Lab1 report

학번 : 20210273
이름 : 하태혁

1. 개요

Verilog 프로그래밍의 기초적인 문법을 이해하고 사용할 줄 알며 각 기능에 따라 Gate level에서 Module을 구성하고 연결할 줄 아는 것을 목표로 한다. Lab1_1에서는, 제작한 Module을 Design source에 추가하고 Testbench을 제작하여 적절한 I/O값을 입력해준다. 이후 Simulation을 진행하여 Verilog module이 적절히 구성되었는지 테스트 해본다. Lab1_2에서는 여러 코드를 통해 Functionally complete set을 직접 디자인하고 Schematic 기능으로 구성한 회로를 확인해 보는 것을 목표로 한다.

2. 이론적 배경

Xilinx Vivado에서 Verilog라는 언어를 사용하여 하드웨어 동작을 표현하며 하드웨어를 표현하는 언어를 HDL(Hardware Description Language)라 한다. Verilog로 모델링 할 수 있는 단계는 High-Level부터 Low-Level까지 있는데 순서대로 Behavioral, Register-Transfer, Gate 가 존재한다. Module에서 정의하는 변수는 한 변수당 하나의 신호를 처리하기 때문에 1-bit scalar이며 각 Gate와 Gate 사이를 연결하고 신호를 전달하기 위해서 wire을 정의하고 사용한다.

Functionally complete set은 집합 내부에 존재하는 Logic operation을 사용하여 어떠한 Boolean expression도 표현할 수 있는 집합을 의미한다. 예를 들어 {and, or, not} 의 경우 모든 Boolean expression을 표현할 수 있으므로 Boolean Algebra에서 functionally complete하다고 할 수 있다.

3. 실험 준비

NOT, OR, AND, NOR, NAND에 대한 Truth table은 다음과 같다.

