

Digital Voting Machine Using Verilog

Tejas Nanasaheb Rahane, Sakshi Vikram Shelke, Shravani Sambhaji Hinge,
Shelke Nikhil Sandip, Assistant Professor Rahane S.L

Electronics & Telecommunication Engineering, Jaihind College Of Engineering Kuran, Maharashtra, India

Abstract- The Digital Voting Machine Using Verilog HDL is a digital system designed to count votes for four candidates automatically. It uses candidate inputs, clock and reset signals, edge-detection logic, and individual vote counters. A Verilog testbench is used to verify vote counting and analyze simulation waveforms using tools such as EDA Playground or Xilinx Vivado. This project helps students understand digital logic design, sequential circuits, counters, and simulation-based verification without requiring physical hardware. It is intended for educational purposes and can be extended with FPGA implementation and additional security features.

Keywords— Digital Voting Machine, Verilog HDL, Vote Counter, Sequential Logic, Testbench, Simulation, FPGA.

I. INTRODUCTION

A Digital Voting Machine is an electronic system designed to register votes and count them automatically using digital logic circuits. Voting systems play an important role in collecting individual choices and determining results accurately. Traditional manual vote counting can require considerable time, effort, and human resources, particularly when the number of votes increases. Digital systems can simplify this process by automating vote registration and maintaining separate vote counts for each candidate. The project "Digital Voting Machine Using Verilog HDL" focuses on designing and simulating a basic electronic voting system using Hardware Description Language (HDL).

Verilog HDL is a hardware description language widely used to design, model, and verify digital circuits. It allows designers to describe the behaviour and structure of electronic systems, including registers, counters, multiplexers, and control logic. In this project, Verilog HDL is used to develop a voting machine that supports four candidates. Each candidate has a separate input signal, and the system maintains an individual vote counter for each candidate. The design demonstrates how digital logic can be used to process inputs and update stored values in a controlled manner.

The proposed system consists of candidate input signals, a clock signal, a reset signal, edge-detection registers, and four individual vote counters. The clock synchronizes the operation of the system, while the reset signal initializes all counters and internal input states. When a new candidate button press is detected, the corresponding counter increments by one. Edge-detection logic prevents the same continuously held input from being counted repeatedly. In the educational implementation, simultaneous candidate inputs are handled using fixed-priority logic, so the testbench applies one candidate input at a time.

The project is developed entirely using software tools, eliminating the need for physical electronic components during the initial design and testing stages. A Verilog testbench is created to generate the clock, apply reset, simulate different voting sequences, and verify the expected vote totals. Simulation tools such as EDA Playground with Icarus Verilog or Xilinx Vivado can be used to compile and test the design. Waveform analysis helps observe the candidate inputs, clock transitions, reset behaviour, and changes in the vote counters.

One of the main objectives of this project is to understand the practical application of sequential logic circuits in a real-world-inspired system. Through this project, students gain knowledge of digital counters, registers, edge detection, synchronous operations, testbench development,

and functional verification. The modular structure also makes the design easier to understand, test, and modify for future improvements.

The proposed system can be extended by implementing it on an FPGA development board, interfacing an LCD or seven-segment display, supporting additional candidates, and introducing improved input handling. Further research could explore voter authentication, secure result storage, and audit mechanisms. However, the current design is intended for academic demonstration and is not a secure or certified voting system suitable for official elections.

Overall, the Digital Voting Machine Using Verilog HDL provides a practical introduction to digital system design and hardware verification. It demonstrates how a simple voting mechanism can be modelled, implemented, and tested through simulation while establishing a foundation for more advanced FPGA-based digital applications.

Keywords: Digital Voting Machine, Verilog HDL, Electronic Voting, Vote Counter, Sequential Logic, Edge Detection, Testbench, Simulation, Waveform Analysis, FPGA.

II. WORKING

The working of the Digital Voting Machine Using Verilog HDL begins with the initialization of the system. The design uses Verilog HDL to describe the digital logic required to register votes and maintain separate vote counts for four candidates. Initially, the reset signal is activated to clear all four vote counters and initialize the previous input states to zero. After the reset signal is deactivated, the voting machine becomes ready to accept inputs from Candidate A, Candidate B, Candidate C, and Candidate D.

