Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Mini Project Report

on

“Digital Vending Machine”

[Code No: EEEG202]

(For partial fulfilment of II Year/ I Semester in Computer Engineering)

Submitted by:

Aditya Pokharel (32)

Submitted to:

Mr. Om Nath Acharya

Department of Electrical and Electronics Engineering

Kathmandu University

26th February, 2025

Acknowledgement

I would like to extend my deepest gratitude to my **EEEG202** instructor, **Mr. Om Nath Acharya**, for his invaluable guidance, encouragement and support throughout this project. His insightful supervision and expertise provided me with a strong foundation to design and implement a functional digital vending machine. This project has been an enriching learning experience, and I truly appreciate the opportunity to apply theoretical knowledge to a practical application.

Abstract

This project presents the design and implementation of a digital vending machine using logic circuits. The system integrates combinational and sequential logic to facilitate amount insertion, product selection and change calculation. The vending machine allows users to insert various denominations, with a comparator ensuring sufficient balance before dispensing the selected item. Each item's availability is managed using a 4-bit asynchronous down counter. Multiple 7 segment displays provides real-time feedback on the inserted amount, required balance, product availability, dispensed product and calculated change. Flip-flops are used as memory device to store inserted amount and product selection ensuring smooth operation. The project demonstrates the practical application of digital logic design, emphasizing its role in automated transaction systems.

Bonafide Certificate

This mini project on

"Digital Vending Machine"

is the bonafide work of

Aditya Pokharel (32)

who carried out this mini project under my supervision.

Subject Instructor

____

Mr. Om Nath Acharya

Assistant Professor

Department of Electrical and Electronics Engineering

Kathmandu University

Table of Contents

List of Figures

Chapter 1 Introduction

Vending machines have become an essential part of modern automated retail systems, offering convenience by dispensing products without human intervention. This project focuses on designing a digital vending machine using logic circuits, incorporating both combinational and sequential logic to ensure smooth functionality.

The system enables users to insert amount in various denominations, with a comparator verifying if the inserted amount meets the required cost. If sufficient, the selected product is dispensed, and the remaining balance is displayed. The vending machine is designed to handle a maximum of 15 units per product, managed by a 4-bit asynchronous down counter, which decrements upon each purchase and allows restocking up to full capacity.

The project utilizes fundamental digital logic components including 4-bit adder/subtractor, 4-bit comparator, 4-bit memory using D-flip flops, 4-bit asynchronous down counter using T-flip flops and 7-segment displays in order to implement features such as amount insertion, product selection, product availability and change calculation. The goal is to apply theoretical knowledge of digital logic design to a practical system that simulates real-world vending machine operations efficiently.

Chapter 2 Circuit Composition and Working

2.1 Components

2.1.1 Built-in Components

The components available in logisim that were directly used in this project are:

1. **7 Segment LED Displays**
   Function: Visually display number or character in human-readable format
   Role: Display inserted amount, required amount, calculated change, dispensed product and number of units per product

2. **Logic Gates (AND, OR, NOT, NOR, XOR)**
   Function: Perform basic logic operations based on input signals
   Role: Implement combinational logic and process input signals in the circuit

3. **Full Adder**
   Function: Perform addition of three binary inputs (A, B and C_{in}) producing a sum and carry out
   Role: Handle multi-bit operation by adding corresponding bits and propagating carry to next stage

4. **Flip Flops (D and T)**
   Function: Store binary states and maintain system memory
   Role: Store inserted amount and selected product using D Flip Flops and implement counter mechanism for tracking product inventory using T Flip Flops

5. **Input and Output Switches**
   Function: Test and debug by manually setting logic levels
   Role: Create input and output terminals for individual custom-designed components

6. **Buttons**

Function: Correspond to inputs according to user actions

Role: Trigger clock, preset and reset for flip flops

7. **Constants**

Function: Implement fixed value i.e. either 0 or 1

Role: Provide fixed values that never change during operations

2.1.2 Custom-Designed Components

The manually created components designed to fulfill specific functions in this project are:

1. **4-bit Binary to 7 Segment Display**

The 4-bit Binary to 7-Segment Display is designed to convert a 4-bit binary input into a decimal representation on two 7-segment displays. It includes four binary inputs (representing values from 0 to 15) and one enable input that controls whether the display is active. The design utilizes NOT gates, AND gates, NOR gates and OR gates to determine the activation of segments for each decimal value.

The binary inputs and their complements are connected to 16 AND gates, each corresponding to a decimal value from 0 to 15. This functions as a decoder, ensuring that only one output is active at a time.

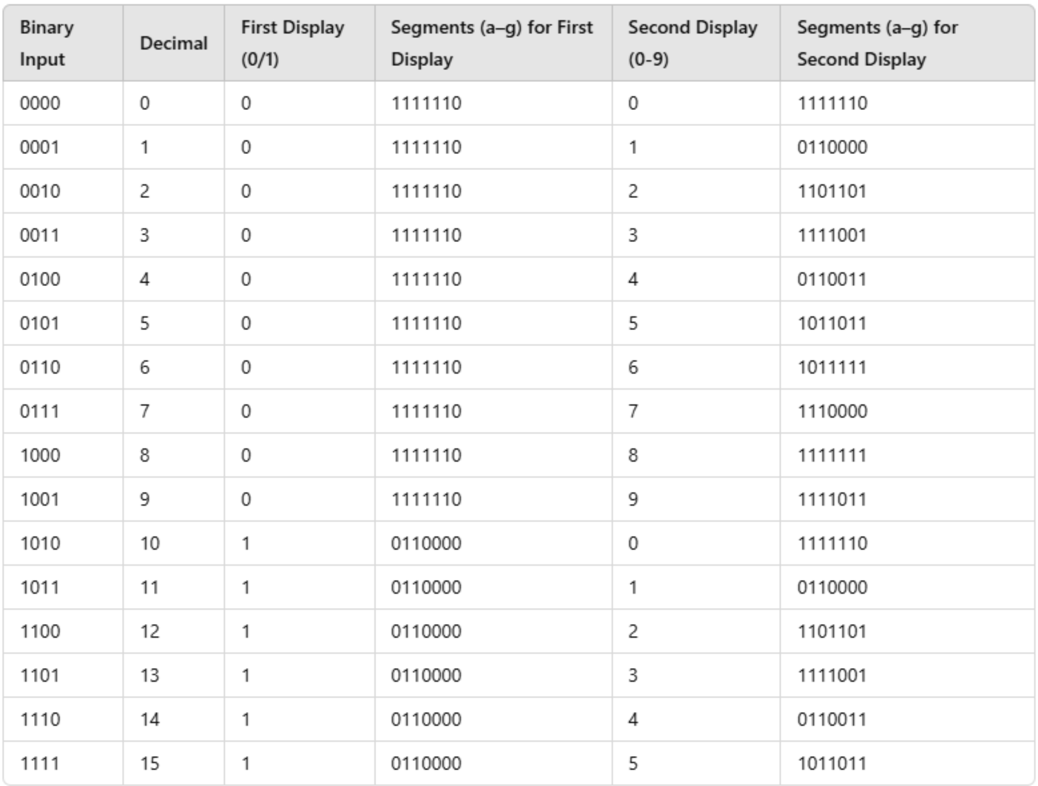
The first 7-segment display represents either 0 or 1, indicating whether the number is in the range 0-9 or 10-15. Segments b and c are always ON (1) since only 0 and 1 need to be displayed. Segment g is always OFF (0) as it is not required. Segments a, d, e and f are controlled by a NOR gate, which turns them OFF when the input is greater than 9.

