Candy Vending Machine

EC204 Project

Submitted to Dr. Ratnamala madam and Sumathra madam

V Abhiram, Vedant darji

Roll N0. 221EC260-261

November 15, 2023

Abstract

This project is based on the concept of the **Finite state machine** where the machine interacts with the user inputs and returns the desired output. The *FSM* uses the **flip flops**.

The *fsm* is controlled by an external clock signal which has a particular frequency of working. The clock signal is used for the change of present state to the next state and display of the outputs of the *FSM*. The project can be implemented in the regions where there is a lot of demand for the products.

The main idea of the project is how to process the data which is very large and get the output.

Like the menu of a hotel contains *100 dishes* and each dish price varies and are not the same so we have a database which is a text file and the costumer will give the input which dish he will choose and then we can get the complete bill of the particular costumer.

Our method of approach is by reading the text file in verilog and then generating the output based on the states of the vending machine.

Contents

1 Introduction **2**
1.1 Background 2
1.2 Problem Statement 2
1.3 Objectives 3

2 Literature Review **4**

3 Methodology **6**
3.1 Data Collection 6
3.2 State diagram and algorithms 7

4 Results **10**
4.1 Text files and Output waveform 10
4.2 Processing codes: 14
4.3 Test bench 17

5 Discussion **18**

6 Conclusion **19**
6.1 Summary of Findings 19
6.2 Limitations 20
6.3 Recommendations 20

Acknowledgments **21**
References 21

Chapter 1

Introduction

The Candy Vending Machine Verilog project aims to design and implement a digital vending machine using the Verilog. The vending machine is a digital system that simulates the behaviour of a simple vending machine, allowing users to insert money, make a selection, and receive a product along with any necessary change.

1.1 Background

We have already studied about single product vending machine in class but we are trying to take a level up and make a multiple product or specifically multiple flavoured candy.
The key advantage of vending machines is their convenience, allowing people to access products quickly and easily without the need for traditional retail transactions. They are commonly found in places with high foot traffic, such as airports, malls, schools, and office buildings.

1.2 Problem Statement

Design and implement a candy vending machine that allows users to select and purchase candies. The vending machine should offer a variety of candy options with different prices. Users should be able to make selections, view the total cost of their choices, and complete the purchase by inserting the appropriate amount of money.

The candy vending machine should have the following features:

- **Candy Selection:** Users should be able to view a list of available candies, each with a corresponding code and price.
- **User Interaction:** The vending machine should provide a user-friendly

interface for selecting candies. Users can input their candy choices using the designated codes.

- **Total Cost Display:** The machine should display the total cost of selected candies as users make their choices.
- **Payment:** Users should be able to insert money into the machine to cover the total cost. The machine should provide change if necessary.
- **Error Handling:** Implement proper error handling for scenarios such as insufficient funds, invalid candy codes, or other potential issues.

1.3 Objectives

The objectives of a candy vending machine are to provide a convenient and efficient means for customers to purchase candies while ensuring the proper functioning.

- **User-Friendly Interface:** Design an intuitive and easy-to-use interface that allows customers to easily browse, select, and purchase candies. The interface should be accessible to a diverse range of users, including those with limited technological familiarity.
- **Candy Selection and Display:** Provide a diverse selection of candies with clear and attractive displays. The machine should clearly present the available options, including prices and any relevant information about the candies.
- **Transaction Handling:** Enable users to make selections and complete transactions seamlessly. The machine should accurately calculate the total cost of selected items and handle payments, including the ability to provide change when necessary.

Chapter 2

Literature Review

Digital logic with verilog (Stephen Brown and Zvonko Vranesic) Finite State Machine (FSM) is a mathematical model used to design and describe the behavior of a system with a finite number of states. It's a conceptual machine with the ability to be in exactly one of a finite set of states at any given time. The system can change from one state to another in response to external inputs or events; this transition is determined by a set of rules or conditions known as transitions.