When a candidate's voting input changes from LOW to HIGH, the edge-detection logic identifies the transition as a new button press. The clock signal synchronizes the operation, and at the next rising clock edge, the corresponding candidate's vote counter is incremented by one. For example, when Candidate A receives a valid vote, Counter A

increases by one, while the counters for Candidates B, C, and D remain unchanged. This process ensures that each new button press is registered as a single vote.

The system uses edge-detection registers to store the previous state of each candidate input. By comparing the current input with its previous state, the design prevents the same continuously held input from being counted repeatedly. When the candidate input returns to LOW, the system becomes ready to detect another button press for that candidate. In the current design, if multiple candidate inputs become HIGH simultaneously, the fixed-priority logic selects the first matching candidate according to the programmed priority. Therefore, the testbench applies one candidate input at a time to ensure the expected voting behaviour.

Each candidate has an individual 8-bit vote counter that stores the total number of votes received. These counters can represent values from 0 to 255 before wrapping around. As the voting sequence continues, the counters update independently according to the candidate inputs. The counter values are provided through the output signals and can be observed during simulation to determine the number of votes registered for each candidate.

After implementing the voting logic, a Verilog testbench is developed to verify the complete system. The testbench generates the clock signal, activates and deactivates the reset signal, and applies a predefined sequence of voting inputs. For example, the testbench may register three votes for Candidate A, three votes for Candidate B, two votes for Candidate C, and two votes for Candidate D. At the end of the simulation, the testbench compares the actual counter values with the expected results and displays whether the test has passed or failed.

Simulation tools such as EDA Playground with Icarus Verilog or Xilinx Vivado can be used to compile and test the design. The simulation waveform displays the clock, reset, candidate inputs, and vote counters over time. By examining these waveforms, the designer can verify that the counters reset correctly, increment only for valid new button presses, and

maintain the expected vote totals. This helps identify logical errors before any potential hardware implementation.

Overall, the system demonstrates how digital logic and Verilog HDL can be used to implement a basic electronic voting mechanism. It provides practical knowledge of sequential circuits, clock synchronization, reset operations, edge detection, counters, testbench development, and waveform analysis. The design is suitable for academic learning and software-based mini-projects. However, it is an educational prototype rather than an election-ready system; real-world use would require additional input debouncing,

voter authentication, security controls, privacy protection, and independent auditing.

Advantages

- **Automatic Vote Counting:** The system counts votes automatically for four candidates, reducing manual effort and saving time.
- **Accurate Results:** Individual digital counters maintain separate vote totals and help reduce manual counting errors.
- **Prevention of Repeated Counting:** Edge-detection logic ensures that a continuously pressed button is counted only once.
- **Low-Cost Implementation:** The design can be developed and tested using software simulation without requiring physical hardware.
- **Easy Verification:** A Verilog testbench and simulation waveforms help verify the system's functionality and identify design errors.

Disadvantages

- **Limited Security:** The basic design does not include voter authentication or advanced security mechanisms.
- **Limited Candidates:** The current system supports only four candidates and requires modifications to support more.
- **Counter Overflow:** Each 8-bit counter can store values from 0 to 255; exceeding this limit causes the counter to wrap around.

- **Simultaneous Input Limitation:** If multiple candidate inputs are activated simultaneously, the fixed-priority logic selects only one candidate.
- **Hardware Dependency for Real-Time Use:** The current project is simulation-based and requires FPGA implementation and suitable input/output interfaces for physical operation.