The second 7-segment display shows digits 0-9 and uses a combination of

NOR and OR gates to activate the correct segments. Each segment (a-g) is controlled based on the 4-bit binary input**.** NOR gates are used to turn off segments for specific numbers, ensuring the correct digit is displayed. Segment e**,** instead of using NOR, is controlled by an OR gate to simplify logic and reduce the number of required inputs. This ensures efficient activation of segments while minimizing unnecessary logic complexity.

The Enable input is globally connected to all 14 segment outputs using 14 AND gates so that the display remains active only when Enable is 1. If the Enable input is 0, all outputs remain off, ensuring no display activity when the circuit is disabled.

This implementation ensures efficient binary-to-decimal conversion while optimizing circuit complexity using combinational logic techniques.



Binary Input	Decimal	First Display (0/1)	Segments (a–g) for First Display	Second Display (0-9)	Segments (a–g) for Second Display
0000	0	0	1111110	0	1111110
0001	1	0	1111110	1	0110000
0010	2	0	1111110	2	1101101
0011	3	0	1111110	3	1111001
0100	4	0	1111110	4	0110011
0101	5	0	1111110	5	1011011
0110	6	0	1111110	6	1011111
0111	7	0	1111110	7	1110000
1000	8	0	1111110	8	1111111
1001	9	0	1111110	9	1111011
1010	10	1	0110000	0	1111110
1011	11	1	0110000	1	0110000
1100	12	1	0110000	2	1101101
1101	13	1	0110000	3	1111001
1110	14	1	0110000	4	0110011
1111	15	1	0110000	5	1011011

Figure 1: Truth table for 4-bit Binary to 7 Segment Display

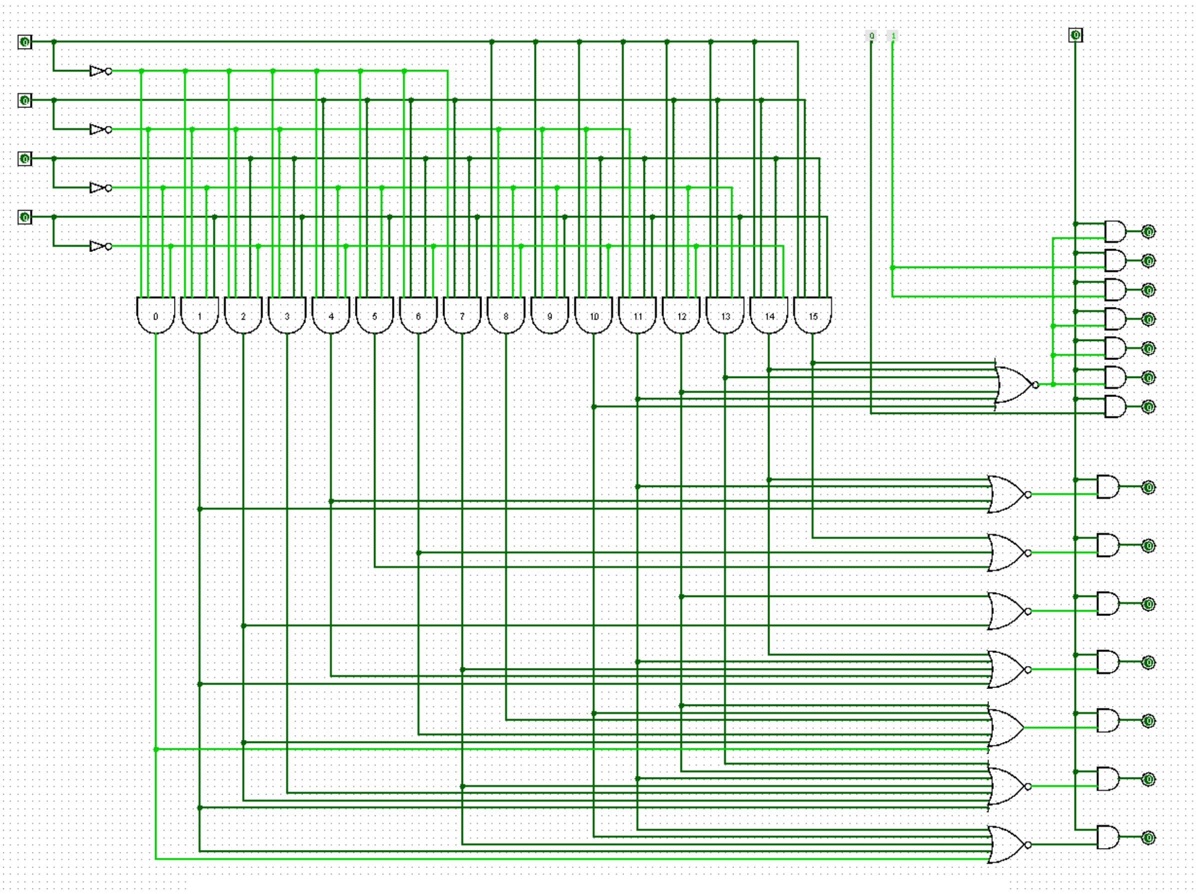


Figure 3: Circuit for 4-bit Binary to 7 Segment Display

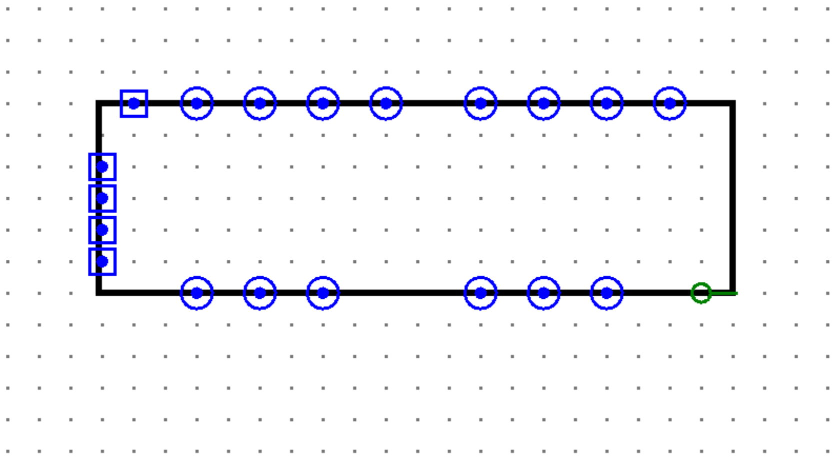


Figure 2: Block for 4-bit Binary to 7 Segment Display

2. 4-bit Adder/Subtractor

The 4-bit Adder/Subtractor is a combinational circuit that performs both addition and subtraction based on a control input. It takes two 4-bit binary numbers (A and B) and produces a 5-bit output (4-bit result + carry). The operation is controlled by a Mode (M) input, where M = 0 performs addition, and M = 1 performs subtraction. To handle subtraction, the circuit first modifies B using four XOR gates, where each bit of B is XORed with M. If M = 0, B remains unchanged and if M = 1, B is inverted, forming the first step of two's complement conversion. The carry-in of the first full adder is also set to M, ensuring proper two's complement subtraction when needed. The modified B and A are then passed through four full adders, where carry is propagated from the least significant bit (LSB) to the most significant bit (MSB). The final result consists of four sum bits and a carry-out, which may indicate overflow in addition or borrow in subtraction. This design efficiently integrates both operations in a single circuit while ensuring correct arithmetic results through minimal hardware.

