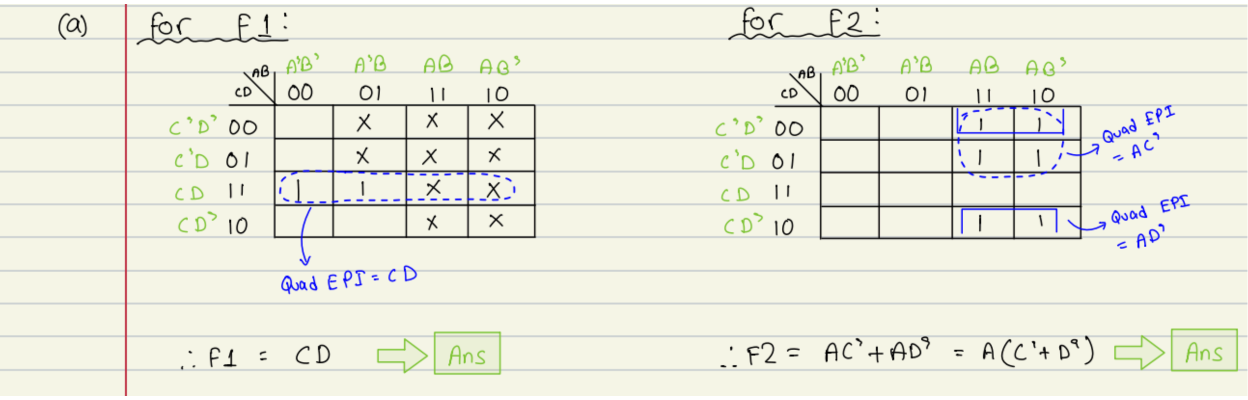
ECE 210 LAB-1 REPORT

Name: Misbah Ahmed Nauman
Student ID: 1830574
CCID: misbahah

Name: Muhammad Abdur Rab Siddiqui
Student ID: 1804733
CCID: msiddiq6

Section Number: D25
Lab Number: 1
Lab Date: 10.08.2024

PRE-LAB ASSIGNMENT



(a) for F1:

CD \ AB	A'B' 00	A'B 01	AB 11	AB' 10
C'D' 00		X	X	X
C'D 01		X	X	X
CD 11	1	1	X	X
CD' 10			X	X

Quad EPI = CD

∴ F1 = CD ⇨ Ans

for F2:

CD \ AB	A'B' 00	A'B 01	AB 11	AB' 10
C'D' 00			1	1
C'D 01			1	1
CD 11				
CD' 10			1	1

Quad EPI = AC'

Quad EPI = AD'

∴ F2 = AC' + AD' = A(C' + D') ⇨ Ans

Fig 1.1: Screen-shot of Pre-lab calculations.

