

# EE303 Lab2 Report

20180155 김준범

20160791 권용빈

## I. Introduction

Lab2는 Lab1의 연장선으로 vending machine을 구현함을 통해 verilog의 syntax에 더 익숙해지는 과제이다. Lab2를 통해 state를 update해주는 machine을 구현하는 방법을 배웠고 두개 이상의 always block을 동시다발적으로 실행하는 방식을 배우고 익숙해졌다.

## II. Design

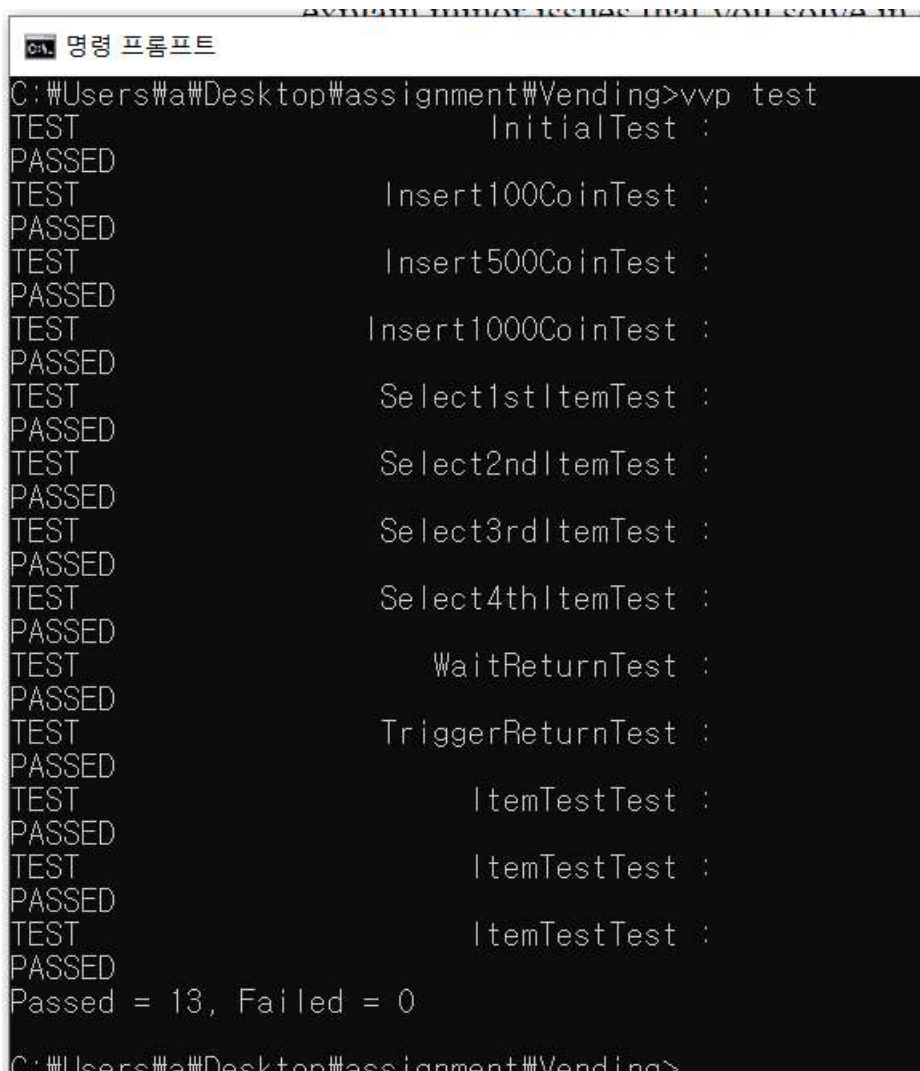
일상생활에서 접할 수 있는 자판기와 동일한 기능을 하는 vending machine program을 구현하였다. Input coin bit를 받아 vending machine이 보유하고 있는 금액을 update하고 select item bit를 통해 구매하고자 하는 item을 구매하고 상황에 따라 잔돈을 반환하는 기능을 구현하였다.

## III. Implementation

본 랩에서 작성한 코드는 한 개의 sequential 구문과 두 개의 combinational 구문으로 구성되어 있다. 첫 번째 sequential 구문은 input signal을 return이 trigger 된 경우와 그렇지 않은 경우로 나누어지며 return이 trigger 되지 않은 경우는 coin이 들어온 경우와 item이 선택된 경우로 나뉜다. 해당 구문에서는 next state에 전달되어야 할 변수들의 value를 지정해준다. 두 번째 sequential 구문은 output 인자, 선택 가능한 item과 실제 item이 선택된 경우의 output에 대한 값을 설정해준다.

available 한 item은 현재의 input된 돈이 item의 가격보다 크며 item의 개수가 남아있을 때 1로 설정된다. 선택한 item이 available 한 경우 해당 item에 대한 output이 1로 설정된다. sequential 구문에서는 reset인 경우 여러 state의 값을 초기화해주고 reset이 아닌 경우 next state에 전달되어야 할 변수들을 현재 state의 값에 대입한다.

#### IV. Evaluation



```
C:\Users\wa\Desktop\assignment\Vending>vvp test
TEST                                InitialTest :
PASSED
TEST                                Insert100CoinTest :
PASSED
TEST                                Insert500CoinTest :
PASSED
TEST                                Insert1000CoinTest :
PASSED
TEST                                Select1stItemTest :
PASSED
TEST                                Select2ndItemTest :
PASSED
TEST                                Select3rdItemTest :
PASSED
TEST                                Select4thItemTest :
PASSED
TEST                                WaitReturnTest :
PASSED
TEST                                TriggerReturnTest :
PASSED
TEST                                ItemTestTest :
PASSED
TEST                                ItemTestTest :
PASSED
TEST                                ItemTestTest :
PASSED
Passed = 13, Failed = 0
C:\Users\wa\Desktop\assignment\Vending>
```

Lab2는 Lab1의 연장선으로 vending machine을 구현하는 과제이다. Lab2를 통해 state를 update해주는 machine을 구현하는 방법을 배웠고 두개 이상의 always block을 동시다발적으로 run하는 방식을 익혔다.

Always block들에 조건문을 추가하지 않은 always@(\*)의 mechanism을 완벽하게 이해하지 못하였다.

#### V. Discussion

저희는 하나의 clock cycle마다 input signal, output signal, state value 들이 변화해야 한다고 생각했다. 그래서 방향으로 코드를 작성했지만 11번 13번에서 예상 못 한 결과를 받았다. 그래서 첫 번째 combinational 구문에서 디버깅해본 결과 output signal이 두 번의 clock cycle 동안 들어온 것을 알게 되었고 이 부분을 적용하기 위해 block item 변수를 사용하여 첫 번째 clock 주기에서 num\_item 값을 변화시켰다면 그 다음 clock cycle에서는 변화시키 않도록 코드를 작성했다.

## VI. Conclusion

Vending machine을 구현하는 과정에서 여러개의 cycle을 concurrently하게 실행하는 방법을 배울 수 있었다. 앞으로의 과제에서 Lab1과 Lab2를 구현하는 과정을 통해 익힌 verilog사용 방법을 보다 실용적으로 적용할 수 있기를 바란다.