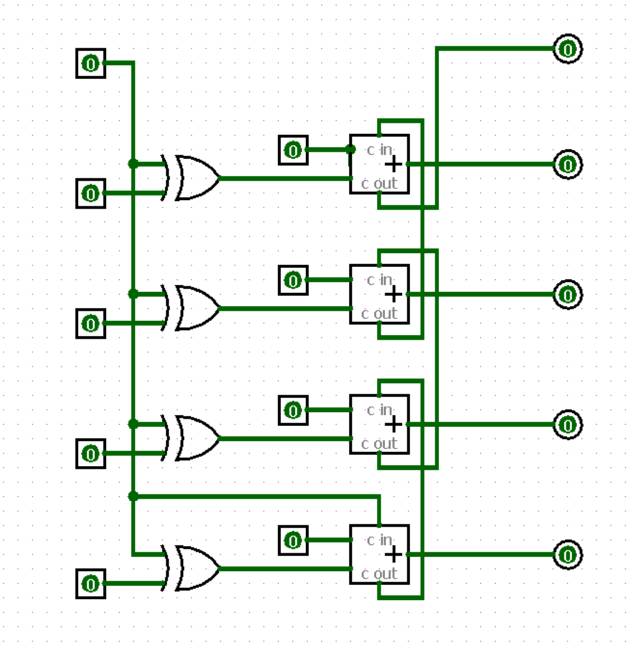


Figure 4: Circuit for 4-bit Adder/Subtractor

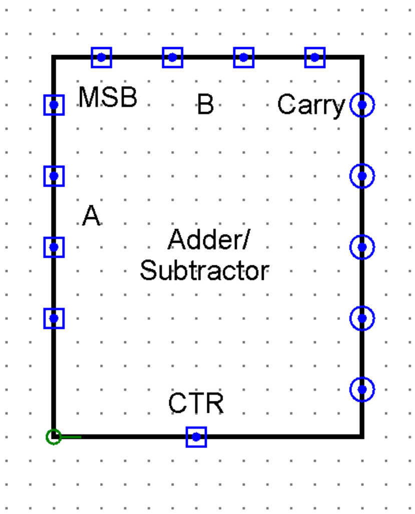


Figure 5: Block for 4-bit Adder/Subtractor

3. 4-bit Comparator

A 4-bit comparator is a combinational logic circuit that compares two 4-bit binary numbers, A (A3, A2, A1, A0) and B (B3, B2, B1, B0), and determines whether A is greater than or equal to B. It outputs 1 if A ≥ B, otherwise, it outputs 0.

The comparison starts from the most significant bit (MSB) and moves to the least significant bit (LSB). If A3 > B3, A is greater, and the output is immediately set to 1. If A3 = B3, the decision depends on the next bit, A2 and B2, and this process continues until either A is determined to be greater or all bits are equal. If A < B at any stage, the output is set to 0, and further comparisons are unnecessary. If all bits are equal, the output remains 1 since A is at least equal to B.

The implementation uses combination of gates forming XNOR gates to check bit equality, AND gates to ensure lower bits are compared only when higher bits match, and OR gates to combine the results for the final decision.