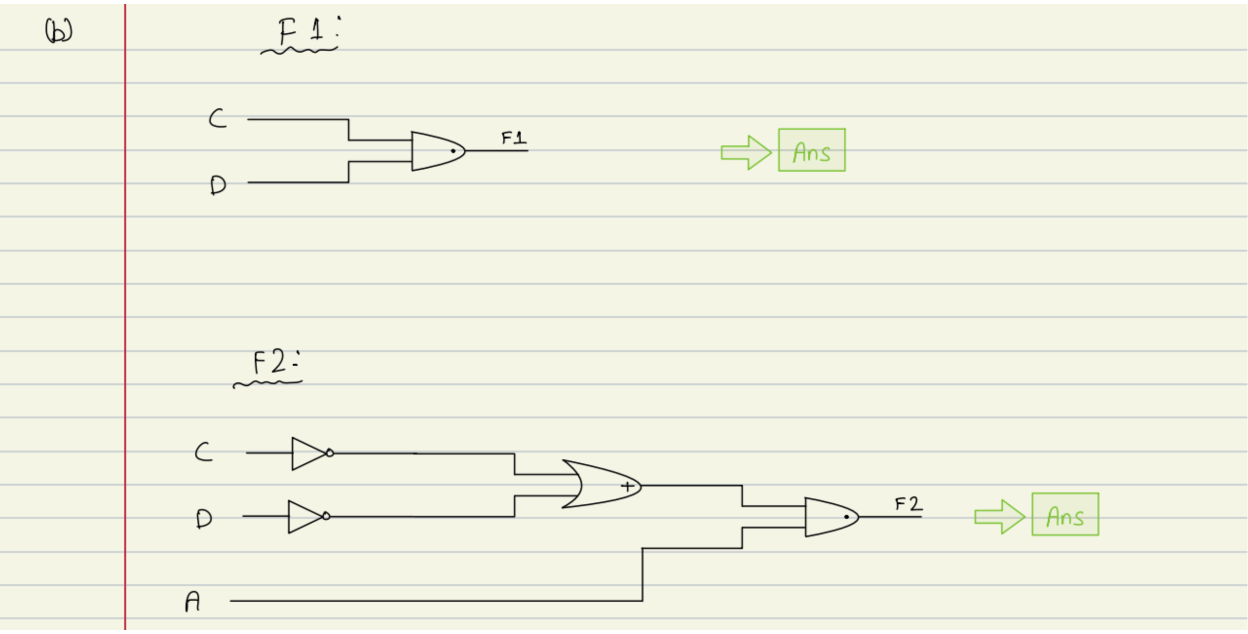


Fig 1.2: Screen-shot of Pre-lab circuit design.

Aforementioned is my solution for the pre-lab assignment for this experiment, and all circuits and calculations in this lab will be based on these results. During the lab, a simplified version of Equation F2 was provided; however, we proceeded with the lab using the equation derived from F2. It is important to note that the designed circuit could have been more efficient and utilized fewer logic gates if the equation had been simplified.

ABSTRACT

This lab focuses on designing and implementing basic combinational logic circuits using AND, OR, and NOT gates. The main objective was to develop an understanding of digital logic design through circuit minimization using Karnaugh Maps (K-maps) and to code the design using VHDL. The circuits were simulated in Vivado software and physically implemented on the Zybo Z7 FPGA development board. The functionality of the circuits was verified through both software simulations and real-time monitoring using the Digilent Analog Discovery 2. The experiment demonstrated the successful minimization of logic expressions, leading to efficient circuit designs.

INTRODUCTION

In this lab we explored the design and analysis of combinational logic circuits. This lab focused on building these circuits using basic AND, OR, and NOT gates, while progressively applying advanced techniques like Karnaugh Maps (K-maps) for optimization. The primary objective was to ease the transition from traditional circuit prototyping with Integrated Circuits (ICs) to sophisticated digital system design using Field-Programmable Gate Arrays (FPGAs).

Combinational logic circuits, which rely on current inputs to produce outputs, are foundational in devices like computers, control units, and data processing systems. We examined IC packaging formats such as DIP (Dual In-line Package) and SOIC (Small Outline Integrated Circuit), analyzing their impact on circuit performance and integration.

Then we moved to FPGAs using the Xilinx Zybo Z7 development board, implementing our designs with Vivado software and VHDL. Applying Boolean algebra and K-map minimization, we optimized our circuits and monitored real-time performance using the Analog Discovery 2 tool.

DESIGN SECTION

The goal of this lab was to design and implement basic logic circuits using AND, OR, and NOT gates. The design had to meet specific logic requirements derived from a given truth table (from pre-lab) and was minimized using K-maps.

CONSTRAINTS

- **Input Signals:** The inputs were binary signals representing the variables from the truth table.
- **Output Requirements:** The outputs were created to match the simplified

logic expressions from the truth tables, making sure the circuits worked as intended.

METHODOLOGY

- **VHDL Implementation:** The minimized logic expressions (from K-maps) were coded in VHDL to generate the required logic circuits.
- **Circuit Simulation:** The design was first simulated in Vivado to ensure correctness of the logic and functionality.
- **FPGA Implementation:** The circuit was then implemented on the Zybo Z7 FPGA development board, and real-time outputs were monitored using the designed circuit and Digilent Analog Discovery 2.

