

Lab1 - RTL design

team21: 20200001조은국, 20200431박현빈

2022년 3월 15일
(CSED 311)

Introduction

ALU는 arithmetic logic module의 약어로, 기본적인 산술연산과 논리연산을 하는 module이다.

vending machine은 시간안에 돈을 넣고 그에 맞는 가격의 상품을 고를 때 상품이 출력되고, 일정 시간이 지나면 남은 금액이 반환되는 방식으로 동작한다.

Design

ALU

ALU에서는 instruction에 따라 주어진 두 숫자의 산술연산과 배타적 논리합, 논리곱, 논리합, shift 같은 논리연산등을 수행한다.

vending machine

vending machine은 vending_machine.v를 뼈대으로 check_time_and_coin, change_state, calculate_current_state의 3가지 모듈이 기능적으로 동작한다.

1. coin이 들어오는 경우
2. item이 select되는 경우 -> (해당 아이템을 구매 가능한 경우, 불가능한 경우가 존재)
3. return_trigger가 동작되는 경우

Implementation

ALU

FuncCode가 변화할 때마다 if문 block에 따라 다른 연산이 수행된다. 특히 덧셈과 뺄셈 연산에 대해서는 msb의 bit를 check하여 overflow를 파악한다.

vending machine

vending machine의 주 기능을 수행하는 check_time_and_coin, change_state, calculate_current_state의 3가지 모듈을 주로 수정하였다.

1. check_time_and_coin(mealy machine)

- sequential logic 방식으로 구현한다.
- wait_time은 기본 상태나 reset_n이 0으로, input이거나 item_select(available)이 kWaitTime으로 초기화한다. 0이 아닌 경우 매 clock마다 1씩 줄게 한다.
wait_time은 moore machine 방식으로 동작하도록 한다.
- i_trigger_return이 1이거나 wait_time이 0인 경우에는 잔고가 0이 될 때까지 계속 반환하도록 설정한다.

2. calculate_current_state(mealy machine)

- combinational logic 방식으로 구현한다.
- o_available_item은 돈이 변할 때마다 변하도록 설정한다.
- o_output_item은 item_select와 available이 모두 1인 경우 한 clock동안 1이 되도록 설정한다. 기본은 4'b0000
- input_total, output_total, return_total은 각각의 사건이 발생할 시 누적되도록 한다.
- current_total_nxt는 input/output/return 신호가 발생한 경우 다음 clock에서 갖게 될 돈을 변화시킨다.

3. change_state(moore machine)

- sequential logic 방식으로 구현한다.
- current_total을 설정한다. 초기와 reset_n의 경우 0으로, 아닌 경우 current_total_nxt로 설정한다.

Discussion

ALU

overflow flag를 설정하는 logic에 있어서 초기에 잘못생각해 예상하지 않은 결과가 나타나는 부분이 있었다. input value들의 msb를 고려해서 flag를 1 또는 0으로 지정해야 했다.

vending machine

calculate_current_state module에서 i_input_coin과 i_select_item을 통해 currnet_total을 계산할 때 예상되는 값보다 current_total의 값이 2배로 산출되는 오류가 있었다. 이에 module 내 always @(*)에서 always@(i_input_coin)과 always@(i_select_item)으로 바꿨을 때 중복해서 계산하는 경우가 사라졌다.

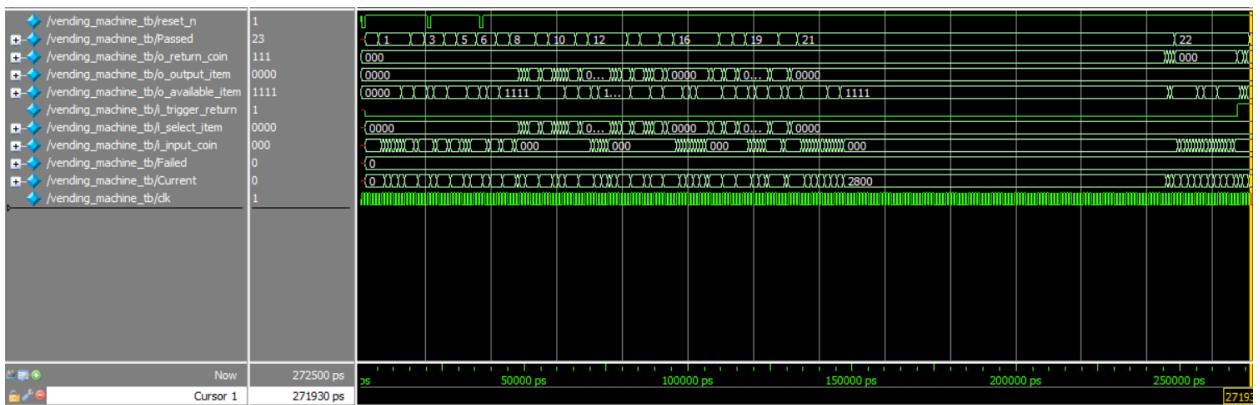
check_time_and_coin module에서 o_return_coin을 언제 계산해야 할지 알아야 했기 때문에 i_trigger_return을 위 module의 input으로 추가하였다. 또한 make_curr_0이라는 integer register를 추가해 잔액을 return해야 하는 경우를 판단하는 용도로 사용했다. 또한 o_return_coin으로 총 잔액을 한번에 반환할 수 없는 경우가 존재했는데, 여러 번에 걸쳐 반환하는 방식을 통해 해결하였다.

reset_n이 0인 상황에서 current_total이 초기화 되지 않아 o_return_coin이 계산되는 오류가 발생했다. change_state module에서 reset_n이 0일 때 current_total이 0이 되는 코드를 추가했고, current_total과 o_return_coin이 reset_n의 의미대로 초기화가 될 수 있었다.

Conclusion

1. 실험 결과

ALU와 vending_machine 모두 주어진 testcase에서는 온전히 동작함을 발견할 수 있었다.



2. 결론

ALU와 vending machine의 동작 방식 및 구현에 대해 심층적으로 알아볼 수 있는 경험 이었다. 더불어 verilog내에서 combinational logic, sequential logic[1] 사용되는 방식 또한 습득할 수 있었다.