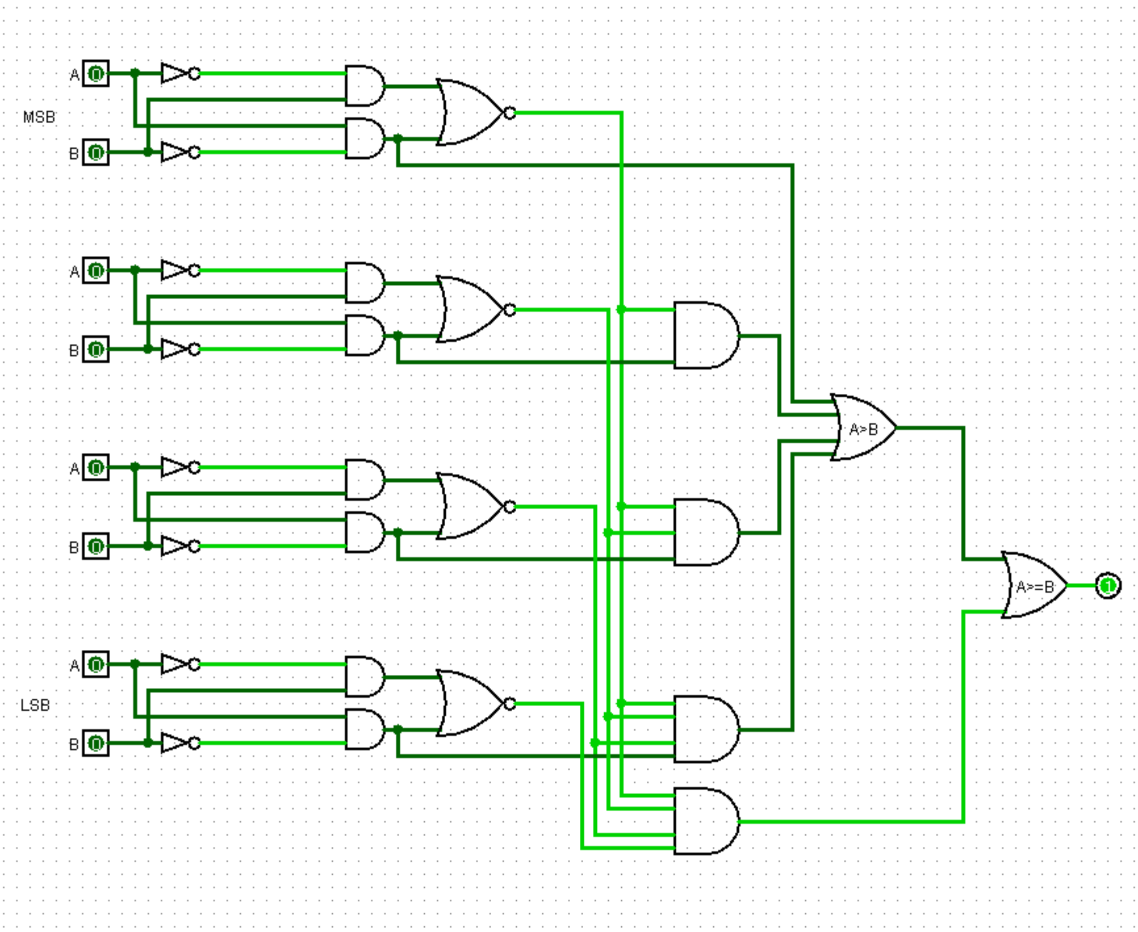


Figure 7: Circuit for 4-bit Comparator

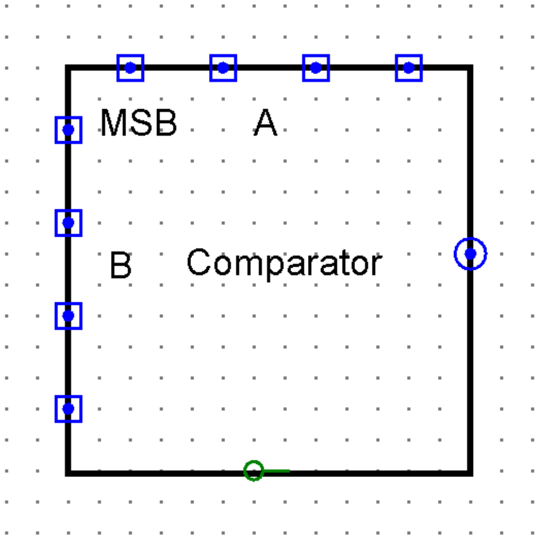


Figure 6: Block for 4-bit Comparator

4. 4-bit Memory

A 4-bit memory unit is designed using four D flip-flops, where each flip-flop stores a single bit of data. This allows the circuit to hold a 4-bit value that can be updated synchronously with a clock signal. The memory operates based on three main control signals: Clock, Clear and Disable.

The clock input controls when new data is stored in the memory. On each rising edge of the clock signal, the data present at the 4-bit data inputs is transferred to the outputs, updating the stored value. However, this update only occurs if the memory is enabled.

The clear input is used to reset the memory. When activated, it forces all outputs to 0, effectively clearing the stored value. This ensures that the memory can be reset whenever necessary.

The disable input determines whether the memory should accept new data or retain its current value. It is connected to a NOT gate, which inverts its logic. When disable is 1, the NOT gate outputs 0, effectively disabling the flip-flops and preventing any changes to the stored data, even if the clock signal is active. Conversely, when disable is 0, the NOT gate outputs 1, allowing the flip-flops to update their values on the next clock pulse.

Overall, this 4-bit memory unit functions as a small storage element for storing inserted amount and selected product’s amount.

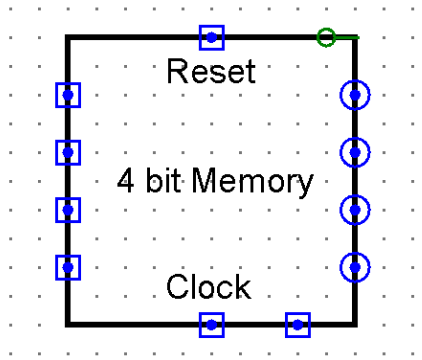


Figure 8: Block for 4-bit Memory

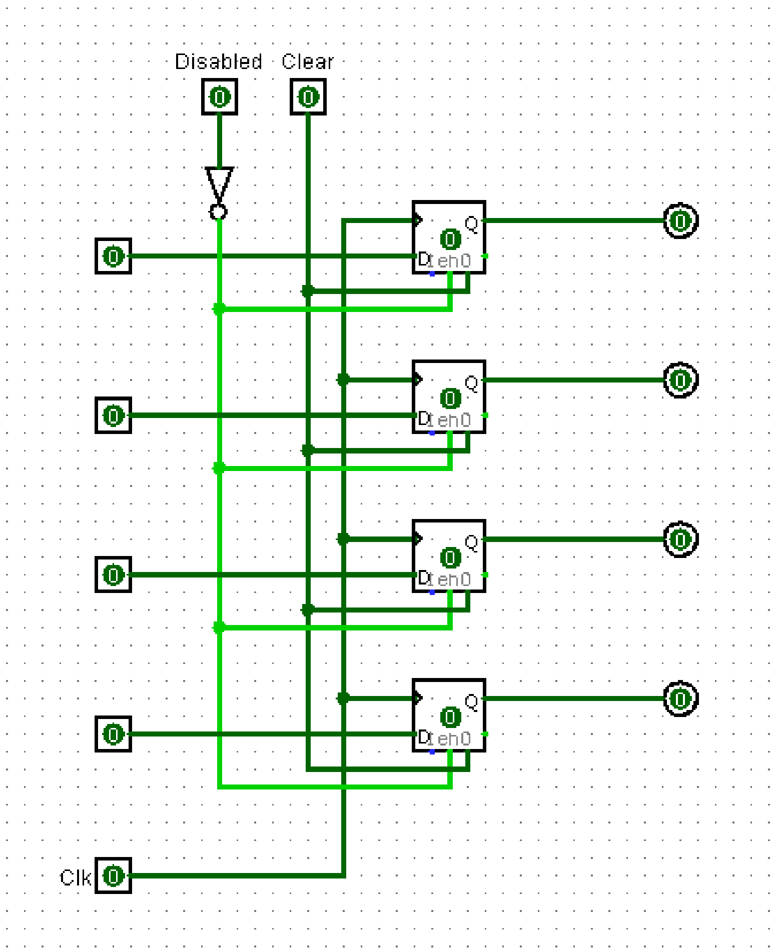


Figure 9: Circuit for 4-bit Memory

5. 4-bit Asynchronous Down Counter

A 4-bit asynchronous down counter with rising edge triggering consists of four T flip-flops, where each flip-flop toggles its state on the rising edge of the clock signal. This counter decrements on each clock pulse, following a binary sequence from 1111 (15) down to 0000 (0).

The first flip-flop receives an external clock input, and the remaining flip-flops take their clock from the output of the previous flip-flop. Since this is an asynchronous counter, only the first flip-flop is triggered directly by the external clock, while the others toggle on the rising edge of the preceding flip-flop's output.

A preset input is used to initialize the counter to 1111 (15). When activated, this forces all flip-flops to output 1, setting the counter to its maximum value.

To determine whether the counter has reached 0000, all flip-flop outputs are fed into an OR gate. When at least one output is 1, the OR gate output is 1, indicating that the count has not reached zero. When all outputs are 0000, the OR gate output becomes 0, signaling that the counter has reached zero.

A disable input is included to control the counter’s operation. The OR gate output is combined with the disable input using an AND gate. When disable is 1, the counter functions normally, decrementing on each clock pulse. When disable is 0, or when the counter reaches 0000, the AND gate output becomes 0, preventing further toggling of the flip-flops and stopping the counter.

