11주차 결과보고서

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

**1.**

**-RS-Flipflop(NOR)**

스크린샷, 텍스트, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NOR) Simulation 결과

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명 스크린샷, 텍스트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NOR) Verilog 코드

블랙, 어둠이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NOR) 회로도

위의 Verilog 코드를 보게 되면 구현한 Flipflop이 RS-Flipflop이며, AND gate와

NOR gate를 이용하여 구현했다는 것을 알 수 있다. 또한 위의 Simulation 결과와 회로도를

토대로 진리표를 만들면 아래와 같고, 진리표 외에도 현재 상태와 입력값이 주어졌을 때

다음 상태를 알려주는 상태표(Characteristic Table)도 존재한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output** |
| E | S | R | Qt+1 |
| 0 | X | X | Qt |
| 1 | 0 | 0 | Qt |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 불가 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output** |
| Qt | S | R | Qt+1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 불가 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 불가 |

\*RS-Flipflop(NOR) 진리표

\*RS-Flipflop(NOR) 상태표

**-Clock Pulse가 0일 때**

Simulation 결과를 보게 되면 E(Clock Pulse)가 0일 때, 현재 상태를 그대로 유지하고

있는 것을 볼 수 있다. 그 이유는 코드와 회로도를 보게 되면 알 수 있는데, 먼저 코드를

보게 되면 E가 0일 때 R 또는 S와 AND 연산하면 0이 되게 되고 그 0과 OR연산하게

되면 그 0과 연산하는 값이 그대로 나오게 된다. 이때 Q값은 Q’값이 OR연산을 하며,

Q’값은 Q값이 OR연산을 하고 마지막에 NOT연산을 해주기 때문에 결국 Q는 Q 그대로,

Q’은 Q’ 그대로 나오게 된다. 회로도를 봐도 마찬가지인데 E가 0이면 앞에 있는

두 AND gate는 모두 출력 0이 나온다. 이때 Q값이 1이면 위에 있는 NOR gate는 Q’값과

AND gate에서 나온 0을 입력으로 받아 1이 나오게 되고, 밑에 있는 NOR gate는 Q값과

AND gate에서 나온 0을 입력으로 받아 0이 나오게 된다. 즉, 현재 Q값과 다음 Q값이

같은 것을 볼 수 있다. 반대도 마찬가지이다.

**-Clock Pulse가 1일 때**

E(Clock Pulse)가 1이라면 그때는 R과 S에 따라서 결과가 다르게 나오는데 R과 S가

모두 0이면 위의 경우와 똑같기 때문에 코드에서 E와 AND연산하면 똑같이 0이 나와서

OR연산하는 값의 NOT이 결과로 나오고, 회로도에서도 똑같이 AND gate에서는 모든

출력이0이 나오기 때문에 현재상태를 그대로 유지한다.

만약 R만 1이라면 코드에서는 Q를 나타내는 식에서 Q’값에 상관없이 언제나 0이 나오게

되고, Q’를 나타내는 식에서는 S가 0이여서 AND연산하면 0이 나오고, OR연산을 해도

Q값이 0이기 때문에 0이 나온다. 따라서 마지막에 NOT연산을 하게 되면 1이 최종적으로

Q’값이 된다.

회로도를 통해 과정을 살펴보면 R이 1이기 때문에 위에 있는 AND gate는 1이 나오게

되고, S는 0이기 때문에 아래에 있는 AND gate는 0이 나오게 된다. 따라서 위에 있는

NOR gate는 Q’가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내고, 그 0과 아래에

있는 AND gate에서 나온 1을 입력으로 받은 아래에 있는 NOR gate는 출력을 1로

내보내게 된다.

만약 S만 1이라면 코드에서는 Q’를 나타내는 식에서 Q값에 상관없이 언제나 0이 나오게 되고, Q를 나타내는 식에서는 R이 0이여서 AND연산하면 0이 나오고, OR연산을 해도 Q’값이 0이기 때문에 0이 나온다. 따라서 마지막에 NOT연산을 하게 되면 1이 최종적으로 Q값이 된다.

회로도를 통해 과정을 살펴보면 R이 0이기 때문에 위에 있는 AND gate는 0이 나오게

되고, S는 1이기 때문에 아래에 있는 AND gate는 1이 나오게 된다. 따라서 아래에 있는

NOR gate는 Q가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내고, 그 0과 위에

있는 AND gate에서 나온 0을 입력으로 받은 위에 있는 NOR gate는 출력을 1로

내보내게 된다.

