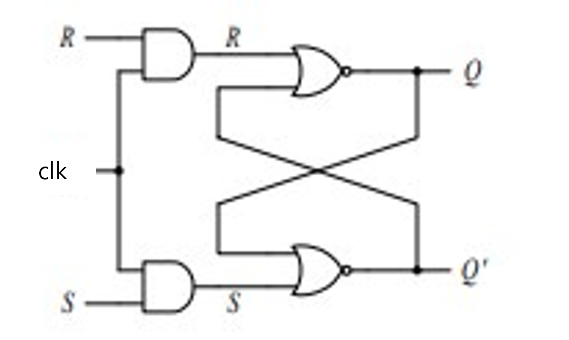
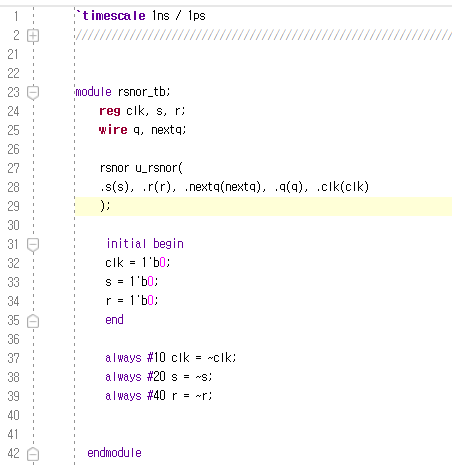
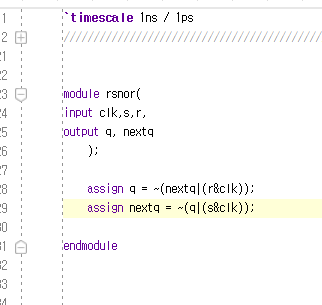
11주차 결과보고서

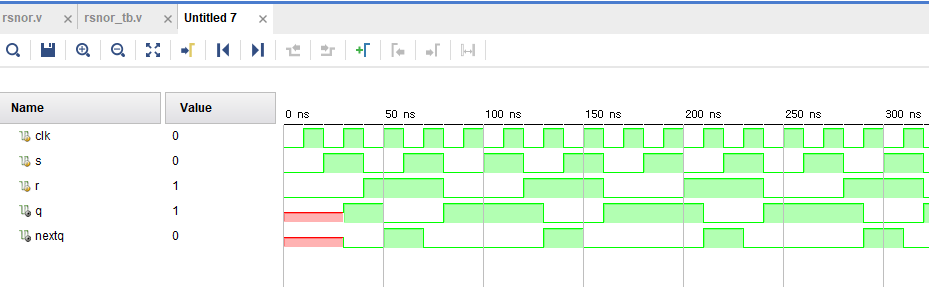
전공 : 아트앤테크놀로지 학년 : 3학년 학번 : 20191098 이름 : 백승주



Rs 풀립 풀롭은 1비트의 정보가 저장 가능한 회로다. 위 그림는 rs 플립 플롭의 회로로 RS 플립 플롭은 위와 같이 두 개의 AND 게이트와 두 개의 Nor 게이트를 통해 구현이 가능하다. 여기서 q 값은 R값과 clk값의 And 게이트를 거친 값을 얻은 후 이 값을 q’ 값과의 Nor 연산으로 구했고 q’값은 s값과 clk값의 And 연산, 그리고 이 값과 q값과의 Nor 연산을 통해 구했다. Q’ 변수의 이름을 nextq라 하고 nor 게이트를 이용한 rs플립플롭의 Verilog 코드를 작성하면 다음과 같다



위 코드와 testbench 코드에 따라 다음과 같은 시뮬레이션 결과가 나온다.

