A project report on

SEQUENCE DETECTOR USING VERILOG HDL

Submitted in partial fulfilment for the award of the degree of

Bachelor of Technology in
Electronics and Computer Engineering

by

25BLC1046	**ETHAN SAMUEL DEVADATTA**
25BLC1285	**AUSTIN SAM ANIL**
25BLC1289	**VIDITYA SANJAY SINGH**

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI
SCHOOL OF ELECTRONICS ENGINEERING

Course: Digital Logic and Computer Architecture

10th April, 2026

DECLARATION

I hereby declare that the project report entitled "Sequence Detector Using Verilog HDL" submitted by Ethan Samuel Devadatta (25BLC1046), Austin Sam Anil (25BLC1285), and Viditya Sanjay Singh (25BLC1289), for the partial fulfilment of the Bachelor of Technology degree in Electronics and Computer Engineering at Vellore Institute of Technology, Chennai, under the guidance of Dr. Annis Fatima, is a record of bonafide work carried out by us.

We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this or any other institute or university.

Place: Chennai

Date: 10th April, 2026

Ethan Samuel Devadatta Austin Sam Anil Viditya Sanjay Singh

(Signatures of the Candidates)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Dr. Annis Fatima, Assistant Professor, School of Electronics Engineering, Vellore Institute of Technology, Chennai, for her expert guidance, constant encouragement, and invaluable support throughout this project. Her deep knowledge of Digital Logic and Computer Architecture provided us with clear direction and helped us navigate the challenges of Hardware Description Language (HDL) design and simulation.

We extend our heartfelt thanks to the visionary leader Dr. G. Viswanathan, Honourable Chancellor, and the entire leadership team of Vellore Institute of Technology — Dr. Sankar Viswanathan, Dr. Sekar Viswanathan, Dr. G.V. Selvam (Vice Presidents), Dr. Sandhya Pentareddy (Executive Director), and Dr. V.S. Kanchana Bhaaskaran (Vice-Chancellor), for fostering an environment that encourages innovation and practical learning.

Our sincere thanks go to Dr. Vijayakumar Peroumal, Head of the Department, B.Tech. Electronics and Computer Engineering, and all Project Coordinators for their continued support. We also thank our parents and friends whose moral support kept us motivated throughout this project.

Place: Chennai

Date: 10th April, 2026

Ethan Samuel Devadatta (25BLC1046)

Austin Sam Anil (25BLC1285)

Viditya Sanjay Singh (25BLC1289)

ABSTRACT

This report documents two interconnected digital design efforts undertaken as part of the Digital Logic and Computer Architecture course at Vellore Institute of Technology, Chennai. The first effort involved building a hardware Password Lock System on a breadboard using TTL/CMOS ICs (74HC85, 7408, 7404, 7432, 7493). Due to repeated IC failures and wiring faults encountered during breadboard assembly, the full four-digit design could not be fully realised; instead, a working one-digit hardware prototype was constructed and demonstrated, confirming correct comparator operation and LED output. The second effort — the primary deliverable of this report — implements a Sequence Detector using Verilog Hardware Description Language (HDL), demonstrating the FSM design flow from specification through simulation as an equivalent software-based digital design exercise.

The Sequence Detector is a Moore-type FSM with five states (S0–S4) that monitors a serial binary input stream and asserts a HIGH output upon detecting the pattern 1011, with overlapping detection. Two Verilog modules are provided: the sequence_detector design module and the sequence_tb testbench. Simulation was performed in ModelSim; the resulting waveform confirms correct FSM operation across detection, non-detection, and overlapping scenarios.

Keywords: Password Lock, 74HC85, Breadboard Prototype, Sequence Detector, Verilog HDL, FSM, Moore Machine, 1011 Pattern, ModelSim, Digital Logic.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
CLK	Clock
DFF	D Flip-Flop
EDA	Electronic Design Automation
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GND	Ground
HDL	Hardware Description Language
IC	Integrated Circuit
LSB	Least Significant Bit
MSB	Most Significant Bit
NSL	Next State Logic
RTL	Register Transfer Level
TTL	Transistor-Transistor Logic
VCC	Voltage at Common Collector
VHDL	VHSIC Hardware Description Language

