11주차 결과보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20191583 이름: 김태곤

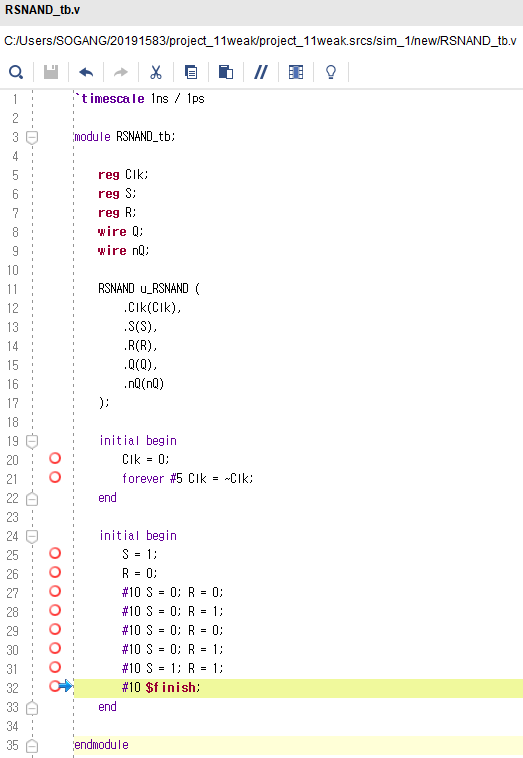
**1. RS Flip-Flop의 결과 및 Simulation 과정에 대해서 설명하시오.**

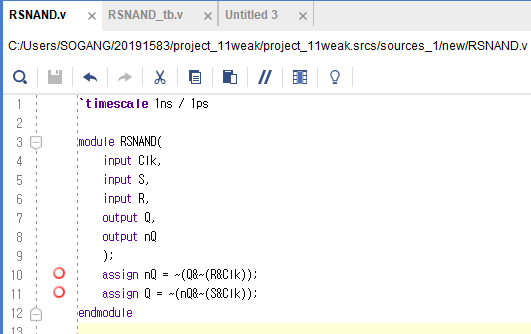
**(verilog source, simulation 출력 예시, table 등의 과정 상세히 적을것)**