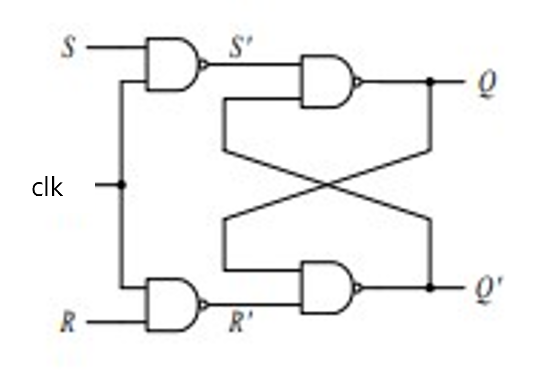


위 시뮬레이션의 결과를 처음부터 살펴보자. 먼저 clk값과 q값, nextq값과의 관계를 살펴보면 q값과 nextq값은 clk가 0일 때는 이전의 값들을 유지하고 clk값이 1이 되면 새로운 값을 업데이트한 것을 확인할 수 있다.  
 맨 처음 s값과 r값이 0인 경우 q값과 nextq값에는 unknown값이 출력된 것을 확인할 수 있다. Q와 nextq는 s와 r이 0일 경우 이전의 값을 출력해야 하지만 이 경우 이전의 값이 존재하지 않기 때문에 위와 같은 결과가 나오는 것을 알 수 있다. 그 이후 s값이 1이 되고 r값이 0이 되며 q값은 1이 되고 nextq값은 0이 된다. 이후 r값이 1이 되고 s값이 0이 되었을 때는 반대로 q값이 0이 되고 nextq값이 1이 된 것을 확인할 수 있다. S와 r이 둘 다 1이 되었을 때는 시뮬레이션 결과를 확인해보면 unknow값으로 출력될 거라는 예상과 다르게 값이 출력된 것을 확인할 수 있었다. 그리고 s값과 r값이 둘 다 0의 값을 가질 경우에는 q와 nextq값이 이전의 값을 유지하고 있는 것을 확인할 수 있다.

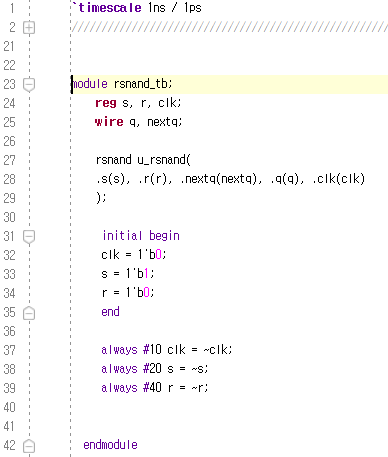
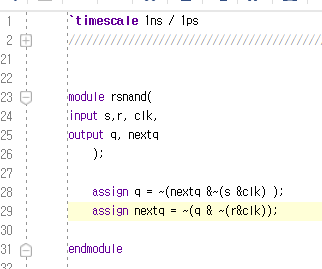
위 시뮬레이션 결과를 토대로 rs플립 플롭의 table을 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Clk** | **S** | **R** | **q** | **Nextq** |
| 0 | X | X | 이전 값 유지 | 이전 값 유지 |
| 1 | 0 | 0 | 이전 값 유지 | 이전 값 유지 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | x | x |

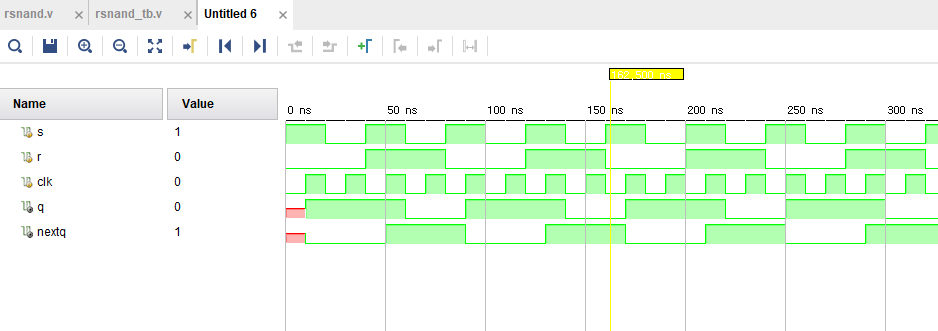
* 1. 앞선 rs플립 플롭은 nor 게이트를 활용한 플립 플롭이다. 이번에는 nand 게이트를 활용해 rs플립 프롧을 구현해보자.