P	not P
0	1
1	0

P	Q	P or Q
0	0	0
0	1	1
1	0	1
1	1	1

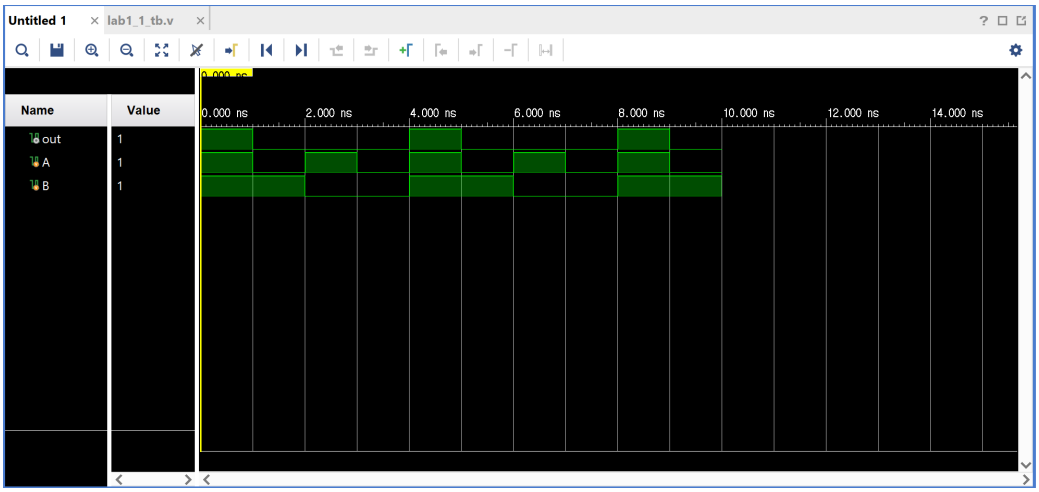
P	Q	P and Q
0	0	0
0	1	0
1	0	0
1	1	1

P	Q	P nand Q
0	0	1
0	1	1
1	0	1
1	1	0

P	Q	P nor Q
0	0	1
0	1	0
1	0	0
1	1	0

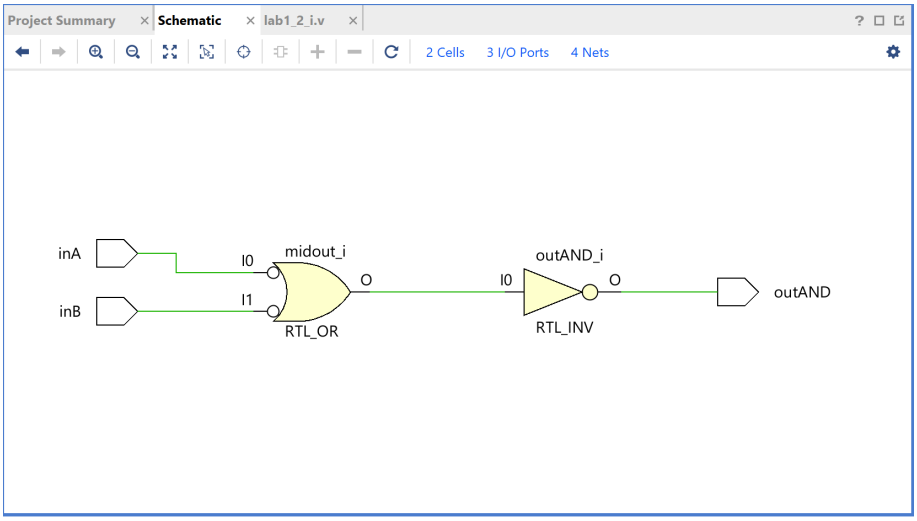
4. 결과

1) lab1_1



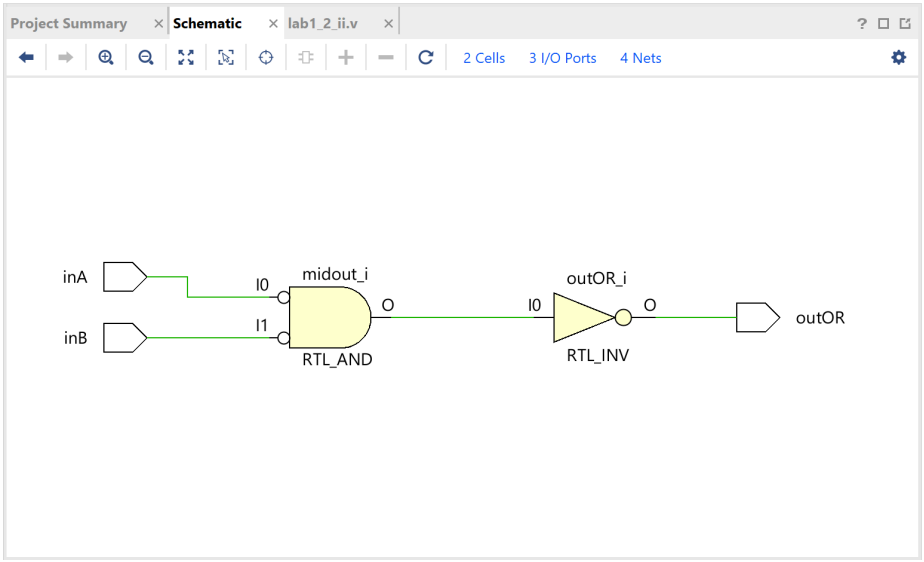
and gate를 구현하였다. A와 B값을 1로 초기화해주기 위해 initial begin에서 각 값을 초기화해 주었고 시뮬레이션 시간도 10ns로 설정하였다. always begin에서 A는 1ns마다 B는 2ns 마다 반전시켰다.

