vending machine

Parsa HaghighatGoo

July 3, 2023

Shiraz University stdnum , 40030644

1 Introduction

A vending machine is a self-service device that dispenses various products when money or a specific form of payment is inserted. These machines are commonly found in public places such as airports, train stations, offices, schools, and shopping centers. Vending machines can offer a wide range of items, including snacks, beverages, cigarettes, lottery tickets, toiletries, and even electronics. They typically operate through a combination of mechanical mechanisms, electronics, and sensors to detect and process payments, which can include cash, coins, credit cards, or mobile payments. Once the payment is made, the vending machine releases the selected item to the user. In this project, I designed a vending machine with the help of Verilog language, which can have 8 different products, which I will explain more about in the following.

2 modules:

In this project, there is a main module, which contains several other modules, each of which gives different access to the owner of the vending machine and the customer.

Modules docs :

- ownerpass
- satisfaction
- gift
- Main
- buying phase
- retrieve the Money
- read stuff
- modify stuff
- checkin
- keyboard
- charging the Supply
- sevenSegment Display
- support phase
- tb
- gift with ai

2.1 ownerpassword

in this doc the owner password for changing detail of product is written.

2.2 satisfaction

in this doc we explain that :
if the customer is satisfied ,he can enter a number between 1 and 10 for rating to the machine.

2.3 gift

in this doc we explain that:
if customer amount of product is bigger than two then we give him one more of that product as a gift

2.4 gift with ai

this module give us an idea for creating a simple ai for grouping the products :
this idea is when we want to give customer a product as a gift . when customer select a product from a group we give him another product of group as a gift.

- for example : if customer select 000 product we give him 111 product as a gift.

there is 4 groups of product with similar specifications or related specifications.

groups :

- 1. 000 111
- 2. 001 110
- 3. 010 101
- 4. 011 100

it works as an simple ai which connect product with similar specifications or related specifications.

2.5 main

The ‘main‘ module is the top-level module that handles the overall functionality of the system. It has several input and output ports defined as follows:

Input Ports: - ‘canclebutton‘: A button used for canceling operations. - ‘supportbutton‘: A button used for providing support. - ‘ownerpass[5:0]‘: A six-bit input representing the owner’s password. - ‘typing[12:0]‘: A 13-bit input representing the user’s typing. - ‘clock‘: The clock signal. - ‘tmps‘: A temporary signal.

Inout Ports: - ‘statisfy‘: A bidirectional signal used for satisfying a condition. - ‘usermode[1:0]‘: A two-bit bidirectional signal representing the user mode. - ‘stuffmode[2:0]‘: A three-bit bidirectional signal representing the stuff mode. - ‘sizein[3:0]‘: A four-bit bidirectional signal representing size input. - ‘cmoney[3:0]‘: A four-bit bidirectional signal representing cash money. - ‘pmoney[3:0]‘: A four-bit bidirectional signal representing product money. - ‘addsize[3:0]‘: A four-bit bidirectional signal representing additional size. - ‘withdrawmoney[3:0]‘: A four-bit bidirectional signal representing the amount of money to be withdrawn.

Input Ports (Continued): - ‘printnumber[3:0]‘: A four-bit input representing the number to be printed.

Output Ports: - ‘redlight‘: A one-bit output indicating the status of a red light. - ‘sevensegment[6:0]‘: A seven-bit output used for displaying on a seven-segment display. - ‘product[10:0]‘: An eleven-bit output representing the product.

Within the module, there is a reset button (‘if (canclebutton != 1) begin‘) that controls the behavior of the system when the cancel button is pressed. If the cancel button is not pressed, the following operations are performed:

1. Keyboard Module: The module ‘keyboard‘ is instantiated to handle user input from the keyboard. It connects the clock signal and various input/output signals related to size, money, etc.

2. User Mode Handling: An always block ‘always @(posedge clock)‘ is used to handle the user mode based on the value of ‘usermode‘.

- If ‘usermode‘ is ‘2’b00‘, it enters the BuyingPhase module (‘BuyingPhase buy‘) for processing buying-related operations. The relevant signals such as ‘stuffmode‘, ‘cmoney‘, ‘csize‘, ‘pmoney‘, ‘product‘, and ‘redlight‘ are connected to this module. Additionally, if the ‘supportbutton‘ is pressed, it instantiates the ‘supportbutton‘ module (‘supportbutton s()‘) for providing support. - If ‘usermode‘ is ‘2’b01‘ and the ‘ownerpass‘ matches ‘6’b001101‘, it enters the charging the supply module (‘chargingthesupply charge‘). The relevant signals such as ‘stuffmode‘, ‘addsize‘, ‘clock‘, ‘redlight‘, and ‘product‘ are connected to this module. Otherwise, the ‘redlight‘ is set to 1 indicating an error. - If ‘usermode‘

is ‘2’b10‘ and the ‘ownerpass‘ matches ‘6’b001101‘, it enters the retrive money module (‘retrivethe-money retrive‘). The relevant signals such as ‘clock‘, ‘withdrawmoney‘, ‘pmoney‘, ‘sevensegment‘, and ‘redlight‘ are connected to this module. Otherwise, the ‘redlight‘ is set to 1 indicating an error. - If ‘usermode‘ is ‘2’b11‘, it enters the seven-segment display module (‘sevensegmentdisplay display‘). The relevant signals such as ‘clock‘, ‘printnumber‘, and ‘sevensegment‘ are connected to this module. - If none of the above cases match, the ‘redlight‘ is set to 1 indicating an error.