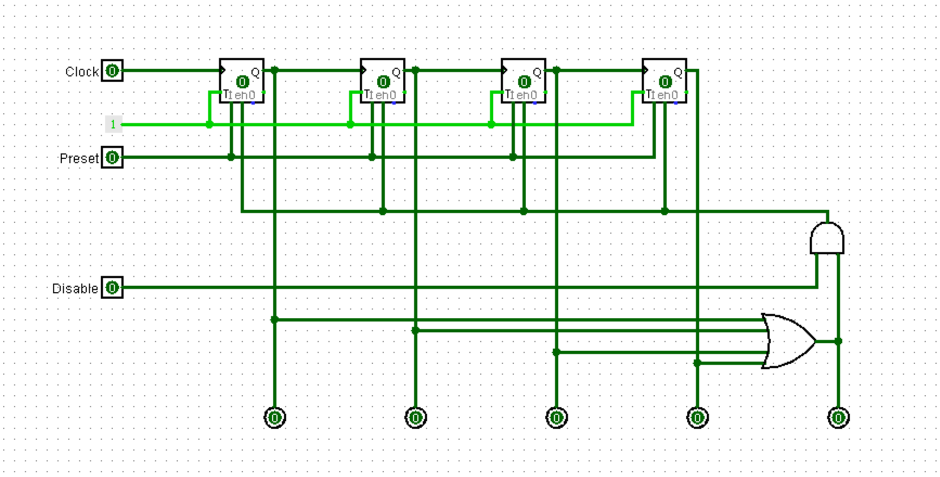


Figure 11: Circuit for 4-bit Asynchronous Down Counter

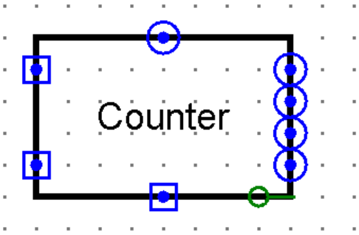


Figure 10: Block for 4-bit Asynchronous Down Counter

6. Amount Insertion Sub-Circuit

The Amount Insertion Sub-Circuit is responsible for processing monetary inputs and maintaining a running total of the inserted amount. It has four inputs corresponding to denominations $1, $2, $5 and $10, which are converted into a 4-bit binary value using OR gates (acting as an encoder). This binary value is then fed into a 4-bit adder/subtractor module, where the control input is set to 0 to perform addition.

The output of the adder is stored in a 4-bit memory module (implemented using D flip-flops), which retains the accumulated value. The memory's stored value is also fed back as the second input to the adder, ensuring continuous summation with each new input. The final output of this system is the 4-bit memory’s value, representing the total inserted amount.

To handle overflow and manual disabling, the carry output from the adder and a manual disable input are connected to an OR gate, which then controls the memory’s disable function. This ensures that the memory module is disabled if the total amount exceeds its limit or if it is manually turned off. Additionally, a clear input is provided to reset the memory when required.

The clock signal for updating the memory is generated using a NOR gate applied to the four money input signals. This setup ensures that the memory updates on the falling edge of any monetary input, allowing for precise synchronization of value accumulation.

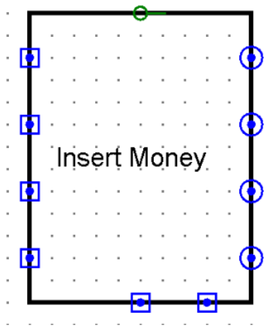


Figure 12: Block for Amount Insertion

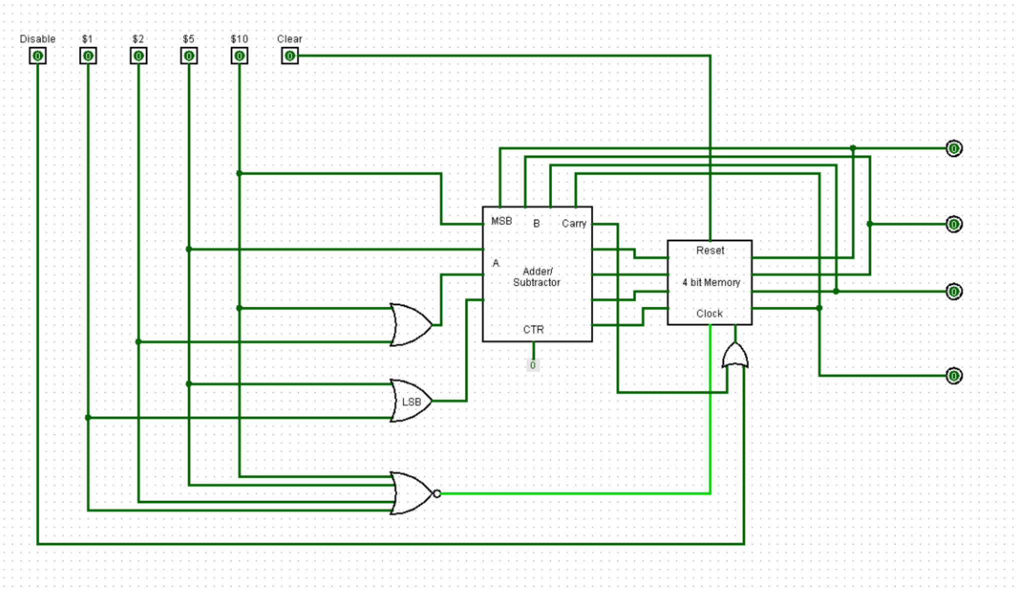


Figure 13: Circuit for Amount Insertion

7. Product Selection Sub-Circuit

The Product Selection Sub-Circuit is responsible for processing product selection amounts based on predefined monetary inputs. It has five inputs corresponding to values $2, $3, $6, $8 and $12, which are converted into a 4-bit binary value using OR gates (acting as an encoder). This binary representation is then stored in a 4-bit memory module (using D flip-flops) to retain the selected amount.

The memory module provides outputs representing the stored binary value. Additionally, the circuit includes a clear input to reset the memory when needed and a disable input that, when activated, prevents updates to the memory.

The clock signal for updating the memory is generated using an OR gate applied to all five money inputs. This setup ensures that the memory updates precisely on the rising edge of any product selection input, allowing for accurate and synchronized operation.

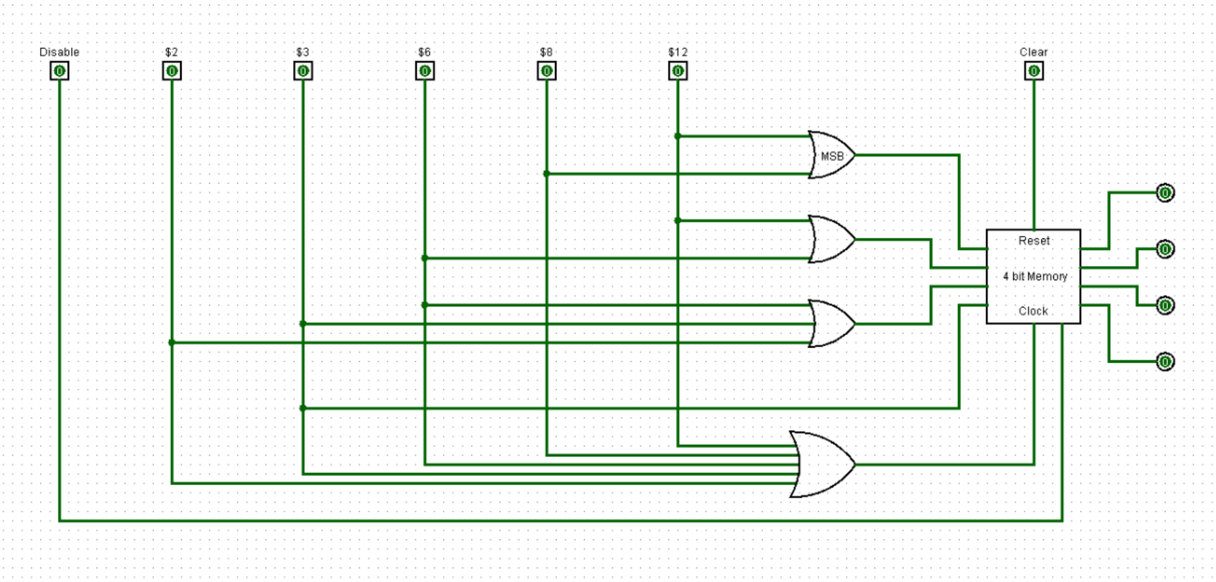


Figure 14: Circuit for Product Selection

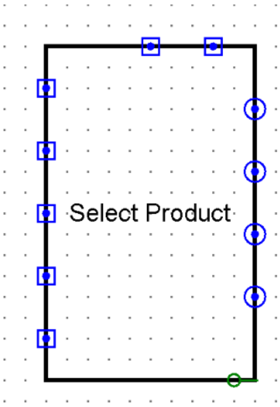


Figure 15: Block for Product Selection

8. Calculation Sub-Circuit

The Calculation Sub-Circuit is responsible for verifying whether the inserted money is sufficient for a selected product and computing the remaining balance.