Here are the key components and characteristics of Finite State Machines:

- States: The possible conditions or situations that the system can be in. States represent different modes or phases of the system's behavior.
- Transitions: The rules or conditions that define how the system moves from one state to another. Transitions are triggered by external inputs or events.
- Events or Inputs: External stimuli or triggers that cause the FSM to transition from one state to another. These events drive the behavior of the system.
- Actions: Activities or operations associated with state transitions. Actions may include updating variables, performing calculations, or triggering external processes.
- Initial State: The starting state of the system before any events occur.
- Final State (optional): A state indicating the end of a process or the completion of a task.

Finite State Machines are widely used in various fields, including computer science, control systems, and digital circuit design. They provide a clear and structured way to model and understand the behavior of systems that exhibit distinct operational modes.

There are two main types of Finite State Machines:

- Deterministic Finite State Machine (DFA): In a DFA, each state has a unique transition for every possible input. The next state is determined solely by the current state and the input.
- Nondeterministic Finite State Machine (NFA): NFAs allow multiple transitions for the same input from a single state. The transition is not uniquely determined by the current state and input.

Finite State Machines are used in various applications, such as designing software, creating digital circuits, modeling communication protocols, and describing the behavior of sequential systems. They are a fundamental concept in computer science and engineering.

For reading text files in verilog we need to use the function $readmemb. this functions reads the data from the file and stores it in an array format.The file specified should contain data in a specific format. Each line of the file corresponds to a word (byte, 16-bit word, etc.) in the memory array. The data in each line is typically represented in hexadecimal format.

The $display statement is used to print the contents of the memory for verification purposes. When you run a simulation, the contents of the memory after reading from the file will be displayed.

Chapter 3

Methodology

We first got the algorithm for the candy vending machine. We created a candy.txt a text document which has the candy flavors and then we created the price.txt text file where the Candy prices are stored.

The created text file must be created in the simulation sources for the reading the text file in verilog.

After reading the text file and we get the input from the user about the candy flavor and then the price will get calculated.

The user will be given the option for to select the mode of payment so that the non cash payments can also be done. Even for the card payments we get different text files so that the payment is possible.

The equation we used are:

$$Total = Candy_{\text{price}} \times Quantity$$

$$Change_{\text{generated}} = Money_{\text{given by user}} - Total$$

3.1 Data Collection

Here the data is collected according to the whether the user is able to pay for the total amount otherwise the data will not be collected.
For the data collected we will then process that data to get to know which flavor of candy is sold better in that area and will introduce that related flavors in nearby areas.

3.2 State diagram and algorithms

The algorithm chart is attached below for the candy vending machine.

Here the candy vending machine is having some delay for the each step as we have to get the response from the user.

If there is no response from the user then the candy machine will get rested to the initial state.

The states are given below:

- State A: Here the reset logic takes place
- State B: When the user reaches the vending machine then the machine detects the presence of the user and then display the Candy menu
- State C: Then the user selects the Candy and the quantity of the candy
- State D:Then the user will get to know the total amount
- State E:Then the user selects the mode of payment either cash or Cash less payment.
- State F:If cash payment it generates the change if necessary
- State G:If cashless no generation of the change

