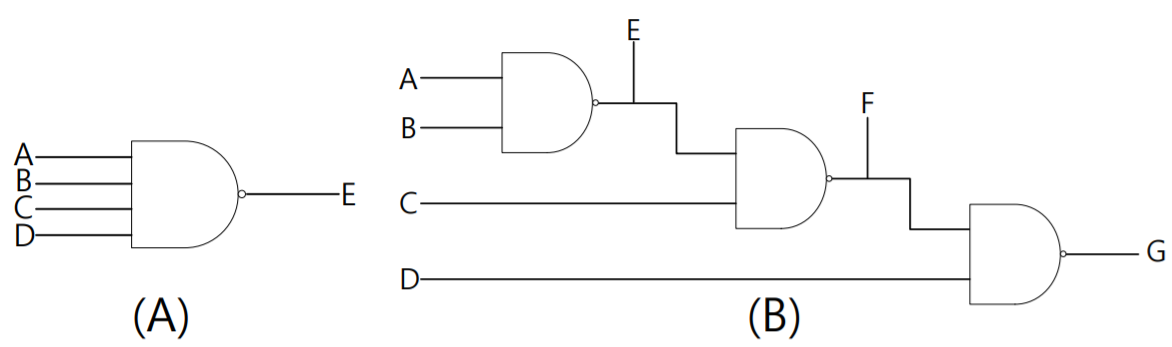
4주차 결과보고서

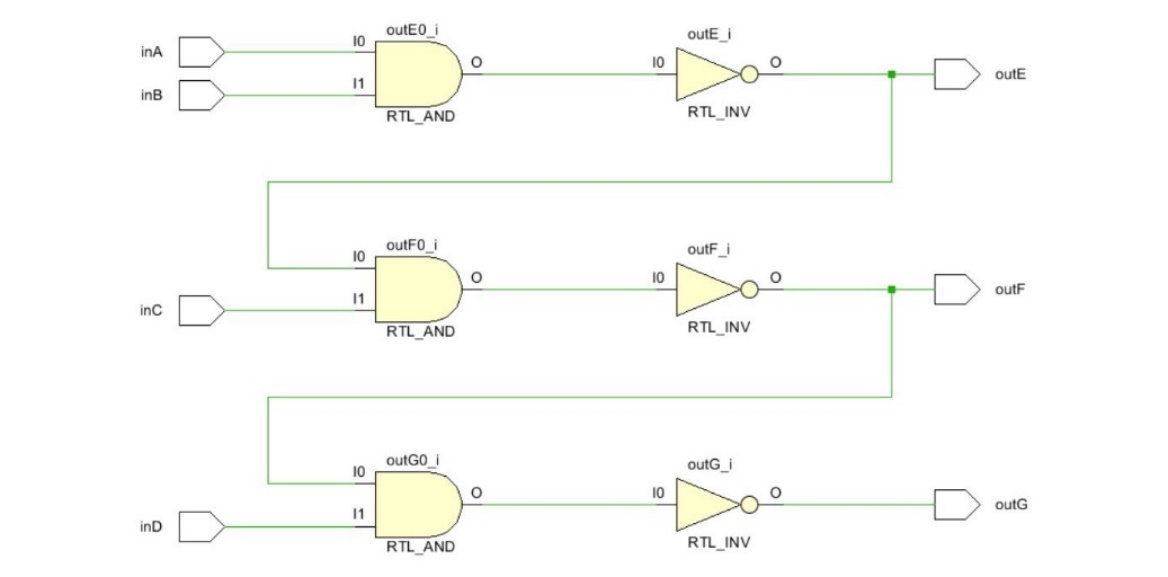
전공: 물리학과 학년: 3학년 학번: 20191286 이름: 김나현

1. 실험 목적

AND, OR, NOT 게이트에 이어, NAND, NOR, XOR 게이트의 동작을 확인해보기 위해 Verilog로 입력 신호 코드를 생성한 후 schematic을 확인하고 testbench를 생성하여 simulation 결과를 확인합니다. 이때, 각 게이트에 대한 simulation 결과를 얻고, 이를 이용하여 진리표를 작성합니다.

1. 4-input NAND gate의 simulation 결과 및 과정에 대해 설명하시오.

<Figure 1> 입력이 네 개일 때, NAND gate를 이용한 두 가지의 회로 구현 방법

위의 <Figure 1>은 입력이 4개일 때, 하나의 출력을 얻기 위해 NAND 게이트를 이용하여 회로를 구현한 두 가지 방법입니다. 이때, (A)는 한 개의 NAND 게이트에 모든 입력이 들어가 하나의 출력 E를 얻은 것이고, (B)는 세 개의 NAND 게이트에 A와 B의 입력이 들어가 출력 E를 얻고, 출력 E와 입력 C가 다시 두 번째 NAND 게이트의 입력으로 들어가 출력 F를 얻고, 마지막으로 출력 F와 입력 D가 세 번째 NAND 게이트의 입력으로 들어가 최종 출력인 G를 얻는 것을 의미합니다.