It has two 4-bit inputs, one representing the product price and the other representing the money inserted. These inputs are fed into a 4-bit comparator, which outputs 1 if the inserted money is greater than or equal to the product price.

Simultaneously, the inputs are also sent to a 4-bit subtractor, where the control signal is set to 1, ensuring that the operation performed is subtraction (money - price). If the price is greater than the inserted money, the result becomes negative.

To handle negative values, the subtractor's carry output is inverted to a second 4-bit adder/subtractor's control input using a NOT gate ensuring it acts as the subtractor only when carry is 0 otherwise the previous value is maintained by neglecting the carry output. This subtractor takes the previous subtraction result as one input and an all-ones (1111) value masked by a NOT gate as the second input. This ensures that if the previous subtraction was negative, the circuit correctly represents the absolute value of the deficit.

The final output represents the difference in price and money which acts as the change to be returned when money is greater or insufficient money when price is greater. The comparator's output determines whether the remaining balance is change to be returned or insufficient money.

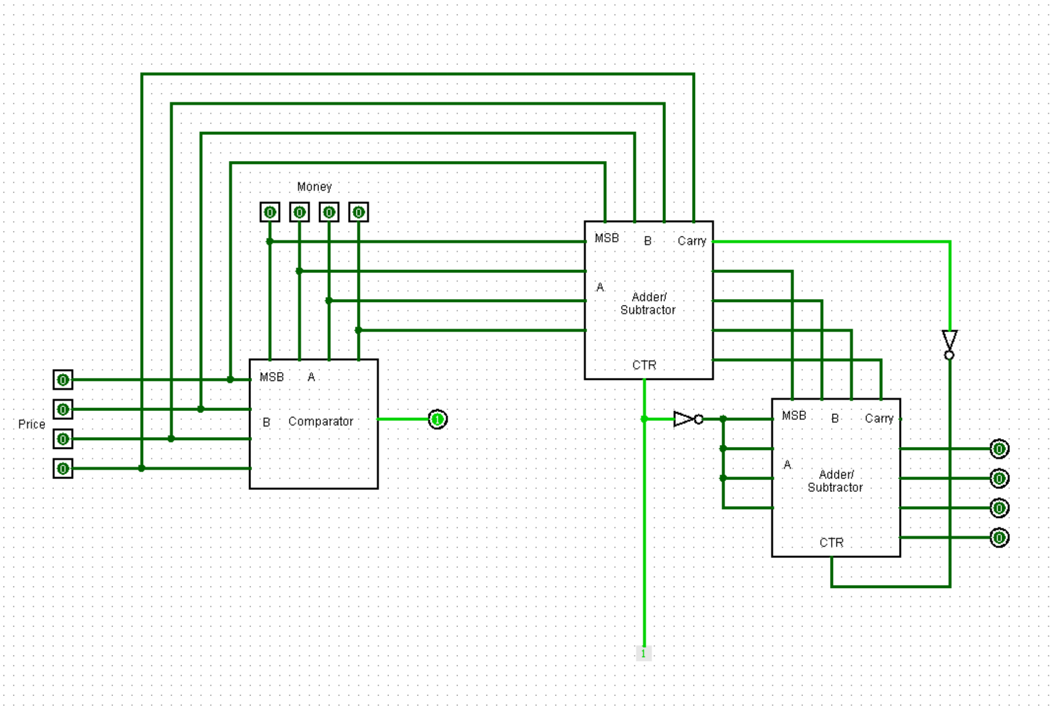


Figure 16: Circuit for Calculation

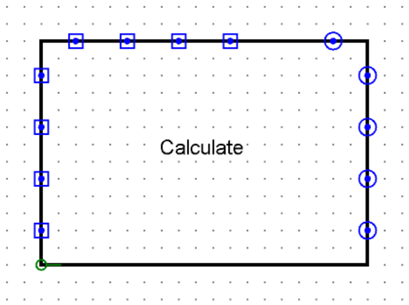


Figure 17: Block for Calculation

9. Product Display Sub-Circuit

The Product Display Sub-Circuit is responsible for identifying the selected product based on its price and enabling the 7-segment display accordingly.

It has 4 binary inputs representing the product's price. These inputs, along with their negations, are fed into five AND gates corresponding to the products priced at $2, $3, $6, $8, and $12. This setup acts as a decoder, ensuring that only one of the five AND gates outputs 1, representing the selected product.

The outputs of these AND gates are used to determine the selected product and are also processed for the 7-segment display signals.

Segments a, e, and f are directly connected to the enable signal since it must be active for all products. Segment g is controlled by the AND of enable and the NOR of products having prices $6 and $8, ensuring it is active only when necessary. Segment d is activated by the AND of enable and the NOT of product having price $2, meaning it turns on for all product except product having price $2. Segments b and c are activated by the AND of enable and the NOR of products having prices $6 and $12, ensuring they are ON when displaying the appropriate product information.

This circuit effectively decodes the selected product and ensures the 7-segment display operates only when the enable is ON and visually represents the valid product.

