5주차 결과보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20212022 이름: 이예준

**1.**

De-Morgan법칙과 Boolean대수 기본정리에 대한 동작을 이해하고 확인한다.

Verilog를 이용해 De-Morgan법칙과 Boolean대수 기본정리를 코드로 구현하고,

Simulation 결과를 확인하고, 그 코드를 FPGA보드에 적용하여 회로 동작을 확인한다.

**2.**

De-Morgan제 1법칙은

변수들의 곱의 NOT이 각각의 변수들의 NOT의 합과 같다는 것이다.

식은이다.

스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

\*좌변식의 Simulation 결과

스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

\*우변식의 Simulation 결과

좌변식과 우변식의 Simulation 결과 A와 B가 모두 1인 경우를 제외하면

모두 결과값이 1인 것을 보면서 De-Morgan제 1법칙이 성립하는 것을 볼 수 있다.

특히 좌변식은 NAND의 식과 결과값 모두 일치하는 것을 볼 수 있다.

De-Morgan제 2법칙은

변수들의 합의 NOT이 각각의 변수들의 NOT의 곱과 같다는 것이다.

스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명식은이다.

스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명\*좌변식의 Simulation 결과

\*우변식의 Simulation 결과

좌변식과 우변식의 Simulation 결과 A와 B가 모두 0인 경우를 제외하면

모두 결과값이 0인 것을 보면서 De-Morgan제 2법칙이 성립하는 것을 볼 수 있다.

특히 좌변식은 NOR 의 식과 결과값 모두 일치하는 것을 볼 수 있다.

**3.**

-

두 Boolean Function식이 주어질 때

두 식이 같다는 것을 보이기 위해 Simulation을 통해 비교해 본다.

스크린샷, 텍스트, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

\*좌변식의 Simulation 결과

스크린샷, 텍스트, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

\*우변식의 Simulation 결과

겉으로 보기에는 두 식이 다르게 보이지만 Simulation을 보게 되면

두 식이 서로 결과가 같은 것을 볼 수 있다.

1. 좌변식을 보게 되면 A의 NOT과 B의 NOT이 OR연산을 하게 되고

그 결과값과 C의 NOT이 서로 AND연산한다.

**A와 B 둘 중 하나라도 0이 나오고 C가 0인 경우에만 결과값이 1이 나오고,**

**A와 B 둘 다 1이 나오거나 C가 1이 되는 경우에는 언제나 결과값이 0이 나온다.**

2. 우변식을 보게 되면 A와 B가 AND연산을 하게 되고 그 결과값과 C를 서로

OR연산한 뒤 NOT을 취해준다. 즉, A와 B의 연산값과 C를 NOR연산하는 것과 같다.

**A와 B 둘 중 하나라도 0이 나오고 C가 0인 경우에만 결과값이 1이 나오고,**

**A와 B 둘 다 1이 나오거나 C가 1이 되는 경우에는 언제나 결과값이 0이 나온다.**

따라서 두 식이 일치하다는 것을 확인할 수 있었다.

-

두 Boolean Function식이 주어질 때

두 식이 같다는 것을 보이기 위해 Simulation을 통해 비교해 본다.

스크린샷, 소프트웨어, 텍스트, 디스플레이이(가) 표시된 사진

자동 생성된 설명

\*좌변식의 Simulation 결과

스크린샷, 텍스트, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*우변식의 Simulation 결과

겉으로 보기에는 두 식이 다르게 보이지만 Simulation을 보게 되면

두 식이 서로 결과가 같은 것을 볼 수 있다.

1. 좌변식을 보게 되면 A의 NOT과 B의 NOT이 AND연산을 하게 되고

그 결과값과 C의 NOT이 서로 OR연산한다.

**A와 B 둘 다 0이 나오거나 C가 0인 경우에는 결과값이 1이 나오고,**

**A와 B 둘 중 하나라도 1이 나오고 C가 1이 되는 경우에만 결과값이 0이 나온다.**

2. 우변식을 보게 되면 A와 B가 OR연산을 하게 되고 그 결과값과 C를 서로

AND연산한 뒤 NOT을 취해준다. 즉, A와 B의 연산값과 C를 NAND연산하는 것과 같다.

