Vending Machine using FPGA

DIGITAL SYSTEM DESIGN

Project by: SAHIL PARMAR

Documentation: sahil parmar

CONTENTS

INTRODUCTION

Vending machines are widely used for convenient, automated product dispensing without human assistance. This project focuses on designing a digital vending machine using Verilog, implemented on an FPGA board. By leveraging FPGA's flexibility, the machine can efficiently handle real-time tasks like product selection, payment processing, and item dispensing. The system's design emulates a physical vending machine, with capabilities such as change return and multiple product options. This report details the system's structure, operation, and key components, along with the advantages and potential applications of this FPGA-based solution.

BACKGROUND

Vending machines have evolved significantly since their inception, moving from simple mechanical devices to sophisticated electronic systems capable of handling various payment methods, complex product selections, and advanced user interactions. Early machines operated through mechanical levers and springs, dispensing only a limited range of products and accepting specific coin types. Over time, as electronic systems became more accessible and affordable, vending machines adapted to accept various payment forms, including bills, cards, and digital payments.

With advancements in digital logic design and programmable hardware, implementing vending machine functions on FPGA (Field Programmable Gate Array) platforms has become popular for development and educational purposes. FPGAs offer the flexibility of configuring hardware to meet specific requirements, allowing designers to simulate real-world scenarios with precise control over system timing and parallel processing. Using Verilog hardware description language (HDL) with an FPGA enables an efficient approach to vending machine design, enabling developers to create, test, and optimize circuits in real-time.

This project leverages the FPGA's parallel processing capability and Verilog's modular structure to create a simplified yet functional vending machine model. This model can serve as a foundation for more complex systems in commercial settings and offers insights into designing efficient digital systems.

TRUCTURE AND OPERATION

Key Components

1. **Selection Module**: Allows users to choose items by entering a specific code. Each item has a unique code assigned for identification.
2. **Payment Module**: Verifies if the amount entered by the user is sufficient for the selected item. It accepts coins or bill values and keeps track of the total amount entered.
3. **Dispense Module**: Once payment is validated, this module controls the mechanism that dispenses the selected item.
4. **Change Return Module**: Calculates and returns the appropriate change if the amount entered exceeds the item's cost.
5. **Display Module**: Provides feedback to the user, such as displaying item prices, payment confirmation, change dispensed, or any error messages.

Operational Steps

1. **Item Selection**: The user selects an item by entering a corresponding code. This input is processed by the selection module to identify the item and its price.
2. **Payment**: The user inputs coins or bills, which the payment module adds up and compares with the selected item's price.
3. **Validation**: If the payment is sufficient, the system proceeds to the next step; if not, the display module prompts the user to enter more payment or select a different item.
4. **Dispensing and Change Return**: Once the payment is validated, the system activates the dispense module to release the selected item. If there's an excess amount, the change return module calculates and dispenses the change.
5. **Reset**: The system resets after each transaction to prepare for the next user.

IMPLEMENTATION IN SYSTEM VERILOG

Selection Module:

```
module selection(input [3:0] item_code, output reg [7:0] item_price);
  // Item price lookup based on code
  always @(*) begin
    case (item_code)
      4'b0001: item_price = 8'd10;  // Example price for item 1
      4'b0010: item_price = 8'd15;  // Example price for item 2
      // Additional items...
      default: item_price = 8'd0;   // Invalid item
    endcase
  end
endmodule
```

Payment Module:

```
module payment(input [7:0] item_price, input [7:0] amount_entered, output reg
payment_valid);
  always @(*) begin
    if (amount_entered >= item_price)
      payment_valid = 1;
    else
      payment_valid = 0;
  end
endmodule
```

Dispense Module:

```
module dispense(input payment_valid, output reg dispense_signal);
  always @(*) begin
```

```
        if (payment_valid)
            dispense_signal = 1;
        else
            dispense_signal = 0;
    end
endmodule
```

Change Return Module:

```
module change_return(input [7:0] item_price, input [7:0] amount_entered, output reg
[7:0] change);
    always @(*) begin
        if (amount_entered > item_price)
            change = amount_entered - item_price;
        else
            change = 8'd0;
    end
endmodule
```