2) lab1_2_i



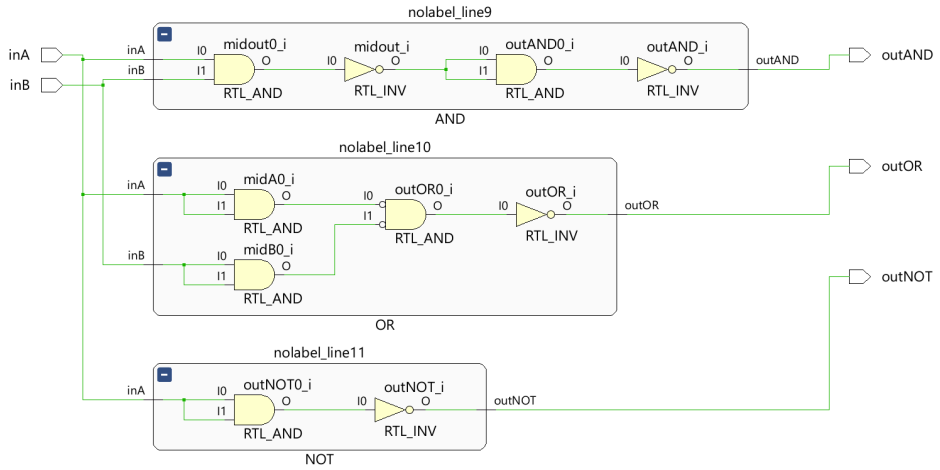
집합 {or, not}이 functionally complete set임을 증명하기 위해 or gate와 not gate를 사용해 AND gate를 구현하였다.

3) lab1_2_ii



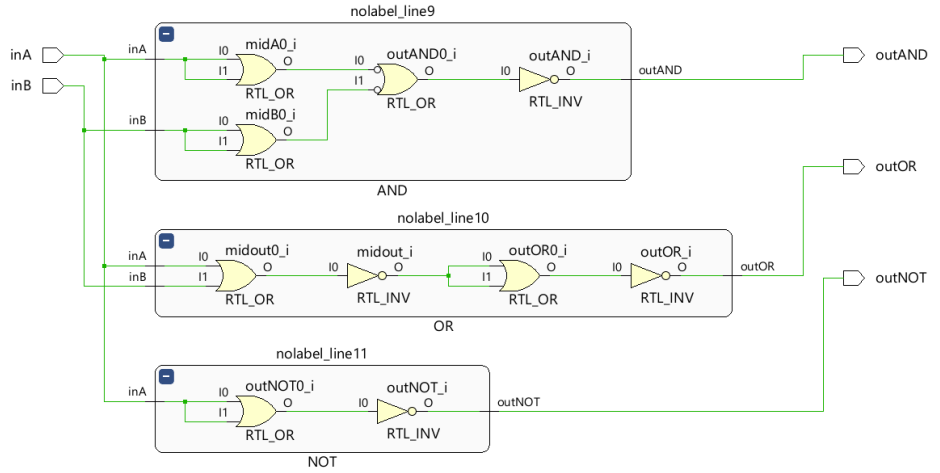
집합 {and, not}이 functionally complete set임을 증명하기 위해 and gate와 not gate를 사용해 OR gate를 구현하였다.

4) lab1_2_iii



집합 {nand}가 functionally complete set임을 증명하기 위해 nand gate만으로 {and, or, not} 집합을 완성하였다.

5) lab1_2_iv



집합 {nor}가 functionally complete set임을 증명하기 위해 nor gate만으로 {and, or, not} 집합을 완성하였다.

5. 논의

Module을 구성하는 과정이 C++의 OPP와 굉장히 비슷하다고 생각했지만, 문법에서는 차이점이 많았다. 먼저 각 gate를 구현하면 output이 바로 return 되는 게 아니라 다른 wire를 통해 받고 그 wire를 다른 gate와 연결하여 신호를 전달한다는 것이었다. 그러나 문법을 이해하니 코드를 짜는 것이 회로를 디자인하는 것과 매우 유사하다는 것을 느낄 수 있었다.

MAC 작업환경을 사용하고 있어서 처음에 베릴로그 작업환경을 구축하는 게 상당히 어려웠고 자일링스도 5번 재설치했다. 처음 보는 수많은 오류를 해결하기 위해 몇 시간 동안 구글링하며 많은 노력을 기울였고 자일링스가 인식하지 못하는 파일은 다른 컴퓨터에서 설치하고 해당 파일을 압축해서 MAC으로 다시 옮기는 등 온갖 방법을 동원하며 오류를 해결하기 시작했다. 처음에는 시간이 아깝다 느껴졌지만, 에러를 해결하는 과정에서 많은 걸 배울 수 있었다. 구글링 실력도 늘었고 작업환경과 파일 경로를 스스로 생각해보며 컴퓨터를 조금 더

이해할 수 있었다.