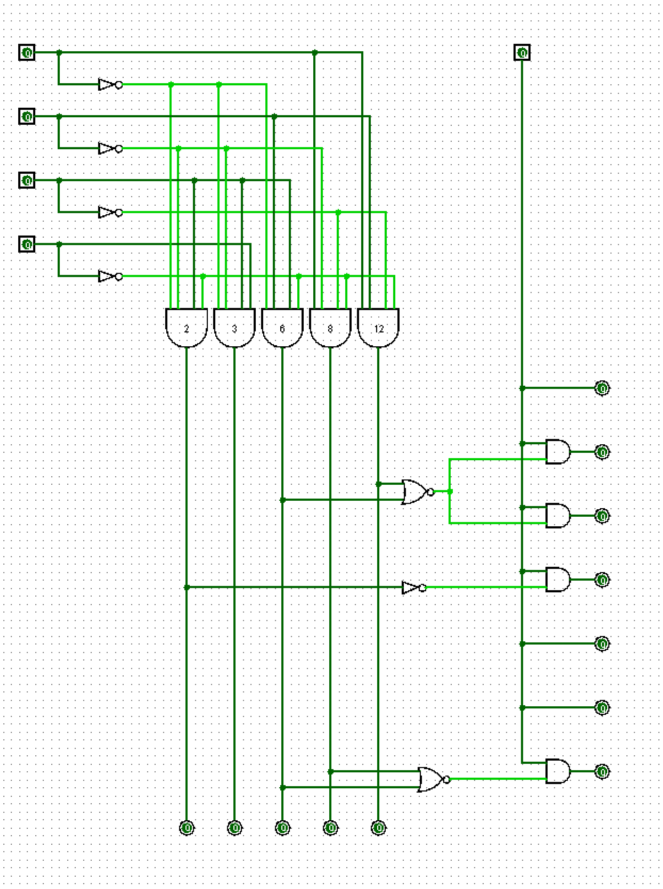


Figure 19: Circuit for Product Display

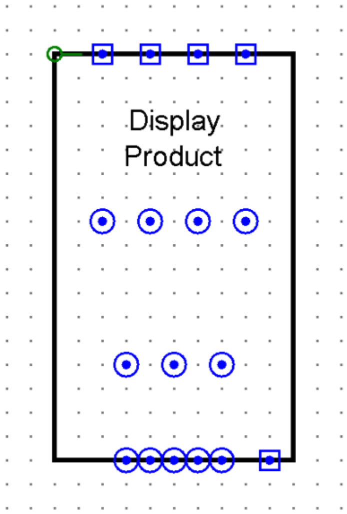


Figure 18: Block for Product Display

2.2 Circuit Functionality

The main circuit integrates multiple sub-circuits to form a digital vending machine system for handling amount insertion, product selection, calculations and display functionalities.

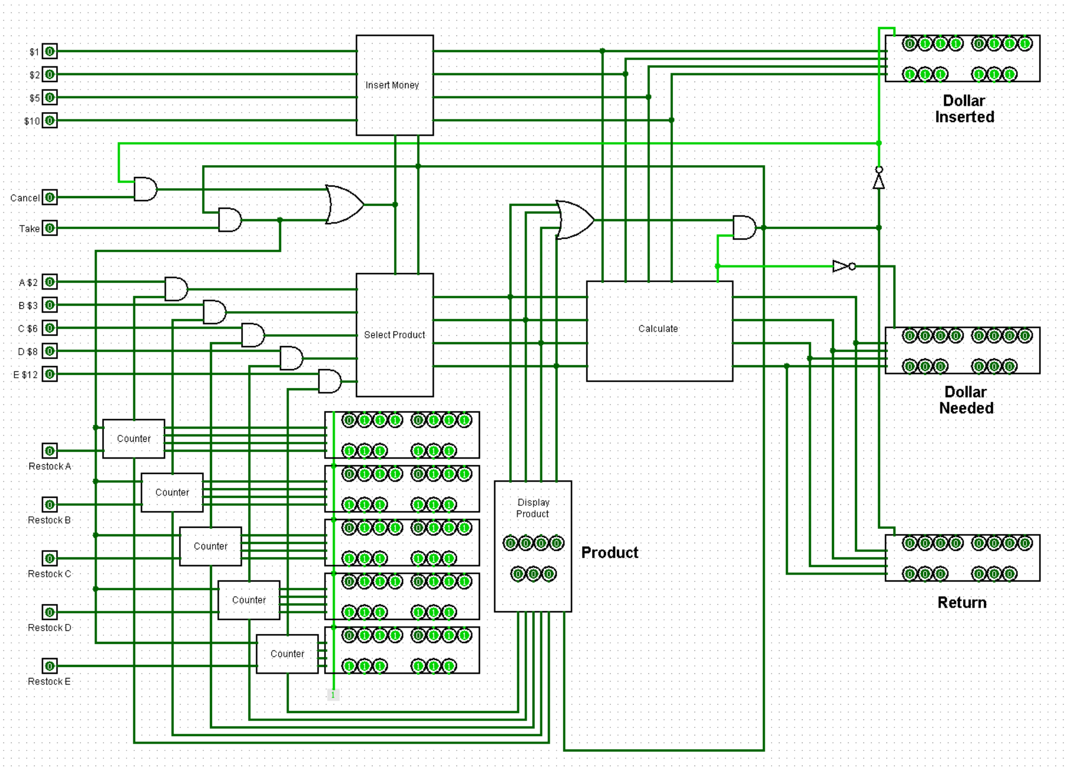


Figure 20: Main Circuit

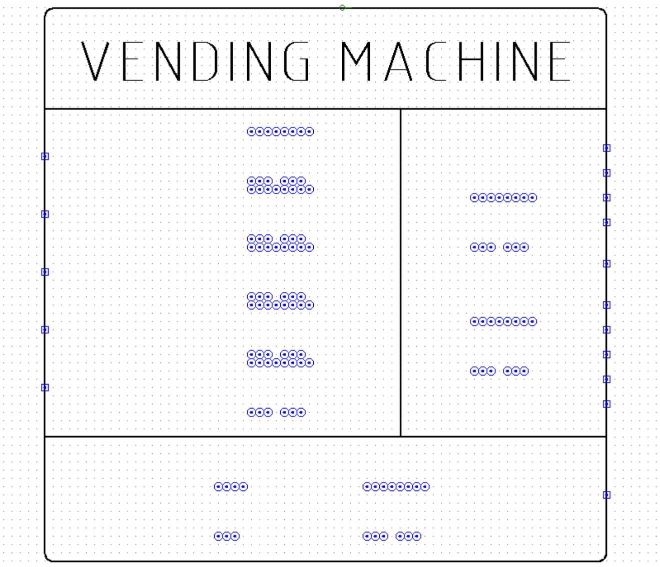


Figure 21: Main Block

2.2.1 Money Insertion and Display

The circuit begins with four inputs representing amount to insert: $1, $2, $5 and $10. These inputs go into the block of Amount Insertion Sub-circuit, which converts them into a 4-bit binary value. This binary value is then sent to a 4-bit binary to 7-segment display block, enabling the representation of the inserted amount on two 7-segment displays.

The display block is only activated when the inversion of the AND gate is 1, ensuring that the inserted amount is not displayed when a product is selected and amount is sufficient for dispensing the product.

2.2.2 Product Selection and Display

For product selection, there are five inputs corresponding to different product prices ($2, $3, $6, $8 and $12). These inputs are fed to separate AND gates along with the corresponding output of the separate counters which indicates the availability of the products, ensuring that the product can only be selected when it is in stock. The outputs of these AND gates each pass through the block of Product Selection Sub-circuit, generating 4-bit binary output, which is fed into the block of Product Display Sub-circuit. This sub-circuit processes the data and controls a 7-segment LED display, indicating the dispensed product.

The display block is only activated when the output of the AND gate is 1, ensuring that the product is displayed only when the product is selected and amount is sufficient for dispensing the product.

2.2.3 Price Comparison and Change Calculation

The inserted money and the selected product price are fed into the block of Calculation Sub-circuit, which first compares the two values using a comparator. If the inserted money is less than the product price, the comparison output is 0. Conversely, if the inserted money is greater than or equal to the product price, the comparison output is 1.

4-bit subtracted output from the block of Calculation Sub-circuit is sent to two different block of 4-bit to 7-segment display sub-circuit. One represents the insufficient amount and activates only when the comparator output's inversion is 1. Another represents the change and activates only when the output of the AND gate is 1, ensuring that the display turns on when the comparator's output is 1 and a product is selected.