그러나 S와 R이 모두 1이라면 코드에서는 Q를 나타내는 식은 Q’값에 상관없이 언제나

0이 나오게 되고, Q’를 나타내는 식도 Q값에 상관없이 언제나 0이 나오게 된다. 따라서

Q값과 Q’값이 서로 보수관계를 갖지 않기 때문에 Flipflop이 정상적인 작동을 할 수 없다.

회로도를 통해 과정을 살펴보면 R이 1이기 때문에 위에 있는 AND gate는 1이 나오게

되고, S도 1이기 때문에 아래에 있는 AND gate도 1이 나오게 된다. 따라서 아래에 있는

NOR gate는 Q가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내고, 위에 있는

NOR gate도 Q가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내기 때문에

위에서 설명한대로 유효한 경우가 될 수 없다.

**-RS-Flipflop(NAND)**

스크린샷, 텍스트, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NAND) Simulation 결과

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NAND) Verilog 코드

도표, 라인, 폰트, 평면도이(가) 표시된 사진

자동 생성된 설명

\*RS-Flipflop(NAND) 회로도

위의 Verilog 코드를 보게 되면 구현한 Flipflop이 RS-Flipflop이며, NAND gate 4개를

이용하여 구현했다는 것을 알 수 있다. 또한 위의 Simulation 결과와 회로도를

토대로 진리표를 만들면 아래와 같고, 진리표 외에도 현재 상태와 입력값이 주어졌을 때

다음 상태를 알려주는 상태표(Characteristic Table)도 존재한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output** |
| E | S | R | Qt+1 |
| 0 | X | X | Qt |
| 1 | 0 | 0 | Qt |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 불가 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output** |
| Qt | S | R | Qt+1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 불가 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 불가 |

\*RS-Flipflop(NAND) 진리표

\*RS-Flipflop(NAND) 상태표

**-Clock Pulse가 0일 때**

Simulation 결과를 보게 되면 E(Clock Pulse)가 0일 때, 현재 상태를 그대로 유지하고

있는 것을 볼 수 있다. 그 이유는 코드와 회로도를 보게 되면 알 수 있는데, 먼저 코드를

보게 되면 E가 0일 때 R 또는 S와 AND 연산하면 0이 되게 되고 그 0을 NOT연산하여

1이 나오게 된다. 그 1과 Q또는 Q’과 또 NAND(AND+NOT)연산을 해주는데, 결과는 1과

NAND연산하는 값 그대로 나온다. 이때 Q값은 Q’값이 NAND연산을 하며, Q’값은 Q값이

NAND연산을 하기 때문에 결국 Q는 Q 그대로, Q’은 Q’ 그대로 나오게 된다. 회로도를 봐도

마찬가지인데 E가 0이면 앞에 있는 두 NAND gate는 모두 출력 1이 나온다. 이때 Q값이

1이면 뒤쪽 위에 있는 NAND gate는 Q’값과 NAND gate에서 나온 1을 입력으로 받아 1이

나오게 되고, 뒤쪽 밑에 있는 NAND gate는 Q값과 NAND gate에서 나온 1을 입력으로

받아 0이 나오게 된다. 즉, 현재 Q값과 다음 Q값이 같은 것을 볼 수 있다.

반대도 마찬가지이다.

**-Clock Pulse가 1일 때**

E(Clock Pulse)가 1이라면 그때는 R과 S에 따라서 결과가 다르게 나오는데 R과 S가

모두 0이면 위의 경우와 똑같기 때문에 코드에서 E와 NAND연산하면 똑같이 1이 나와서

NAND연산하는 값이 결과로 나오고, 회로도에서도 똑같이 NAND gate에서는 모든 출력이 1이 나오기 때문에 현재상태를 그대로 유지한다.

만약 R만 1이라면 코드에서는 Q’를 나타내는 식에서 Q값에 상관없이 언제나 1이 나오게

되고, Q를 나타내는 식에서는 S가 0이여서 NAND연산하면 1이 나오고, 그 1과 Q’값 1을

다시 NAND 연산을 하면 Q값이 0이 나온다.