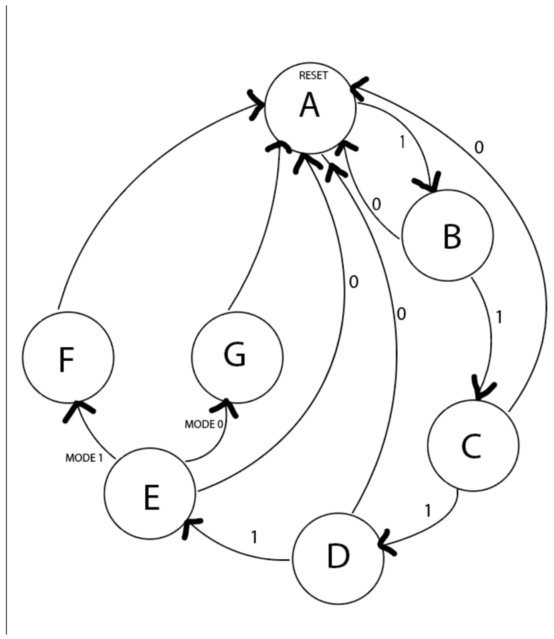


Figure 3.1: state diagram

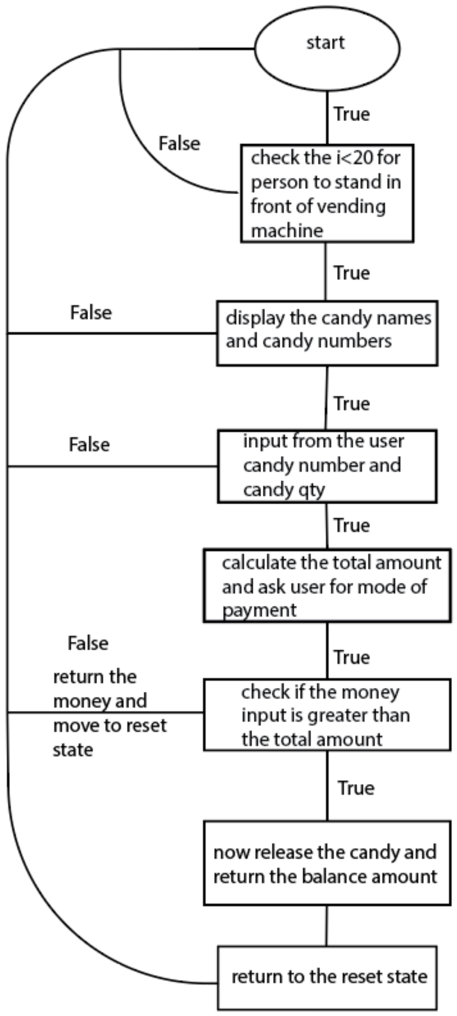

start
True
False
check the i<20 for person to stand in front of vending machine
True
False
display the candy names and candy numbers
True
False
input from the user candy number and candy qty
True
calculate the total amount and ask user for mode of payment
False
return the money and move to reset state
True
check if the money input is greater than the total amount
True
now release the candy and return the balance amount
return to the reset state

Figure 3.2: Algorithm

Chapter 4

Results

4.1 Text files and Output waveform

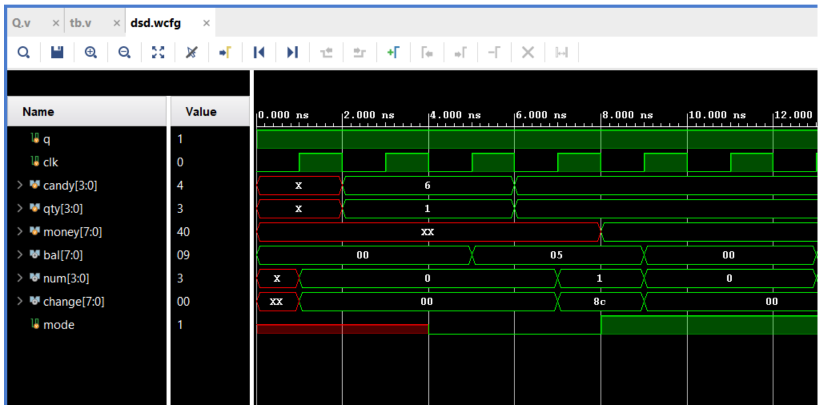
The output wave forms are given as the output will be the num which gives the quantity of the number of Candy dispensed and the change will be generated only when we input the mode of the payment as the cash.

If the user selects the mode of payment as cashless payment then it reads the text file where the bank details are mentioned.

After all the transactions the user will be given the Candy otherwise if the balance is not sufficient it will display the comment as the