**※NAND로 구성된 RS Flip-Flop과 NOR로 구성된 RS Flip-Flop 2가지 모두 작성할 것**

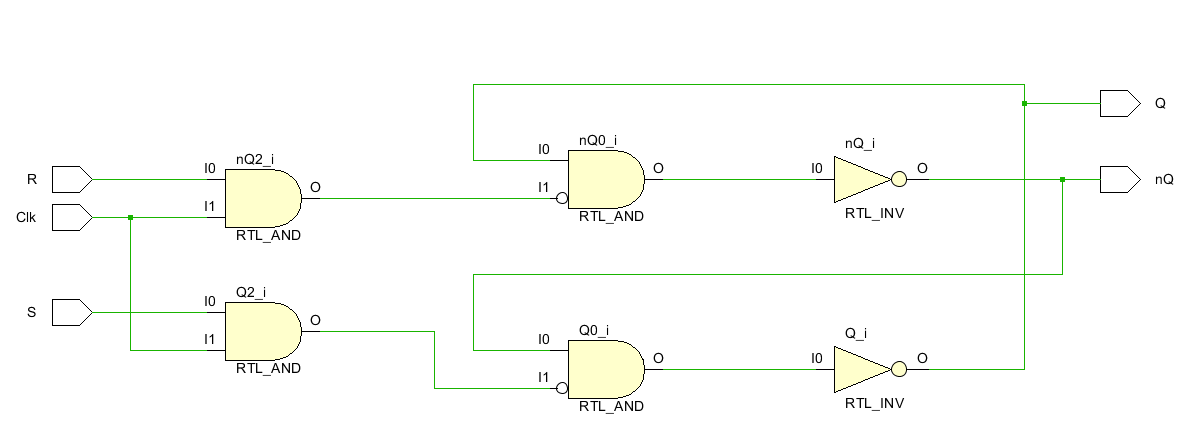
**1) NAND RS Flip-Flop**

**1-1) Source Code**

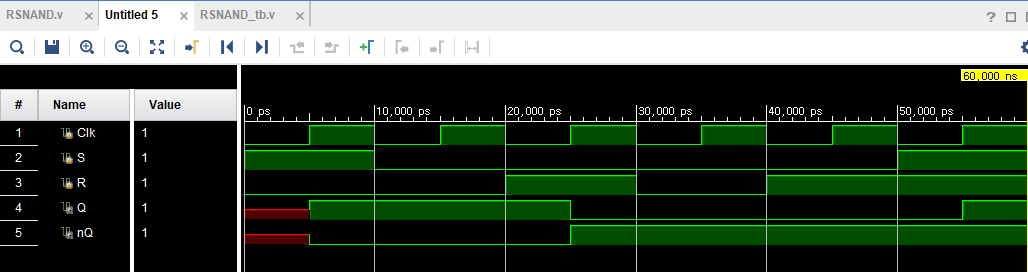
****

****

**1-2) Schematic**

****

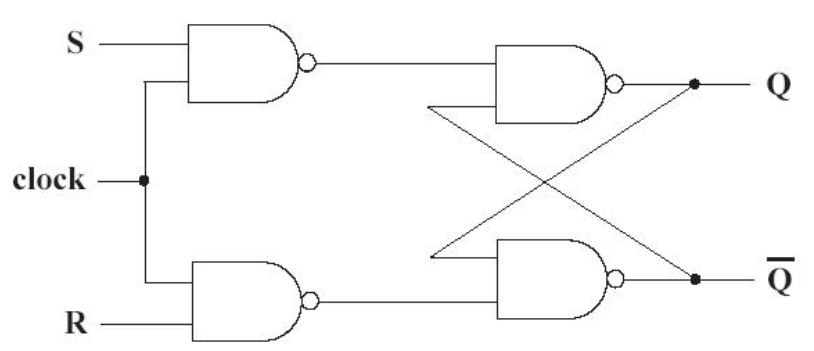
**1-3) Simulation**

****

**1-4) Truth Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | | | Output | |
| 입력 순서 | R | S | Q | ~Q |
| 1 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 1 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 1 |
| 5 | 1 | 0 | 0 | 1 |
| 6 | 1 | 1 | X | X |

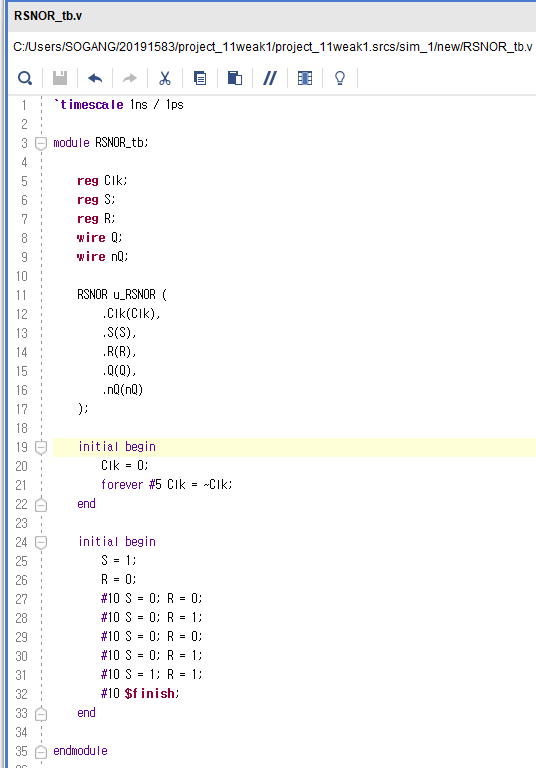
**1-5) 결과 및 과정**

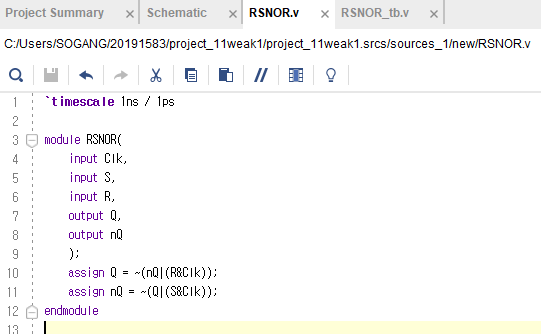
****

Design Source 코드에서는 Q는 S와 Clk의 NAND값과 nQ와의 NAND계산을 통해 나타내었고, nQ는 R과 Clk의 NAND값과 Q의 NAND계산을 통해 나타내었다. Simulation은 5ns마다 Clk의 값을 변화시켰고, 10ns마다 이번 실습에서 주어진 input값을 지정하였다. Simulation 결과 Q와 nQ의 값은 위의 표와 같은 결과가 나왔고, 제대로된 RS Flip Flop with NAND를 구현한 것을 알 수 있다. Clk가 1인 경우 R,S,Q,nQ의 값을 살펴보면 알 수 있다. 01인 경우 Q는 1로 Set되었다. 10인 경우 nQ로 Reset되었으며, 00인 경우 이전 상태를 유지하는 것을 확인할 수 있다. 11의 입력의 경우 simulation 결과 11이 나왔지만, 11은 불확정 값이므로 XX로 나타내었다.