회로도를 통해 과정을 살펴보면 R이 1이기 때문에 앞쪽 위에 있는 NAND gate는 0이

나오게 되고, S는 0이기 때문에 앞쪽 아래에 있는 NAND gate는 1이 나오게 된다. 따라서

뒤쪽 위에 있는 NAND gate는 Q가 0이든 1이든 상관없이 언제나 1을 출력으로 내보내고,

그 1과 앞쪽 아래에 있는 NAND gate에서 나온 1을 입력으로 받은 뒤쪽 아래에 있는

NAND gate는 출력을 0으로 내보내게 된다.

만약 S만 1이라면 코드에서는 Q를 나타내는 식에서 Q’값에 상관없이 언제나 1이 나오게

되고, Q’를 나타내는 식에서는 R이 0이여서 NAND연산하면 1이 나오고, 그 1과 Q값 1을

다시 NAND 연산을 하면 Q’값이 0이 나온다.

회로도를 통해 과정을 살펴보면 R이 0이기 때문에 앞쪽 위에 있는 NAND gate는 1이

나오게 되고, S는 1이기 때문에 앞쪽 아래에 있는 NAND gate는 0이 나오게 된다. 따라서

뒤쪽 아래에 있는 NAND gate는 Q’가 0이든 1이든 상관없이 언제나 1을 출력으로

내보내고, 그 1과 앞쪽 위에 있는 NAND gate에서 나온 1을 입력으로 받은 뒤쪽 위에 있는

NAND gate는 출력을 0으로 내보내게 된다.

그러나 S와 R이 모두 1이라면 코드에서는 Q를 나타내는 식은 Q’값에 상관없이 언제나

1이 나오게 되고, Q’를 나타내는 식도 Q값에 상관없이 언제나 1이 나오게 된다. 따라서

Q값과 Q’값이 서로 보수관계를 갖지 않기 때문에 Flipflop이 정상적인 작동을 할 수 없다.

회로도를 통해 과정을 살펴보면 R이 1이기 때문에 앞쪽 위에 있는 NAND gate는 0이

나오게 되고, S도 1이기 때문에 앞쪽 아래에 있는 NAND gate도 0이 나오게 된다. 따라서

뒤쪽 위에 있는 NAND gate는 Q’가 0이든 1이든 상관없이 언제나 1을 출력으로

내보내고, 뒤쪽 아래에 있는 NAND gate도 Q’가 0이든 1이든 상관없이 언제나 1을

출력으로 내보내기 때문에 위에서 설명한대로 유효한 경우가 될 수 없다.

**2.**

스크린샷, 텍스트, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*D-Flipflop Simulation 결과

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

\*D-Flipflop Verilog 코드

블랙, 어둠, 천체, 우주이(가) 표시된 사진

자동 생성된 설명

\*D-Flipflop 회로도

위의 Verilog 코드를 보게 되면 구현한 Flipflop이 D-Flipflop이며, NAND gate와

NOR gate를 이용하여 구현했다는 것을 알 수 있다. 또한 위의 Simulation 결과와 회로도를

토대로 진리표를 만들면 아래와 같고, 진리표 외에도 현재 상태와 입력값이 주어졌을 때

다음 상태를 알려주는 상태표(Characteristic Table)도 존재한다.

|  |  |  |
| --- | --- | --- |
| **Input** | | **Output** |
| E | D | Qt+1 |
| 0 | X | Qt |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| **Input** | | **Output** |
| Qt | D | Qt+1 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 불가 |

\*D-Flipflop 진리표

\*D-Flipflop 상태표

**-Clock Pulse가 0일 때**

Simulation 결과를 보게 되면 E(Clock Pulse)가 0일 때, 현재 상태를 그대로 유지하고

있는 것을 볼 수 있다. 그 이유는 코드와 회로도를 보게 되면 알 수 있는데, 먼저 코드를

보게 되면 E가 0일 때 D 또는 D’와 AND 연산하면 0이 되게 되고 그 0과 OR연산하게

되면 그 0과 연산하는 값이 그대로 나오게 된다. 이때 Q값은 Q’값이 OR연산을 하며,

Q’값은 Q값이 OR연산을 하고 마지막에 NOT연산을 해주기 때문에 결국 Q는 Q 그대로,

Q’은 Q’ 그대로 나오게 된다. 회로도를 봐도 마찬가지인데 E가 0이면 앞에 있는

두 AND gate는 모두 출력 0이 나온다. 이때 Q값이 1이면 위에 있는 NOR gate는 Q’값과

AND gate에서 나온 0을 입력으로 받아 1이 나오게 되고, 밑에 있는 NOR gate는 Q값과

AND gate에서 나온 0을 입력으로 받아 0이 나오게 된다. 즉, 현재 Q값과 다음 Q값이

같은 것을 볼 수 있다. 반대도 마찬가지이다.

