컴퓨터 구조 Lab1

20190439 컴퓨터공학과 오승훈
20190782 컴퓨터공학과 최서영

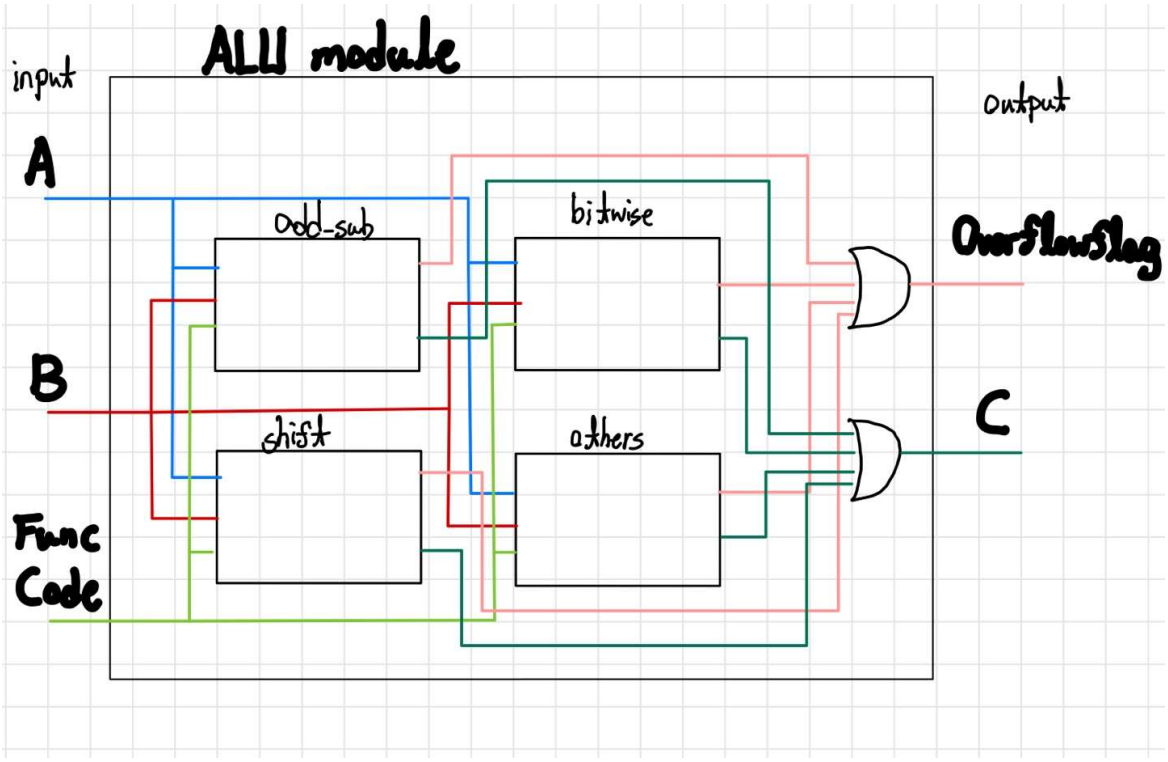
Introduction

이번 lab에서는 ModelSim을 이용하여 Verilog의 기본적인 기능과 문법을 학습해보고, add, sub, shift, bitwise 등의 기본적인 operator들의 사용법을 익히며 ALU를 만드는 것이 최종 목표이다. 또한 이 모든 것을 만든 후 최종적으로 우리가 만든 ALU가 정상적인 동작을 하는지에 대해서 테스트를 돌려보는 것까지가 목표가 된다.