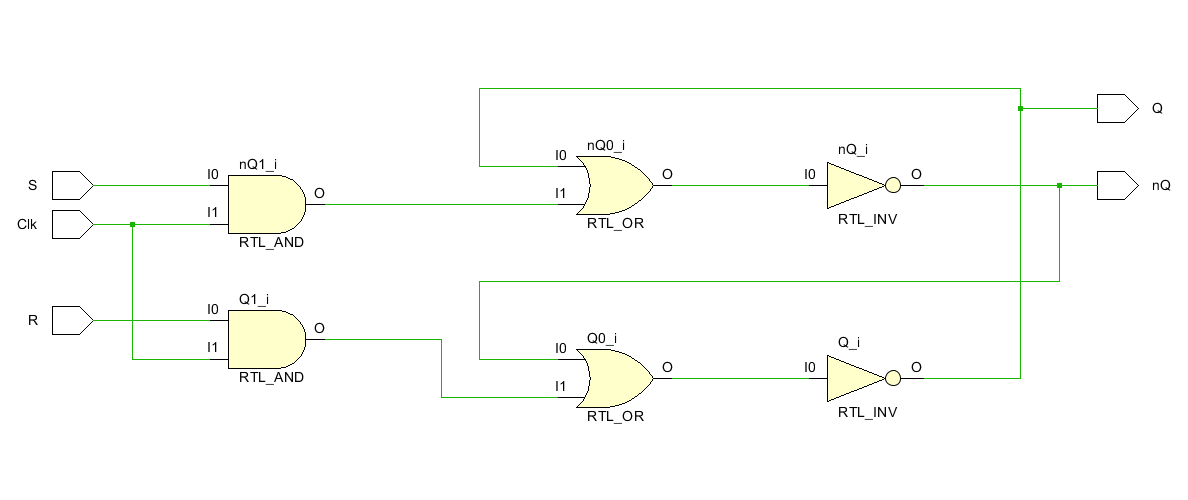
**2) NOR RS Flip-Flop**

**1-1) Source Code**

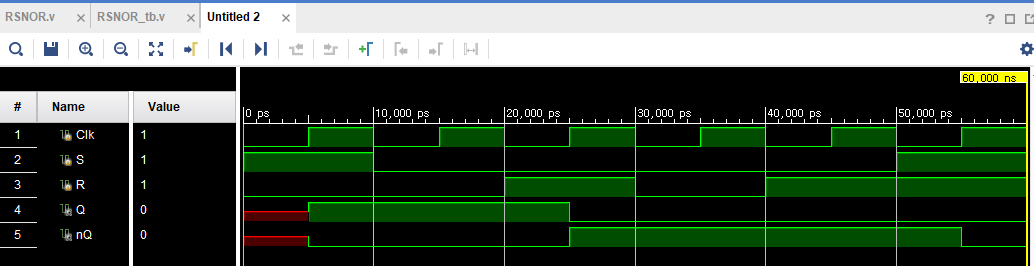
****

****

**1-2) Schematic**

****

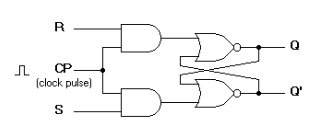
**1-3) Simulation**

****

**1-4) Truth Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | | | Output | |
| 입력 순서 | R | S | Q | ~Q |
| 1 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 1 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 1 |
| 5 | 1 | 0 | 0 | 1 |
| 6 | 1 | 1 | X | X |