**-Clock Pulse가 1일 때**

E(Clock Pulse)가 1이라면 그때는 R과 S에 따라서 결과가 다르게 나오는데 RS-Flipflop과 동일한 구조를 가지고 있고 입력값을 보게 되면 D와 D’이 S와 R 위치에 있는 차이점이 있다. 따라서 S가 1이고 R이 0인 경우와 S가 0이고 R이 1인 경우만 있는 RS-Flipflop과 동일하다고 볼 수 있다.

만약 D가 1이라면 코드에서는 Q’를 나타내는 식에서 D와 E가 AND연산해서 1이 나오고,

따라서 Q값에 상관없이 언제나 0이 나오게 되고, Q를 나타내는 식에서는 D’와 E가 AND연산해서 0이 나오고, 그 0과 Q’값 0을 OR연산과 NOT연산을 해서 최종적으로

Q값이 1이 된다.

회로도를 통해 과정을 살펴보면 D가 1이기 때문에 위에 있는 AND gate는 0이 나오게

되고, 아래에 있는 AND gate는 1이 나오게 된다. 따라서 아래에 있는 NOR gate는 Q가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내고, 그 0과 위에 있는 AND gate에서 나온 0을 입력으로 받은 위에 있는 NOR gate는 출력을 1로 내보내게 된다.

만약 D가 0이라면 코드에서는 Q를 나타내는 식에서 D’와 E가 AND연산해서 1이 나오고,

따라서 Q’값에 상관없이 언제나 0이 나오게 되고, Q’를 나타내는 식에서는 D와 E가 AND연산해서 0이 나오고, 그 0과 Q값 0을 OR연산과 NOT연산을 해서 최종적으로

Q’값이 1이 된다.

회로도를 통해 과정을 살펴보면 D가 0이기 때문에 위에 있는 AND gate는 1이 나오게

되고, 아래에 있는 AND gate는 0이 나오게 된다. 따라서 위에 있는 NOR gate는 Q’가 0이든 1이든 상관없이 언제나 0을 출력으로 내보내고, 그 0과 아래에 있는 AND gate에서 나온 0을 입력으로 받은 아래에 있는 NOR gate는 출력을 1로 내보내게 된다.

**3.**

이번 실험에서 AND gate와 NOR gate로 구현한 RS-Flipflop과 NAND gate로 구현한

RS-Flipflop의 작동방식에 대해 알아봤고 각각 Verilog코드를 구현하여Simulation결과와 FPGA 보드 구현 결과가 정상적으로 나오는 것을 확인할 수 있었다.

또한 AND gate와 NOR gate로 구현한 D-Flipflop의 경우도 Verilog코드를 구현하여

Simulation결과와 FPGA 보드 구현 결과가 정상적으로 나오는 것을 확인할 수 있었다.

이번 실험의 논의사항으로는 RS-Flipflop의 한계에 대해 말하고자 한다. S와 R이 모두

1일 때 플립플롭이 오작동을 할 수 있다는 가능성은 결국 근본적으로 구조 자체가 해소될

수 없는 문제점을 갖고 있기 때문에 RS-Flipflop의 신뢰성을 떨어뜨린다. 따라서 R,S가 모두

1이여도 작동할 수 있는 JK-Flipflop에 대해 더 공부하고 연구할 필요성을 느낀다.

**4.**

플립플롭을 효율적으로 사용하기 위해서는 전력소모에 대해서도 신경 써야 한다.

플립플롭의 전력소모는 공급전압과 평균 공급 전류를 곱한 값이 되는데

예를 들어 하나의 플립플롭이 15V 전원으로 작동하고 10mA의 전류가 흐른다면,

전력소모는 가 된다. 이런 플립플롭이 20개가 연결되어 작동한다면 총 소비전력은 1.5W가 된다. 또한 이 회로에 공급되어야 할 전류는 이다.