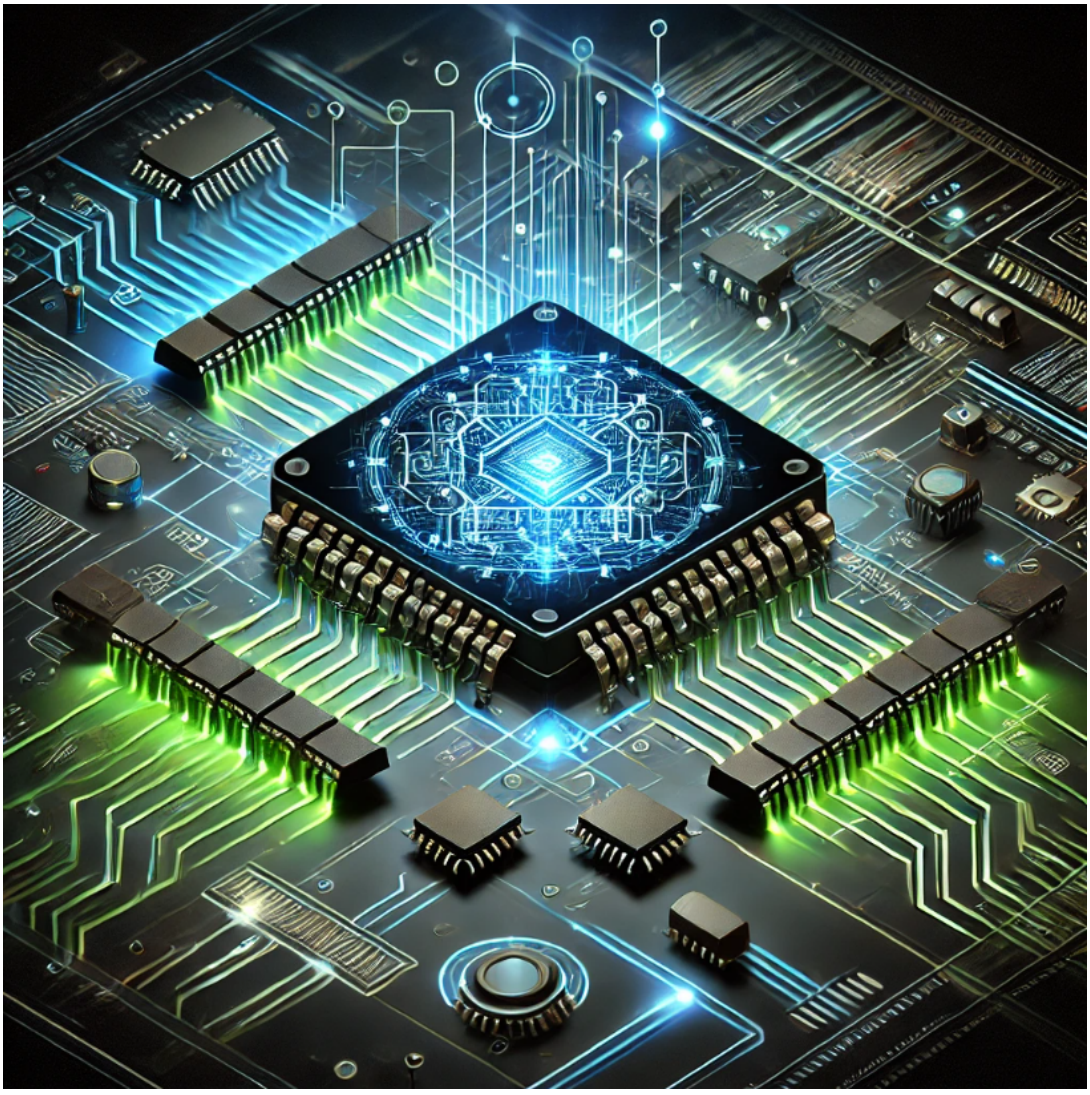
Logic Design Lab Project

Project Manual - Logic-Controlled Board



Prepared By:
Charbel Beaini , Ehab Shayya

Spring 2025

Lebanese American University

0.1 Purpose

This project serves as an opportunity to apply and integrate all the techniques and skills acquired throughout the semester in logic design. Provides a practical implementation of theoretical concepts that reinforces the principles of Boolean algebra, combinational and sequential logic, state machines, and circuit optimization.

