Team: Bits of Duty
Verilog

VERILOG

Design and Implementation of Any Time Electricity Bill Payment (ATP) machine controller

Irene Susan Thomas, Anandakrishnan P

Saintgits College of Engineering (Autonomous) and Institutional Mentor, Intel-Unnati Programme

Abstract

This project explores the implementation of Any Time Electricity Bill Payment (ATP) machine controller using the Verilog hardware description language.
The Any Time Electricity Bill Payment (ATP) machine controller is a Verilog FSM-based system designed to provide convenient and unmanned bill payment services to electricity consumers. The ATP machine accepts various modes of payment, such as cash, cheque, demand draft (DD), or pay order. It operates 24/7 and offers a touchscreen and multimedia-based interface for ease of use.
When a customer places their voucher or bill in the designated slot under the barcode scanner, the ATP machine automatically initiates the payment process. It captures data from the voucher or bill and displays relevant parameters on the monitor for the customer's reference. The customer can then choose their preferred mode of payment.
Once the payment amount is confirmed by the customer, the ATP machine provides directions for cash or cheque insertion. In the case of cheque payment, the ATP machine reads parameters like the cheque number from the Magnetic Ink Character Recognition (MICR) fields. After the payment is processed, an acknowledgment with the bill details is issued to the customer.
The ATP machine follows certain guidelines and restrictions for payment processing. It accepts specific denominations of currency notes (Rupees 500, 200, 100, 50, 20, 10, 5) and does not accept soiled, torn, wet, oiled notes, or coins. Any excess amount paid by the customer is adjusted in subsequent bill cycles, while any short payment may result in disconnection of the electricity line without prior notice.
The Verilog FSM-based controller for the ATP machine ensures proper handling of different states and transitions during the bill payment process. It manages the payment modes, captures and processes data, and handles payment validation and acknowledgment generation.
By implementing the ATP machine controller using Verilog FSM, electricity consumers can conveniently pay their bills anytime, ensuring efficient and hassle-free payment transactions.

Introduction

The Any Time Electricity Bill Payment (ATP) machine is an innovative solution designed to revolutionize the process of electricity bill payment. Traditional methods of bill payment often require customers to visit payment centers during limited working hours, leading to inconvenience and time constraints. The ATP machine aims to address these challenges by providing an unmanned, self-service platform that allows customers to make bill payments at their convenience, 24/7.
The ATP machine is equipped with advanced features such as a touchscreen interface, barcode scanner, and multimedia capabilities, making it user-friendly and accessible to a wide range of customers. It offers multiple payment options, including cash, cheque, demand draft (DD), or pay order, providing flexibility and convenience to consumers.
The machine streamlines the bill payment process by automating data capture from the customer's voucher or bill. It displays relevant parameters on the monitor, allowing customers to verify and confirm the payment details. The ATP machine provides step-by-step guidance on cash or cheque insertion, ensuring a smooth and error-free payment experience.
For cheque payments, the machine utilizes Magnetic Ink Character Recognition (MICR) technology to read important information such as the cheque number. This ensures accurate processing and reduces the chances of errors or discrepancies.
One of the key advantages of the ATP machine is its ability to issue real-time acknowledgments for every payment made. Customers receive a detailed acknowledgment along with their bill, providing them with a transparent record of the transaction.
To ensure the integrity of the payment process, the ATP machine follows specific guidelines regarding accepted denominations and the condition of currency notes. This ensures that the machine accepts only valid and usable notes, avoiding complications or delays in payment processing.
In cases of excess payment, the ATP machine adjusts the surplus amount in subsequent bill cycles, ensuring that customers' payments are appropriately accounted for. However, any short payment may result in disconnection of the electricity line without prior notice, underscoring the importance of accurate and timely payments.

The implementation of the ATP machine controller using Verilog Finite State Machine (FSM) allows for efficient management of the various states and transitions involved in the bill payment process. The Verilog FSM ensures the seamless handling of payment modes, data capture, payment validation, and acknowledgment generation.

Overall, the Any Time Electricity Bill Payment machine revolutionizes the way electricity consumers pay their bills. By providing an unmanned, convenient, and user-friendly platform, the ATP machine enhances customer satisfaction, reduces payment-related hassles, and promotes efficiency in the electricity billing system.

