4주차 결과보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20191629 이름: 이주헌

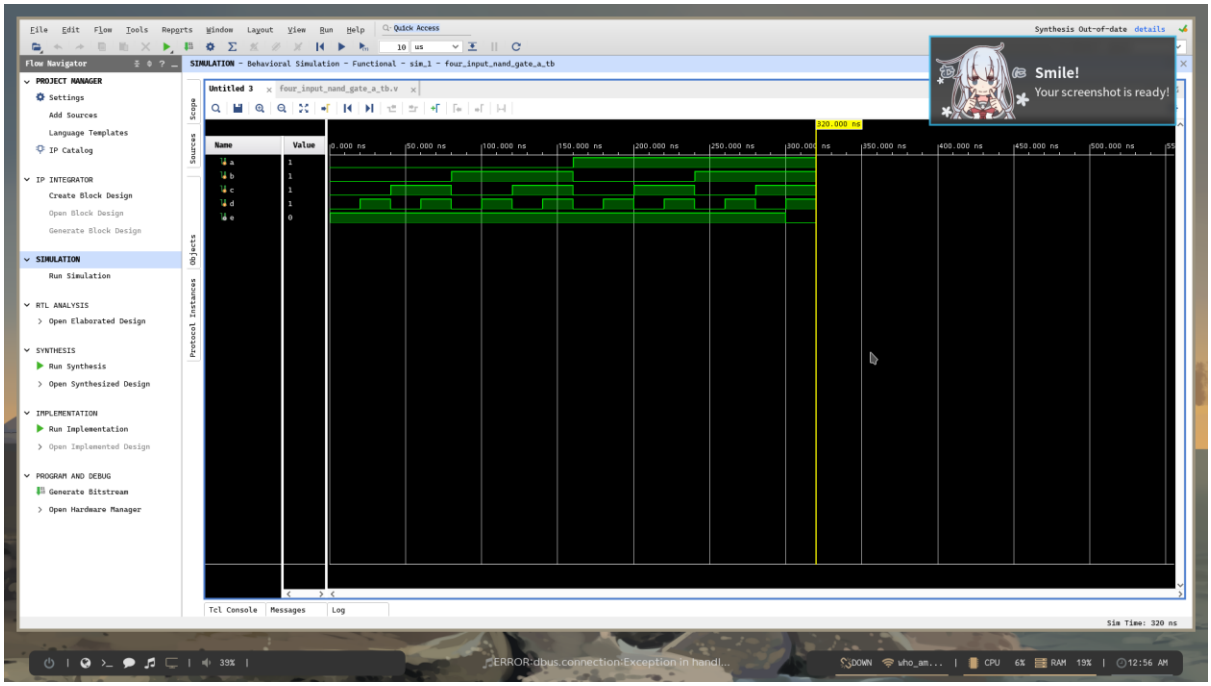
1. 실험 목적

NAND, NOR, XOR 게이트의 원리와 그 동작을 눈으로 확인하고, Verilog를 이용하여 각 게이트를 구현할 수 있다.

2. 4-input NAND gate의 simulation 결과 및 과정에 대해서 설명하시오.

Input A	Input B	Input C	Input D	Output E
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

4개의 입력을 갖는 NAND 게이트의 진리표



4개의 입력을 갖는 NAND 게이트의 진리표

4개의 입력을 갖는 NAND 게이트의 경우, (A)의 회로와 (B)의 회로는 같다고 볼 수 없다.

$$\begin{aligned}&\neg(abcd)\\&=\neg(ab)+\neg(cd)\\&=\neg a+\neg b+\neg c+\neg d\end{aligned}$$

$$\begin{aligned}&\neg(\neg(\neg(ab)c)d)\\&=\neg(\neg((\neg a+\neg b)c)d)\\&=\neg((\neg(\neg a+\neg b)+\neg c)d)\\&=\neg(\neg(\neg a+\neg b)+\neg c)+\neg d)\end{aligned}$$

따라서, 4개의 입력을 갖는 경우에도 (A) 회로와 (B) 회로는 같다고 볼 수 없다.