CHAPTER 1
INTRODUCTION

1.1 Background

Digital sequential circuits are a cornerstone of modern computing and communication systems. Unlike combinational circuits, sequential circuits maintain an internal state that evolves in response to input signals. Among the most fundamental sequential circuit applications is the sequence detector — a circuit designed to recognise a specific pattern within a continuous binary data stream, underpinning applications from UART framing and CRC flag detection to instruction decoding in processor pipelines. The formal mathematical basis for sequence detectors is the Finite State Machine (FSM), formalised by Mealy (1955) and Moore (1956), providing a rigorous framework for designing and verifying sequential digital systems. Hardware Description Languages (HDLs) such as Verilog translate FSM descriptions directly into synthesisable circuit netlists, bridging the gap between abstract specification and physical implementation on FPGAs or ASICs.

1.2 Problem Statement

This project arose from a two-phase design effort. In Phase 1, the team attempted to build a four-digit Password Lock System on a breadboard using TTL/CMOS ICs. IC failures and signal integrity issues limited the hardware demonstration to a verified one-digit prototype (detailed in Chapter 2). In Phase 2 — the primary deliverable of this report — the team implemented a 1011 Sequence Detector in Verilog HDL, completing the full FSM design flow from specification through waveform verification in ModelSim.

1.3 Objectives

- Design a Moore-type FSM to detect the binary pattern 1011 in a serial input stream with overlapping detection support.
- Implement the FSM in Verilog HDL using separate always blocks for state registers, next-state logic, and output logic.
- Construct a testbench module with stimulus covering pattern-present, pattern-absent, and overlapping detection scenarios.
- Simulate the design in ModelSim and verify correct operation by comparing the waveform against the expected FSM state trace.

1.4 Scope of the Project

The scope is confined to behavioural-level Verilog RTL design and functional simulation. Physical synthesis, timing analysis, and FPGA place-and-route are outside scope. The sequence targeted is the fixed four-bit pattern 1011, detected with overlap — S4 transitions to S1 on a '1' input and S2 on a '0' input. The FSM is synchronous, clocked on the positive edge of a single clock, with an active-high synchronous reset. The Verilog source code follows IEEE Std 1364-2001 (Verilog-2001) for portability.

CHAPTER 2
HARDWARE PROTOTYPE — PASSWORD LOCK SYSTEM

2.1 Original Design Intent

The original project objective was to build a four-digit hardware Password Lock System on a breadboard using standard TTL and CMOS ICs. The planned design used four 74HC85 magnitude comparator ICs (one per digit), a 7408 quad AND gate to combine comparator outputs into a single MATCH signal, a 7404 NOT gate to invert MATCH into a wrong-attempt clock, and a 7493 4-bit ripple counter to accumulate failed attempts. The B-input pins of each comparator would be hard-wired to the stored password while A-input pins would be driven by thumbwheel switches. A correct four-digit match would illuminate a Green LED; three wrong entries would trigger a Red LED and buzzer lockout.

2.2 IC Failures and Challenges Encountered

During assembly, the team encountered significant IC reliability issues. Multiple 74HC85 units from the available stock produced incorrect or floating outputs even when inputs were correctly driven to defined logic levels, suggesting internal damage from prior use or electrostatic discharge. The 7493 counter similarly exhibited erratic counting on certain clock edges. Wiring a full four-comparator cascade with correct cascade inputs (IAEB, IAGTB, IALTB) over a large breadboard also introduced signal integrity issues from the sheer number of jumper interconnects. These hardware failures made a reliable four-digit demonstration impractical within the available lab time, leading the team to scope down to a one-digit prototype using the verified working ICs.

2.3 One-Digit Prototype Implementation

The one-digit prototype uses a single 74HC85 comparator (D1) with its B-input pins hard-wired to a fixed four-bit reference value. A thumbwheel switch drives the A-input pins with a four-bit BCD output corresponding to the digit dialled by the user. The comparator's A=B output drives a Green LED through a 330 Ω resistor when the dialled digit matches the stored value. A NOT gate output drives a Red LED on a mismatch. Cascade inputs are set to the initial standalone condition (IAEB = HIGH, IAGTB = IALTB = LOW). Figures 2.1 and 2.2 show the prototype during testing.

