**CENG 3430 : Rapid Prototyping of Digital Systems**

**Project Topic: Vending Machine**

Shivangi Singh 1155075848

Ajay Singh 1155083332

# 

# 

# 

# 

# 

# 

# 

# Abstract

# 

*A vending machine is a machine which dispenses items such as snacks and beverages after the customer inserts money into it. This report describes the modeling of a Finite State Vending Machine implemented on the Xilinx ZedBoard. Apart from the usual features, the Vending Machine also has a cancel order option and makes use of the LED’s and HEX 7 segment display to output the money change and show money input respectively.*

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Section** | **Title** | **Page Number** |
| 1 | Introduction | 3 |
| 2 | Theory and Design | 3 |
| 2.1 | Design and Specifications of Vending Machine | 3 |
| 2.2 | FSM | 4 |
| 2.3 | Design of FSM | 5 |
| 2.4 | Generating a clock for the digit selection | 7 |
| 3 | Implementation and Experimental Result | 8 |
| 4 | Discussion | 11 |
| 4.1 | Expectations | 11 |
| 4.2 | Difficulties | 11 |
| 4.3 | Further improvement and possibilities | 11 |
| 5 | Conclusion | 12 |
| 6 | References | 12 |

1. **Introduction**

In recent years, there has been a fast growth of FPGA (Field Programmable Gate Array) devices. Many types of electronic devices that fulfill the need of the society have been programmed using FPGA. VHDL (Very High Speed Integrated Circuit Hardware Description Language) is one of the most common hardware descriptive language and we have made use of it in this vending machine project.

Vending machines are coin operated machines that dispense items based on the user selection. These machines are quite common today due to how convenient they are. They are an important outlet for automatic retailing. In this project, as there was a hardware limitation of the type of coin input, the input has been simulated using buttons. Inputs of 1 HKD, 2 HKD and 5 HKD are allowed and output items of 5 HKD and 6 HKD are available. The user can select any item and the exceeding amount, with the item, will be dispensed. The user also has the option to cancel his/her request if they have not chosen an item.

**2. Theory and Design**

**2.1 Design and Specifications of Vending Machine**

Products sold:

1. Candy - HKD5 (Collect\_in = “‘01”)
2. Chips - HKD6 (Collect\_in = “10”)

Coins Accepted:

1. HKD1
2. HKD2
3. HKD5

Input signals:

1. Reset : bit

* User presses the appropriate Reset button on the ZedBoard

1. Clk : STD\_LOGIC
2. One\_in : STD\_LOGIC

* Insert One dollar coin; User presses the appropriate One\_in button on the ZedBoard

1. Two\_in : STD\_LOGIC

* Insert Two dollar coin; User presses the appropriate Two\_in button on the ZedBoard

1. Five\_in : STD\_LOGIC

* Insert Five dollar coin; User presses the appropriate Five\_in button on the ZedBoard

1. Collect\_in : STD\_LOGIC\_VECTOR(1 downto 0)

* User inputs “01” or “10” on the ZedBoard Switch depending on whether he wants candy or chips

1. Cancel : STD\_LOGIC

* User presses the appropriate Cancel button on the ZedBoard

Output signals:

1. One\_out : STD\_LOGIC

* Output 1 One dollar coin; LED 0 blinks

1. Two\_out : STD\_LOGIC

* Output 1 Two dollar coin; LED1 blinks

1. Five\_out

* Output 1 Five dollar coin; LED 2 blinks

1. One\_out2 : STD\_LOGIC

* Output 2 One dollar coins; LED 3 blinks

1. Two\_out2 : STD\_LOGIC

* Output 2 Two dollar coins; LED 4 blinks

1. Five\_out2 : STD\_LOGIC

* Output 2 Five dollar coins; LED 5 blinks

1. Dispense : STD\_LOGIC\_VECTOR(1 downto 0)

* To show that the product has been dispensed, “01” if candy and “10” if chips; LED 6 and LED 7 blinks