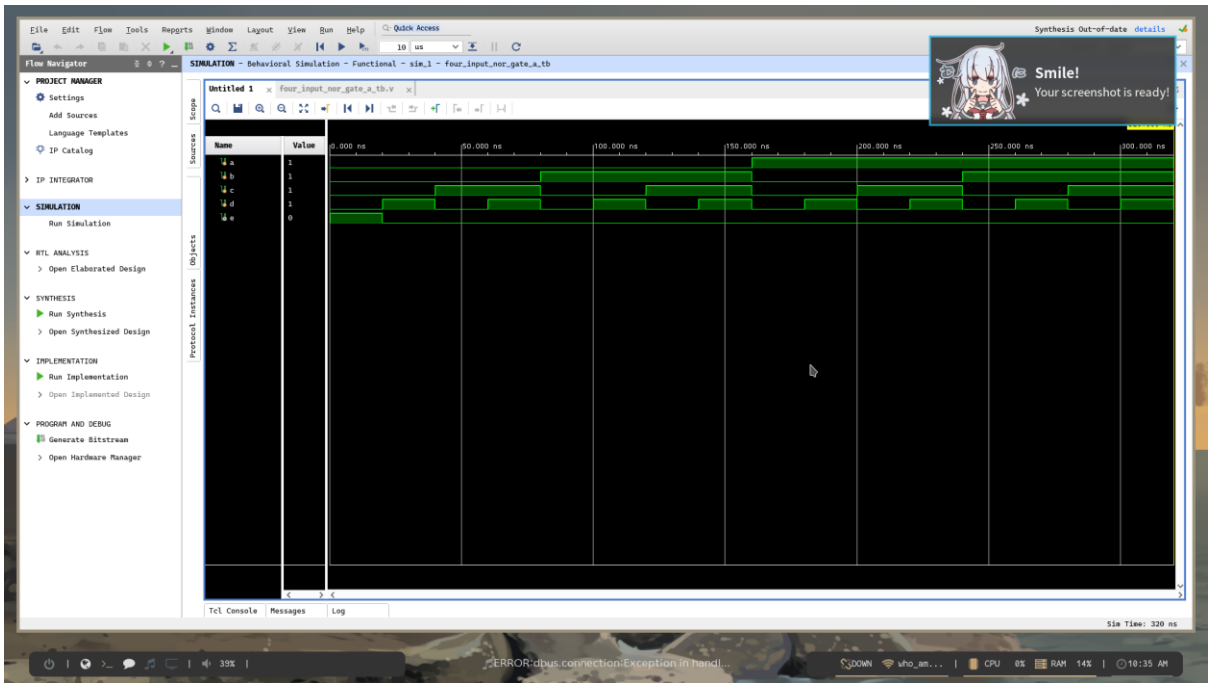
NAND를 반전한 게이트인 AND 게이트의 경우는 입력 신호 중 두 개를 먼저 묶어 연산하여도 출력 값은 변하지 않았다. 그러나 NAND 게이트의 경우는 다중 입력 게이트의 "정의"에 따라 연산의 값이 달라지기 때문에 다중 입력 NAND 게이트가 무엇인지 확실하게 정의하여야 한다.

3. 4-input NOR gate의 simulation 결과 및 과정에 대해서 설명하시오.

Input A	Input B	Input C	Input D	Output E
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0

0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

4개의 입력을 갖는 NOR 게이트의 진리표



4개의 입력을 갖는 NOR 게이트의 실행 결과

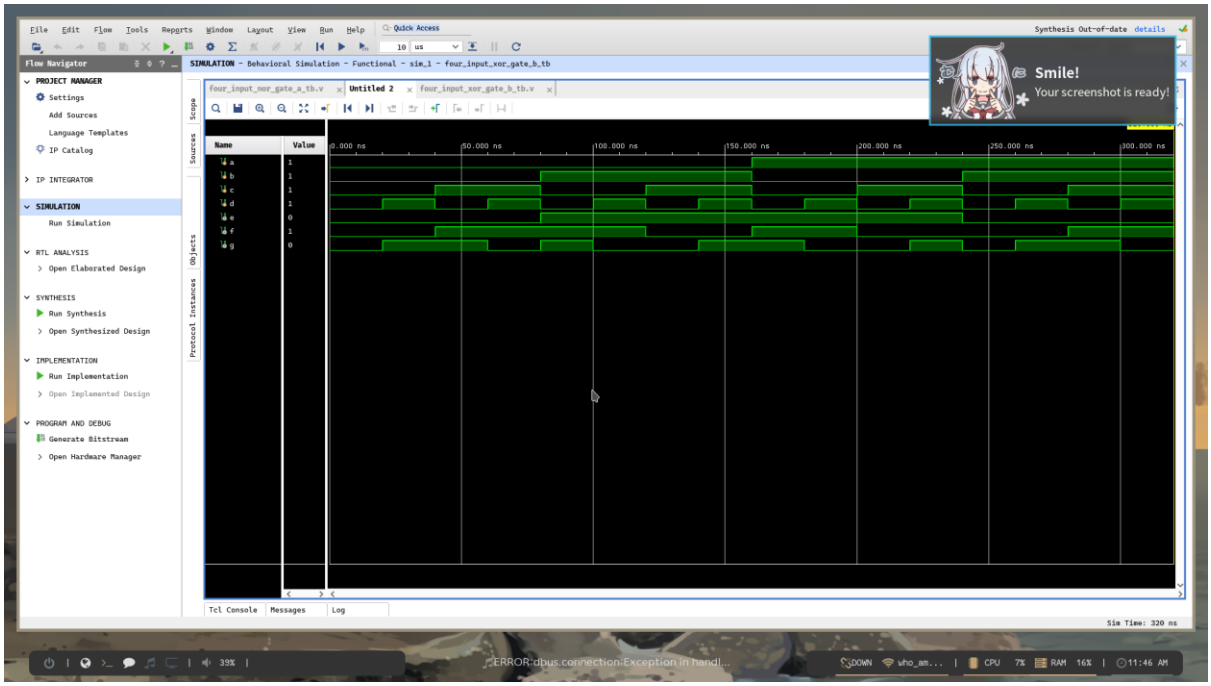
4개의 입력을 갖는 NOR 회로 역시 마찬가지로 (A)의 회로와 (B)의 회로가 서로 같다고 볼 수 없다. NOR 게이트를 테스트하는 코드는 NAND 게이트와 전혀 차이가 없는 코드였으나, 위에서 알 수 있듯이 NAND의 결과를 "반전"한 듯한 느낌을 준다.