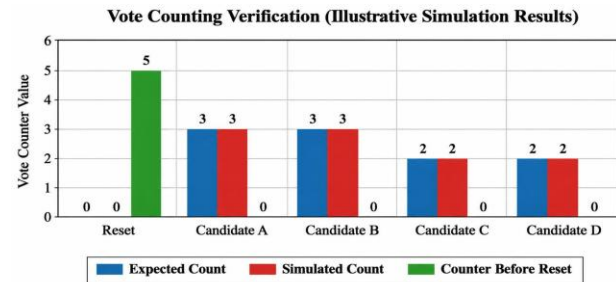


Chart -1: Comparative Processing Performance of FPGA-Based Image Edge Detection

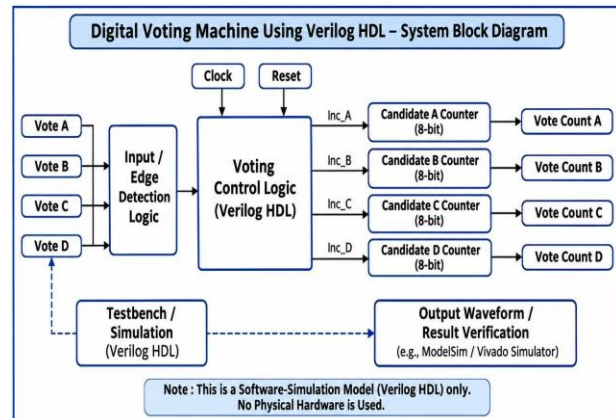


Figure 1: FPGA-Based Image Edge Detection System Using the Sobel Operator

III. CONCLUSION

The Digital Voting Machine Using Verilog HDL project demonstrates the design and simulation of a basic electronic voting system for four candidates. It uses digital logic, edge detection, clock and reset signals, and individual vote counters to register and count votes automatically. The design is verified using a Verilog testbench and simulation waveforms to check the correctness of the voting process and final results.

This project provides practical knowledge of Verilog programming, sequential circuits, digital counters, and functional verification. It is cost-effective and suitable for software-based academic projects without physical hardware. In the future, it can be improved through FPGA implementation, display interfacing, and additional security features.

REFERENCES

1. L. C. Harsha, S. D. Bharatula, B. N. K. Reddy, and K. Sarangam, "Design and Implementation of a Secure and Accurate Electronic Voting Machine Using Verilog on Zynq FPGA," *Journal of Mobile Multimedia*, vol. 21, nos. 3–4, pp. 505–520, 2025, doi: 10.13052/jmm1550-4646.213410. [Online]. Available: <https://doi.org/10.13052/jmm1550-4646.213410>.
2. V. S. Sriraam, V. R. Raghavi, and S. R. Ramesh, "Implementation of Electronic Voting Machine in FPGA," in *Proc. IEEE International Conference on Computing, Electronics and Electrical Engineering (IC2E)*, 2024, doi: 10.1109/IC2E362166.2024.10826813.
3. P. P. M. Prasad et al., "Electronic Voting Machine Using FPGA," *Engineering Research: Perspectives on Recent Advances*, vol. 6, pp. 80–89, 2025, doi: 10.9734/bpi/erpra/v6/2915.
4. Y. Srinivasa Rao, G. N. B. P. Rao, G. Sravanthi, and G. Karunya, "Implementation of Internet Voting Machine for Avoiding Rigging Using Verilog HDL," *International Journal of Versatile Research and Analysis*, vol. 4, no. 3, pp. 786–790, Mar. 2026. [Online]. Available: <https://ijpub.org/ijvra/papers/IJVRA2603618.pdf>.
5. E. Rajitha, S. Kishore Reddy, and K. Dharma, "Xilinx Based Electronic Voting Machine," *International Journal of Versatile Research and Analysis*, 2026. [Online]. Available: <https://www.ijpub.org/ijvra/viewpaperforall.php?paper=IJVRA2607548>.
6. IEEE, *IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language*, IEEE Std. 1800-2023, Feb. 2024, doi: 10.1109/IEEESTD.2024.10458102.
7. G. Howell et al., *Cybersecurity Framework Election Infrastructure Profile*. Gaithersburg, MD, USA: National Institute of Standards and Technology, Jan. 2024, doi: 10.6028/NIST.VTS.200-1.
8. AMD, *Vivado Design Suite User Guide: Logic Simulation*, UG900, version 2024.2, Nov. 2024. [Online]. Available: <https://docs.amd.com/r/2024.2-English/ug900-vivado-logic-simulation>.
9. AMD, *Vivado Design Suite Tutorial: Design Flows Overview*, UG888, version 2025.1, Jun. 2025. [Online]. Available: <https://docs.amd.com/r/2025.1-English/ug888-vivado-design-flows-overview-tutorial>.
10. AMD, *UltraFast Design Methodology Guide for FPGAs and SoCs*, UG949, version 2026.1, Jun. 2026. [Online]. Available: <https://docs.amd.com/r/en-US/ug949-vivado->