1. Ssd : STD\_LOGIC\_VECTOR(6 downto 0)

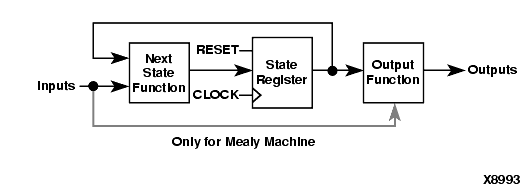
* To display the money inputted into the Vending Machine on the HEX 7 Segment Display

Steps:

1. The customer selects the product they want, by inputting “01” or “10” on the Switch.
2. Then they use the buttons to input coins.
3. With each button press, the input is shown on the HEX 7 Segment Display.
4. The appropriate LEDs will blink when
   1. A product has been dispensed, and
   2. There is change to be outputted
5. While inputting coins using the buttons, if the customer presses the Cancel button then all the money inputted will be outputted via the blinking LEDs.
6. The customer can press the Reset button anytime to start the transaction over.

**2.2 FSM**

FSM stands for Finite state machine. It is a model of computation used to design sequential logic circuits. It is an abstract model of machine that can be in one of the finite number of states at a time. The state at any given time is called the current state. A particular FSM is given by a set of states and the triggering condition for each transition. There are two types of machines, Mealy Machine and Moore Machine. We implemented our Vending Machine using the Mealy Machine model in which the output depends on the present state as well as the input.



**2.3 Design of FSM**

States: s0, s1, s2, s3, s4, s5, s6, s7, s8, s9, s10

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cur\_state** | **one\_in** | **two\_in** | **five\_in** | **Next\_state** | **Collect\_in** | **one\_out** | **two\_out** | **five\_out** | **one\_out2** | **two\_out2** | **five\_out2** | **Cancel** |
| s0 | 0 | 0 | 0 | s0 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s1 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s2 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s5 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| s1 | 0 | 0 | 0 | s1 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s2 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s3 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s6 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s1 | 00 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| s2 | 0 | 0 | 0 | s2 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s3 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s4 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s7 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s2 | 00 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| s3 | 0 | 0 | 0 | s3 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s4 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s5 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s7 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s3 | 00 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| s4 | 0 | 0 | 0 | s4 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s5 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s6 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s9 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s4 | 00 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| s5 | 0 | 0 | 0 | s5 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s6 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s7 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s10 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 1 | 0 | 0 | s6 | 01 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 1 | 0 | s7 | 01 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 1 | s10 | 01 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s5 | 00 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| s6 | 0 | 0 | 0 | s6 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 01 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 00 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| s7 | 0 | 0 | 0 | s7 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 01 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 10 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 00 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| s8 | 0 | 0 | 0 | s8 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 01 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 10 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 00 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| s9 | 0 | 0 | 0 | s9 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 01 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 10 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 00 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| s10 | 0 | 0 | 0 | s10 | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 01 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 10 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
|  | 0 | 0 | 0 | s0 | 00 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**2.4 Generating a clock for the digit selection**

*if rising\_edge(clk) then*

*if (count=49999) then*

*ms\_pulse <= not ms\_pulse;*

*count<=0;*

*else*

*count <= count + 1;*

*end if;*

*end if;*

This will give us a wave (ms\_pulse) of period = one millisecond. We used this for switching between the first and the second digit by turning on and off the ssdcat signal:

*when s1 =>*

*if ms\_pulse = '1' then*

*digit <= “0000”; -- Display MSB at left 7-segment*

*else*

*digit <= “0001”; -- Display LSB at right 7-segment*

*end if;*

**3. Implementation and Experimental Result**

The code for this project has been implemented in 4 sections:-

* FSM state declaration and transition
* Clock division
* Port Mapping
* LED display

In total 11 states are used :

*type state\_type is (  
 S0,  
 S1,  
 S2,  
 S3,  
 S4,  
 S5,  
 S6,  
 S7,  
 S8,  
 S9,  
 S10  
);*

As the highest item price is 6 HKD, the maximum money input is 10 as a 5 HKD coin is accepted after 5 HKD has already been input. The state transition occurs depending on the money input.

