Lab 2: Vending Machine Lab

EE312 Computer Architecture
TA: 조해윤 / Haeyoon Cho (EMAIL: haeyoon.cho@kaist.ac.kr)
Assigned date: 2019 / 09 / 17
Due date: 2019 / 09 / 26 13:00 (before class)

1. Overview

Lab 2 teaches you more advanced concepts and skills of Verilog language. You have learned the basics of Verilog language in *Lab 1*, now it's time to improve it. In *Lab 2*, you are required to implement a vending machine. After you finish *Lab 2* by yourself, you will have a good understanding of register-transfer level, synchronous circuit, and finite state machine which you have learned from the lectures. Have fun!

2. Backgrounds

Register-Transfer Level (RTL)

Register-Transfer Level (RTL) is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (i.e., data) between hardware registers, and the logical operations performed on those signals. The RTL abstraction is used in Hardware Description Languages (HDLs) like Verilog and VHDL to create high-level representations of a circuit, from which lower-level representations and ultimately actual wiring can be derived. [1]

Finite State Machine (FSM)

A Finite State Machine (FSM) is a machine that can be in exactly one of a finite number of states at any given time. The FSM can transit from one state to another state in response to external inputs; the *transition* between states is predefined in the FSM. An FSM is defined with a list of state, conditions for each transition, and initial state. FSMs are of two types of FSM, which are the Moore machine and Mealy machine. A Moore machine is an FSM whose outputs are determined only by its current state, as shown in Fig. 1. This is in contrast to a Mealy machine whose outputs are determined by both its current state and inputs, as shown in Fig. 2.

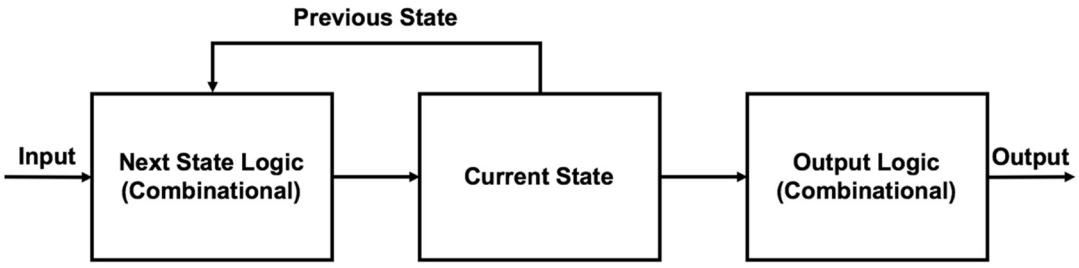

Figure 1. The Moore Machine

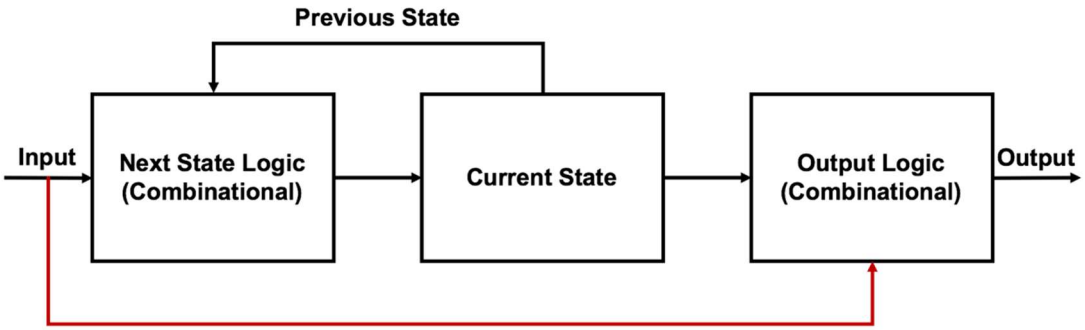

Figure 2. The Mealy Machine

An FSM in an RTL design consists of three parts:

1. A combinational logic that calculates the next state
2. A sequential logic that saves the current state
3. A combinational logic that calculates the output

In *Lab 2*, you are required to implement all three parts of a vending machine by yourself.

3. Files

You are given three files:

1. *vending_machine.v*: The file where you implement the vending machine module

2. *vending_machine_def.v*: The file where constants needed to implement the vending machine are defined

3. vending_machine_TB.v: The file which you can test and grade your module with

4. Vending Machine Lab

In *Lab 2*, you are required to implement a vending machine, which is an example of FSMs. There might be several corner cases while you implement the vending machine, so you have to make sure that your implementation of the vending machine functions properly for all use cases.

You are given a template of a vending machine in *vending_machine.v*. The template defines the interface of the vending machine, as shown in Table 1.

Input		
Signal	**Description**	**# of bits**
i_input_coin	Insert a coin	1 for each coin
i_select_item	Select an item	1 for each item
i_return_trigger	Return	1
clk	Clock	1
reset_n	Reset	1
Output		
o_output_item	Get an item	1 for each item
o_available_item	Available items	1 for each item
o_return_coin	Return coins	1 for each item

Table 1. Interface of the vending machine

5. Grading

The TAs will grade your lab assignments with *vending_machine_TB.v* that is already given to you. If you pass all the test in *vending_machine_TB.v*, you will get a full score. Your score is determined how many tests you pass.

6. Lab Report Guidance

You are required to submit a lab report for every lab assignment. You can write your report either in Korean or English.

Your lab report **MUST** include the following sections:

1. Introduction
    a. *Introduction* includes what you think you are required to accomplish from the lab assignment and a brief description of your design and implementation.
2. Design
    a. *Design* includes a high-level description of your design of the Verilog modules (e.g., the relationship between the modules).
    b. Figures are very helpful for the TAs to understand your Verilog code.
    c. The TAs recommend you to include figures because drawing the figures helps you how to *design* your modules.
3. Implementation

a. *Implementation* includes a detail description of your implementation of what you design.

b. Just writing the overall structure and meaningful information is enough; you do not need to explain minor issues that you solve in detail.

c. **Do not copy and paste your source code.**

4. Evaluation

   a. *Evaluation* includes how you evaluate your design and implementation and the simulation results.

   b. *Evaluation* must include how many tests you pass in vending_machine_TB.v

5. Discussion

   a. *Discussion* includes any problems that you experience when you follow through the lab assignment or any feedbacks for the TAs.

   b. Your feedbacks are very helpful for the TAs to further improve *EE312* course*!*

6. Conclusion

   a. *Conclusion* includes any concluding remarks of your work or what you accomplish through the lab assignment.

7. Requirements

You **MUST** comply with the following rules:

- You should implement the lab assignment in **Verilog**.
- You should only implement the **TODO** parts of the given template.
- You should name your lab report as **Lab2_YourName_StudentID.pdf**.
  One report per team
  You do not have to use the same teams as Lab 1
- You should compress the lab report, simulation results, and source code, then name the compressed zip file as **Lab2_YourName_StudentID.zip**, and submit the zip file on the KLMS.
- You should not change the interface of the vending machine defined in *vending_machine.v*.

Reference

1. Frank Vahid (2010). *Digital Design with RTL Design, Verilog and VHDL* (2nd ed.). Jorn Wiley and Sons. p. 247. ISBN 978-0-470-53108-2