<Figure 2> <Figure 1>의 (B)를 구현한 schematic diagram

위의 <Figure 2>는 <Figure 1>의 두 가지 구현 방법 중 (B)를 구현한 schematic diagram입니다. NAND 게이트를 AND와 NOT 게이트를 이용해서 구현할 수 있다는 것을 보여주기 위해, 다음 <Figure 3>의 verilog 코드와 같이 작성하여 synthesis한 후, schematic을 확인한 것이기 때문에 NAND 게이트 없이 AND와 NOT 게이트만을 사용하여 <Figure 1>의 (B)를 구현하였다는 것을 알 수 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<Figure 3> four-nand.v



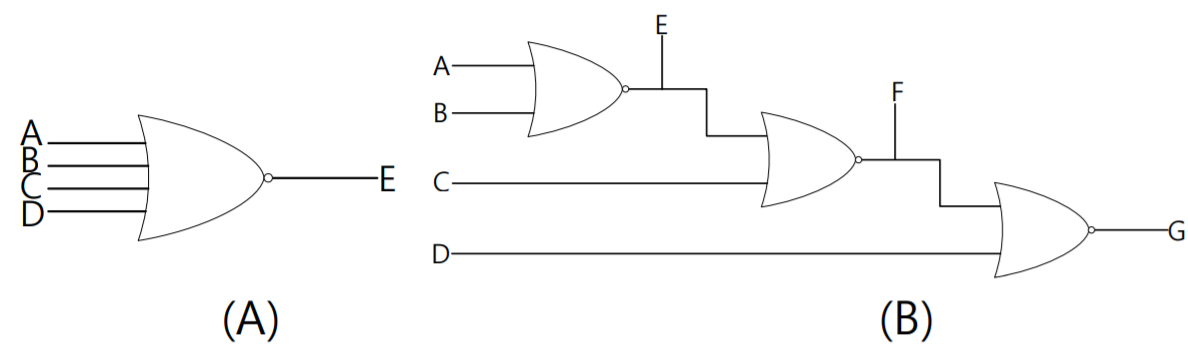
<Figure 4> <Figure 3>의 simulation 결과

위의 <Figure 4>는 <Figure 3>와 testbench를 이용하여 simulation한 결과로, 이를 통해 <Figure 1>의 (B) 논리 회로의 진리표를 작성해보면 다음 <Table 1>과 같습니다.

<Table 1> <Figure 1>의 (B) 진리표

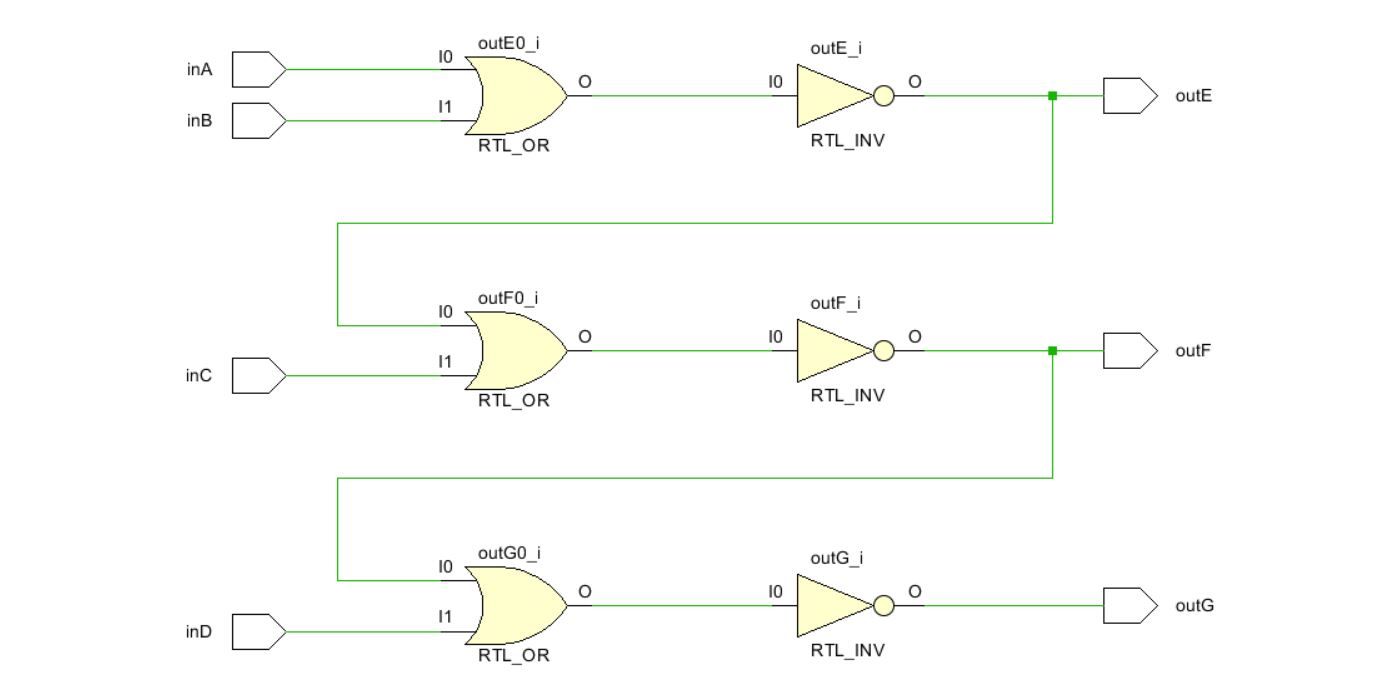
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| A | B | C | D | G |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

