CSED311 Lab2: RTL Design

컴퓨터공학과 강덕형 20170684
컴퓨터공학과 이승준 20170735

Introduction

설계 및 구현 목표

이번 Lab 과제는 RTL(Register-Transfer Level)을 이용하여 간단한 vending machine을 구현하는 것이 목표이다. 이 과정에서 FSM(Finite State Machine) 을 설계하고, 이를 combinational logic 과 sequential logic 을 이용해 구현한다.

학습목표

이번 Lab 과제를 통해 FSM을 이해하고, 이를 구현할 때 사용하는 Combinational logic과 Sequential logic의 특성에 대해 학습하는 것이 목표이다. 이것들을 응용해 vending machine 을 구현하며, 각 모듈별 역할을 적합하게 나누며 모듈화를 익힌다.

Design

1. wait time 을 체크하고 업데이트하는 하나의 모듈을 설계한다. 만약 wait time 이 0 보다 작거나 같으면, 돈을 모두 거슬러주라고 명령하는 return_flag 를 다른 모듈들에 propagate 한다.
2. 현재 state 정보와 input을 받아서 다음 state를 계산하는 하나의 모듈을 설계한다. 해당 모듈은 다음 state를 state를 update하는 모듈에 전달만 해준다. 그러므로, 해당 모듈은 combinational logic 만으로 구성한다.
3. clock signal에 맞게, 머신의 state를 update하는 모듈을 설계한다. 따라서, 해당 모듈은 sequential logic 만으로 구성되어 있다.
4. 위 세 개의 모듈을 submodule 로 가지는 vending machine 모듈은, 외부로부터 input 을 받고, input 과 내부 변수들을 이용해 submodule 들을 연결하고, output 을 submodule 들로부터 받아온다.

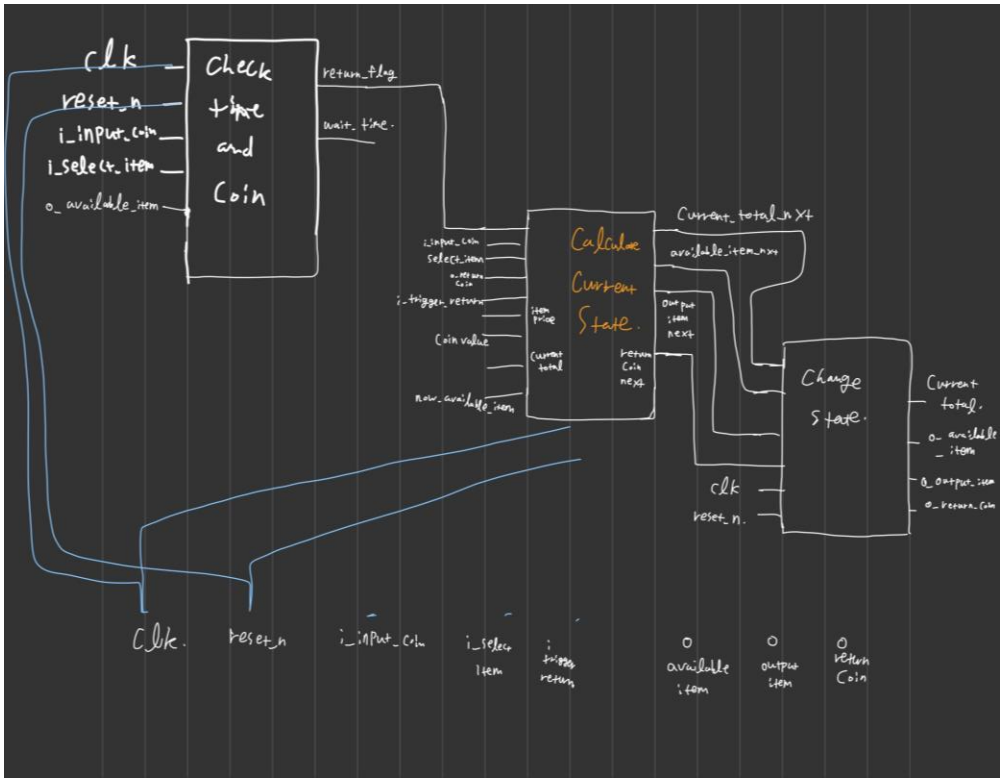

Figure 1. Vending machine module diagram

Implementation

우선 machine의 state와 action을 정의하겠다. state는 **{current_total, o_available_item, o_ouput_item, o_return_coin}** 으로 정의된다. action은 **i_input_coin | i_select_item | i_trigger_return**로 정의된다.

check_time_and_coin.v: coin이 새로 들어오거나 available한 item을 선택했을 때, wait time을 초기화한다. wait time이 0 보다 작아지는 걸 감지하며 이 때 return flag를 활성화한다. return flag는 calculate_current_state 모듈에 전달되며, 돈을 거슬러 주라고 명령하는 flag이다.

calculate_current_state.v: 현재 state 정보와 action, 그리고 그 외 input 들을 받아서 next state 를 계산만 한다. 그리고 next state를 change_state 모듈에 전달한다. 계산만 하는 모듈이기에, combinational logic 으로만 구성된다.

change_state.v: calculate_current_state 모듈에게 전달받은 next state 값들을, clk 의 posedge 마다 머신의 현재 state로 업데이트해준다. 업데이트만 해주는 모듈이므로, sequential logic 으로만 구성된다.

vending_machine.v: 위에 정의된 submodule들을 instantiation 한다. 그 후, vending machine 모듈의 input과 내부 변수들을 각 submodule instance들에 적합하게 연결해준다. 그 후, 각 모듈들에서 나오는 output 들을 vending machine 의 output 에 연결해준다.

Discussion

Why we did modularization in this way

초기 제공받은 skeleton 코드의 check_time_and_coin module에는 o_return_coin이 output으로 있었다. 그런데 우리는 모듈이 한 가지 역할만 수행해야 된다고 생각했다. check_time_and_coin module의 경우 time과 coin을 check만 해야 하고, calculate_current_state module에서는 state들을 계산만 해야 한다고 생각하였다. 그러므로 우리는, o_return_coin 을 check_time_and_coin module 에서 제거하고, return_flag 를 대신 추가해주었다. 그리고 check_time_and_coin module 에서 나오는 return flag 와, action 인 i_trigger_return 을 calculate_current_state module 에서 받아, 거스름돈 계산은 이 모듈이 전담하여 수행하도록 설계했다. 그리고, machine의 state가 pos clock에 synchronous하게 update 되도록 하기 위해, change_state 모듈을 설계하였다.

Use of `$signed` in check_time_and_coin module

한 번에 return할 수 있는 돈의 양이 정해져 있고 현재 가지고 있는 돈보다 많은 경우가 있기 때문에, vending machine은 return을 여러번 해줘야 한다. 이를 위해 wait time이 0보다 작거나 같아져 돈을 거슬러줄 경우, 매 pos clk 마다 wait_time 이 계속 작아지며 return을 수행하도록 설계하였다.
따라서 우리 모듈은 wait_time이 0보다 작거나 같아지는 상황을 감지할 수 있어야 하였고, 이를 위해 wait_time이 음수로 표현 될 수 있어야 한다. 따라서 wait_time에 $signed를 사용하였다.

Conclusion

우리는 본 과제에서 Combinational logic과 sequential logic을 이해하고 이를 이용하여 FSM을 설계하여 vending machine RTL 을 verilog 로 구현하였다. 또한 구현 과정에서 modularization을 적용하여 코드의 readability 와 재사용성을 향상시켰다.