또, NAND 게이트와 OR 게이트, NOR 게이트와 AND 게이트의 출력 결과를 비교하여 드모르간의 정리를 눈으로 직접 확인할 수 있었다.

4. 4-input XOR gate의 simulation 결과 및 과정에 대해서 설명하시오.

Input A	Input B	Input C	Input D	Output E	Output F	Output G
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	0
0	1	0	0	1	1	1
0	1	0	1	1	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	0
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	0	0
1	1	0	1	0	0	1
1	1	1	0	0	1	1
1	1	1	1	0	1	0

4개의 입력을 갖는 XOR 게이트의 진리표



4개의 입력을 갖는 XOR 게이트의 실행 결과

XOR 게이트는 기본적으로 2개의 입력을 갖는, 즉 “binary” 연산자이다. 만약 입력이 두 개일 때에는, 두 입력의 논리값이 **다를 때**에만 논리적 참을 출력한다. 그러나 이렇게 하나의 XOR 게이트가 여러 개의 입력값을 가지게 하면 XOR 게이트의 정의는 다음과 같이 확장된다.

“입력 중 논리적 참의 개수가 홀수이면 참을 출력한다.”

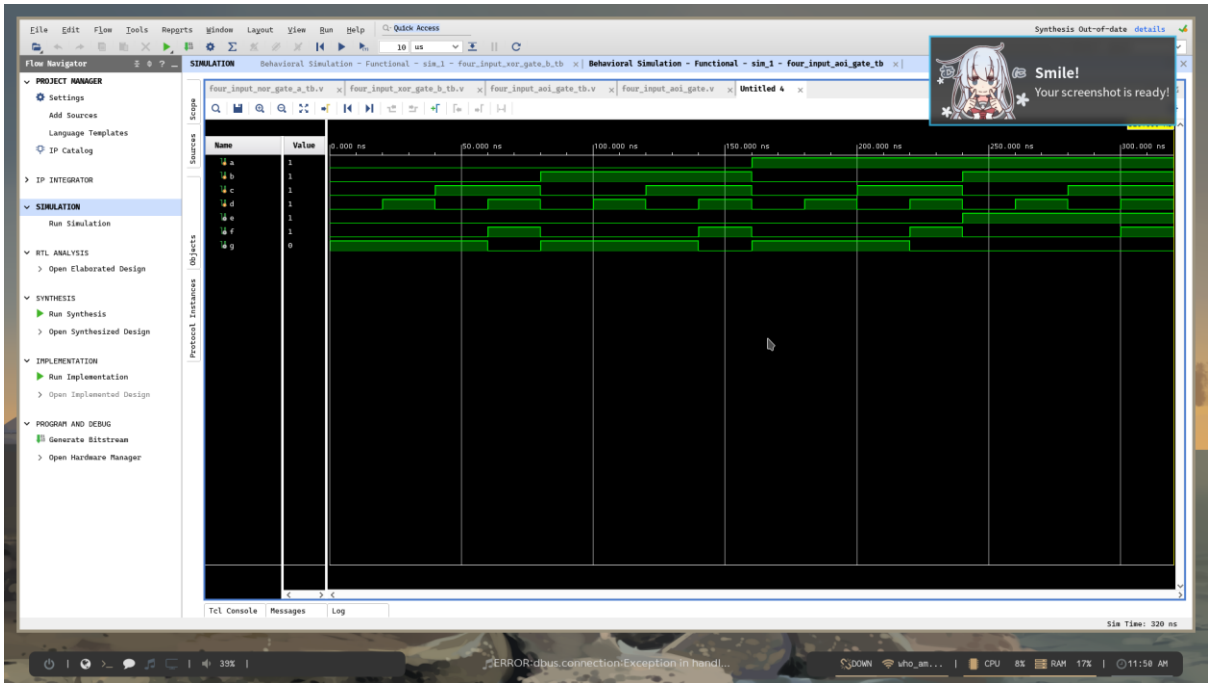
FPGA 시뮬레이션은 이 정의의 확장을 아주 잘 보여주는 예시라고 할 수 있다.