NAND 게이트는 모든 입력이 1일 때를 제외하고는 모두 1을 출력한다는 사실과 달리, <Table 1>을 보면 <Figure 1>의 (B) 논리 회로는 NAND 게이트의 진리표에서 관찰할 수 있는 특성이 보이지 않는다는 것을 확인할 수 있습니다. 이는 <Figure 1>의 (A)처럼 모든 입력이 하나의 NAND 게이트에 입력되었을 때만 simulation 결과를 확인하면 우리가 이미 알고 있는 NAND 게이트의 특성이 나타나는 진리표를 얻을 수 있고, <Figure 1>의 (B)처럼 여러 개의 NAND 게이트를 거치면 최종 출력(<Figure 4>의 TestOut과 <Figure 1>의 (B)에서 G를 의미함.)과 simulation의 결과를 이용하여 진리표를 작성하여도 NAND 게이트의 특성이 드러나지 않는다는 사실 때문입니다. 이로부터 NAND 게이트는 3주차 실험에서 다루었던 AND 게이트와 OR 게이트와 달리, 여러 개의 입력이 존재할 때 하나의 게이트에 모든 입력이 인가되었을 때 나오는 출력 결과와 여러 번의 동일한 게이트를 거쳐서 나오는 출력 결과가 상이하다는 사실을 알 수 있습니다.

1. 4-input NOR gate의 simulation 결과 및 과정에 대해 설명하시오.

<Figure 5> 입력이 네 개일 때, NOR gate를 이용한 두 가지의 회로 구현 방법

위의 <Figure 5>은 입력이 4개일 때, 하나의 출력을 얻기 위해 NOR 게이트를 이용하여 회로를 구현한 두 가지 방법입니다. 이때, (A)는 한 개의 NOR 게이트에 모든 입력이 들어가 하나의 출력 E를 얻은 것이고, (B)는 세 개의 NOR 게이트에 A와 B의 입력이 들어가 출력 E를 얻고, 출력 E와 입력 C가 다시 두 번째 NOR 게이트의 입력으로 들어가 출력 F를 얻고, 마지막으로 출력 F와 입력 D가 세 번째 NOR 게이트의 입력으로 들어가 최종 출력인 G를 얻는 것을 의미합니다.

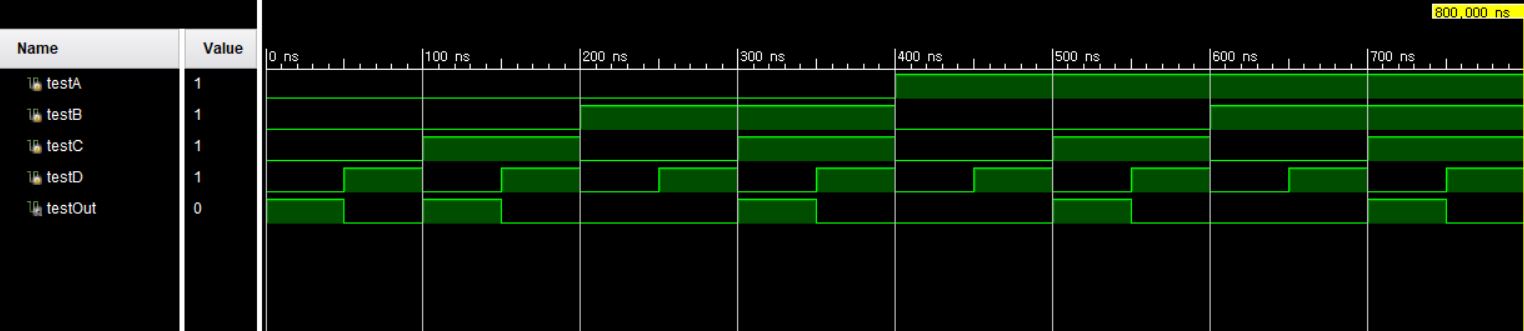
앞서 NAND 게이트 진리표, <Table 1>을 통해 NAND 게이트는 AND, OR 게이트와 달리 여러 개의 입력이 있을 때, 모든 입력이 하나의 게이트를 거쳐 나오는 출력과 여러 개의 게이트를 거쳐 나오는 출력이 다르다는 사실을 알게 되었는데 본 실험에서도 NOR 게이트가 NAND 게이트와 같은 특성을 보이는지 아니면 AND나 OR 게이트와 같은 특성을 보이는지에 초점을 맞춰서 결과를 분석해보고자 합니다.

<Figure 6> <Figure 5>의 (B)를 구현한 schematic diagram

위의 <Figure 6>는 <Figure 5>의 두 가지 구현 방법 중 (B)를 구현한 schematic diagram입니다. NAND 게이트의 경우와 마찬가지로, NOR 게이트를 OR와 NOT 게이트를 이용해서 구현할 수 있다는 것을 보여주기 위해 다음 <Figure 7>의 verilog 코드와 같이 작성하여 synthesis한 후, schematic을 확인하였습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<Figure 7> four-nor.v

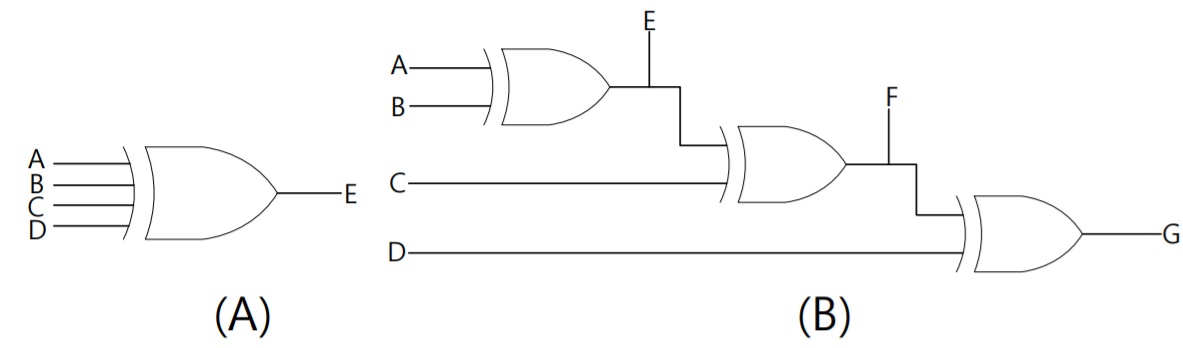
<Figure 8> <Figure 7>의 simulation 결과

위의 <Figure 8>는 <Figure 7>과 testbench를 이용하여 simulation한 결과로, 이를 통해 <Figure 5>의 (B) 논리 회로의 진리표를 작성해보면 다음 <Table 2>과 같습니다.