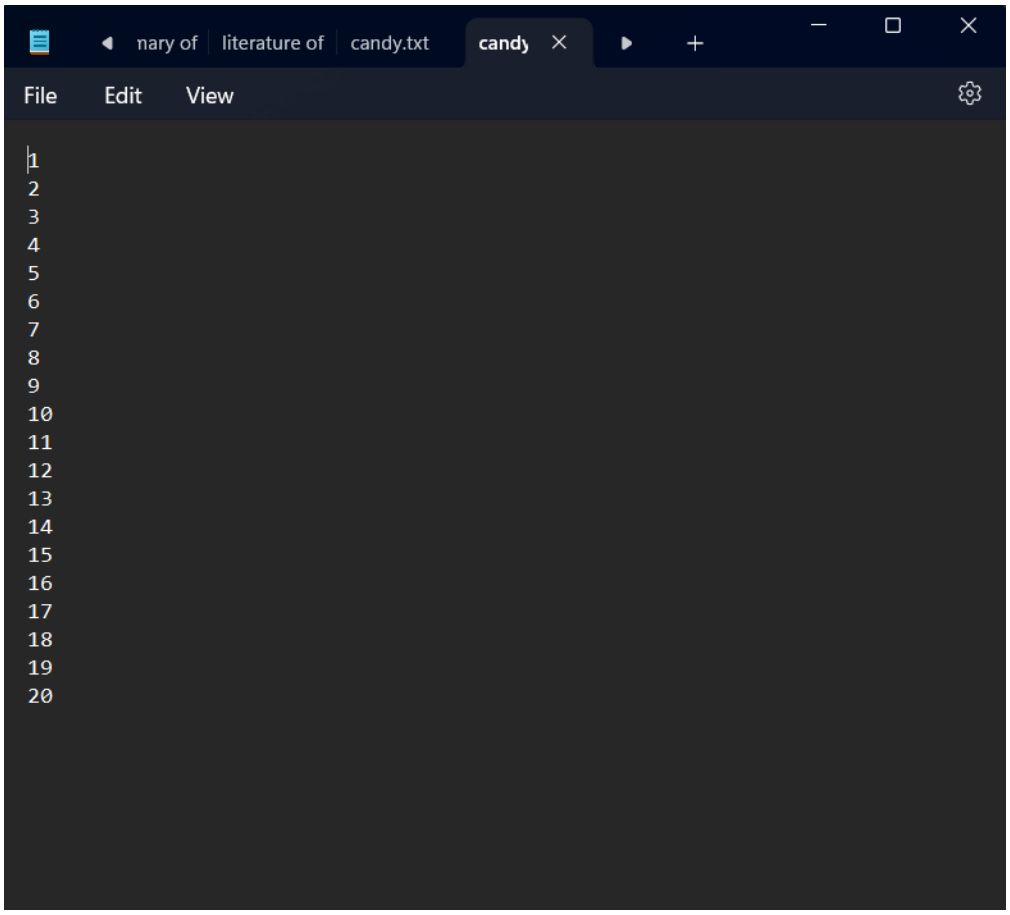
insufficient balance

.



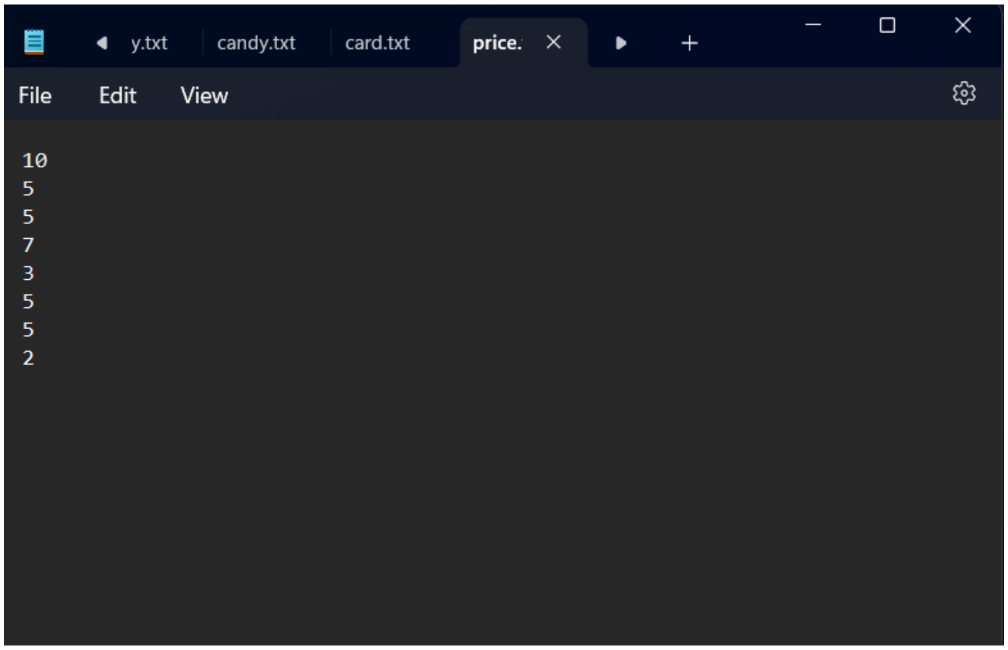
Name	Value
q	1
clk	0
candy[3:0]	4
qty[3:0]	3
money[7:0]	40
bal[7:0]	09
num[3:0]	3
change[7:0]	00
mode	1