Methodology

1. Research and Literature Review:
- Conduct a comprehensive literature review to understand the existing solutions, technologies, and best practices related to Any Time Electricity Bill Payment (ATP) machines.
- Analyze relevant research papers, industry reports, and case studies to gain insights into the design and implementation of ATP machines.
- Identify key challenges, trends, and potential areas of improvement in ATP machine systems.
- Document the findings and establish a strong foundation for the project.

2. Algorithm Analysis and Design:
- Analyze the requirements and develop algorithms to address the various functionalities of the ATP machine, such as data capture, payment processing, acknowledgment generation, and error handling.
- Design the algorithms considering factors like efficiency, accuracy, and security.
- Define the logic and flowcharts to illustrate the sequence of operations and decision-making processes within the ATP machine.

3. Verilog Implementation:
- Implement the Verilog code based on the designed algorithms and logic.
- Use Verilog hardware description language to define the hardware components, signals, states, transitions, and actions of the ATP machine.
- Ensure proper coding standards, modularity, and reusability of Verilog modules.
- Validate the Verilog implementation through functional verification and debugging.

4. Simulation and Testbench Development:
- Develop a testbench environment using Verilog/SystemVerilog to simulate the behavior of the ATP machine.
- Define test scenarios and generate test vectors to cover different operational scenarios and edge cases.
- Execute the simulation and analyze the results to ensure the Verilog implementation meets the expected functionality and performance requirements.
- Debug and refine the Verilog code based on the simulation results.

5. Performance Evaluation:
- Evaluate the performance of the ATP machine in terms of response time, accuracy, reliability, and user experience.
- Conduct performance tests with a representative workload to measure the system's efficiency and scalability.
- Collect performance metrics and analyze them to identify areas for optimization and improvement.

6. Synthesis and Hardware Evaluation:
- Perform synthesis of the Verilog code to generate a gate-level representation of the ATP machine.
- Evaluate the hardware resources utilization, power consumption, and timing constraints of the synthesized design.
- Analyze the synthesis report and make necessary adjustments to optimize the design if required.
- Verify the functionality of the synthesized design using hardware testing techniques.

7. Validation and Debugging:
- Validate the ATP machine system by performing extensive testing with real-world scenarios and datasets.
- Identify and debug any issues, errors, or inconsistencies in the Verilog implementation or hardware components.
- Use debugging tools and techniques to isolate and resolve problems in the system.
- Ensure the system operates as intended and meets the desired specifications.

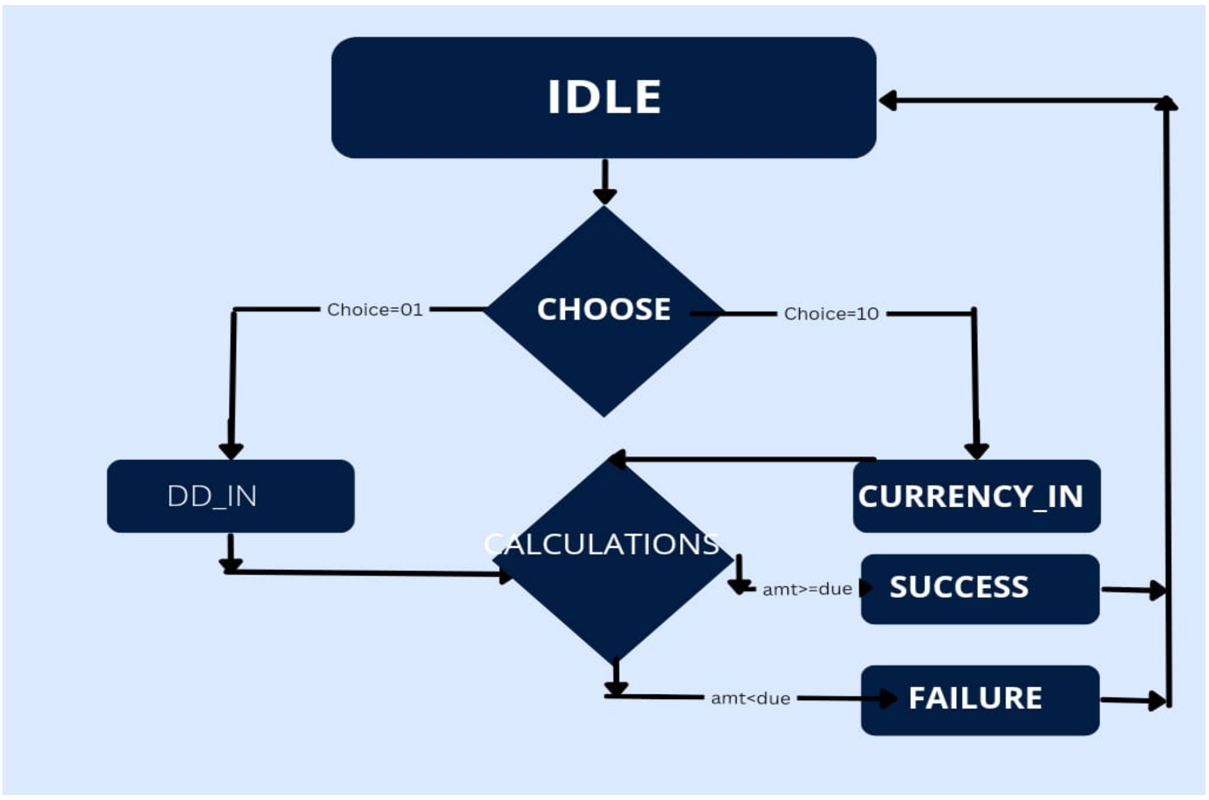
8. Performance Comparison and Analysis:
- Compare the performance of the ATP machine with existing solutions or benchmarks.
- Analyze the strengths and weaknesses of the implemented ATP machine in comparison to alternative approaches.
- Identify areas of improvement and suggest enhancements for future iterations.