위와 같이 nand 게이트를 이용한 rs 플립 플롭은 s 값이 두 개의 nand 게이트를 거쳐 q로 입력되고 r값은 두 개의 nand 게이트를 거쳐 q’ 로 입력된다. 위 rs 플립 플롭은 nor 게이트로 만든 rs 플립 플롭과 그 작동 원리는 동일하다. 이에 따른 Verilog 코드를 작성하면 다음과 같다.



Testbench 코드를 활용해 시뮬레이션 결과를 얻으면 다음과 같다.

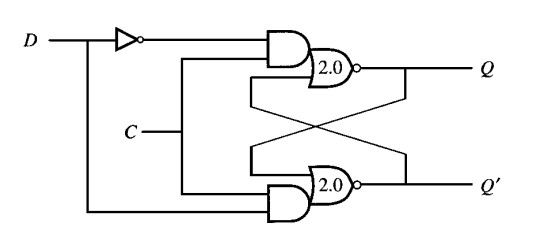


위 시뮬레이션 결과를 확인하면 clk 값이 0일 때에는 이전 값을 유지하는데 맨 처음에는 이전 값이 존재하지 않으므로 unknown 표시된 것을 확인할 수 있다. 이외에도 이전 nor 게이트를 활용한 rs 플립 플롭과 동일하게 r 값이 0, s 값이 1이면 q와 nextq는 1, 0 r값이 1이고 s값이 0이면 q,nextq가 0,1, s와 r이 둘 다 0 이면 이전 값 유지라는 결과가 나타나는 것을 확인할 수 있었다.

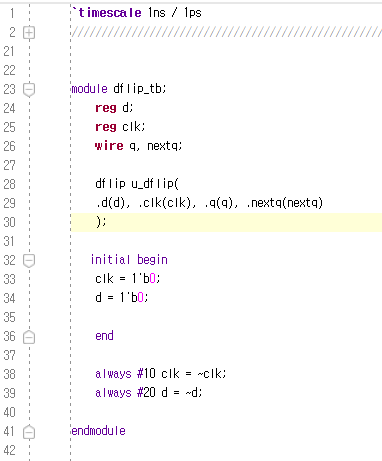
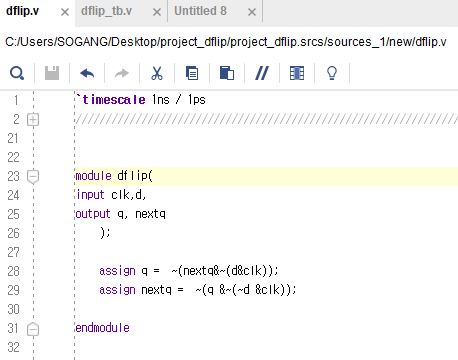
단 이번에도 r과 s 가 둘 다 1일 경우 원하는대로 unknown 값이 도출되지는 않았다. Table은 앞서 살펴본 nor rs 플립 플립과 동일하게 나타난다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Clk** | **S** | **R** | **q** | **Nextq** |
| 0 | X | X | 이전 값 유지 | 이전 값 유지 |
| 1 | 0 | 0 | 이전 값 유지 | 이전 값 유지 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | x | x |

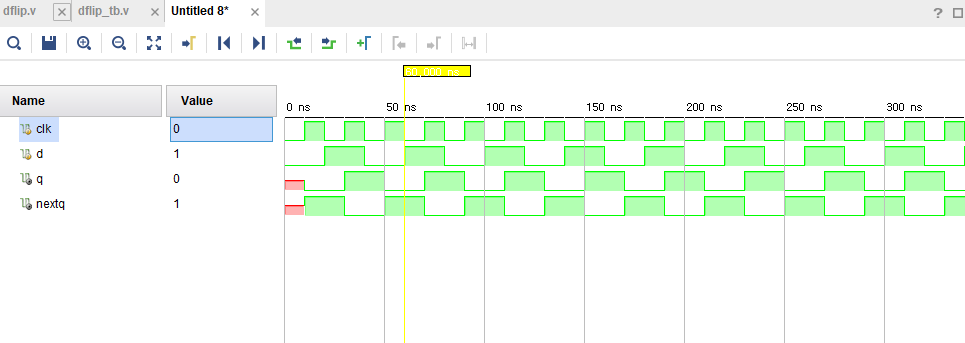
3.