Figure 4.1: Output Waveform



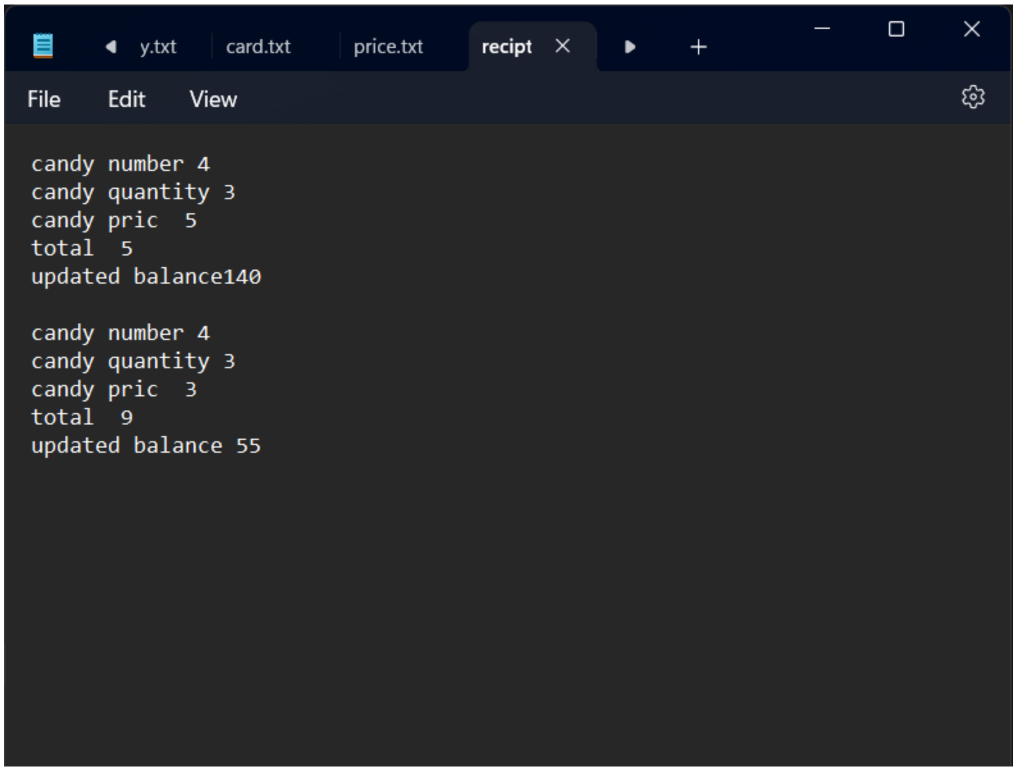
```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

Figure 4.2: candy menu



```
10
5
5
7
3
5
5
2
```

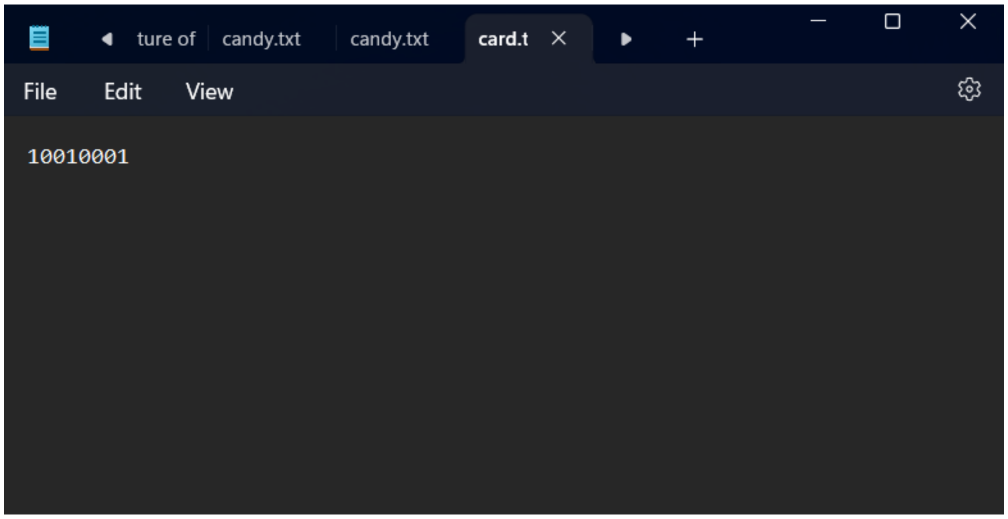
Figure 4.3: Price file