2.2.4 Cancellation and Retrieval

A Cancel button allows the user to cancel a transaction before dispensing a product or if the amount is insufficient. This input is connected to an AND gate that ensures cancellation is possible only if a product is not already selected or the amount is insufficient.

A Take button allows the user to receive the dispensed product and change. This input is connected to an AND gate that ensures retrieval is possible only if the product is selected and amount is sufficient.

The outputs of both the Cancel and Take AND gates are combined using an OR gate, which then clears the memories of both the Amount Insertion Block and Product Selection Block.

2.2.5 Disabling Amount Insertion and Product Selection

The AND gate output (which enables product and change display) is also connected to the disable inputs of both the Amount Insertion and Product Selection blocks. This prevents further money insertion or product selection until the product dispensed and change is taken.

2.2.6 Stock Management

Once a product is dispensed, the Product Display Block generates outputs specifying which product was dispensed. These outputs are connected to the enable inputs of five separate counters, ensuring that only the counter for the dispensed product decrements.

Each counter has a preset input, allowing products to be restocked and a clock input, connected to the Take AND gate output, ensuring that the product count decreases only after the user takes the product and change.

Each product counter has a 4-bit binary output representing the remaining stock. These outputs are sent to a 4-bit binary-to-7-segment display block, enabling real-time visibility of stock levels.

2.3 Working

Step 1: User inserts money, which is displayed on 7-segment displays.

Step 2: User selects a product, the Calculation Sub-circuit determines if the money is sufficient or insufficient.

Step 3: If the money is insufficient, insufficient amount is displayed on 7-segment displays after which more money can be inserted or the transaction can be cancelled.

Step 4: If the money is sufficient, the product and change amount are displayed and dispensed.

Step 5: The dispensed product and change are taken by the user which reduces the stock count.

Step 6: The system resets, ready for the next transaction.

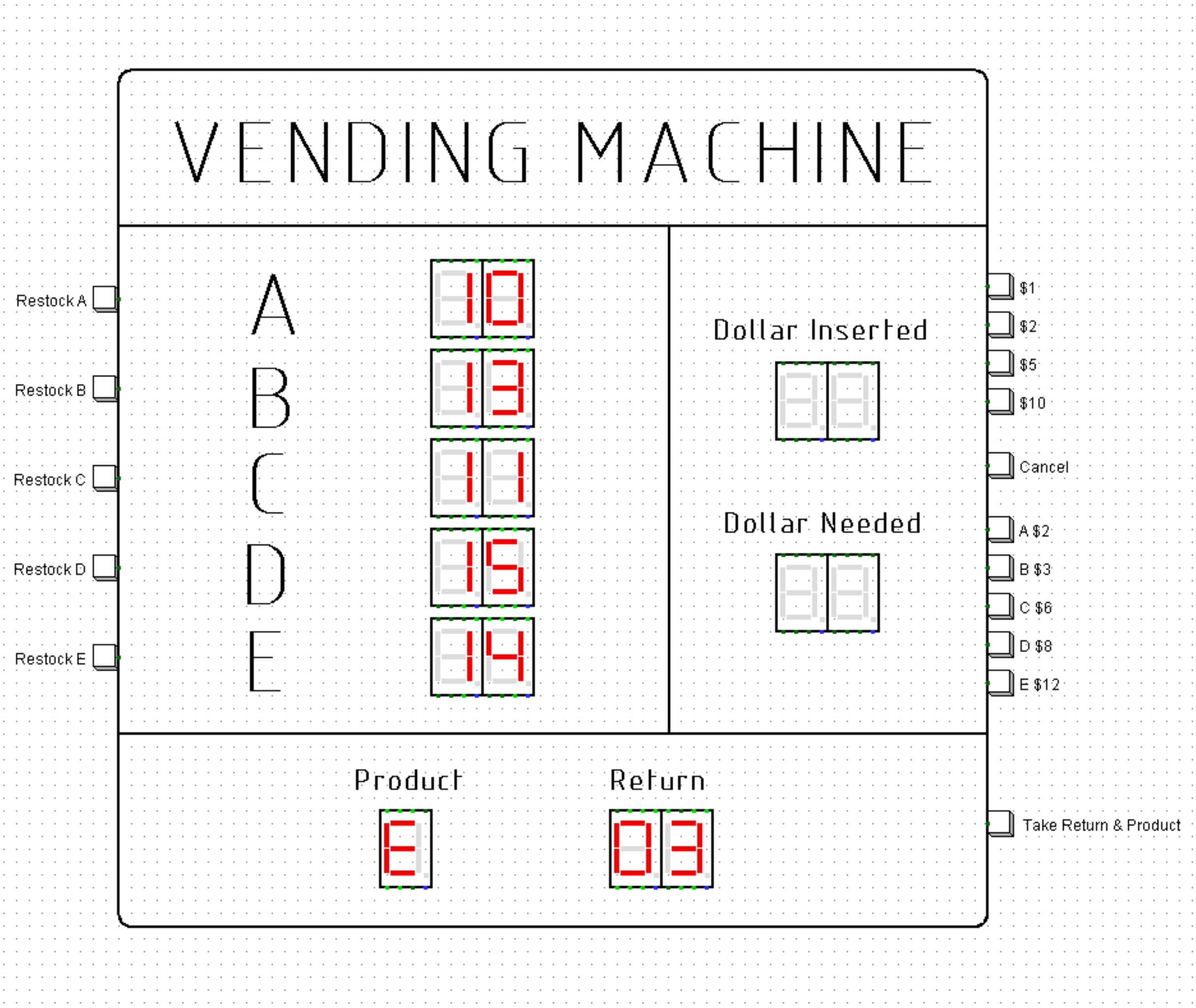


Figure 22: Digital Vending Machine

2.4 Project Link

GitHub Repository: https://github.com/PokharelAditya/Digital-Vending-Machine.git

Chapter 3 Result and Conclusion

3.1 Result

The designed vending machine control system successfully processes transactions by handling amount insertion, product selection and product dispensing. The system ensures that the inserted money is correctly accumulated and displayed using a binary-to-7-segment decoder. The calculation sub-circuit accurately determines whether the inserted amount is sufficient for the selected product and enables the appropriate output, either the insufficient amount or the change display. Additionally, the product display sub-circuit effectively identifies and displays the selected product while ensuring that products are dispensed only when they are available in stock. The counters accurately track the stock levels of each product, allowing restocking when needed. The cancellation mechanism enables users to abort transactions before dispensing. The system also prevents further money insertion or product selection when a product is dispensed until it is taken by the user.

3.2 Conclusion

This vending machine logic circuit effectively integrates digital components like adders, subtractors, comparators, memory blocks, counters and display drivers to implement a functional and efficient transaction system. The logical design ensures smooth operation, preventing issues like product selection when out of stock or dispensing change incorrectly. The use of synchronous and asynchronous logic elements, combined with control logic gates, ensures that all processes, from inserting amount to dispensing product, are properly coordinated. The system's modular design allows easy modifications or enhancements, such as expanding the product range or integrating new payment methods. This project demonstrates the effectiveness of digital logic circuits in automating vending machine operations, ensuring user-friendly interaction.