0.2 Expected Learning Outcomes

- **Applying and Learning New Concepts**
  - Explore new techniques beyond classroom concepts through practical implementation.
  - Adapt to troubleshooting and debugging challenges in circuit design.
- **Problem-Solving and Design Thinking**
  - Break down complex problems using structured approaches such as divide-and-conquer.
  - Optimize circuit design for efficiency and minimal resource use.
- **Practical Implementation and Hands-On Skills**
  - Gain proficiency in soldering, wiring, and assembling digital circuits.
  - Systematically debug and test circuits to identify and resolve faults.
- **Bridging Theory and Practical Applications**
  - Connect theoretical logic principles with real-world hardware applications.
  - Implement digital circuit concepts in practical hands-on projects.
- **Efficient Presentation and Documentation**
  - Communicate design choices effectively through structured documentation.
  - Present circuit designs with block diagrams, truth tables, and schematics.

For a visual representation of the learning outcomes, refer to the following mind map.

0.3 Project Description

A Logic-Controlled Board is a dynamic switching system that features four switches and four colored lamps: red, green, blue, and yellow. Each switch is initially connected to a lamp of the same color, creating an apparent direct connection. However, the uniqueness of this system lies in its ability to adapt dynamically based on the last switch turned off, allowing for reconfiguration of the lamp activation sequence.

This project aims to design and implement a fully functional logic-controlled board, where the sequence of lamp activations is determined by the last switch turned off. Users can rearrange the lamp positions and switch caps freely, yet the system will still maintain the correct color-to-switch mapping. In addition, the project incorporates an advanced feature, in which the removal of a switch cap temporarily disables its functionality, preventing it from operating. Once the cap is placed back, the switch regains its functionality, adding an interactive and engaging component to the system.

The system operates by tracking the last switch turned off, storing the sequence logic, and dynamically adjusting the behavior based on user interactions. This creates an illusion for the audience, making it appear as though the switches and lamps are somehow intelligently connected. Your design will be based on a combination of digital logic circuits, memory elements, and finite state machines (FSMs) to ensure seamless operation.

This project not only showcases the power of logic design and sequential circuits but also highlights how we can implement any digital system we observe in our surroundings or online. The same fundamental principles applied in this project can be extended to real-world applications such as elevator control systems, vending machines, traffic light controllers, digital games, or interactive/fun projects like this one. Understanding how logic circuits govern these everyday systems is crucial for bridging theoretical learning with practical engineering applications.

For a visual demonstration of the concept, refer to the following video.

0.4 Solution Approach

0.4.1 System Behavior

The system operates based on a predefined lamp activation sequence, dynamically adjusted according to the last switch turned off. The behavior follows the rules outlined below.

Lamp Activation Sequences
By default, the lamp activation order is 1-2-3-4. However, once a switch is turned off last and after a timeout, the system updates to a new sequence:

- If Switch 1 was turned off last: The system follows Sequence 1: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$.
- If Switch 2 was turned off last: The system follows Sequence 2: $2 \rightarrow 3 \rightarrow 4 \rightarrow 1$.
- If Switch 3 was turned off last: The system follows Sequence 3: $3 \rightarrow 4 \rightarrow 1 \rightarrow 2$, including Special Trick #3.
- If Switch 4 was turned off last: The system follows Sequence 4: $4 \rightarrow 3 \rightarrow 2 \rightarrow 1$.

NB: A 7-segment display should be added to indicate the current sequence number for practice purposes.This module allows the user to easily track the active sequence, making it useful when learning and practicing the trick. The 7-segment display should be implemented as a separate module that can be easily removed to switch between practice mode and performance mode, ensuring seamless functionality without affecting the core design.

Reset Condition: The system resets when:

1. All switches are turned off.
2. At least one switch has been learned.
3. The reset timer (4 seconds) has elapsed.

Once reset, the system updates to the new sequence based on the last switch turned off.

Special Trick #3 (Sequence 3)
Special Trick #3 is activated when at least two lamps are ON in Sequence 3. The sequence involves the following steps:

1. **Step A: Initial Lamp Turn Off**
   - One of the two active lamps is switched OFF.
   - A 4-second timer starts, during which the user must remove the color cap of the corresponding switch.