**1-5) 결과 및 과정**

****

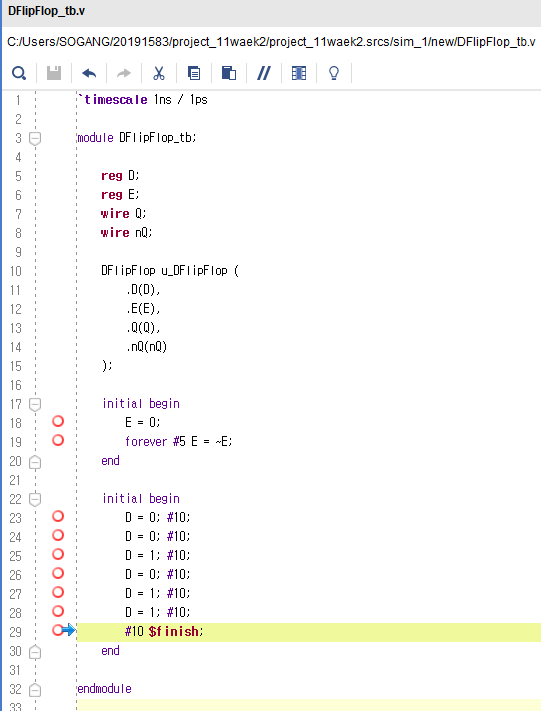
Design Source 코드에서는 Q는 R과 Clk의 AND값과 nQ와의 NOR계산을 통해 나타내었고, nQ는 S과 Clk의 AND값과 Q의 NOR계산을 통해 나타내었다. Simulation은 5ns마다 Clk의 값을 변화시켰고, 10ns마다 이번 실습에서 주어진 input값을 지정하였다. Simulation 결과 Q와 nQ의 값은 위의 표와 같은 결과가 나왔고, 제대로된 RS Flip Flop with NOR를 구현한 것을 알 수 있다. Clk가 1인 경우 R,S,Q,nQ의 값을 살펴보면 알 수 있다. NOR 게이트로 구현한 RS Flip Flop과 NAND 게이트로 구현한 RS Flip Flop 모두 같은 결과값이 나온 것을 확인할 수 있다. 이는 결과적으로 같은 논리식을 NAND로 표현하냐, NOR 표현하냐의 차이일 뿐이다. 11의 입력의 경우 simulation 결과 11이 나왔지만, 11은 불확정 값이므로 XX로 나타내었다.

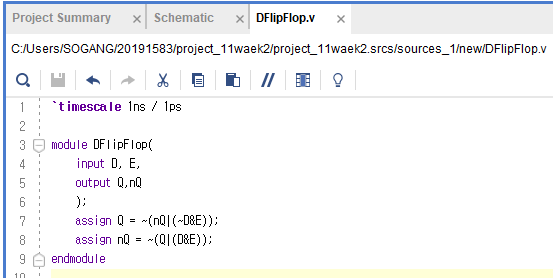
NAND와 NOR 게이트를 통해 구현한 RS Flip Flop을 통해 RS Flip Flop의 동작 원리를 알 수 있다. 먼저 R은 Reset, S는 Set, Q는 현재 상태, nQ는 현재 상태의 반대값을 나타낸다. 이때 R=0, S=1이면 Set 상태가 돈다. 이 경우 Q는1로 설정되고 nQ는 0이 된다. R=0, S=0일때는 이전 상태를 유지한다. Q가 1이었다면 계속 1이고, 0이었다면 계속 0을 유지한다. R=1, S=0일 때는 Reset 상태가 된다. 이때 Q는 0이되고, nQ는 1이된다. 마지막으로 R=1, S=1일 때는 RS 플립플롭에서는 허용되지 않는 입력값이다. 즉 이론적으로 불가능한 상태이다. 이처럼 위의 simulation을 통해 이와 같은 RS Flip Flop의 동작 원리에 대해 알아볼 수 있다.