<Table 2> <Figure 5>의 (B) 진리표

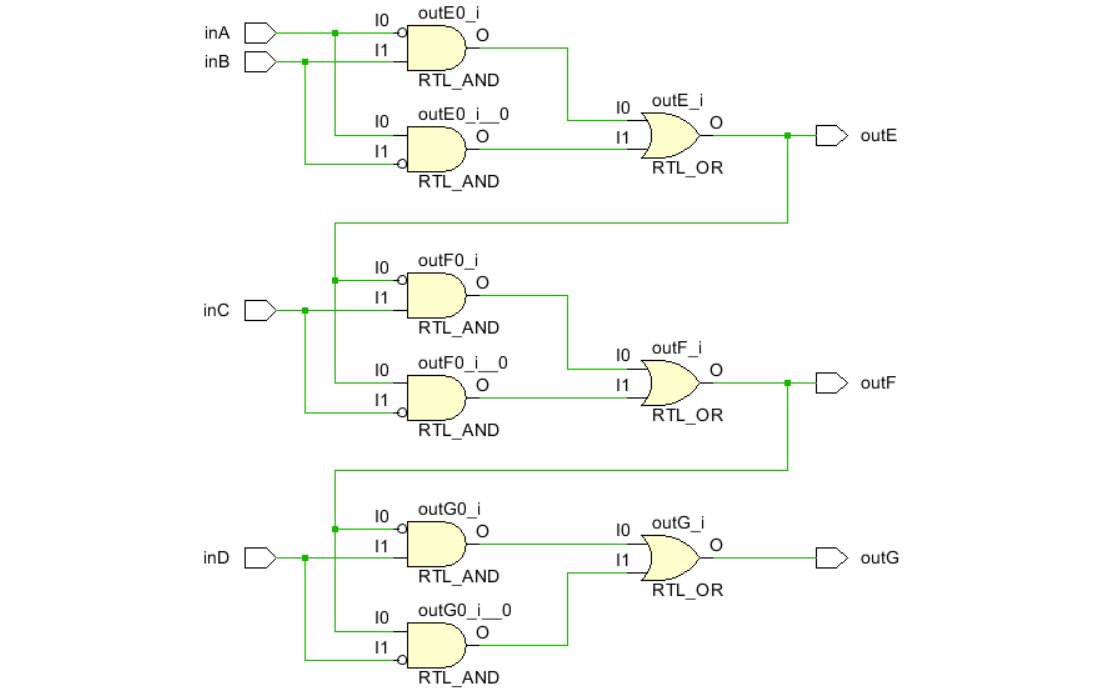
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| A | B | C | D | G |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

<Table 2>를 보면, NOR 게이트는 모든 입력이 0일 때를 제외하고는 모두 0을 출력한다는 사실과 달리, <Figure 5>의 (B) 논리 회로의 진리표에서는 NOR 게이트의 진리표에서 관찰할 수 있는 특성이 보이지 않는다는 것을 확인할 수 있습니다. 이는 NAND 게이트와 마찬가지로 <Figure 5>의 (A)처럼 모든 입력이 하나의 NOR 게이트에 입력되었을 때만 simulation 결과를 통해 우리가 익히 알고 있는 NOR 게이트의 특성이 나타나는 진리표를 얻을 수 있고, 똑같이 여러 개의 입력이 있다고 해도 <Figure 5>의 (B)처럼 여러 개의 NOR 게이트를 거치면 최종 출력(<Figure 8>의 TestOut과 <Figure 5>의 (B)에서 G를 의미함.)과 simulation의 결과를 이용하여 진리표를 작성하여도 NOR 게이트의 특성이 한 눈에 드러나지 않는다는 사실 때문입니다. 이로부터 NOR 게이트 역시, 3주차 실험에서 다루었던 AND 게이트와 OR 게이트와 달리 여러 개의 입력이 존재할 때 하나의 게이트에 모든 입력이 인가되었을 때 나오는 출력 결과와 여러 번의 동일한 게이트를 거쳐서 나오는 출력 결과가 상이하다는 것을 알 수 있습니다.

1. 4-input XOR gate의 simulation 결과 및 과정에 대해 설명하시오.

<Figure 9> 입력이 네 개일 때, XOR gate를 이용한 두 가지의 회로 구현 방법

앞선 두 개의 게이트와 동일하게 위의 <Figure 9>은 입력이 4개일 때, 하나의 출력을 얻기 위해 NOR 게이트를 이용하여 회로를 구현한 두 가지 방법입니다. 이때, (A)는 한 개의 XOR 게이트에 모든 입력이 들어가 하나의 출력 E를 얻은 것이고, (B)는 세 개의 XOR 게이트에 A와 B의 입력이 들어가 출력 E를 얻고, 출력 E와 입력 C가 다시 두 번째 XOR 게이트의 입력으로 들어가 출력 F를 얻고, 마지막으로 출력 F와 입력 D가 세 번째 XOR 게이트의 입력으로 들어가 최종 출력인 G를 얻는 것을 의미합니다.