2. **Step B: Capless Switch Behavior**
    - The capless switch can be turned ON, but the LED does not illuminate.
    - The switch can be toggled ON/OFF multiple times, but it must not remain OFF for more than 4 seconds.
    - Another LED can be turned ON and then OFF, allowing the capless trick to be performed on multiple switches.

3. **Step C: Re-Enabling Capless Switches**
    - If a capless switch remains OFF for more than 4 seconds, it will become active again.
    - Before reactivating the switch, the cap must be placed back on top of it.
    - The process can be repeated with multiple switches.

Locking and Unlocking Mechanism

The system can be locked to ensure the audience cannot detect any patterns in the switch-lamp sequence.

- **Locking the Device:**
    - Remove the battery.
    - Turn ON Switch 2 only.
    - Reinsert the battery.
    - The system enters a locked state where each switch is directly mapped to its corresponding LED (1-1, 2-2, 3-3, 4-4).
- **Unlocking the Device:**
    - Restart the system without Switch 2 being ON.
    - The system returns to normal operation with dynamic sequence behavior.

For a visual demonstration of the solution, refer to the following video.

0.4.2 FSM Design

It is required to design a finite state machine and a state table for the design above. The state table is built up using current states bits, inputs, outputs, and next state bits. The control logic for the Logic-Controlled Board is modeled using a finite state machine (FSM). This FSM governs the dynamic behavior of the system, including how switch activations affect the LED sequence and how the system handles advanced features such as the capless switch trick.

Inputs and Outputs

The system is designed around a combination of physical user inputs and visual outputs. Clearly defining the system's inputs and outputs helps establish the foundation for the FSM design and the implementation logic.

Inputs:

- **Switches (SW1–SW4):** Four toggle switches act as the primary user inputs. Each switch is associated with a specific color (Red, Green, Blue, Yellow), and their ON/OFF states drive the system behavior.
- **Reset Timer Signal ($\mathbf{T_{reset}}$):** A logic-level input that goes HIGH after 4 seconds of all switches being OFF. This signal is used to trigger the FSM transition into the appropriate sequence based on the last switch that was turned off (Will also be used in Special Trick #3).

Outputs:

- **LED Indicators (L1–L4):** Each switch controls an LED of matching color. The logic determines which LEDs turn ON or OFF based on the current sequence and cap status.
- **7-Segment Display:** When operating in *Practice Mode*, the system outputs the currently active sequence number (1 to 4) on a 7-segment display. This module can be detached in *Performance Mode* to enhance the illusion.

FSM State Definitions

The FSM consists of several key states that define the behavior of the system. These states are designed to be modular, allowing students flexibility in how they implement the control logic.

- **Boot:** Initial power-on state. The system checks whether Switch 2 is ON to determine whether to enter the normal FSM flow or locked mode.
- **Locked:** Activated when the system is booted with Switch 2 ON. In this state, each switch is directly mapped to its corresponding LED (1→1, 2→2, etc.), and no dynamic behavior occurs.

- **Sequence Detector:** Transitional state that determines which switch was turned off last after a reset condition and assigns the appropriate lamp activation sequence accordingly.
- **Sequence 1–4:** Represent the four different operation modes (lamp activation sequences), selected based on which switch was turned off last. Each sequence follows a unique LED activation order (4 different states).
- **Switch Not Working:** A temporary state used in Sequence 3 to represent a switch with its cap removed. In this state, the switch can be toggled, but its corresponding LED will not respond. Once the timeout expires and the cap is restored, the switch reactivates.

Note: Timeouts and delays (e.g., 4-second timers) are treated as external inputs to the FSM.

0.4.3 Deriving the Output Functions

- Use the state table to derive the output functions.
- Apply a minimization technique (such as Karnaugh maps or Boolean algebra) to optimize the functions.

0.4.4 Circuit Implementation

- Draw the circuit diagram based on the optimized logic functions.
- Minimize the number of logic gates used.
- Reduce the number of integrated circuits (ICs) required.

0.4.5 Quartus Design

- Implement the design using Quartus.
- Construct modular blocks for different parts of the system to integrate them into the final design.
- Simulate the circuit and analyze the results.

0.4.6 Clock Design

- Conduct a detailed study on clock circuit design.
- Implement a 555 Timer to generate the required clock signal.
- Document all design steps and analyze the clock performance.

0.4.7 Soldering and Breadboard Implementation

- Breadboard soldering is necessary for practical implementation.
- Soldering should be clean and precise.
- Use a chip chair for soldering to prevent heat damage to the components.
- Design the project in modular parts so that partial functionality can still be graded even if the entire system is not fully operational.