**A와 B 둘 다 0이 나오거나 C가 0인 경우에는 결과값이 1이 나오고,**

**A와 B 둘 중 하나라도 1이 나오고 C가 1이 되는 경우에만 결과값이 0이 나온다.**

따라서 두 식이 일치하다는 것을 확인할 수 있었다.

**4.**

1-Bit 비교기는 A와 B 두개의 input을 받아 4개의 output을 낸다.

output 4개는 각 C, D, E, F이며, A==B, A!=B, A>B, A<B 비교를 한 결과값이다.

C는 A와 B가 서로 같을 때만 1을 반환하고 그 외에는 0을 반환한다.

D는 A와 B가 서로 다를 때만 1을 반환하고 그 외에는 0을 반환한다.

E는 A와 B 중 A가 더 클 때만 1을 반환하고 그 외에는 0을 반환한다.

F는 A와 B 중 B가 더 클 때만 1을 반환하고 그 외에는 0을 반환한다.

아래는 각각 1-Bit 비교기의 Simulation결과와 진리표이다.

스크린샷, 텍스트, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

\*1-Bit 비교기 Simulation 결과

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | A=B (C) | A≠B (D) | A>B (E) | A<B (F) |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |

\*1-Bit 비교기 진리표

C는 A와 B가 서로 같을 때만 1을 반환하는 XNOR연산을 하고,

D는 A와 B가 서로 다를 때만 1을 반환하는 XOR연산을 한다.

E는 A가 1이고 B가 0일 때만 1을 반환하고 그 외에는 0을 반환하며,

F는 A가 0이고 B가 1일 때만 1을 반환하고 그 외에는 0을 반환한다.

**5.**

De-Morgan 제 1법칙, 2법칙에 대해 Verilog 코드를 통한Simulation 결과와 FPGA보드 구현 결과를 봤을 때 해당 법칙들이 성립함을 확인할 수 있었다.

또한 Boolean Function 식이 De-Morgan 법칙에 의해 바뀌었을 때 결과는 그대로 유지 된다는 것도 Simulation을 통해 확인할 수 있었다.

마지막으로 1-Bit 비교기에서 각 비트들을 비교하는 연산들이 어떤 논리연산 과정을 거쳤는지에 대해서도 알아봤다.

논의 사항으로 De-Morgan법칙이 전자회로에서 어떻게 쓰이는지에 대해 탐구해본다.

전자회로는 일반적으로 회로식을 SOP 형식(곱들의 합)으로 구성한다.

그런데 De-Morgan법칙을 이용하면 NAND 게이트만으로도 구성할 수 있다.

예를 들어  라고 할 때, De-Morgan법칙을 이용해서 변환하면

로 바꿔 쓸 수 있는데, 이는 NAND 게이트만으로

회로 구성이 가능하다는 것을 의미한다. 전자회로에서 NAND 게이트는 구조가 가장 간단하고 입력이 많아져도 큰 지장이 없기에 가장 널리 쓰인다. 이것만으로 만든 메모리가 바로 우리가 일반적으로 부르는 낸드 [플래시 메모리](https://namu.wiki/w/%ED%94%8C%EB%9E%98%EC%8B%9C%20%EB%A9%94%EB%AA%A8%EB%A6%AC)(NAND flash memory)이다.

**6.**

어떤 함수가 SOP 형태나 POS 형태로 나타나게 되면 그러한 함수를 표준형이라고 한다.

예를 들어 F=AB+BC와 같이 나타나면 곱들이 합해진 형태이기 때문에 SOP가 되고

 F=(A+B)⋅(B+C) 와 같이 나타나면 합들이 곱해진 형태이기 때문에 POS이다.

이렇게 SOP, POS로 나타나지 않는 경우는 표준형이 아니다.

\*SOP = Sum Of Product (곱들의 합)

\*POS = Product Of Sum (합들의 곱)