EXPERIMENTAL PROCEDURE AND EQUIPMENTS

EQUIPMENTS

- Xilinx Zybo Z7 FPGA development board
- Vivado software
- Digilent Analog Discovery 2
- Breadboard, resistors, jumper wires
- Logic ICs (74HC04 for NOT gates, 74HC08 for AND gates, 74HC32 for OR gates)

PROCEDURE

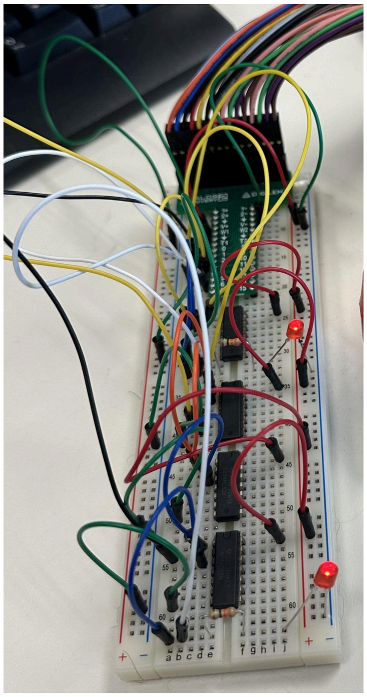


Fig 1.3: Photo of the breadboard designed for Requirement 1.

1. We set up the circuit from the expression derived in our pre-lab using a breadboard, resistors, jumper wires, and optional LEDs for visual output, along with ICs (as shown in **Fig. 1.3**)
2. The circuit was connected to the Digilent Analog Discovery 2, which interfaced with the Waveform software on our computer to supply power, provide input through the DIO ports, and observe the output. The output was also monitored using LEDs.
3. Input and output signals (DIO0 to DIO5) were added to the logic window of Waveform, allowing live patterns to be observed (**Fig. 1.4**).
4. Next, we Implemented our pre-lab work using VHDL in the Vivado software. We opened a new project in Vivado and added the downloaded files: lab1.vhd, lab1_tb.vhd, and const.xdc from e-class to our project.
5. We then completed the source file lab1.vhd, assigning the switches as follows: sw(0) = A, sw(1) = B, sw(2) = C, and sw(3) = D. For the outputs, we designated led(0) = F1 and led(1) = F2.
6. In the VHDL code, we implemented the logic for the outputs as shown in **Fig. 1.5**.
7. To verify our results, we ran a simulation test using the Vivado software to ensure that the logic performed as expected (**Fig. 1.6)**.
8. Once the design was completed, we demonstrated the implementation on the Zybo Z7 board, ensuring that the logic performed as expected.

RESULTS

INPUT A	INPUT B	INPUT C	INPUT D	OUTPUT F1	OUTPUT F2
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	1	1	1	0
0	1	0	0	X	0
0	1	0	1	X	0
0	1	1	0	0	0
0	1	1	1	1	0
1	0	0	0	X	1

1	0	0	1	X	1
1	0	1	0	X	1
1	0	1	1	X	0
1	1	0	0	X	1
1	1	0	1	X	1
1	1	1	0	X	1
1	1	1	1	X	0

Table 1.1: The table above summarizes the observed outputs of F1 and F2 for various combinations of inputs A, B, C, and D.