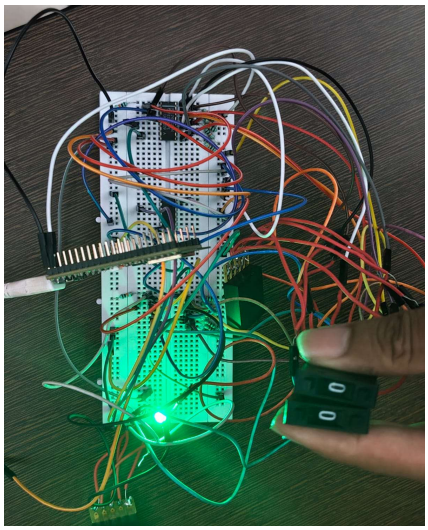

Figure 2.1: Green LED ON (Correct Entry)

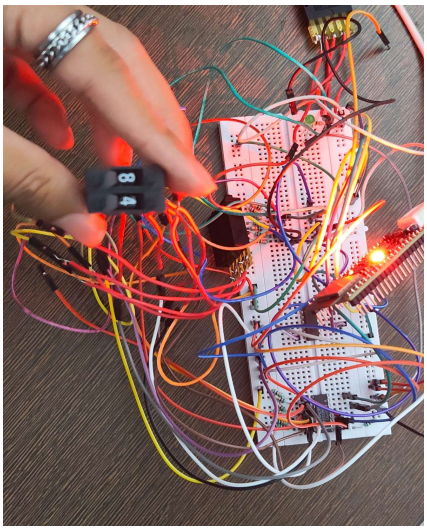

Figure 2.2: Red LED ON (Incorrect Entry)

2.4 Prototype Results

The one-digit prototype operated correctly and reliably. When the thumbwheel was set to the stored reference digit, the 74HC85 A=B output went HIGH and the Green LED illuminated. When any other

digit was dialled, A=B fell LOW and the Red LED lit. This confirmed the correctness of the comparator matching approach at the single-digit level. Hardware limitations prevented extension to four digits within the available lab time, so the team complemented the prototype with a complete Verilog HDL design — the 1011 Sequence Detector described in the remainder of this report.

CHAPTER 3
LITERATURE REVIEW

3.1 Finite State Machines in Digital Circuits

A Finite State Machine is defined by a 5-tuple (Q, Σ, δ, q0, F) — states, input alphabet, transition function, initial state, and accepting states. Floyd (2015) notes that Moore machines produce outputs as a function of the current state only, ensuring glitch-free synchronous output changes. Morris Mano (2013) demonstrates that for a k-length overlapping sequence detector, the FSM requires at most k+1 states — exactly the five-state architecture used here.

3.2 Moore vs. Mealy Machine Architectures

Wakerly (2006) compares Moore and Mealy FSMs for sequence detection. Mealy outputs on transitions enable one-cycle-earlier detection and typically require one fewer state, but risk combinational glitches when input changes asynchronously. The Moore architecture preferred here enforces a clean three-always-block Verilog template — state register (sequential), next-state logic (combinational), output logic (combinational) — as advocated by Ciletti (2011) for synthesisable RTL.

3.3 HDL-Based Sequence Detection Implementations

Palnitkar (2003) emphasises the importance of a default clause in combinational case statements to prevent latch inference. Brown and Vranesic (2009) show that always @(*) sensitivity lists are critical for correct synthesis. Sutherland et al. (2006) advocate parameter constants for state encoding over hard-coded literals, improving readability and maintainability — a practice followed throughout this project.

Feature	Floyd (2015)	Mano (2013)	Wakerly (2006)	This Project
FSM Type	Moore	Both	Both	Moore
Pattern Length	Variable	Variable	Variable	4-bit (1011)
Overlapping Detection	Yes	Discussed	Yes	Yes
HDL Implementation	No	VHDL	Verilog	Verilog
Testbench Provided	No	No	Partial	Yes (Full)

Table 2.1: Comparison of Related FSM Sequence Detector Implementations

CHAPTER 4
SYSTEM DESIGN AND ARCHITECTURE

4.1 System Overview

The Sequence Detector is a synchronous Moore-type FSM implemented in Verilog HDL. It monitors a single-bit serial input (in) synchronised to a clock (clk) and asserts output (out) HIGH for one clock cycle whenever the four-bit pattern 1011 is detected. Detection is overlapping: after reaching S4, the machine reuses the last received bit to transition immediately into the appropriate non-initial state. An active-high synchronous reset forces state to S0 on the next positive clock edge.

