LAB 2. RTL Design

201800038 박형규, 20180480 성창환

1. Introduction

이번 Lab에서는 Verilog를 통해 주어진 조건들을 정확하게 실행할 수 있는 간단한 vending machine을 FSM으로 구현해본다. 3가지 종류의 동전을 받고, 4가지 종류의 아이템을 가지고 있는 vending machine을 구현할 것이다. 이번 랩을 통해 Verilog의 사용법을 익히고 FSM의 구조 및 구현방법을 익히게 될 것이다.

2. Design

Vending Machine을 FSM을 통해서 구현하기위해 5개의 state로 분리해서 디자인했다.

first: 초기상태, 동전이 들어오면 coin으로 넘어감.

coin: 들어온 동전의 값을 _nxt들에 저장하고 stay로 넘어가면서 waitTime을 초기화.

stay: 현재 선택가능한 item들을 보여준다. 시간을 초과하면 change로, 아이템을 선택하면 print로 동전을 넣으면 coin으로 이동한다.

print: 입력된 아이템을 위한 돈이 충분한지 판단해서 충분하면 아이템을 보내고 값들을 변경한 뒤에 coin으로 돌아가고 충분하지 않으면 바로 coin으로 돌아간다.

change: 잔액을 반환하고 first로 이동.

State를 이동할 때 현재 state와 input에 따라 정해지므로 mealy machine이다. 현재 state를 나타내는 CS, 다음 state를 알려주는 NS를 선언해서 구현한다.

3. Implementation

주어진 레지스터 외에 추가로 다음과 같은 레지스터를 선언하였다. 먼저 각 state를 나타내기 위해 parameter first, coin, stay, print, change를 선언하였다. 각 parameter는 Design에서 설명한 각 state를 나타낸다. reg[2:0] CS는 현재의 state를 나타내는 레지스터로 A~F의 값 중 하나를 가진다. Reg[2:0] NS는 다음 state를 나타내는 레지스터로 A~F의 값 중 하나를 가진다. reg [`kNumCoins-1:0] input_coin은 i_input_coin이 들어오면 그 값으로 변경되고, B state에서 코인 투입이 반영되면 0으로 초기화된다. reg [`kNumItems-1:0] select_item은 i_select_item이 들어오면 그 값으로 변경된다. reg [1:0] output_item은 몇 번째 item이 선택되었는지를 나타내

는 register이다. reg [1:0] return_coin는 어떤 coin이 output으로 선택되었는지를 나타내며 trigger_return은 i_trigger_return값이 활성화되면 trigger_return값도 활성화된다.

Vending_machine을 구현함에 있어서 combinational logic과 sequential circuit을 나누어 작성하였다. 또한 combinational logic에 있어서도 next states와 관한 combinational logic과 outputs에 관한 combinational logic을 따로 작성하였다.

다음은 Combinational logic중 next state에 관한 것들에 대한 것들이다. 먼저 CS가 first일 경우 input_coin이 0이 아닐 경우에는 NS를 coin으로 보내주고, 이외의 경우에는 NS를 first로 설정해준다. CS가 coin일 경우에는 input으로 들어온 코인의 액수를 current_total_nxt에 더해주고 input_coin[i]값을 초기화해준다. 또한 waitTime을 'kWaitTime으로 재초기화 시켜주며 Next State를 stay로 설정한다. CS가 stay일 경우에는 input_coin이 들어오는 경우에는 NS를 coin으로, item이 selected되는 경우에는 NS를 print로, waitTime이 0인 경우에는 NS를 change로 설정한다. CS가 print인 경우에는 현재 투입한 총 금액이 사려고하는 item의 금액보다 높다면 총 금액에서 사려고 하는 물건의 금액을 빼주고, output_item을 설정해주며 select_item[i]를 초기화해주고 waitTime을 재초기화시켜준다. 그 뒤에 다시 Next State를 stay로 설정한다. 마지막으로 CS가 change일 경우에는 current_total_nxt가 0이 될때까지 return_coin을 변경하면서 잔액을 return해주고 current_total_nxt가 0이 되면 NS를 first로 설정한다. 여기서 coin와 print에서만 wait Time을 초기화 시켜주는 이유는 coin와 print의 NS가 stay로 설정되므로 stay에서 최대 kwaittime만큼을 새롭게 대기하기 때문이다.

Combinational logic중 outputs와 관한 코드는 각각 current_total, output_item, return_coin에 따라 output을 설정해준다.

마지막으로 Sequential circuit에 관한 코드에서는 reset버튼이 눌린다면 모든 항을 처음과 같이 reset시키고 다시 first state로 돌아간다. reset버튼이 눌리지 않았다면 current_total을 current_total_nxt로 업데이트 시키고, waitTime을 줄인다. 그 뒤에 CS를 NS로 설정한다. 이 때, 리턴 버튼이 눌렸다면 NS를 change state로 설정해준다.

4. Discussion

Always @(*) begin에서 *을 사용하니 오류가 생겨서 필요한 값들에 대해서만 구현해서 오류를 해결했다.

처음에는 6개의 state로 진행했지만 출력 값에 자꾸 오류가 생겼다. 굳이 필요 없는 state때문에 clock이 불필요하게 낭비되어서 그런 것 같아 5개의 state로 줄였더니 문제가 해결되었다.

5. Conclusion

Verilog를 이용해서 simple vending machine을 구현해보면서 Verilog의 사용법을 더 잘 이해하게 되었다. 또한 simple vending machine과 같은 FSM을 구현할 때 이용되는 combinational logic과 sequential logic의 차이를 이해하고, Moore Machine과 Mealy Machine의 차이를 복습할 수 있었다. 마지막으로 디지털시스템설계 강의에서 배우고 프로젝트로 진행했던 FSM을 코드를 통해서 구현하는 과정을 통해 실제 회로가 Verilog에서는 어떻게 구현되는지 더 잘 이해할 수 있었다.