5. 4-input AOI gate의 simulation 결과 및 과정에 대해서 설명하시오.

Input A	Input B	Input C	Input D	Output E	Output F	Output G
0	0	0	0	0	0	1
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	0	1
0	1	1	0	0	0	1
0	1	1	1	0	1	0
1	0	0	0	0	0	1

1	0	0	1	0	0	1
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	1	0

4개의 입력을 갖는 AOI 게이트의 진리표



4개의 입력을 갖는 AOI 게이트의 실행 결과

AOI는 a, b가 모두 참이거나, c, d가 모두 거짓이면 논리적 거짓을 반환하는 논리 게이트이다. 비록 진리표와 신호 그래프를 보면 유용성이 바로 보이지 않지만, AOI 게이트를 사용하면 NAND 논리 회로나 NOR 논리 회로보다 적은 양의 트랜지스터를 사용해서 같은 논리 회로를 구성할 수 있다는 장점이 있다.

그러나, 우리가 사용하는 FPGA는 트랜지스터를 사용하여 논리회로를 구성하지 않고, LUT 유닛을 프로그래밍하여 논리 회로를 구성하므로 이 경우에는 일반 게이트를 사용하는 것과 같은 양의 LUT가 사용되어 효율성에 큰 차이가 없다.

6. 결과 검토 및 논의사항

지금까지 여러 논리 게이트를 살펴보았는데, OR, AND, XOR과 같은 기본 논리 게이트는 교환법칙과 결합법칙이 성립하여 입력의 순서를 바꾸어도 같은 결과를 출력한다는 것을 알 수 있었다. 반대로, NOT 인버터를 사용하여 출력을 반전시키는 게이트인 NAND, NOR, AOI 게이트와 같은 경우는 교환법칙 및 결합법칙이 성립하지 않아 “게이트의 정의”를 확실하게 하여야 한다.

그 첫 번째 예시로 NAND 게이트의 (A)의 경우는 모든 입력이 논리적 참일 때에만 논리적 거짓을 출력하는 비교적 간단한 진리표를 볼 수 있었지만, (B)의 경우는 (A)와 달리 매우 복잡한 진리표와 신호 그래프를 관찰할 수 있었다. NOR 게이트 역시 (A)의 경우가 (B)의 회로보다 훨씬 간단한 진리표를 가지고 있었다.

반대로 XOR 게이트와 같은 경우는 (A)와 같이 모든 입력을 한 번에 XOR으로 묶는 것과 (B)와 같이 두 개씩 따로따로 분리하여 XOR하는 것 사이의 차이가 존재하지 않았다.

그러나 분리하여 계산하면 다른 결과값이 나온다는 것이 단점이라고 할 수만은 없다. AND, OR, XOR 게이트는 이들만을 사용하여 모든 논리 회로를 나타내는 것이 불가능하지만 NAND, NOR, AOI 게이트는 하나의 종류의 게이트만을 사용하여 모든 논리회로를 나타낼 수 있는 “범용” 게이트이기 때문이다.

7. 추가 이론 조사 및 작성

OR, AND 게이트가 NOR, NAND와 같이 자기 자신을 반전한 게이트를 가지고 있듯, XOR 게이트 역시 자기 자신을 반전한 XNOR 게이트가 존재한다. XNOR 게이트는 두 입력이 같으면 논리적 참을, 그렇지 않으면 논리적 거짓을 출력하는 게이트이다. XOR 게이트는 두 입력이 다르면 논리적 참을 반환하기에 OR과 비슷하다 하여 연산자는 $\oplus$를 사용하며, XNOR 게이트는 두 입력이 같으면 논리적 참을 반환하기에 AND와 비슷하여 연산자는 $\odot$를 사용한다.