That’s a high-level overview of the main module and its functionality.

2.6 buying phase

In this module, the customer can give the number or code of the desired product to the machine, and after selecting the product, he can enter the quantity he needs of that product, and then enter the amount and pay. Now, if there is a problem in any of the steps The red light will be shown on the screen.

2.7 retrieve the Money

In this module, the owner of the vending machine can take the amount of money in the machine to the desired amount.

2.8 charging the Supply

In this module, the owner of the vending machine gets access to the ability to charge the needed product in the desired amount and it is necessary to mention that no more than 15 of each product can be available.

2.9 read and modify stuff

In these two modules, we read and save the desired entries from a text file

2.10 checkin

The ‘checkin‘ module is designed to check if a task went wrong or not based on the value of the input signal ‘check‘. It provides an output indication of the task status and also interfaces with a seven-segment display.

Here’s the step-by-step explanation of what this module does:

1. The module takes a 2-bit input signal called ‘check‘, which likely represents the status of the task being checked.

2. Inside the ‘always‘ block, there is an ‘if-else‘ statement that checks the value of ‘check‘.

3. If the value of ‘check‘ is ‘2’b00‘ (binary 00), it means that the task did not encounter any error. In this case, the ‘result‘ output is set to ‘4’b1111‘ (binary 1111), indicating a successful completion of the task.

4. If the value of ‘check‘ is anything other than ‘2’b00‘, it suggests that the task encountered an error. In this case, the ‘result‘ output is set to ‘4’b0000‘ (binary 0000), indicating a failure in the task.

5. Additionally, when the task encounters an error (i.e., when ‘check‘ is not ‘2’b00‘), the ‘redlight‘ output is set to 1. This can be used as a signal to indicate the error by turning on a red light.

6. There is a submodule called ‘sevensegment‘ instantiated as ‘ss‘. It appears to handle the display of the ‘result‘ value on a seven-segment display. However, there seems to be a typo in the code where the output ‘sevensegment‘ is used again as an input for this submodule instantiation. This line should likely use a different signal name representing the output of the seven-segment display module.

2.11 keyboard

The code you provided defines a Verilog module called ‘keyboard‘. This module appears to be modeling a keyboard input system with various modes and outputs for displaying values on a seven-segment display.

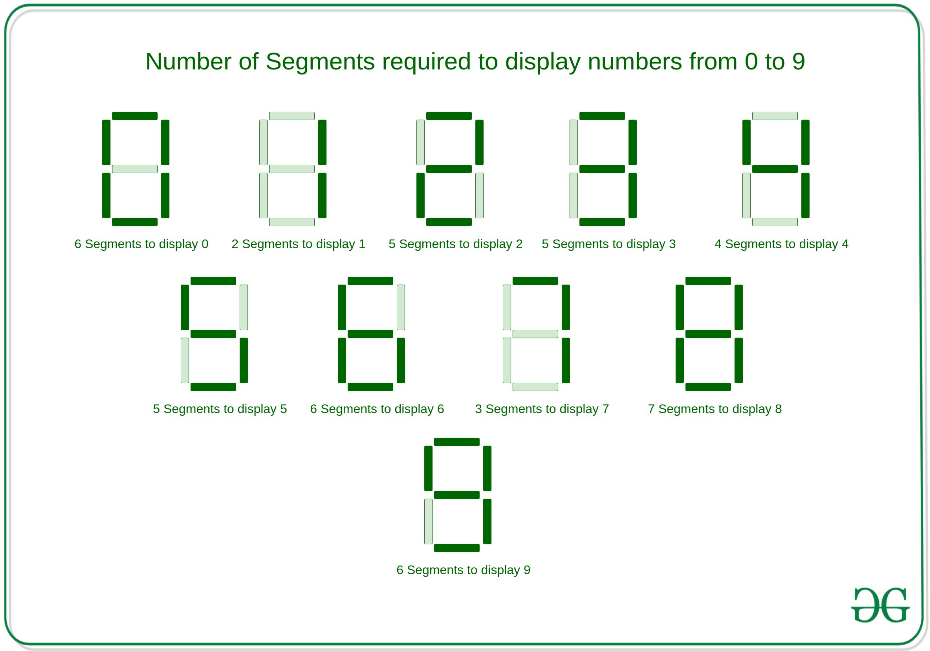

Figure 1: A picture of sevensegment's working way for 2.13 section

In summary, the 'keyboard' module takes input signals and assigns them to various output signals representing different modes and values. It also instantiates multiple 'sevensegmentdisplay' modules to display these values on a seven-segment display.

2.12 tb

this is a test case for main module

2.13 support phase

The code you provided defines a Verilog module called 'supportphase'. This module does not have any inputs or outputs. It contains an initial block that displays a string message. the message is about how customer can contact with owner and give him an example email.

2.14 sevensegment display

A seven-segment display is a form of electronic display device used to represent decimal numbers and some alphabets. It consists of seven segments that can be arranged in different configurations to display different characters. Each segment is a light-emitting diode (LED) or a light-emitting diode array (LEDA) that can be turned on or off to form different patterns. The seven segments in a typical display are arranged in a rectangular pattern, with each segment labeled A, B,C, D, E, F, and G, as shown below.
To display a particular digit or character on a sevensegment display, the appropriate segments are turned on or off using electrical signals. For example, to display the digit 0, segments A, B, C, D, E, and F would be turned on, while segment G would be turned off. To display the digit 1, only segments B and C would be turned on.

2.15 Good luck!

We hope you find this doc useful.