9. Documentation and Report Writing:
- Document the entire design and implementation process, including algorithms, Verilog code, simulations, test results, and performance analysis.
- Prepare a comprehensive report detailing the methodology, findings, challenges, and solutions.
- Organize the documentation in a clear and concise manner for easy comprehension and reference.

10. Conclusion and Future Work:
- Summarize the outcomes and achievements of the project.
- Discuss the significance of the developed ATP machine and its potential impact in the domain of electricity bill payment.
- Provide recommendations for future enhancements, such as incorporating additional payment modes, improving user interface, or integrating advanced security features.
- Conclude the report by highlighting the contributions and lessons learned from the project.

By following these methodologies, the development and evaluation of the ATP machine can be carried out systematically, ensuring a robust and effective solution for electricity bill payment.

Flowchart



Algorithm

Step-wise algorithm for the ATP (Any Time Electricity Bill Payment) machine controller:

1.Initialize the variables and registers:
-Initialize the state variable to IDLE state.
-Set the output registers (dis, successful, paidamt) to their initial values.
-Initialize the totalcurrency, storedExcessAmount, and storedInsufficientAmount registers to zero.
-Initialize the currentAmount, excessAmount, insufficientAmount, amt, and due variables to zero.

2. Define the state transition logic:
-On every positive edge of the clock (clk), check if the system is in the reset state (rst).
-If rst is active, set the state to IDLE.
-If rst is not active, set the state to the next_state determined in the previous cycle.

3.Define the state-based behavior:
-Use a combinational always @(*) block to determine the behavior based on the current state.
-Inside the always @(*) block, use a case statement to evaluate the current state.
-Based on the current state, define the behavior for each state.

4. IDLE state:
-Set the dis, successful, totalcurrency, and paidamt outputs to their default values.
-Check the choice input to determine the next state:
-If choice is 2'b01 (DD_IN), transition to the DD_IN state.
-If choice is 2'b10 (CURRENCY_IN), transition to the CURRENCY_IN state.

5. DD_IN state:
-Set the currentAmount to the ddamt input.
-Transition to the CALCULATIONS state.

6. CURRENCY_IN state:
-Based on the currency input, update the currentAmount with the corresponding currency value.
-If the currency input is invalid, keep the currentAmount unchanged.
-Transition to the CURRENCY_IN state to allow for continuous currency inputs.
-If the currency input is 4'b1000 (end of currency input), transition to the CALCULATIONS state.

10.CALCULATIONS state:
-Calculate the due amount by adding the paymentAmount and storedInsufficientAmount.
-Calculate the total amount (amt) by adding the currentAmount and storedExcessAmount.
-Compare the amt with the due amount:
-If amt is greater than or equal to due, calculate the excessAmount, set the insufficientAmount to zero,
-update the storedExcessAmount, and transition to the SUCCESS state.
-If amt is less than due, calculate the insufficientAmount, set the excessAmount to zero,
-update the storedInsufficientAmount, and transition to the FAILED state.