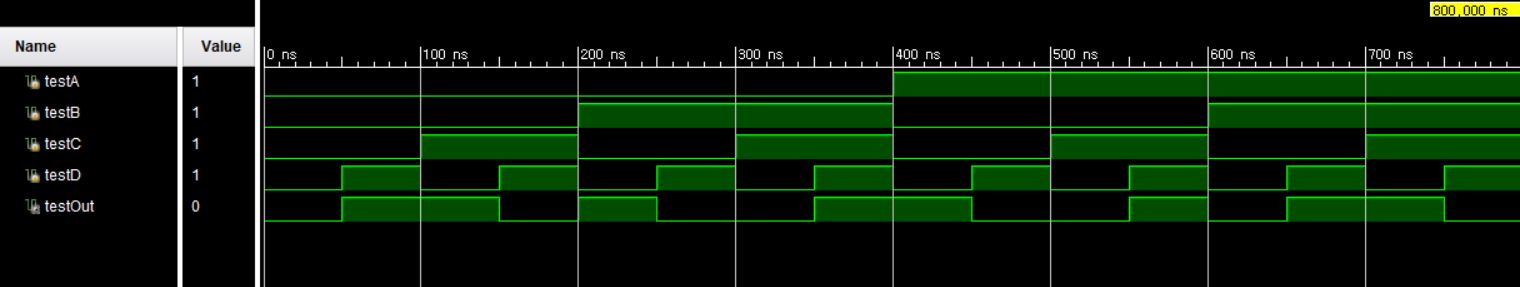
<Figure 10> <Figure 9>의 (B)를 구현한 schematic diagram

위의 <Figure 10>는 <Figure 9>의 두 가지 구현 방법 중 (B)를 구현한 schematic diagram입니다. 위의 두 게이트 회로와 마찬가지로, NOR 게이트를 AND와 OR, NOT 게이트를 이용해서 구현할 수 있다는 것을 보여주기 위해 다음 <Figure 11>의 verilog 코드와 같이 작성하여 XOR 게이트의 기호 없이 AND와 OR 게이트만이 사용된 것을 알 수 있습니다. <Figure 10>을 통해, 두 개의 입력 A와 B가 있을 때, A’B+AB’로 XOR 연산을 한다는 것을 확인할 수 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<Figure 11> four-xor.v



<Figure 12> <Figure 11>의 simulation 결과

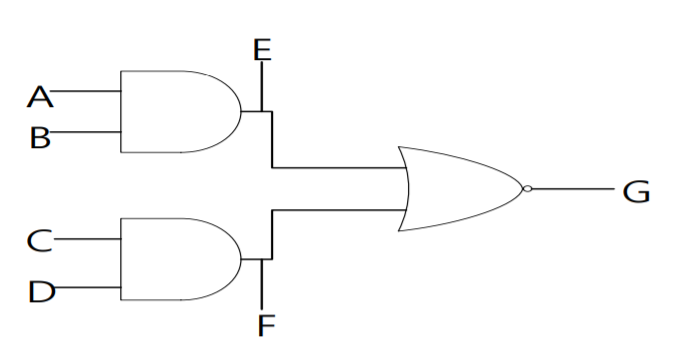
위의 <Figure 12>는 <Figure 11>과 testbench를 이용하여 simulation한 결과로, 이를 통해 <Figure 9>의 (B) 논리 회로의 진리표를 작성해보면 다음 <Table 3>과 같습니다.

<Table 2> <Figure 9>의 (B) 진리표

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| A | B | C | D | G |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