위 그림은 D 플립 플롭의 회로로 앞선 rs 플립 플롭과 비교했을 때 D에 not 을 취한 값이 r 역할,을 하고 s 역할을 d가 하고 있음을 확인할 수 있다. Q 값은 ~d와 c 값의 and 연산 값과 q’ 값의 nor 연산을 통해 구했으며 q’ 값은 d와 c값의 and 연산 값과 q 값과의 nor 연산을 통해 구했다. 이에 따른 Verilog 코드는 다음과 같다.



위 코드들에 의한 시뮬레이션 결과는 다음과 같다.



먼저 clk 값이 0인 겨우에는 이전 값이 유지되기 때문에 맨 처음 값에서는 q와 nextq 값이 unknown처리 된 것을 확인할 수 있다. 이후 clk 값이 1일 때 d 값이 1인 경우에는 q값은 nextq값은 각각 1, 0 의 값을 가지게 되고 d값이 0인 경우에는 q와 nextq는 각각 0,1의 값을 가지는 것을 확인할 수 있다. 즉 clk 값이 1로 유지되는 한, q값은 항상 d값을 그대로 갖는 것을 확인할 수 있다. 또한 항상 q값과 nextq 값은 서로 반대값을 가지는 것 역시 시뮬레이션을 통해 확인할 수 있었다. 이에 따른 d 플립 플롭의 테이블을 작성하면 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Clk** | **D** | **q** | **Nextq** |
| 0 | X | 이전 값 유지 | 이전 값 유지 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

3. 실습을 통해 nor 게이트로 이루어진 rs 플립 플롭, nand 게이트로 이루어진 rs 플립 플롭, d 플립 플롭을 Verilog 코드로 구현하고 그 시뮬레이션 결과를 확인할 수 있었다. 이러한 모든 플립 플롭들에서 공통적으로 초기의 출력값은 unknown 처리된 것을 확인할 수 있었는데 그 이유는 먼저 clk 값이 0일 경우 이전의 출력값을 유지하게 되는데 초기값의 경우 이전의 값이 존재하지 않아 출력값을 정의할 수 없기 때문이었다.

또한 rs 플립 플롭에서 r과 s 값이 모두 1일 경우 nor 플립 플롭에서는 q와 nextq 값이 모두 0으로 출력되고 nand 플립 플롭에서는 q가 1, nextq 가 0을 출력한 것을 확인했는데 같은 결과를 도출해야할 회로들임에도 불구하고 다른 결과, 즉 원치 않는 결과를 출력한 것을 확인했다. 이를 방지하기 위해 d 플립 플롭의 d와 ~d를 이용해 r역할과 s 역할을 대신하게 해 모두 0이 되는 경우를 방지했다. 덕분에 d 플립 플롭의 시뮬레이션 결과는 초기값을 제외하고는 모두 유효한 결과들이 나타났음을 확인할 수 있었다.

**4.** 디지털 시스템에서 정확한 동작을 보장하는데 필수적인 setup time과 holdup time에 대해 알아보자.

Setup Time

Setup Time 은 디지털 시스템에서 중요한 parameter로, 플립플롭이나 레지스터 등에 입력 데이터가 클럭 신호가 변화하기 전에 얼마나 오래 유지되어야 하는지를 나타낸다. Setup Time 이 충족되지 않으면, 입력 데이터가 불안정하게 될 수 있으며, 이는 플립플롭이나 레지스터의 출력이 예상치 못한 값으로 변할 수 있음을 의미한다. 따라서, Setup Time 을 충족하는 것은 정확한 동작을 위해 중요하다.

Hold Time

Hold Time은 Setup Time 과 밀접하게 연관되어 있다. Hold Time은 클럭 신호의 변화 이후에 입력 데이터가 얼마나 오래 유지되어야 하는지를 나타낸다. 이러한 Hold Time은 플립플롭이나 레지스터가 입력 데이터를 올바르게 캡처하고 유지할 수 있도록 보장하는 데 필요하다. Hold Time을 준수하지 않으면 출력이 예측할 수 없는 상태로 전환될 수 있는데 이는 결국 시스템 오류로 이어지게 된다.