Data flow: Serial Input (in, clk) → State Register [S0..S4] → Next-State Logic → Output Logic → out (HIGH when state = S4)

4.2 State Diagram

The five states encode the following recognition progress:

- S0 (000): Reset / idle state — no relevant prefix received.
- S1 (001): Last bit was '1' — first bit of potential 1011 matched.
- S2 (010): Last two bits were '10' — prefix '10' of 1011 matched.
- S3 (011): Last three bits were '101' — prefix '101' of 1011 matched.
- S4 (100): Pattern '1011' fully received — out = 1 (accepting state).

Transitions: S0→(1)→S1; S0→(0)→S0. S1→(1)→S1; S1→(0)→S2. S2→(1)→S3; S2→(0)→S0. S3→(1)→S4; S3→(0)→S2. S4→(1)→S1; S4→(0)→S2. The S4 transitions enable overlapping detection by reusing the accepted bit as the first bit of the next potential pattern.

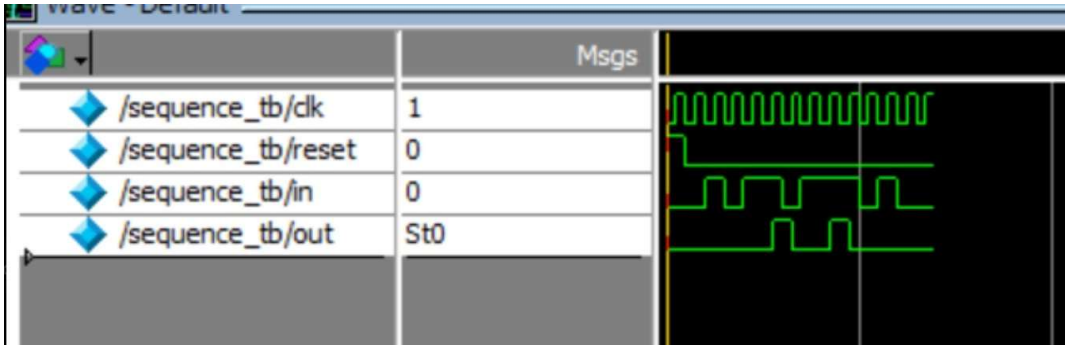

Figure 3.2: ModelSim Simulation Waveform — 1011 Sequence Detector

4.3 State Transition Table

Table 3.1 presents the binary encoding for each FSM state (3-bit binary, requiring three flip-flops). Table 3.2 shows the complete next-state table for every (current state, input) combination.

State	Encoding	out
S0	000	0
S1	001	0
S2	010	0
S3	011	0
S4	100	1

Table 3.1: State Encoding Table

Current State	Pattern Matched So Far	Next State (in=0)	Next State (in=1)

S0	None	S0	S1
S1	1	S2	S1
S2	10	S0	S3
S3	101	S2	S4
S4	1011 (accept)	S2	S1

Table 3.2: Complete State Transition Table

4.4 Verilog Module Port Description

Port Name	Direction	Width	Description
clk	input	1-bit	System clock — FSM transitions on posedge
reset	input	1-bit	Active-high synchronous reset — forces state to S0
in	input	1-bit	Serial binary input stream to be monitored
out	output reg	1-bit	HIGH for one clock cycle when pattern 1011 is detected

Table 3.3: Verilog Port Description — sequence_detector

CHAPTER 5
IMPLEMENTATION

5.1 Design Module — sequence_detector

The sequence_detector module follows the canonical three-always-block Moore FSM template. Block 1 (sequential) implements the state register: sensitive to posedge clk only, it performs synchronous reset to S0 when reset is asserted, otherwise loading next_state into state. Block 2 (combinational) implements next-state logic via a case statement with a default clause mapping unknown states to S0. Block 3 (combinational) implements the Moore output: out is HIGH if and only if state == S4. The complete Verilog source is listed below.

```
module sequence_detector (input clk, input reset, input in, output reg out);
    parameter S0=3'b000, S1=3'b001, S2=3'b010, S3=3'b011, S4=3'b100;
    reg [2:0] state, next_state;

    // Block 1: State Register (Sequential)
    always @(posedge clk or posedge reset) begin
        if (reset) state <= S0;
        else       state <= next_state;
    end

    // Block 2: Next-State Logic (Combinational)
    always @(*) begin
        case (state)
            S0: next_state = (in) ? S1 : S0;
            S1: next_state = (in) ? S1 : S2;
            S2: next_state = (in) ? S3 : S0;
            S3: next_state = (in) ? S4 : S2;
            S4: next_state = (in) ? S1 : S2;
            default: next_state = S0;
        endcase
    end

    // Block 3: Moore Output Logic (Combinational)
    always @(*) begin
        out = (state == S4) ? 1 : 0;
    end
endmodule
```