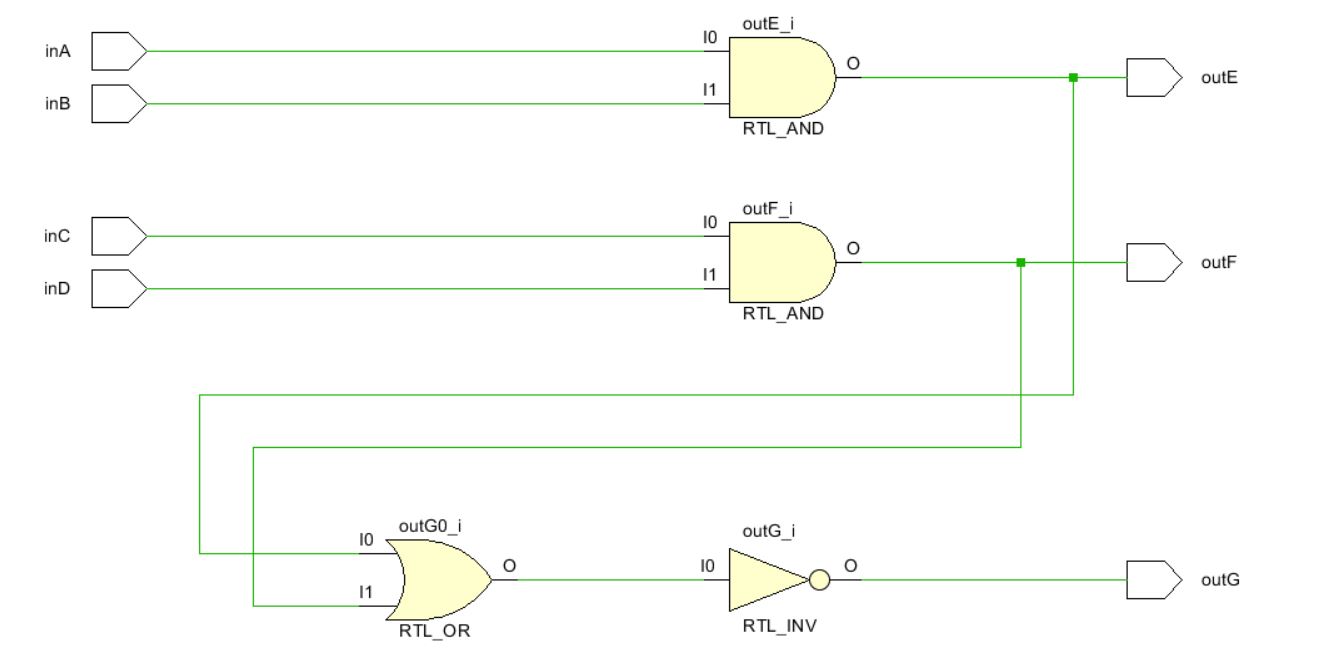
<Table 3>를 보면, XOR 게이트는 참 입력의 개수가 홀수일 때 참을 출력하는 논리 게이트라는 사실과 같이, <Figure 9>의 (B) 논리 회로의 진리표에서도 네 개의 입력 중 1인 입력의 개수가 1개나 3개일 때, 1을 출력하는 모습을 확인할 수 있습니다. 이는 앞선 NAND, NOR 게이트와 달리, <Figure 9>의 (A)처럼 모든 입력이 하나의 NOR 게이트에 입력되었을 때나 <Figure 9>의 (B)처럼 여러 개의 XOR 게이트를 거쳐서 최종 결과를 얻었을 때나 XOR 게이트의 특성이 반영된 진리표를 얻을 수 있다는 사실을 보여줍니다. 이로부터 XOR 게이트는 3주차 실험에서 다루었던 AND 게이트와 OR 게이트와 마찬가지로, 여러 개의 입력이 존재할 때 하나의 게이트에 모든 입력이 인가되었을 때 나오는 출력 결과와 여러 번의 동일한 게이트를 거쳐서 나오는 출력 결과가 동일하다는 것을 알 수 있습니다.

1. 4-input AOI gate의 simulation 결과 및 과정에 대해 설명하시오.



<Figure 13> 입력이 네 개일 때, AND, OR, NOT 게이트를 이용한 AOI 게이트 회로

위의 <Figure 13>은 입력이 4개일 때, 하나의 출력을 얻기 위해 AND, OR, NOT 게이트를 이용하여 AOI 게이트 회로를 구현한 모습입니다. 이때, 위의 회로를 논리식으로 정리해보면 E=AB, F=CD이고, G=(E+F)’이므로 최종 출력 G는 G=E’F’=(AB)’(CD)’=(A’+B’)(C’+D’)입니다.



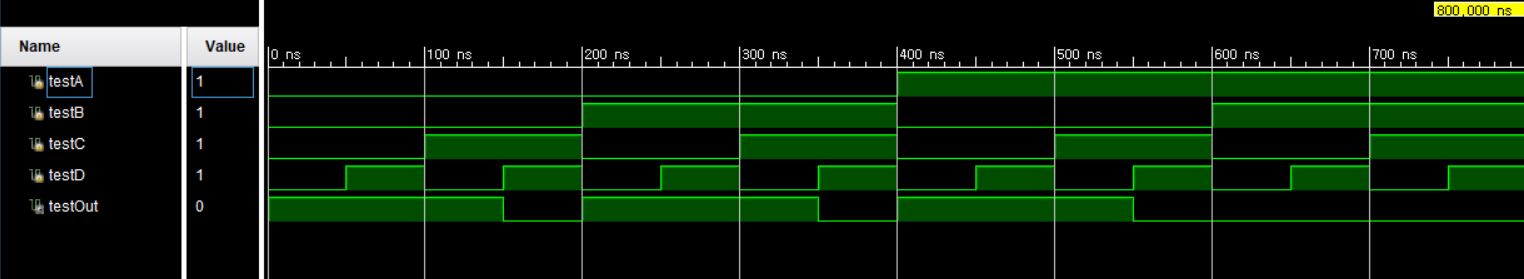
<Figure 14> <Figure 13>을 구현한 schematic diagram

위의 <Figure 14>는 <Figure 13>을 구현한 schematic diagram입니다. 3주차 실험을 통해, AND 게이트의 동작을 공부했기 때문에 위의 <Figure 14>에서 입력 A, B, C, D에 따른 출력 E와 F는 직접 verilog를 이용하지 않고서도 예측이 쉽게 가능하지만 입력 A, B, C, D와 출력 G의 관계는 직관적으로 파악이 어렵기 때문에 다음 <Figure 15>와 같이 verilog 코드를 작성하여 4-input AOI 게이트의 동작을 알아보았습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<Figure 15> four-aoi.v



<Figure 16> <Figure 15>의 simulation 결과

위의 <Figure 16>는 <Figure 15> verilog 코드와 testbench를 이용하여 simulation한 결과로, 이를 통해 4-input AOI 게이트의 진리표를 작성해보면 다음 <Table 4>과 같습니다.