```
candy number 4
candy quantity 3
candy pric  5
total  5
updated balance140

candy number 4
candy quantity 3
candy pric  3
total  9
updated balance 55
```

Figure 4.4: receipt text



```
10010001
```

Figure 4.5: card text file

4.2 Processing codes:

```
module Q(
input wire q,
input wire clk,
input wire [3:0]candy,
input wire [3:0]qty,
input wire [7:0]money,
input wire mode,
output [7:0]bal,
output [3:0]num,
output [7:0]change
);
// outputs
reg [7:0]bal=0;
reg [7:0]num;
reg [7:0]change;
//output ports connection
reg [7:0]bal1=0;
reg [7:0]num1;
reg [7:0]change1;
// parameters
parameter a=1,b=2,c=3,d=4,e=5;
integer ps=a;
integer ns;
// file reading
reg [7:0]buffer[0:7];
reg [7:0]buffer1[0:7];
reg [7:0]buffer2[0:1];
//integers
integer j;
integer i;
// calculating parameters
reg [7:0] selected;
reg [7:0] total=0;
// writing file parameter
integer file;
// for all reading taking place
initial begin
$readmemh(”candy.txt”,buffer);
$readmemh(”price.txt”,buffer1);
$readmemb(”card.txt”,buffer2);
end
// start of the processing
always@(ps or q)begin
case(ps)
```

```
a:if(q)begin
ns=b;
change1=0;
bal1=0;
num1=0;
end
else
ns=a;
b:if(q)begin
ns=c;
for(j=0;j¡7;j=j+1)begin
$display(buffer[j]);
end
end
else begin
if(i¡20)
i=i+1;
else begin
ns=a;
i=0;
end
end
c:if(q)begin
ns=d;
selected=buffer1[candy];
bal1=selected*qty;
$display(selected,bal1);
end
else begin
if(i¡20)
i=i+1;
else begin
ns=a;
i=0;
end
end
d:if(q)begin
if(mode)begin
total=money;
if(total¿=bal)begin
change1=total-bal;
num=qty;
ns=a;
end
else begin
$display(”insufficient balance”);
```

```
change1=total;
ns=a;
end
end
else begin
total=buffer2[0];
$display(total);
if(total¿=bal)begin
change1=total-bal;
num1=qty;
ns=a;
end
else begin
$display(”insufficient balance”);
change1=total;
ns=a;
end
end
end
else begin
if(i¡20)
i=i+1;
else begin
ns=a;
i=0;
end
end
endcase
end
// changing of the state with the clock
always@(posedge clk) begin
//change of the state
// writing into the file for the recipt and the record
num¡=num1;
change¡=change1;
bal¡=bal1;
if(ps==d)begin
file=$fopen(”recipt.txt”,”a”);
$fwrite(file,”candy number
$fwrite(file,”candy quantity
$fwrite(file,”candy pric
$fwrite(file,”total
$fwrite(file,”updated balance
$fwrite(file,”\n”);
$fclose(file);
$display(change,total,bal);
```

```
end
ps¡=ns;
end
// writing into the file initially as we move to the next costumer we will append
the text file and get the information about the data which we receive
endmodule
```

4.3 Test bench