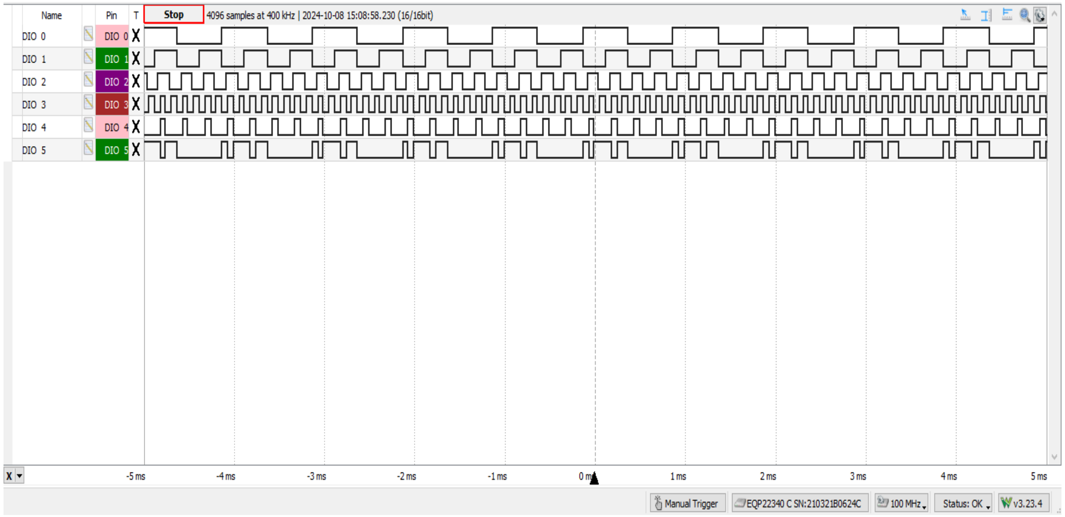
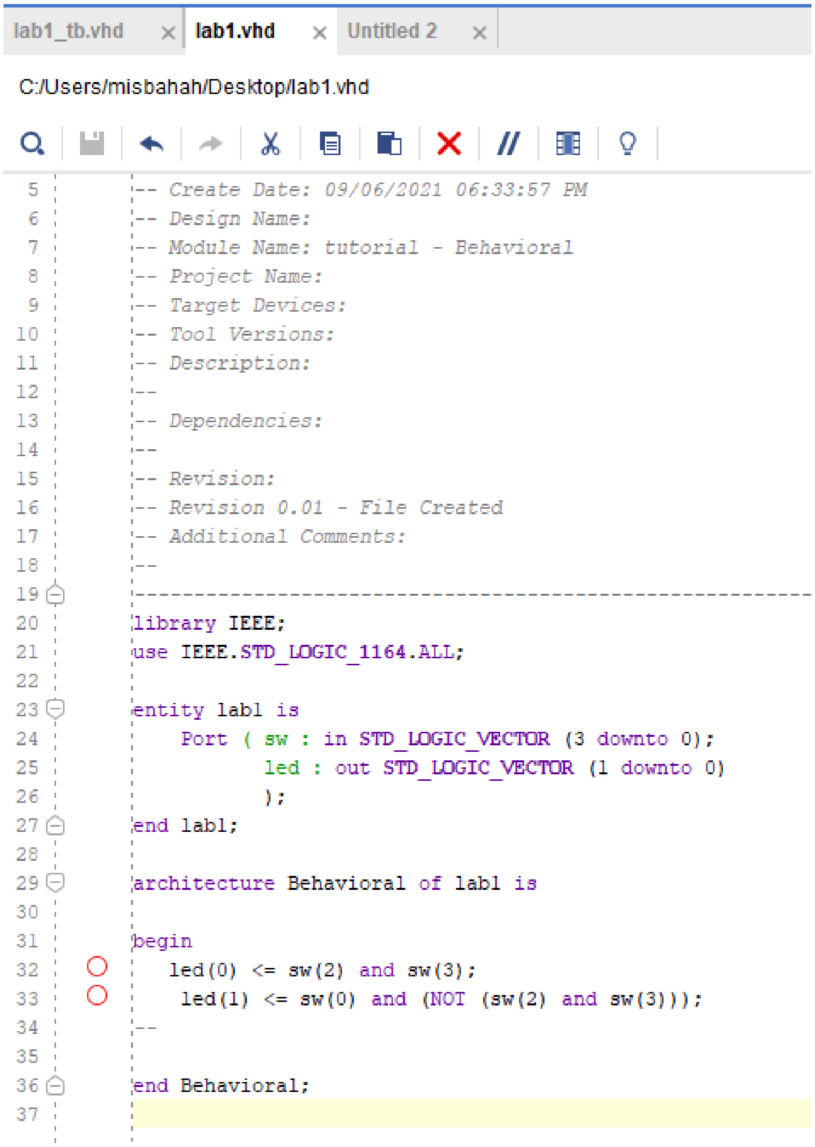
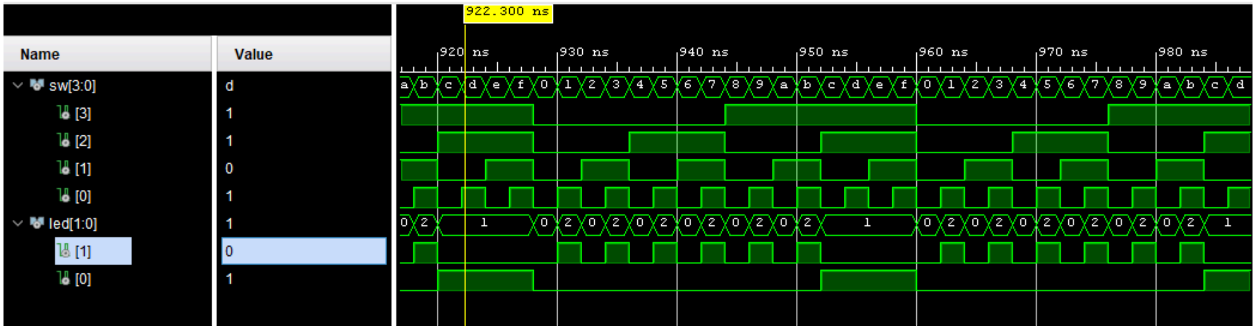