For example:-

*when S2 =>*

*if ms\_pulse='1' then*

*digit <= "0000"; -- Display MSB at left 7-segment*

*else*

*digit <= "0010"; -- Display LSB at right 7-segment*

*end if;*

*if one\_in='1' then*

*next\_state <= S3;*

*elsif two\_in='1' then*

*next\_state <= S4;*

*elsif five\_in ='1' then*

*next\_state <= S7;*

*elsif cancel ='1' then*

*two\_out <= '1';*

*next\_state <= S0;*

*else*

*next\_state <= S2;*

*end if;*

A problem faced in this project was state transition. As button inputs are used for this project, if an input button is held for too long the input might be taken into account more than once. This causes the state to transition multiple times. To solve this problem a slower clock was used for the state transition. The board clock has been divided using *“counter = 4999999”.*

The port mapping for this design:-

BTNC:- cancel

BTND:- one\_in

BTNL:- two\_in

BTNR:- five\_in

BTNU:- reset

SW0:- collect\_in[0]

SW1:- collect\_in[1]

LD0:- one\_out

LD1:- two\_out

LD2:- five\_out

LD3:- one\_out2 // two 1 HKD out

LD4:- two\_out2

LD5:- five\_out2

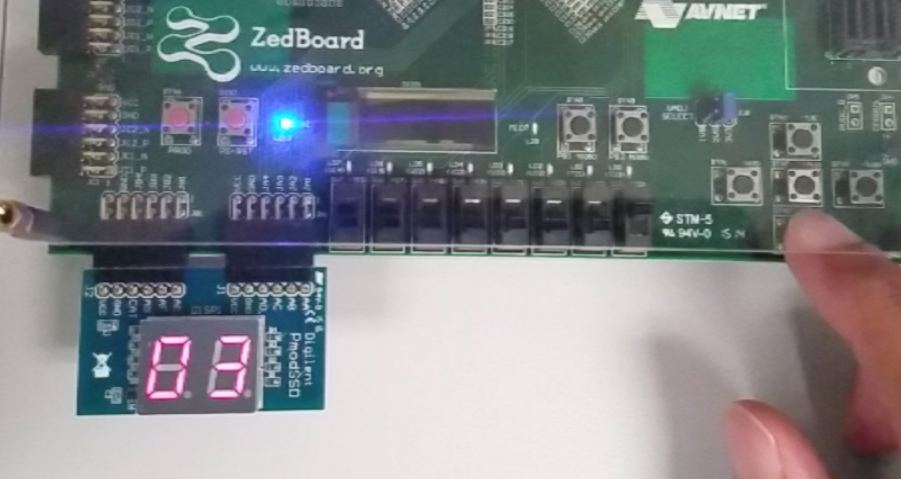
LD6:- dispense[0]

LD7:- dispense[1]

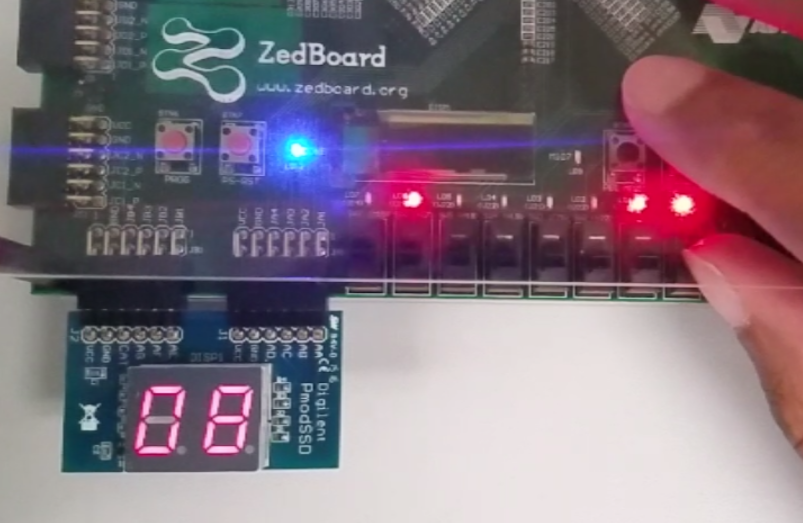
Finally for testing and user friendly purposes, an led display is used to show the amount of input money. The display is hardcoded for each state thus, when a particular state is reached it is displayed on the board. As stated previously, an ms\_pulse counter is used to switch between the 2 led displays on board.