Design

이번 lab의 경우 따로 모듈화가 필요가 없지만 모듈화를 연습을 해보는 것을 의의로 두어 ALU top module이 있고, 이 모듈 내에는 총 4개의 module이 존재한다. 기능적으로 add_sub, shift, bitwise, others로 구분을 하여 진행을 하였다. Top module인 ALU module과 sub module인 add_sub, shift, bitwise, other를 따로 설명을 하자면 ALU module에서는 input 값인 A, B, FuncCode, output인 C, OverflowFlag가 존재하는데 이 모듈에서는 sub_module로 wire를 이용하여 A, B, FuncCode 값을 sub_module로 연결을 해주고, sub_module의 output C, OverflowFlag를 받아온다. 그 후 받아온 모든 sub_module의 output 들인 4개의 C, 4개의 OverflowFlag를 C는 C끼리, OverflowFlag는 OverflowFlag끼리 OR 연산을 한 뒤 최종적인 Top_module의 output C, OverflowFlag에 assign을 해줌으로써 구현이 완료가 된다.

Sub_module에서는 input 값인 A, B, FuncCode, output인 C, OverflowFlag가 존재하는데 A, B, FuncCode는 모두 Top_module에서 wire를 통해 연결받아온 값이다. C는 연산이 진행되고 난 후의 값이 Top_module로 내보내지는 값이며, OverflowFlag는 add, sub 연산에서 overflow가 발생한 경우 체크하는 값이 된다. 각 모듈에서는 if 문을 이용하여 해당하는 FuncCode가 들어온 경우에 필요한 연산을 진행하고, output 값인 C와 OverflowFlag를 Top module로 output을 보내는 역할을 진행한다.



위의 그림은 ALU module의 design을 스케치한 것이다.

Implementation

이번 lab에서 총 16개의 기능을 구현하여야 했는데, 이를 비슷한 것들끼리 나누어 4개의 모듈로 구현하였다. 이 4개의 모듈에 input 값들을 모두 넣고, if 연산으로 FuncCode에 해당하는 연산을 찾고 output 값을 계산한다. 만약 없다면, C와 OverflowFlag값을 모두 0으로 해주었다. 이 output 값들을 main 모듈에서 or 연산하여 최종적인 output을 계산하였다.

- ADD_SUB 모듈

  Signed Addition, Signed Subtraction 연산을 하는 모듈이다. 이때, overflow를 check해주기 위해 A, B의 sign bit와 C의 sign bit를 이용하여 계산하였다. A, B의 sign bit가 같은 때 C의 sign bit가 다르면 overflow가 일어난 것으로 판단하였다.

- Bitwise 모듈

  Bitwise NOT, AND, OR, NAND, NOR, XOR, XNOR 연산을 하는 모듈이다. OverflowFlag는 0으로 고정하고, 나머지는 베릴로그 문법을 통해 C에 결과값을 저장하였다.

- Shift 모듈

  Logical Left Shift, Right Shift와 Arithmetic Left Shift, Right Shift 연산을 하는 모듈이다. OverflowFlag는 0으로 고정하고, 나머지는 베릴로그 문법을 통해 C에 결과값을 저장하였

다. 이때, Arithmetic Left Shift, Right Shift의 경우 베릴로그 문법을 사용하기 위해선 signed로 선언되어 있어야 제대로 작동하기 때문에 signed로 선언한 wire에 A 값을 넣은 후 연산하였다.

- Others 모듈

  Identity, Two's Complement, Zero 연산을 하는 모듈이다. OverflowFlag는 0으로 고정하고, 나머지는 베릴로그 문법을 통해 C에 결과값을 저장하였다.

Discussion

ARS 기능을 구현할 때 Verilog 문법에 있는 것을 사용하였는데, 되지 않아 당황하였었다. 검색을 해보니, signed과 unsigned 경우 다르게 연산을 하고, 변수 선언 당시 signed라고 지정하지 않으면 unsigned로 저장된다는 사실을 알게 되었다. 이를 통해 변수 선언 당시 상황에 맞는 변수를 선언해야 한다는 것을 알게 되었다.

모듈을 나누어 연산을 할 때, if 문을 나누어 상황에 따라 다른 module을 부르면 다른 module은 작동하지 않는 것 인지 궁금하였다. 작년 디지털시스템 설계 수업 때는 다른 module이 작동하였던 기억이 있어 이번엔 모든 output을 받아 or로 연산하였다.

Conclusion

결과적으로 ALU의 16가지 기능을 모두 구현할 수 있었다. 이를 테스트 코드를 통해 확인하였었다.