4096 samples at 400 kHz | 2024-10-08 15:08:58.230 (16/16bit)

Fig 1.4: Photo of a logic window taken in the Waveform software for Requirement 1.



```vhdl
-- Create Date: 09/06/2021 06:33:57 PM
-- Design Name:
-- Module Name: tutorial - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity lab1 is
    Port ( sw : in STD_LOGIC_VECTOR (3 downto 0);
           led : out STD_LOGIC_VECTOR (1 downto 0)
           );
end lab1;

architecture Behavioral of lab1 is

begin
   led(0) <= sw(2) and sw(3);
    led(1) <= sw(0) and (NOT (sw(2) and sw(3)));
--

end Behavioral;

```

Fig 1.5: Code written in VHDL on the Vivado software for Requirement 2.



Name	Value
sw[3:0]	d
[3]	1
[2]	1
[1]	0
[0]	1
led[1:0]	1
[1]	0
[0]	1

Fig 1.6: Screen-shot of a simulation window taken in the Vivado software for Requirement 2.

These figures provide visual confirmation of the circuit's operation and the effectiveness of the implemented VHDL code. The logic window in Fig. 1.4 shows the real-time behavior of the circuit, while the VHDL code in Fig. 1.5 illustrates the logic design. Fig. 1.6 confirms the correct functionality of the design through simulation results.

DISCUSSION

1. **Conclusions from Results:** The experiment showed that the combinational logic circuits we designed using AND, OR, and NOT gates worked just as we expected. The outputs F1 and F2 matched the predicted values for all possible input combinations, confirming that our logic design was correct. Our successful simulation in Vivado, followed by the implementation on the Zybo Z7 board, deepened our understanding of the theoretical concepts behind digital logic design.
2. **Uncertainties and Ambiguities:** While the circuits performed well overall, we did encounter some uncertainties during the breadboard assembly. The complexity of the circuit, due to not fully simplifying the expressions for F1 and F2, made it challenging to ensure reliable connections among the various ICs and wires. This may have introduced slight inconsistencies in the observed outputs, such as variations in signal timing or voltage levels.
3. **Comparison to Expected Results:** The results we observed lined up nicely with our theoretical expectations. The ability of K-maps to minimize the logic expressions was proven by the reduction in the number of gates used in our final circuit design. However, because of the complexity of the original expressions, we ended up using more components than necessary. Moving forward, fully simplifying the logic before implementation will likely lead to a more efficient design, reducing the chance of error and enhancing the overall reliability of the circuit.

In conclusion, the experiment was quite successful in meeting its objectives and effectively demonstrating combinational logic circuit design. For future experiments, we should aim to simplify circuit complexity and improve assembly techniques to boost reliability and accuracy. Additionally, using simulation tools to verify circuit performance before physical implementation could streamline the design process and help avoid potential errors.

CONCLUSION

In this lab, we successfully designed and implemented combinational logic circuits using basic AND, OR, and NOT gates. The experimentation allowed us to confirm that the derived logic expressions effectively produced the expected outputs when simulated in Vivado and tested on the Zybo Z7 board.

Key observations included the effectiveness of Karnaugh Maps (K-maps) in minimizing logic expressions, leading to more efficient circuit designs. The real-time monitoring with the Digilent Analog Discovery 2 provided valuable insights into the circuit's behavior, confirming that the live input-output patterns matched our expectations.

One challenge we faced was that, since we did not simplify our expressions for F1 and F2, our breadboard setup became quite tedious. This resulted in the use of two extra ICs; instead of utilizing four ICs, we could have effectively used only two to create the same circuit. By doing so, we would have required fewer wires and resistors, resulting in a cleaner and more efficient circuit design with fewer components.

For future work, we should ensure that we fully simplify our expressions before proceeding to build the circuit.

Overall, the lab reinforced our understanding of digital logic design principles and the practical application of theoretical concepts in real-world scenarios.

REFERENCES

1. *ECE 210 Lab Manual 1*, University of Alberta, 2024. [Online]. Available: eClass. [Accessed: Oct. 19, 2024].