Following all these steps yielded experimental results that match with expected results:-

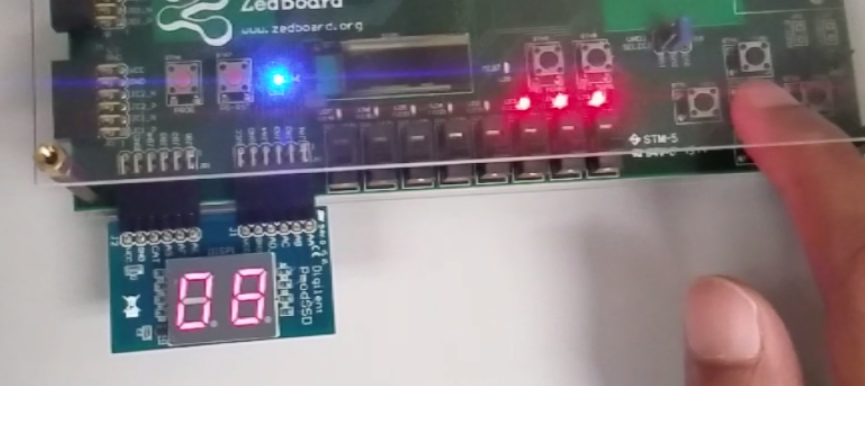
Test Case 1:- Wait state where nothing happens if the amount(5 HKD for collect\_in = 01) is not reached:-



Test Case 2:- Dispensing item and correct change when input amount exceeds the price of the item. In this case, price of item is 5HKD and amount input is 8 HKD. The change output should be a total of 3 HKD:-



Test Case 3:- When an item is not selected and money is input, cancel should output the money input. In this case 8 HKD is used:-



**4. Discussion**

**4.1 Expectations**

We were successful in meeting our aims and objectives. Our Vending Machine fulfills all the necessary requirements, has an additional cancel transaction option and is also very easy to use. The HEX 7 segment display makes it very user-friendly. We initially planned to have a coin collector for our project but we needed a coin collector which would collect, HKD1, HKD2, AND HKD5 coins instead of just HKD1. So we decided to make one of our own but due to time constraints, we couldn’t.

**4.2 Difficulties**

* Designing the FSM took a long time and we had to change the specifications of our design repeatedly to make sure that the FSM does not get too complicated.
* As the clock of the FPGA board is quite fast, state transition was occurring too fast causing one coin input to be taken as multiple coin inputs. This problem was solved by using a clock divider for state transition.
* Because of time constraints, we weren’t able to make our own coin collector.

**4.3 Further improvement and possibilities**

* Make a coin collector, or buy one that accepts a greater variety of coins.
* Increase the range of products and prices.

**5. Conclusion**

The overall design and logic used produced correct results and was an evident success. The problems that were faced made us reflect on the design of the system many times. We learned how as normal people we assume how simple such a project sounds, but when working on it we understood that in reality the vending machine is quite complex. Staring from the state explosion problem when including too many items or too many input variables to the timing issue of the system. We were able to understand and solve these problems, reaching a middle ground and simulate a vending machine by not making it too complex or too simple.

**6. References**

1. [www.csit-sun.pub.ro](http://www.csit-sun.pub.ro/courses/Masterat/Xilinx%20Synthesis%20Technology/toolbox.xilinx.com/docsan/xilinx4/data/docs/xst/hdlcode15.html)