<Table 4> <Figure 13>의 진리표

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| A | B | C | D | G |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 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 |

<Figure 13>의 최종 출력 G에 대한 논리식 G=E’F’=(AB)’(CD)’=(A’+B’)(C’+D’)을 통해서 알 수 있듯이, A와 B를 AND 연산하여 얻은 출력 E와 C와 D를 AND 연산하여 얻은 출력 F는 OR 게이트와 NOT 게이트를 순차적으로 거치므로 NOR 게이트를 거친다고 생각하여도 무방합니다. 따라서, 최종 출력 G는 E와 F가 모두 0일 때만 1을 나타내게 되는데 E와 F가 모두 0이려면 반드시 A와 B 중에서 하나 이상이 0이라는 조건과 C와 D 중에서 하나 이상이 0이라는 조건을 모두 만족해야 합니다. 위의 <Table 4>를 보면, 출력 G가 1인 경우는 실제로 앞서 언급한 두 조건을 모두 만족하고 있다는 사실을 알 수 있고, 출력 G가 0인 경우는 두 조건 중 최소 하나 이상을 만족하지 않고 있다는 사실을 알 수 있습니다.

1. 결과 검토 및 논의사항

앞서 언급하였다시피 NAND 게이트와 NOR 게이트를 이용한 회로에서는 입력의 개수가 여러 개일 때 몇 개의 게이트를 거치느냐에 따라 다른 진리표를 얻을 수 있다는 사실을 알 수 있었습니다. 이는 네 개의 입력이 하나의 NOR 게이트를 거칠 때는, E=(A+B+C+D)’의 논리식을 계산해보면 (A+B+C를 X로, A+B를 Y로 치환한다고 생각하면) E=(A+B+C+D)’=(X+D)’=X’D’=(A+B+C)’D’= (Y+C)’D’=Y’C’D’=(A+B)’C’D’=A’B’C’D’가 되어 모든 입력이 0일 때만 E가 1이 되는, NAND 게이트의 특성을 따르지만 세 개의 NOR 게이트를 거칠 때는, E=(((A+B)’+C)’+D)’이므로 E=((A+B)’+C)D’= (A’+B’+C)D’가 되어 NOR 게이트의 특성을 전혀 따르지 않는 결과를 얻는다는 사실 때문입니다. 마찬가지로, 네 개의 입력이 하나의 NAND 게이트를 거칠 때는, E=(ABCD)’의 논리식을 계산해보면 (ABC를 X로, AB를 Y로 치환한다고 생각하면) E=(ABCD)’=(XD)’=X’+D’=(ABC)’+D’=(YC)’+D’= (Y’+C’)D’=((AB)’+C’)+D’=A’+B’+C’+D’가 되어 모든 입력이 1일 때만 E가 0이 되는, NAND 게이트의 특성을 따르지만 세 개의 NAND 게이트를 거칠 때는, E=(((AB)’C)’D)’이므로 E=(((AB)’C)’D)’ =((AB)’C)+D’=((A’+B’)C)+D=A’C+B’C+D가 되어 NAND 게이트의 특성을 전혀 따르지 않는 결과를 얻는다는 사실 때문입니다.

반면, XOR 게이트의 경우, 3주차 실험에서 다룬 AND, OR 게이트와 마찬가지로 여러 개의 입력이 있을 때 몇 개의 게이트를 거치느냐에 무관하게 동일한 진리표를 얻을 수 있었습니다. 본 실험의 상황과 동일하게 네 개의 입력이 있을 때, 하나의 XOR 게이트를 거칠 때는, E=A⊕B⊕C⊕D라고 표현할 수 있지만 Boolean 연산의 교환, 결합법칙에 의해 E=(A⊕B)⊕(C⊕D)의 순서로 계산하든, E=((A⊕B)⊕C)⊕D의 순서로 계산하든 항상 같은 결과를 얻기 때문에 여러 개의 XOR 게이트를 거쳐도 하나의 XOR 게이트를 거칠 때와 같은 출력을 얻을 수 있는 것이다.