**2. D Flip-Flop 의 결과 및 Simulation 과정에 대해서 설명하시오.**

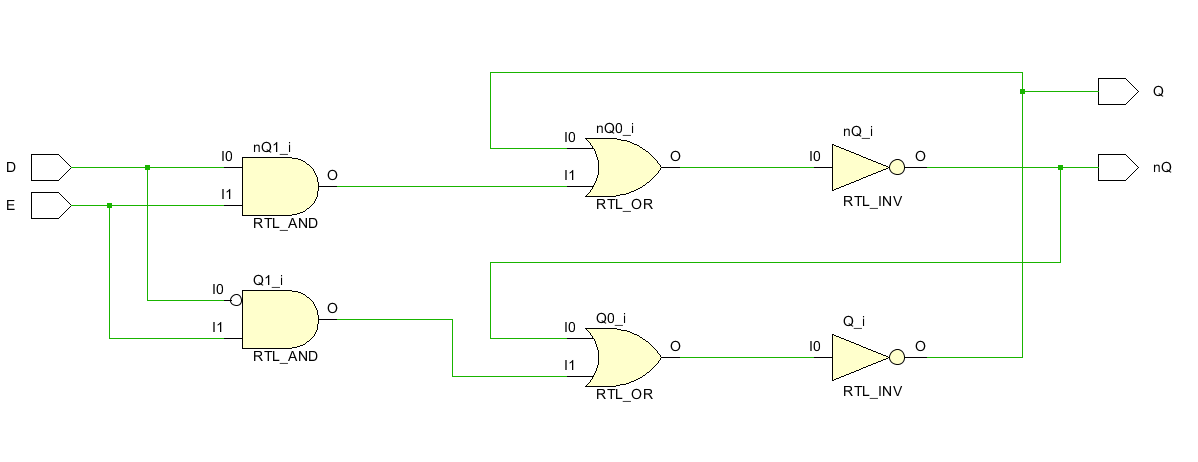
**(verilog source, simulation 출력 예시, table 등의 과정 상세히 적을것)**

1. **Source Code**

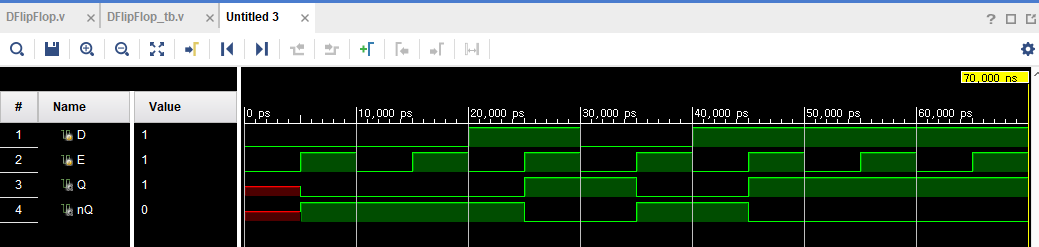
****

****

1. **Schematic**

****

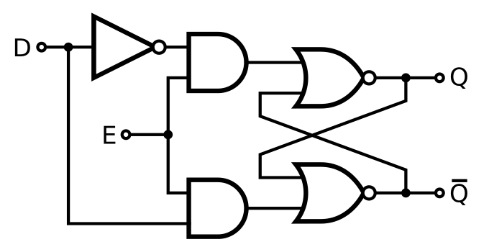
1. **Simulation**

****

1. **Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Input | | Output | |
| 입력 순서 | D | Q | ~Q |
| 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 |
| 3 | 1 | 1 | 0 |
| 4 | 0 | 0 | 1 |
| 5 | 1 | 1 | 0 |
| 6 | 1 | 1 | 0 |

1. **결과 및 과정**

****

Design Source 코드에서는 Q는 D'과 E의 AND값과 nQ와의 NOR계산을 통해 나타내었고, nQ는 D와 E의 AND값과 Q의 NOR계산을 통해 나타내었다. Simulation은 5ns마다 E의 값을 변화시켰고, 10ns마다 이번 실습에서 주어진 input값을 지정하였다. Simulation 결과 Q와 nQ의 값은 위의 표와 같은 결과가 나왔고, 제대로된 D Flip Flop을 구현한 것을 알 수 있다. E가 1인 경우 R,S,Q,nQ의 값을 살펴보면 알 수 있다. E는 Enable 신호로, 이 신호가 활성화되어 있을 때만 데이터가 입력 D에서 출력 Q로 전송된다. D Flip Flop의 동작원리는 간단하다. E신호가 활성화 되어있을 때, Q의 출력값은 D의 입력값에 따라 간다. Simulation 결과에서 E가 활성화 되어있을 때를 살펴보면 Q의 값은 D의 값과 동일함을 알 수 있다.

**3. 결과 검토 및 논의 사항.**

RS Flip Flop과 D Flip Flop의 동작 원리를 게이트 구현 방식으로 이해하는 것은 디지털 회로 설계에서 중요하다. 각 플립플롭은 디지털 회로에서 핵심적인 역할을 수행하며, 그들의 동작은 입력 신호의 조합과 사용된 게이트의 유형에 따라 달라진다.