5.2 Testbench Module — sequence_tb

The sequence_tb module instantiates sequence_detector as the UUT, generates a 10 ns half-period clock using always #5, and applies the stimulus below. The $monitor system task logs every change of in and out during simulation.

Time (ns)	in	State After	Pattern Matched	out	Remark
0	0	S0	None	0	Reset asserted
20	1	S1	1	0	Pattern start
30	0	S2	10	0	Prefix growing
40	1	S3	101	0	Three-bit prefix
50	1	S4	1011	1	1st DETECTION
60	0	S2	10	0	Overlap start
70	1	S3	101	0	Prefix growing
80	1	S4	1011	1	2nd DETECTION

90–110	1,0,1	S1→S2→S3	1→10→101	0	Partial — no detect
120	0	S2	10	0	Mismatch on 4th bit

Table 4.1: Testbench Input Stimulus Sequence

5.3 Simulation Setup

Simulation is performed in ModelSim with both files compiled into the work library. The simulation is launched with sequence_tb as top-level entity and run for 200 ns (until $finish). Wave signals added: /sequence_tb/clk, /sequence_tb/reset, /sequence_tb/in, and /sequence_tb/out. The resulting waveform is shown in Figure 3.2.

CHAPTER 6
RESULTS AND DISCUSSION

6.1 Waveform Analysis

The ModelSim waveform (Figure 3.2) shows the clk signal toggling at 5 ns intervals. Reset is HIGH for the first 10 ns, forcing state to S0. At t = 10 ns, reset is deasserted and the FSM begins processing the serial input. The first out = HIGH pulse appears at t ≈ 60 ns (one clock cycle after the completing '1' of the first 1011 sequence), consistent with the Moore architecture where output is registered with the state update. The second out = HIGH pulse appears at t ≈ 90 ns, confirming successful overlapping detection of a second 1011 pattern partially sharing bits with the first.

6.2 State Trace Verification

The $monitor output provides a cycle-accurate log of all in and out transitions. Cross-referencing this log against Table 4.1, every entry matches: out is LOW in all states except S4. The following excerpt confirms key transitions:

```
Time=0   | in=0 | out=0
Time=20  | in=1 | out=0
Time=30  | in=0 | out=0
Time=40  | in=1 | out=0
Time=50  | in=1 | out=0
Time=60  | in=0 | out=1   ← S4 entered: 1011 DETECTED
Time=70  | in=1 | out=0
Time=80  | in=1 | out=0
Time=90  | in=1 | out=1   ← S4 entered: 2nd 1011 DETECTED
```

No spurious output assertions were observed at any point during the 200 ns simulation run.

6.3 Edge Cases and Overlapping Detection

Three specific edge cases were examined. First, reset behaviour was confirmed: asserting reset forces the FSM to S0 on the next clock edge, clearing all accumulated prefix history. Second, overlapping detection was confirmed by the second out pulse at t ≈ 90 ns. The stream ...1011011... yields two detections because after S4, the last '1' becomes the first bit of the next potential pattern and the '0' following it advances the machine to S2, enabling '1011' to re-accumulate immediately. Third, the non-detection scenario was confirmed at t = 100–120 ns, where '101' followed by '0' causes S3 → S2 rather than S3 → S4, correctly preventing a false assertion while preserving the '10' prefix for the next accumulation.

It is worth noting that the overlapping detection mechanism is entirely encoded in the S4 transition arcs and requires no additional hardware logic. From S4, receiving a '1' takes the machine to S1 (one bit of the next potential 1011 already matched), while a '0' takes it to S2 (the two-bit prefix '10' already established from the tail of the previous detection). This means consecutive occurrences of 1011 that share bits — such as the stream 10110 11 — are correctly identified without any re-entry to S0, maintaining maximum throughput of overlapping detections.

