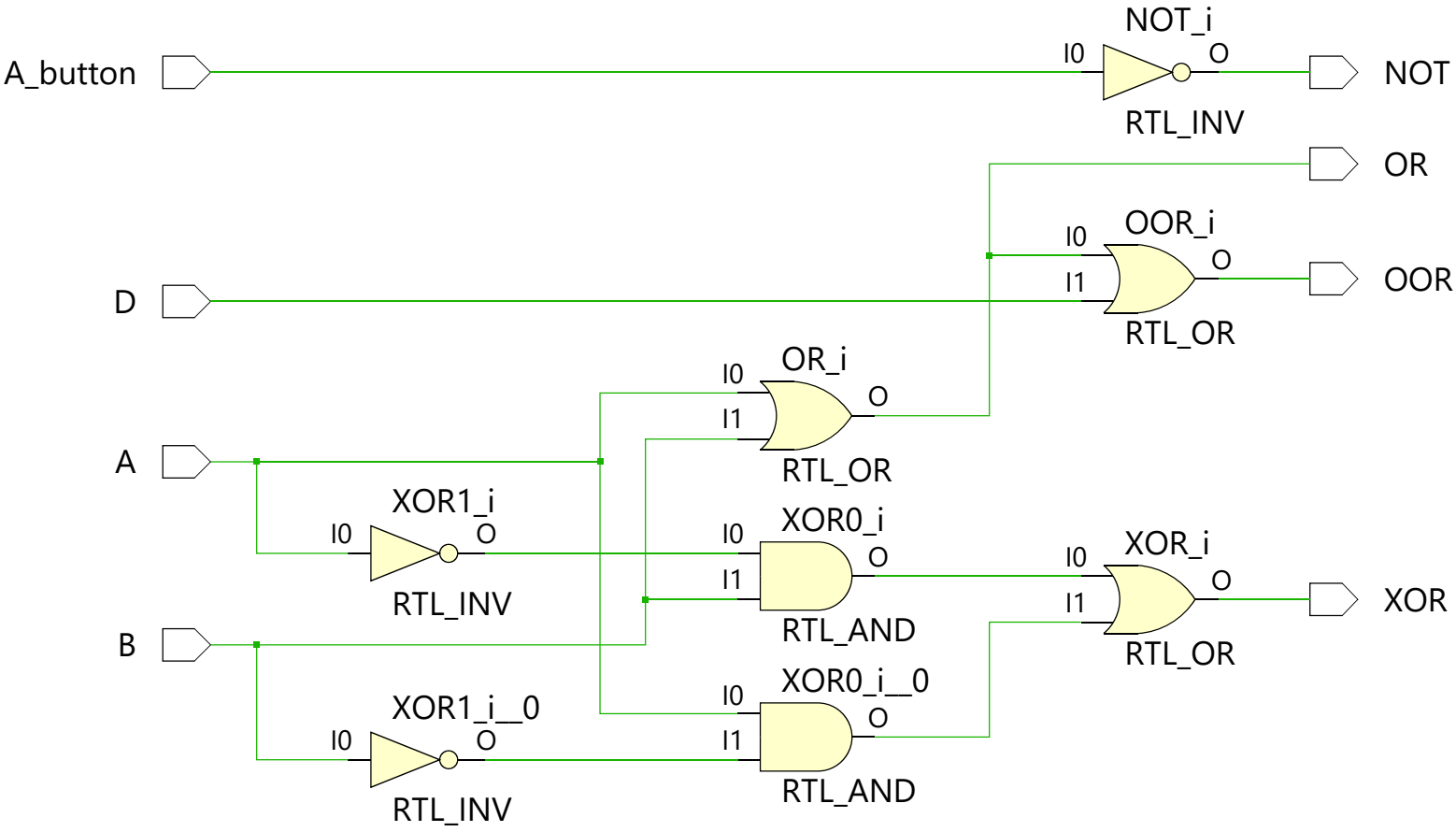
- Complete Verilog Code

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////
//////////////////////////////
// Company: UCSC CSE 100L
// Engineer: Gabriel Dimas
//
// Create Date: 03/29/2022 06:13:05 PM
// Design Name: HelloWorld
// Module Name: myOR
// Project Name: HelloWorld
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////
//////////////////////////////


module myOR(
     input A,
     input A_button,
     input B,
     input D,
     output OR,     //LED LD1
     output NOT, //LED LD0

    output XOR,    //LED LD2
    output OOR  //led ld3
    );
    assign NOT = !A_button;                  //I:
sw0, O: led ld0
    assign OR = A | B;                       //I: sw0
& sw1, O: LED LD1
    assign XOR = (!A & B) | (A & !B);          //I: sw0
&sw1, O: led ld2
    assign OOR = A | B | D;
    //I: sw0 & sw1 & sw2, O: led ld3
endmodule
```

- Part One Wave Diagram




Push_button
LED LD0
SW0 || SW1
LED LD1
SW2
SW0 || SW1 || SW2
LED LD3
SW0
SW1
SW0 ⊕ SW1
LED LD2


A_button
NOT_i
I0
O
RTL_INV
NOT
OR
OOR_i
I0
O
I1
D
OOR
RTL_OR
OR_i
I0
O
I1
A
RTL_OR
XOR1_i
I0
O
XOR0_i
I0
O
XOR_i
I0
O
RTL_INV
I1
XOR
B
RTL_AND
RTL_OR
XOR0_i__0
I0
O
XOR1_i__0
I0
O
I1
RTL_INV
RTL_AND

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////////////////////////
// Company: UCSC CSE 100L
// Engineer: Gabriel Dimas
//
// Create Date: 03/29/2022 06:13:05 PM
// Design Name: HelloWorld
// Module Name: myOR
// Project Name: HelloWorld
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////////////////////////////


module myOR(
    input A,
    input A_button,
    input B,
    input D,
    output OR,    //LED LD1
```

```
    output NOT, //LED LD0
    output XOR,    //LED LD2
    output OOR  //led ld3
    );
    assign NOT = !A_button;           //I: sw0, O:
led ld0
    assign OR = A | B;                //I: sw0 & sw1,
O: LED LD1
    assign XOR = (!A & B) | (A & !B); //I: sw0 &sw1,
O: led ld2
    assign OOR = A | B | D;
    //I: sw0 & sw1 & sw2, O: led ld3
endmodule
```