RS Flip Flop의 경우, NAND와 NOR 게이트를 통한 구현에서 비슷한 동작 패턴을 보인다. Set 동작은 R이 0이고 S가 1일 때 Q를 1로 설정하고, nQ를 0으로 만든다. Reset 동작은 R이 1이고 S가 0일 때 Q를 0으로 설정하고, nQ를 1로 만든다. 두 입력이 모두 0인 경우 플립플롭은 이전 상태를 유지하며, R과 S가 모두 1일 때는 불확정 값인 XX를 출력하여 입력이 허용되지 않음을 나타낸다.

D Flip Flop은 Enable 신호(E)가 중요한 역할을 한다. E가 활성화된 상태에서 D의 값에 따라 Q의 출력이 결정되며, nQ는 Q의 보수 값이다. E가 1일 때, D의 입력이 직접 Q로 전송되며, E가 0일 때는 Q와 nQ가 이전 상태를 유지한다.

이러한 동작 원리는 각 플립플롭이 다른 디지털 회로 설계에 어떻게 적용될 수 있는지를 이해하는 데 도움을 준다. RS Flip Flop은 간단한 메모리 요소로 사용될 수 있으며, D Flip Flop은 데이터 버스에서 데이터의 동기화와 버퍼링에 이상적이다.

종합적으로, RS와 D Flip Flop의 동작 원리를 이해하고 각각의 장점을 적절히 활용하면, 효율적이고 안정적인 디지털 시스템 설계가 가능하다. 회로의 요구 사항과 환경에 따라 최적의 플립플롭 유형을 선택하는 것이 중요하며, 이는 디지털 회로 설계의 성공에 결정적인 요소가 될 수 있다.

**4. 추가 이론 조사 및 작성.**

Flip Flop을 사용하여 신호 무결성을 보장하는 방법은 통신 시스템에서 매우 중요하다. 이는 신호의 지터(jitter)와 노이즈가 데이터의 정확성과 신뢰성에 미치는 영향을 최소화하는 데 핵심적인 역할을 한다.

지터(Jitter)는 신호가 예상 시간에 정확히 도착하지 않고 약간의 시간적 변동이 있는 현상을 말한다. 지터는 데이터 비트가 언제 도착할지 예측하기 어렵게 만들어 데이터 통신의 정확성을 떨어뜨릴 수 있다. 노이즈(Noise)는 외부 또는 내부 소스로부터 발생하는 원하지 않는 전기적 신호로, 원래 데이터 신호의 해석을 방해하고 오류를 일으킬 수 있다.

Flip Flop을 이용한 신호 무결성 보장 방법에는 여러가지가 있다.

1. Clock Synchronization

Flip Flop은 클록 입력에 따라 데이터 입력을 캡쳐하고 저장한다. 클록 신호의 정확한 타이밍을 사용하여 데이터 비트를 정확한 시점에 캡쳐함으로써 지터의 영향을 줄일 수 있다. 이는 입력 데이터의 변동이 클록의 엣지에 정확히 맞춰질 때 최대 효과를 발휘한다.

1. Data Latching

데이터 레칭은 데이터 신호가 Flip Flop에 도달한 후 일정 시간 동안 그 값을 유지하는 과정이다. 이 방법은 신호가 다음 클록 사이클 동안 안정적인 상태로 유지되도록 보장하여, 노이즈나 다른 시간적 변동이 데이터 전송 중에 발생하는 오류를 줄일 수 있다.

1. Clock Buffering

클록 신호를 여러 Flip Flop에 분배하기 전에 클록 버퍼를 사용하여 클록 신호를 강화하고 안정화시킬 수 있다. 이는 클록 신호가 여러 Flip Flop에 동시에 도달하도록 보장하고, 각 Flip Flop이 동시에 동작하도록 함으로써 전체적인 시스템의 신호 무결성을 향상시킨다.

1. Metastability Management

메타스테빌리티는 입력 신호가 클록의 엣지에 매우 근접할 때 발생할 수 있는 현상으로, Flip Flop 출력이 불확정 상태에 머무를 수 있다. 이를 관리하기 위해 디자이너들은 Flip Flop의 셋업 및 홀드 타임을 최적화하여 입력 신호가 불확정 상태에 머물 가능성을 줄인다.

이러한 기법들을 통해 설계자들은 신호 무결성을 개선하고, 데이터 전송 및 처리 과정에서의 오류를 크게 줄일 수 있다.