1 장

기본 게이트: AND, OR, NOT, NAND, NOR, XOR, XNOR

1. 실험 목표

▣ 모든 디지털 논리회로 설계에서 기본적인 논리 함수인 AND, OR, NOT 게이트의 동작 특성을 이해하고, 실제 설계 시 적용되는 NAND, NOR 게이트의 동작 특성을 파악한다. 그리고 추가 논리함수로써 XOR, XNOR 게이트의 동작 특성도 분석해본다.

▣ 기본적인 논리 게이트를 이용한 간단한 회로의 설계구성과 동작을 verilog HDL 로 구현하여 검증한다.

2. 실험 이론

디지털 논리회로는 크게 조합회로와 순차회로로 나뉘어진다. 조합회로는 출력이 입력의 함수로만 정의되는 회로이며, 순차회로는 출력이 현 입력과 내부 상태의 함수로 정의되는 회로이다. 1 장에서 10 장까지는 조합회로를 설계하는 실험이며, 11 장부터는 순차회로에 대한 실험이 진행된다.

디지털 논리회로를 구성하는 가장 기본적인 3 개의 게이트는 AND, OR, NOT 이며, 이는 기본적인 조합회로의 설계 예이다. 그리고 모든 디지털 논리회로는 이들 3 개의 기본 게이트를 조합 구성하여 설계/표현이 가능하다. 이러한 3 개의 기본 게이트의 특성을 그대로 유지하며 출력 값만 반전 (invert)시키는 게이트로 NAND 와 NOR 게이트를 정의할 수 있다. NAND 게이트는 AND 게이트 출력의 값을 반전시키는 함수로 NOT(AND)이며, NOR 게이트는 OR 게이트의 출력의 값을 반전시키는 NOT(OR) 함수이다.

또한 AND, OR, NOT 로 표현 가능한 모든 디지털 논리회로는 NAND 와 NOR 의 조합으로 재표현 가능한 이중적 관계를 갖는 게이트들이다. 이들 기본 게이트 외에 추가로 정의 사용하는 함수로 XOR 와 XNOR 게이트의 동작 특성 또한 본 장에서 실험을 수행한다.

가) AND 게이트

AND 논리 게이트는 모든 입력인 '1'인 경우에만 출력이 '1'이 되는 논리함수이다. 따라서 AND 게이트는 주어진 문제에서 모든 입력 조건들이 동시에 모두 참인 경우에 출력을 '1'로 구동하는 것이 기본 목표이다. 그림 1-1 은 AND 게이트의 기호와 진리표를 나타낸다. AND 게이트 함수를 논리식으로 표현하면 Z = A ・ B 으로 표현된다. 그림 1-1 (c)에서 Verilog HDL 로 AND 게이트를 기술하였다.

본 실험에서는 AND 게이트를 구현하는 것이므로 모듈명을 AND 로 하고 입력신호와 출력신호 리스트를 괄호 안에 기술한다. 모듈명과 함께 기술된 신호리스트는 모듈 본체에서 입력신호와 출력신호로 명확하게 구분되어 기술된다. 모듈 내부의 신호의 연결은 할당문 (**assign**)을 이용하여 데이터 플로우 모델링 기법으로 기술되어 있다.

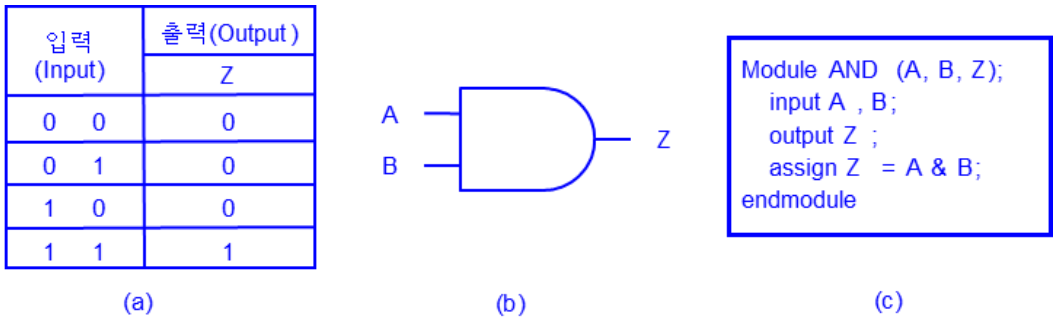


그림 1-1. AND 게이트 (a) 진리표, (b) 심볼 기호, (c) HDL 코드.

AND 게이트는 기본적으로 두 개의 입력을 지원하며, 입력의 수에 따라 3 입력, 4 입력 등, 여러 개의 입력을 지원하는 게이트 구성이 가능하며, 아래 그림 1-2 (a)는 3 입력를 지원하는 AND 게이트 기호이며, 그림 1-2(b)는 3 입력 AND 게이트를 2 입력 게이트 2 개를 2 단계로 연결 구성한 예이다.