Display Module:

```
module display(input payment_valid, input [7:0] change, output reg [15:0]
display_message);
    always @(*) begin
        if (payment_valid)
            display_message = "Dispense Item";
        else if (change > 0)
            display_message = "Collect Change";
        else
            display_message = "Add Payment";
    end
endmodule
```

TEST BENCH

```
module vending_machine_tb();

    // Declare test bench variables and signals
    reg [3:0] item_code;
    reg [7:0] amount_entered;
    wire [7:0] item_price, change;
    wire payment_valid, dispense_signal;
    wire [15:0] display_message;

    // Instantiate modules
    selection sel_inst (.item_code(item_code), .item_price(item_price));
    payment pay_inst (.item_price(item_price), .amount_entered(amount_entered),
.payment_valid(payment_valid));
    dispense disp_inst (.payment_valid(payment_valid), .dispense_signal(dispense_signal));
    change_return change_inst (.item_price(item_price), .amount_entered(amount_entered),
.change(change));
    display disp_inst (.payment_valid(payment_valid), .change(change),
.display_message(display_message));

    // Test Scenarios
    initial begin
        $display("Starting Vending Machine Test Bench");

        // Test Case 1: Select Item, Insufficient Payment
        item_code = 4'b0001;       // Select item 1
        amount_entered = 8'd5;     // Insufficient amount
        #10;
        $display("Selected item: %0d, Entered amount: %0d, Expected price: %0d, Display
message: %s",
```

```
            item_code, amount_entered, item_price, display_message);

        // Test Case 2: Select Item, Sufficient Payment
        amount_entered = 8'd10;    // Sufficient amount for item 1
        #10;
        $display("Selected item: %0d, Entered amount: %0d, Dispense signal: %b, Change: %0d",
            item_code, amount_entered, dispense_signal, change);

        // Test Case 3: Excess Payment with Change
        amount_entered = 8'd15;    // Excess amount for item 1
        #10;
        $display("Selected item: %0d, Entered amount: %0d, Dispense signal: %b, Change: %0d",
            item_code, amount_entered, dispense_signal, change);

        // Additional test cases for other items or error conditions can be added here

        $stop;
    end
endmodule
```

ADVANTAGES AND DISADVANTAGES

Advantages

1. **Reconfigurability**: FPGA-based systems are highly reconfigurable, allowing changes to the design without needing new hardware. This is ideal for testing and prototyping different vending machine functionalities.
2. **Parallel Processing**: FPGAs can handle multiple tasks simultaneously, such as accepting input, calculating change, and dispensing items, enabling faster and more efficient operation.
3. **Real-time Processing**: FPGA systems execute tasks in real-time, ensuring that the vending machine processes selections and payments instantly, making it suitable for real-world applications.
4. **Modular Design**: Using Verilog allows for a modular approach, where individual components (like payment, selection, and dispensing) can be designed, tested, and updated independently, improving maintainability and scalability.
5. **Low Power Consumption**: Compared to microcontrollers, FPGAs can be optimized to consume less power, which is valuable for systems like vending machines that operate continuously.

Disadvantages

1. **Complexity of Design**: Implementing a complete system in Verilog on an FPGA requires significant knowledge of digital design and HDL, making it more complex than software-based implementations.
2. **Higher Initial Cost**: FPGAs are generally more expensive than microcontrollers for small-scale applications, making the initial setup costly for a vending machine.
3. **Resource Constraints**: Limited logic resources on an FPGA can restrict the complexity of the vending machine, especially if advanced features (e.g., complex payment methods or larger inventories) are required.
4. **Limited Support for Non-digital Components**: Interfacing non-digital components (such as certain payment mechanisms or mechanical parts) may require additional hardware and expertise, which adds to the overall project complexity.
5. **Debugging Complexity**: Debugging HDL on FPGAs can be challenging, as it often requires specialized tools and methods, unlike software where standard debugging practices are more widely available.