```
module tb;
reg q,clk;
reg [3:0]candy;
reg [3:0]qty;
reg [7:0]money;
wire [7:0]bal;
wire [3:0]num;
wire [7:0]change;
reg mode;
Q question(q,clk,candy,qty,money,mode,bal,num,change);
initial begin
clk=1'b0;
forever #1 clk= clk;
end
initial #1000 $finish;
initial begin
q=1'b1;
#2 candy=4'b0110;qty=4'b0001;
#2 mode=1'b0;
#2 candy=4'b0100;qty=4'b0011;
#2 mode=1'b1;money=8'b01000000;
#15 $finish;
end
endmodule
```

Chapter 5

Discussion

Our objective for the project was to make a vending machine which can be used for multiple products and keeping a track on which product is maximum sold accordingly telling the seller to boost the production os the particular product.

Also in our project we can change the products or its price by just altering the data in the text file which make the vending machine multi-purpose.

we wanted our project to be user friendly for which it displays the contents in the machine with price and a code so that the user can type the code and pay to get the desired product.we wanted our project to accepts multiple modes of payment such as cash ,card ,online payment.

Now Comparing our project with the present day vending machines all over the world it performs all the usual task that a normal vending machine does on top of that is also tracks the sales of the products and makes the owners job easy to make profit.

The only difficult part will be making the real life model of this vending machine which we would like to work on in future.

Chapter 6

Conclusion

The Vending Machine Verilog project successfully demonstrates the design and implementation of a simple digital vending machine using Verilog. The finite state machine architecture provides a clear structure for handling user interactions and managing the vending process. The project lays the foundation for further enhancements and serves as an educational tool for learning digital design concepts.

6.1 Summary of Findings

1) we learnt FSM and working of vending machine Finite State Machines are mathematical models used to design and represent systems with a finite number of states and transitions between these states. They consist of a set of states, a set of transitions between these states, and an initial state. FSMs are widely used in various fields, including computer science, engineering, and automation.

2) using python to Read a text file in verilog applying the previous knowledge of files in python with read and write functions and connecting it with verilog by using OS we are using the function *$readmemh* which reads the hexadecimal values written inside the text file. the text file must be created inside the verilog file which contains the simulation sources. if the file is not created the reading of the file is not possible. apart from reading hexadecimal values we can also read binary values using the function *$readmemb* .

3) processing a text file in verilog when the file is open the file variable is integer type which creates the stream of data which can be manipulated for writing,reading,appending, writing and reading together, reading and appending together. If we want to display the result we use the function *$display* which displays the result in the TLC console.

4) writing the output of products in verilog If the file is not created and if we want to write in the file then the file will get created automatically in the stimulation sources of the verilog file in which the data is written. Once the file is created and written we need to close the file so that the interference of the file during processing is avoided. We close the file by the command *$fclose*.

5) Usage of Arduino for lcd display Arduino is an open-source electronics platform based on easy-to-use hardware and software. It consists of a physical programmable circuit board (often referred to as a micro controller) and a development environment for writing and uploading code to the board. Arduino boards are widely used for various projects, including robotics, home automation, and interactive art installations.
We use *arduino lcd display* for displaying the candy flavours names and its corresponding prices and candy number.

6) Bill generation: After processing the data from the user we write into the text file *recipt.txt* and then print it so that the customer can get to know about its expenditure and we will also get a know which flavour candy is mostly consumed.

7)Data analysis: After getting the data from each vending machine we can get to know which flavour is liked the most in a particular region according to that we can increase the production of that candy flavour. For data analysis we use python module *Matplotlib*.

6.2 Limitations

1) The input text file must be created and written with some products if not the there will be no output or reading of products.
2) Finite Storage Capacity
3) User Input Handling

6.3 Recommendations

It would be great if the machine to record the number of sale per day keeping a track on how much stock of candy is left in the storage

Acknowledgments

I would like to express my sincere gratitude to...

- Our advisor,Dr. Ratnamala madam and Sumathra madam, for teaching the concepts related to flip flops and the **fsm** topics.

References

- Books used:
  1) Digital logic with verilog
  2) Digital circuits and logic design
  3) Fundamentals of digital logic
- Videos referred:
  1) Verilog tutorials by component byte
  2) all about circuits
  3) NESO achedemy