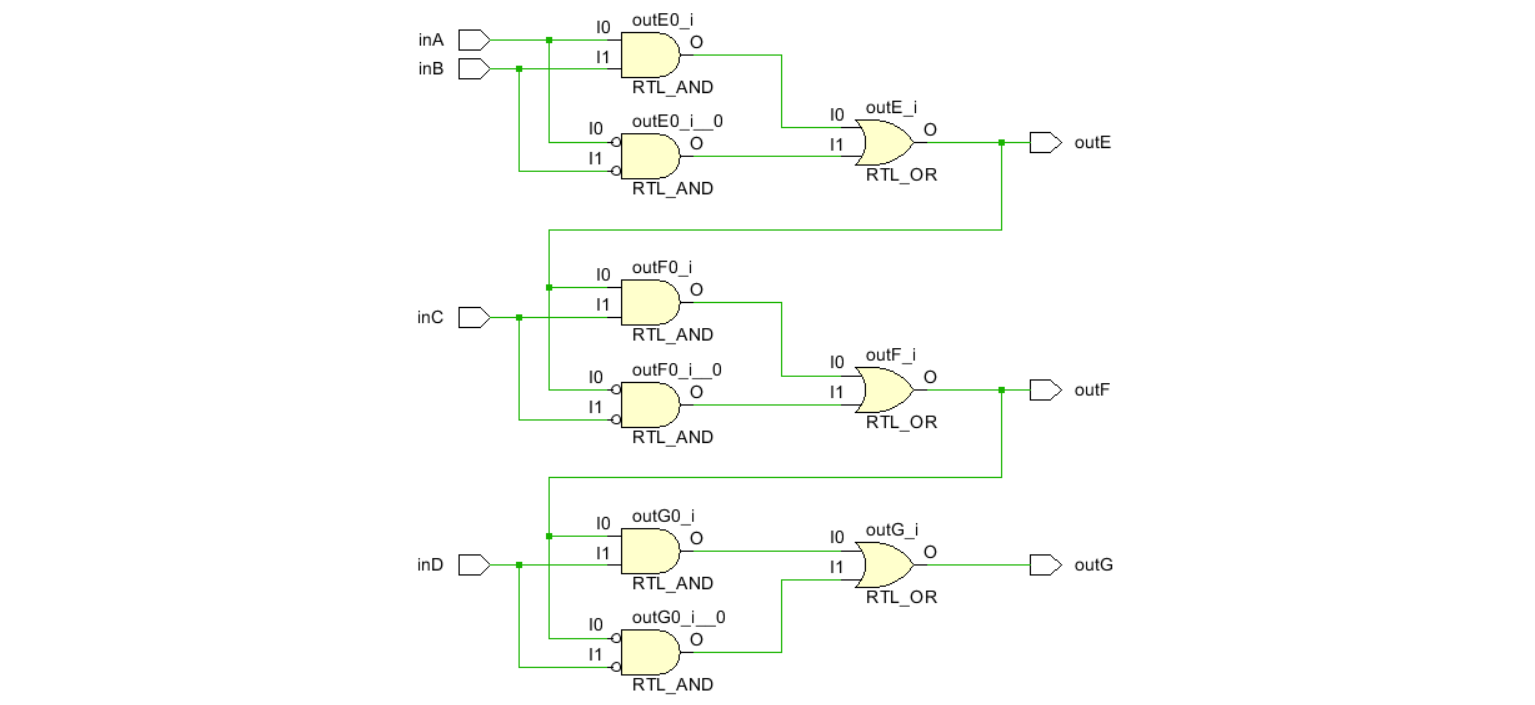
1. 추가 이론 조사 및 작성

텍스트이(가) 표시된 사진

자동 생성된 설명

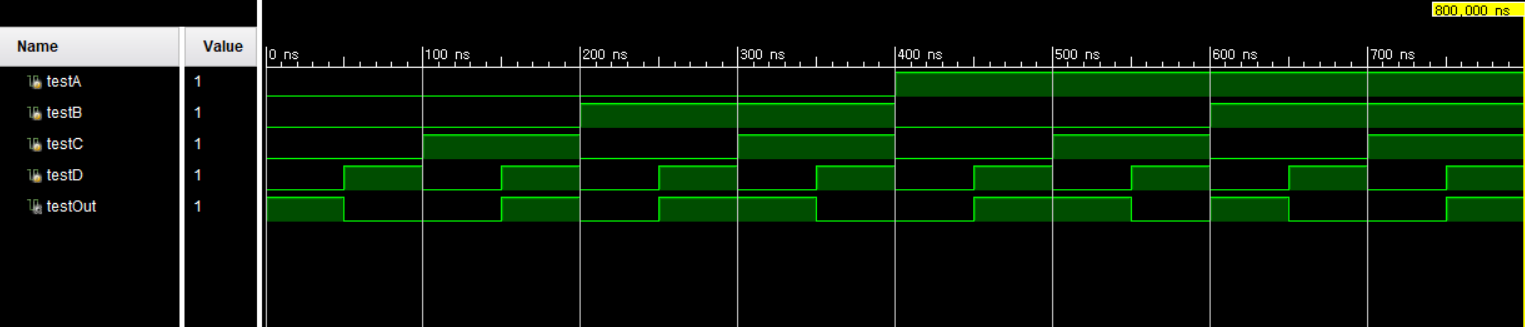
<Figure 17> four-xnor.v

위의 <Figure 17>은 XNOR 게이트를 구현한 Verilog 코드입니다. XNOR 게이트 역시 AND, OR, NOT 게이트만을 이용하여 구현할 수 있기 때문에 위와 같이 코딩을 하였고, 그 결과 <Figure 18>과 같은 schematic diagram을 얻을 수 있었습니다.



<Figure 18> <Figure 17>을 구현한 schematic diagram

<Figure 18>로부터 XNOR 게이트는 두 개의 입력 A와 B가 있다고 할 때, E=AB+A’B’라는 연산을 하는 게이트라는 사실을 알 수 있습니다. E=AB’+A’B의 연산을 하는 XOR 게이트에 NOT 연산을 하면 E’=(AB’+A’B)’=(A’+B)(A+B’)=AB+A’B’이므로 XNOR 게이트와 같은 역할을 한다는 것 또한 알 수 있습니다.



<Figure 19> <Figure 17>의 simulation 결과

<Table 5> <Figure 19>의 진리표

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| A | B | C | D | G |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

<Figure 19>의 simulation 결과를 이용하여 <Table 5>와 같이 진리표를 작성할 수 있습니다. 입력의 개수 중 참인 입력이 홀수 개일 때, 1을 출력하는 XOR 게이트와 반대로 네 개의 입력이 있을 때, 1인 입력의 개수가 0개, 2개, 4개일 때만 1을 출력하는 모습을 확인할 수 있었습니다. XNOR 게이트도 XOR 게이트와 마찬가지로, 여러 개의 입력이 있을 때 하나의 게이트에 모든 입력을 인가하여 얻은 출력이나 여러 개의 게이트를 거쳐 얻은 출력이 동일하다는 것을 확인할 수 있습니다.