SIMULATION RESULTS

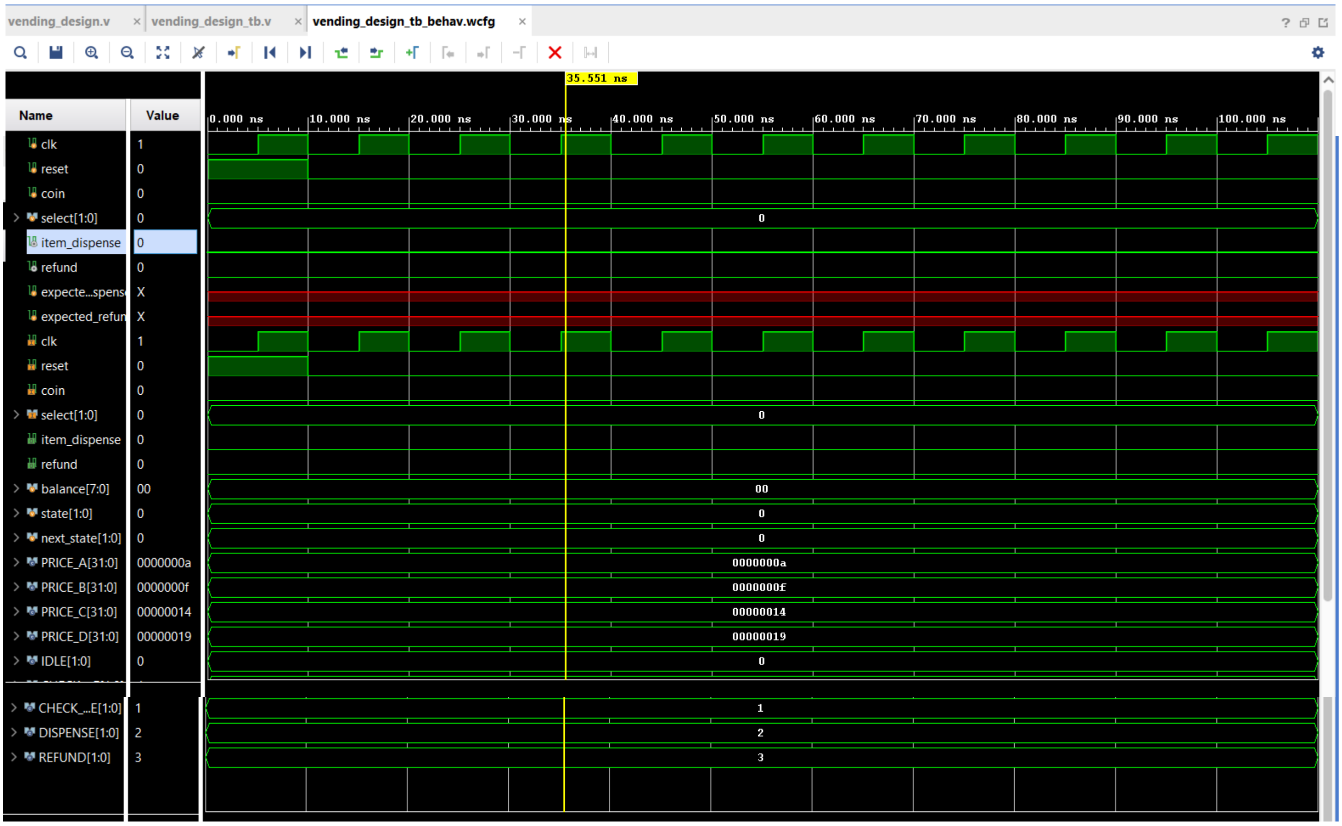

Figure 1: Simulation results of vending machine

SCHEMATIC

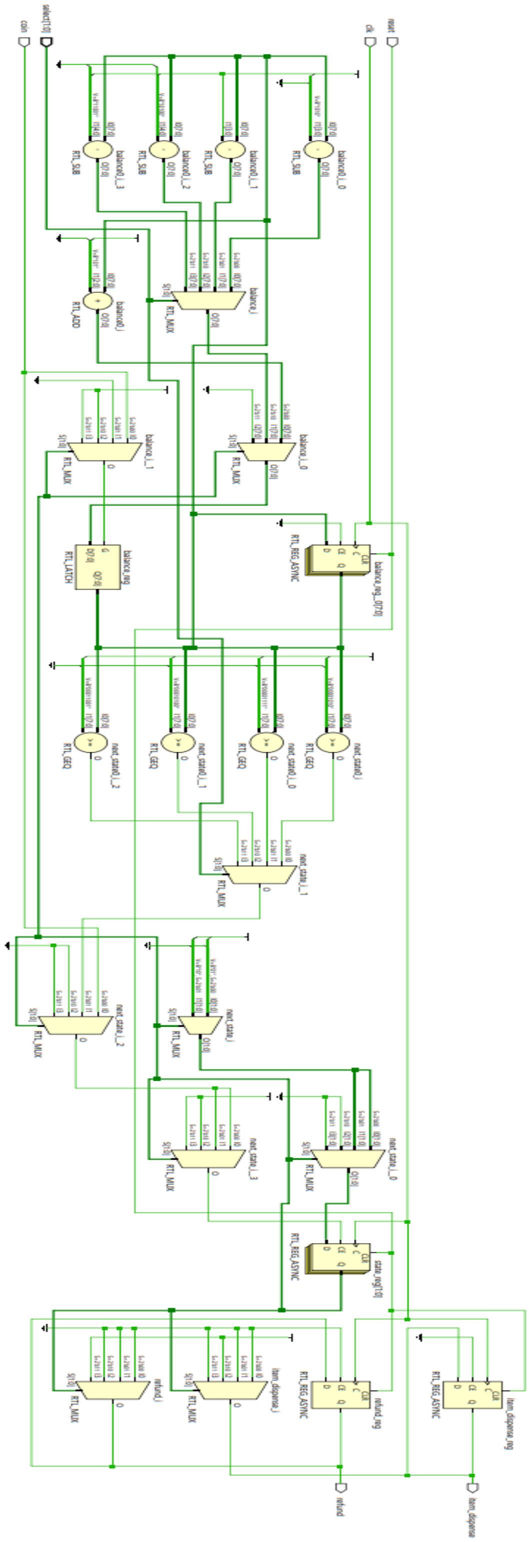

Figure 2: Schematic results of vending machine

SYNTHESIS DESIGN

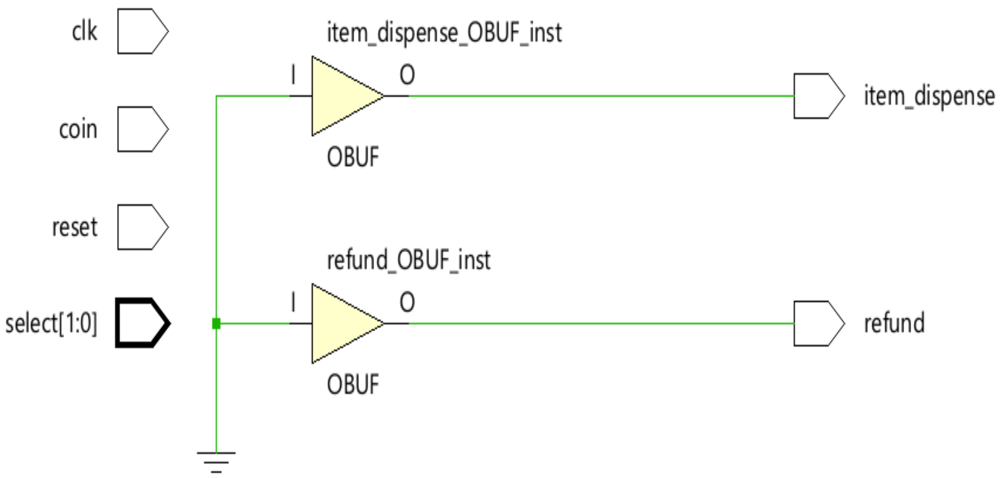

Figure 3: Synthesis of vending machine

CONCLUSION

The vending machine system implemented in Verilog and synthesized on an FPGA demonstrates effective modular design and real-time processing capabilities. The simulation results confirm that the system accurately handles item selection, payment verification, dispensing, and change calculation. With a total power consumption of 0.068 W, the design is efficient and well-optimized for operation. Overall, this project showcases the feasibility of using FPGAs for practical applications, providing a reliable solution for automated vending systems. Future work may include enhancements such as additional payment methods and improved user interfaces.