0.4.8 Packaging the Final Design

- Ensure the project is neatly packed and well-assembled.
- The best packaging will receive the highest evaluation in this section.

0.4.9 Key Considerations

- Clearly present the system analysis.
- Clearly identify inputs and outputs.
- Write down the truth table for the design.
- Apply a minimization technique to derive the simplest form of outputs.
- Draw the full system schematic on paper before implementation.
- Minimize the number of ICs used.
- Implement the design on Quartus, run simulations, and analyze the results.

0.5 Bill of Materials

Item	Quantity
Soldering Iron	1
Solder Iron Wire	1
Perforated Circuit Board / Breadboard	Based on need
Voltage Source	1
Wires	Based on need
555 Timer	1
Resistors and Capacitors (if needed)	Based on design
IC Chips	Based on design
Toggle Switches	4
LEDs (Red, Blue, Yellow, Green)	1 of each
7-Segment Display	At least 1

Table 1: List of Required Components and Quantities

This list may vary depending on your specific design choices and whether you decide to implement additional features or attempt the bonus objectives. Ensuring a well-organized final design with clean soldering and proper packaging will positively impact your final grade.

0.6 Deliverables

0.6.1 Final Required Components

Each team is required to submit and demonstrate the following project components:

Final Project Implementation:

- Fully assembled and functional logic-controlled board.
- Correctly implemented switching logic, including all sequences and special tricks.
- Properly integrated 7-segment display module for sequence indication.
- Functional clock generation circuit using a 555 timer.

Hardware Requirements:

- The final hardware must be neatly packaged, ensuring structural integrity and ease of handling.
- The clock-generating circuit must be soldered on circuit boards. The use of printed circuit boards (PCBs) is not allowed.

0.6.2 Project Report

Each team is required to submit one report. The report should be well-organized and include:

- All design steps, decisions, and assumptions
- A thorough analysis of the design
- The theoretically simulated design using Altera Quartus II

The report should include the following sections:

1. Abstract
2. Introduction
3. Components and equipment used
4. Analysis
5. Paper design
6. Quartus design and analysis
7. Breadboard design and analysis
8. Financial study, focusing on minimizing the cost of the design
9. Delay calculation
10. Power consumption analysis
11. Problems faced during the project
12. Key design points that present advantages over alternative designs

0.6.3 Teamwork

Each group will remain the same as assigned at the beginning of the semester and must work together throughout the project. Any resemblance or common work between different teams will result in a zero for both projects.

0.6.4 Demonstration

Each group must present its work, and all members should be prepared to answer any questions related to the design and implementation.

0.6.5 Due Date

The submission deadline will be announced on blackboard.

0.6.6 Grading and Evaluation

This project constitutes 20% of the total lab grade, distributed as follows:

- 15% common grade for the entire team
- 5% individual effort

0.7 Bonus

This section is for ambitious students who wish to enhance their projects with additional features. Completing any of these challenges will allow you to earn up to 20 extra points on your final grade.

0.7.1 Scrolling 7-Segment Display for Custom Messages (5 pts)

Design a dynamic scrolling message on a 7-segment display to show the word "LOGIC" or any other word of your choice. This requires precise control over multiplexing and display timing to create a smooth scrolling effect.

0.7.2 GitHub Repository with Comprehensive Documentation (5 pts)

Create a well-structured GitHub repository for your project. The repository should:

- Include all necessary project files (schematics, Quartus files, simulations, and documentation).
- Maintain clear directory management for easy navigation.
- Provide a detailed README file explaining:
  - The purpose of the project.
  - A description of each included file.
  - Setup and usage instructions.
  - Additional insights into the project's design choices.
- Upload a demonstration video on YouTube and hyperlink it in the README.

0.7.3 Recreating the 555 Timer Design and Documentation in LaTeX (2.5 pts)

Implement the design and theoretical description of the 555 timer circuit in LaTeX. This should include:

- A detailed explanation of how the 555 timer works.
- The mathematical formulas governing its timing operation.
- A LaTeX-rendered circuit diagram illustrating the 555 timer connections.

0.7.4 Innovative Special Trick or New Functionality (7.5 pts)

Introduce a unique feature or trick of your own design that enhances the illusion and makes the system even more deceptive to the spectator.