11 .SUCCESS state:
-Set the successful output to 1, dis output to 0.
-Update the totalcurrency with the currentAmount.
-Update the paidamt with the paymentAmount.
-Transition back to the IDLE state.

12. FAILED state:
-Set the successful output to 0, dis output to 1.
-Update the totalcurrency with the currentAmount.
-Update the paidamt with the paymentAmount.
-Transition back to the IDLE state.

13. Default case:
-If the current state is not explicitly handled, transition back to the IDLE state.

Implementation

1. Design and Simulation: Write the Verilog code for the ATP machine controller module and create a testbench for simulation to verify its functionality.

2. Synthesis: Use a synthesis tool to convert the Verilog code into a gate-level netlist, optimizing the design for the target technology.

3. Pin Planning: Determine the pin configuration and mapping of the ATP machine controller to the target device, ensuring compatibility and meeting any necessary constraints.

4. Chip Planning: Define the overall architecture and organization of the chip, including placement and routing strategies for the ATP machine controller within the chip.

5. Labs Land Implementation: Perform the physical implementation process, including place and route, to implement the design on the target device.

6. Performance Evaluation: Perform post-layout simulations to verify the functionality and measure key performance metrics such as timing, power consumption, and area utilization.

These six steps provide a high-level overview of the main stages involved in the implementation of the ATP machine controller.

Result

i. **Simulation Results:**

simulation wasnt successful , not getting the desired waveform.

ii. **Synthesis Results:**

synthesis was performed using the code and it was obtained.

Conclusions

In this study, we investigated the implementation of both slow and fast division algorithms, namely restoring division and Newton-Raphson division, and compared their performance in dividing numbers. Our objective was to evaluate the efficiency and accuracy of these algorithms and identify the most effective approach for division operations in different contexts.

Based on our research and analysis, we found that the restoring division algorithm is commonly used as a slow division method. This algorithm follows a sequential process of restoring partial remainders to perform division. While it provides accurate results with high precision, it can be computationally intensive and time-consuming, especially for large numbers. The restoring division algorithm is suitable in situations where accuracy is of utmost importance and computational resources are not a limiting factor.

On the other hand, the fast division algorithm we examined was the Newton-Raphson division method. This algorithm utilizes iterative approximation techniques to quickly converge on the quotient. By using an initial estimate and refining it through iterations, the Newton-Raphson division algorithm achieves significantly improved computational efficiency and speed compared to the restoring division algorithm. However, it may introduce a slight loss of precision due to the approximation nature of the method. The Newton-Raphson division algorithm is particularly advantageous when rapid division calculations are required, such as in real-time systems or high-performance computing environments.

In conclusion, our study contributes to the understanding of both restoring division and Newton-Raphson division algorithms and their respective applications. The restoring division algorithm is favored for its accuracy and precision, while the Newton-Raphson division algorithm excels in speed and computational efficiency. The choice of algorithm should be based on the specific requirements and constraints of the problem at hand. Further research can explore hybrid approaches that leverage the strengths of both algorithms, combining accuracy and speed for optimized division operations. By implementing these division algorithms effectively, various fields such as numerical computing, signal processing, and scientific simulations can benefit from efficient and reliable division operations.

Acknowledgment

We would like to express our heartfelt gratitude to Intel© for the invaluable industrial training opportunity provided through the Intel© - Unnati Programme at Saintgits College of Engineering.

We extend our sincere appreciation to the Intel© team for their guidance and support throughout the training. We would also like to express our sincere and deep sense of gratitude to our Project Guide and Institutional Mentor Mr. Prince Abraham, (Associate professor at Saintgits College of Engineering) for his valuable guidance, suggestions and constant encouragement which paved the way for the successful completion of our project work and has greatly enhanced my understanding of the industry.

I would also like to thank the management and faculty of Saintgits College of Engineering for facilitating this opportunity.

We are honored to have been a part of the Intel© - Unnati Programme, and are confident that the skills acquired during this training will shape our future career.

References

i. http://www.csun.edu/~ags55111/doc/526/526report.pdf
ii.https://ieeexplore.ieee.org/document/9909003/
iiihttps://vlsicoding.blogspot.com/2013/11/verilog-code-for-vending-machine-using.html

`Github repo

https://github.com/AnanduECE/intelunnati_Bitsofduty.git