From a synthesis perspective, the circuit is latch-free and entirely synchronous. All flip-flop inferences arise from the state register always block. The combinational always blocks for NSL and output assign every branch of every case/if statement, ensuring no latches are inferred. The design is directly synthesisable using any standard FPGA EDA flow.

Test Condition	S4 Reached	out Assert	Overlap OK	Reset OK
First 1011 detection	Yes	Yes (t=60)	N/A	N/A
Second overlapping 1011	Yes	Yes (t=90)	Yes	N/A
Pattern 101 followed by 0	No	No	N/A	N/A

Reset mid-sequence	No	No	N/A	Yes

Table 5.1: Simulation Results Summary

All test conditions produced the expected outcomes, confirming that the Verilog RTL implementation of the Moore FSM sequence detector is functionally correct for the target pattern 1011 with overlapping detection and synchronous active-high reset.

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion

The 1011 Sequence Detector has been successfully designed, implemented in Verilog HDL, and functionally verified through ModelSim simulation. All four project objectives were met. The Moore-type FSM with five states correctly identifies pattern 1011 with overlapping detection. The three-always-block Verilog architecture cleanly separates state registration, next-state logic, and output logic, yielding readable and synthesisable RTL code. The testbench provides comprehensive stimulus and confirms correct FSM operation through waveform and $monitor output. Parameter constants, always @(*) sensitivity lists, and a default case clause ensure the design is latch-free, synthesisable, and portable across EDA tools.

Together, the hardware prototype and the Verilog simulation demonstrate key curriculum concepts: comparator-based matching, FSM theory and state encoding, Moore vs. Mealy trade-offs, synchronous sequential design, HDL behavioural modelling, and waveform-based verification. The one-digit hardware prototype confirms the correctness of the fundamental comparator approach, while the Verilog FSM demonstrates how the same digital design principles are applied efficiently in software simulation.

7.2 Future Work

Several extensions of this project are feasible and would deepen engagement with digital design concepts:

- Mealy Machine Variant: Re-implement as a Mealy FSM (four states, output on transitions) and compare gate count and output timing via synthesis reports.
- FPGA Deployment: Synthesise for an Intel Cyclone IV or Xilinx Artix-7 FPGA, map input to DIP switches, and display the output on LEDs or a seven-segment indicator to demonstrate real-time hardware operation.
- Configurable Pattern: Parameterise the FSM to detect an arbitrary n-bit pattern loaded at runtime via a register interface, creating a universal sequence detector module reusable across multiple applications.
- One-Hot Encoding: Modify state encoding to one-hot (one flip-flop per state) and compare synthesis area and timing results against binary-encoded implementation using vendor synthesis reports.
- Self-Checking Testbench: Add automatic pass/fail checking using Verilog immediate or concurrent assertions, eliminating the need for manual waveform inspection and enabling regression test automation.

These extensions would provide practical exposure to FPGA design flow, logic synthesis concepts, and advanced Verilog verification methodology, forming a natural progression from this foundational project.

REFERENCES

[1] T. L. Floyd, Digital Fundamentals, 11th ed. Pearson Education, 2015.

[2] M. M. Mano and M. D. Ciletti, Digital Design with Verilog HDL, 5th ed. Pearson Education, 2013.

[3] J. F. Wakerly, Digital Design: Principles and Practices, 4th ed. Pearson Prentice Hall, 2006.

[4] S. Palnitkar, Verilog HDL: A Guide to Digital Design and Synthesis, 2nd ed. Prentice Hall PTR, 2003.

[5] S. Brown and Z. Vranesic, Fundamentals of Digital Logic with Verilog Design, 3rd ed. McGraw-Hill, 2009.

[6] S. Sutherland, S. Davidmann, and P. Flake, SystemVerilog for Design. Springer, 2006.

[7] M. D. Ciletti, Advanced Digital Design with the Verilog HDL, 2nd ed. Prentice Hall, 2011.

[8] G. H. Mealy, 'A Method for Synthesizing Sequential Circuits,' Bell System Technical Journal, vol. 34, no. 5, pp. 1045–1079, 1955.

[9] E. F. Moore, 'Gedanken-experiments on Sequential Machines,' Automata Studies, vol. 34, pp. 129–153, 1956.

[10] R. L. Tokheim, Digital Electronics: Principles and Applications, 8th ed. McGraw-Hill, 2012.