Lab1_yujineuna_20200927_20200445

Report should include (1)introduction (2)design (3)implementation (4)discussion (5) conclusion

1) Introduction

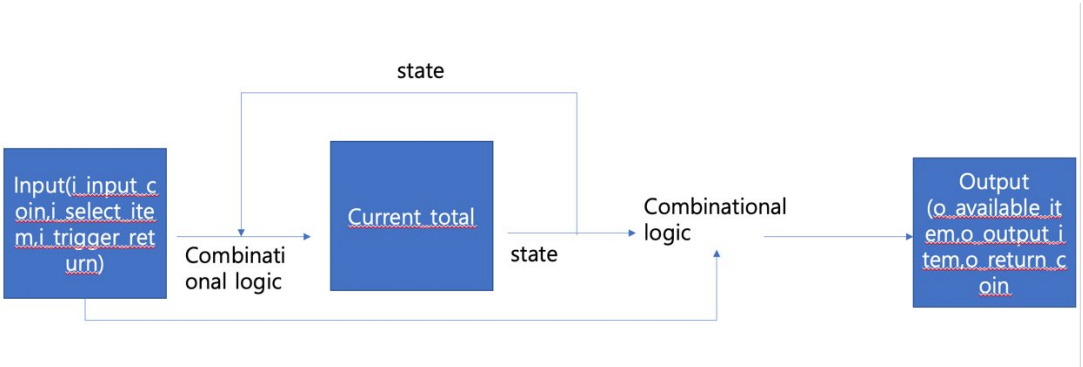
RTL을 design 하는 과정을 통해 vending machine의 알고리즘을 설계하는 과제였다. Input으로 i_input_coin, i_select_item, i_return_trigger, clk, reset_n이 주어지고 output으로 o_output_item, o_available_item, o_return coin이 나와야 한다. 즉 특정한 동전과 아이템, 강제반환 버튼, 클럭 사이클이 입력으로 주어질 때 구매 가능한 아이템과, vending machine에서 제공된 item, 반환되는 코인이 출력으로 나와야 한다.

알고리즘은 다음과 같다.

1. 돈을 넣으면 coin의 type에 따라 i_input_coin의 값이 달라지고 waiting time이 초기화 된다.
2. Vending machine은 돈을 넣으면 현재 money 보다 값이 싼 모든 item을 보여주면서 waiting time을 1만큼 감소시킨다.
3. 2의 상태에서 다시 돈을 넣으면 1로 간다.
4. 2의 상태에서 가능한 아이템을 선택할 시 item이 나오고 2로가면서 waiting time을 초기화한다.
5. 2에서 가능하지 않은 아이템을 선택할 시 아무일도 일어나지 않는다.
6. Waiting time내로 input이 들어오지 않으면 잔돈이 반환된다.

- Return button을 누를 시 잔돈이 반환되고 1로 간다.

2) Design



Mealy machine 이용하여 Input과 current_total이라는 현재 state를 이용해 output을 계산할 수 있도록 디자인하였다.

Submodules>

*Check_time_and_coin :
Coin return time을 update하고 o_return_coin을 계산할 수 있도록 해야한다.
들어오는 input item이 바뀌거나 item이 dispensed되면 wait time을 초기화 해준다.
Wait time이 0이거나 i_trigger_return이 참이면 Current_total을 이용해 현재의 o_return_coin을 계산한다. Wait time이 매 clk마다 1씩 줄어들도록 한다.

*Calculate_current_state :
돈이 들어오거나 item이 선택되면 current state(현재 machine이 가지는 돈)와 input에 따라 next state를 계산한다. Wait time이 0이될 경우 current state가 0이 될 때까지 돈을 반환한다. 이때 o_return_coin을 input으로 해서 current state에서 상품 가격만큼 빼주고, next state를 계산한다. Current state와 i_select_item을 이용해 o_available_item을 계산한다. i_select_item에서 선택된 item을 확인하고, o_available_item일 경우 o_output_item을 계산한다.

*Change_state :
매 clk마다 Calculate_current_state에서 next state를 계산하면 next state로 현재 state를 바꾼다.

Top-level module(vending machine)

3) Implementation

***Calculate_current_state :**
Input: i_input_coin, o_return_coin,i_select_item,i_trigger_return,current_total, o_return_coin, wait_time

-Combinational logic for Current_total_nxt
Wait time이 0이거나 i_trigger_return이 1인데 current_total이 0이 아닌 경우 남은 돈이 반환되어야 한다. 따라서 return_total을 check_time_and-coin 모듈의 output인 o_return_coin을 통해 구하고 현재 총액인 current_total에서 return_total을 빼줌으로써 current_total_nxt를 구한다.
반환될 필요가 없는 경우 i_input_coin이나 i_select_item이 달라질 경우 각각 해당하는 값을 Input_total과 output_total에 대입한다. Current_total에서 input_total은 더해주고 output_total은 빼주면서 current_total-nxt를 구한다.

-Combinational logic for output
Current_total 보다 싼 item을 확인하면서 o_available_item을 구해주었다.
I_select_item과 o_available_item을 확인하면서 o_output_item을 구해준다.

***Check_time_and_coin**

-Combinational logic for wait time

I_input_coin과 o_output_item이 바뀔 때, wait_time을 초기화해줘야한다.

-Sequential logic for o_return_coin

Current_total이 1000보다 큰지, 500보다 큰지, 100보다 큰지 각각 확인하여 o_return_coin을 계산하였다.

-Sequential logic for wait time

Reset_n이 눌리면 wait_time을 초기화 하고 o_return_coin도 0으로 초기화 한다.
만약, 그 외의 경우 wait_time을 1만큼 감소시킨다.

***Change_state**

-Sequential circuit to reset or update states

Reset_n이 눌리면 current_total을 0으로 바꿔주고, 아닌 경우에는 current_total을 current_total_nxt로 한다.

4) Discussion

Wait_time이 음수일 경우에 반환이 작동하지 않는데 이를 방지하기 위해 wait_time이 0보다 클 경우만 1감소시켜주었다.

O_return_coin을 계산하기 위해서 sequential logic을 이용했는데, 이는 combinational logic을 사용할 경우 testbench에서 o_return_coin을 기다리는 동안 while문을 도는데 o_return_coin이 바뀌기 전에 while문을 한바퀴 돌아 current가 빠르게 감소하는 문제점이 있었기 때문이다. 따라서 x, y, z를 non_blocking을 통해 값을 assign하고 이를 통해 o_return_coin을 계산했다.

5) Conclusion

Vending machine을 설계하는 과정을 통해 combinational logic design과 sequential logic design에 대해 알게 되어 비동기적으로 처리하는 상황을 이해할 수 있었다. moore machine과 mealy machine 같은 fsm에 대해 더 깊이 알게 되었다.