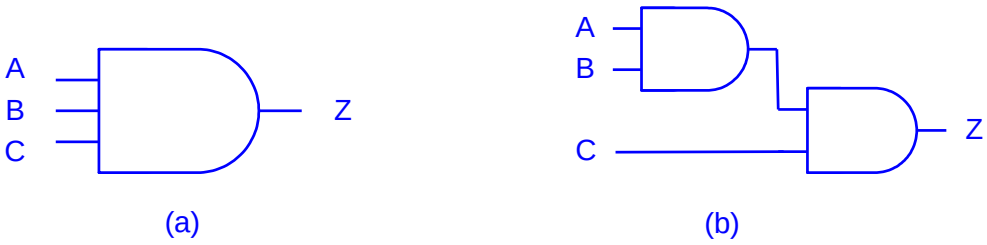


그림 1-2. 3 입력 AND 구성: (a) 3 입력 AND 게이트, (b) 2 단계 3 입력 AND 게이트.

나) OR 게이트

OR 게이트도 모든 논리 기능을 구성하는 데 필요한 기본 게이트 중의 하나로서 논리적 덧셈연산을 수행한다. 즉 입력 중에 하나 이상만 '1'이면 출력이 '1'이 되는 논리함수이다. 그림 1-3 은 OR 게이트의 기호 (b)와 진리표 (a)를 나타낸다. OR 게이트 함수를 논리식으로 표현하면 Z = A + B 으로 표현된다. 그림 1-3 (c)에서 Verilog HDL 로 OR 게이트를 기술하였다.

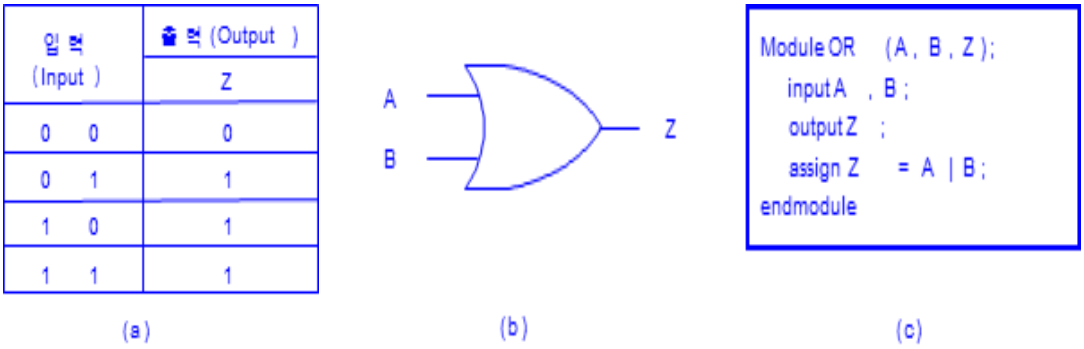


그림 1-3. OR 게이트 (a) 진리표, (b) OR 게이트 기호, (c) HDL 코드.

그림 1-3(c)에서와 같이 모듈명을 OR 로 하고 2 개의 입력신호 A, B 와 1 개의 출력신호 Z 리스트를 괄호 안에 기술한다. 모듈명 내에 기술된 입출력 신호리스트는 모듈 본체에서 입력신호와 출력신호로 정의, 구분되어 기술된다. 모듈 내부의 신호의 연결은 비트 논리 연산자 "|"을 이용하여 출력 Z 에 입력신호 A 와 B 를 데이터 플로우 모델링 기법으로 기술하였다.

OR 게이트도 기본적으로 두 개의 입력을 지원하며, 입력의 수에 따라 3 입력, 4 입력 등, 여러 개의 입력을 지원하는 게이트 구성이 가능하며, 아래 그림 1-4 (a)는 3 입력를 지원하는 OR 게이트 기호이며, 그림 1-4(b)는 3 입력 OR 게이트를 2 입력 게이트 2 개로 2 단계로 구성한 예이다.

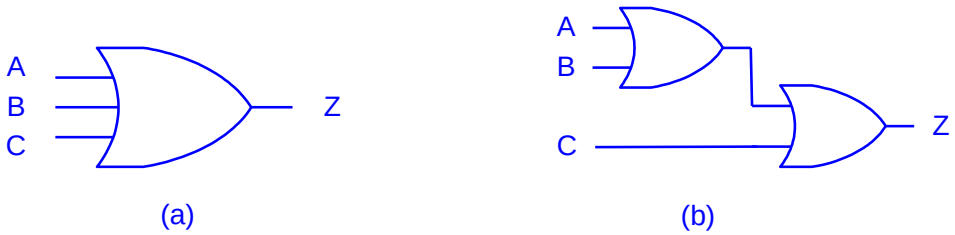


그림 1-4. 3 입력 OR 게이트: (a) 3 입력 OR 게이트. (b) 2 레벨 3 입력 OR 게이트.

다) NOT 게이트

3 번째 기본 게이트는 NOT 게이트로 주어진 입력 값을 반전 또는 보수화 (complement)라 불리는 연산을 수행하는 논리함수로 어떤 주어진 입력 논리 값을 반대의 값으로 변환시킨다. 즉 '1'을 '0'으로 '0'을 '1'로 변환시킨다. 그림 1-5 (c)에서 verilog HDL 로 NOT 게이트 모델을 기술하였다. NOT 게이트의 모듈 명은 NOT 으로 하고 입력신호 A 와 출력신호 B 를 괄호 안에 기술한다. 모듈 내부의 신호의 연결은 보수화 비트 논리 연산자 "~"을 이용하여 기술하였다.

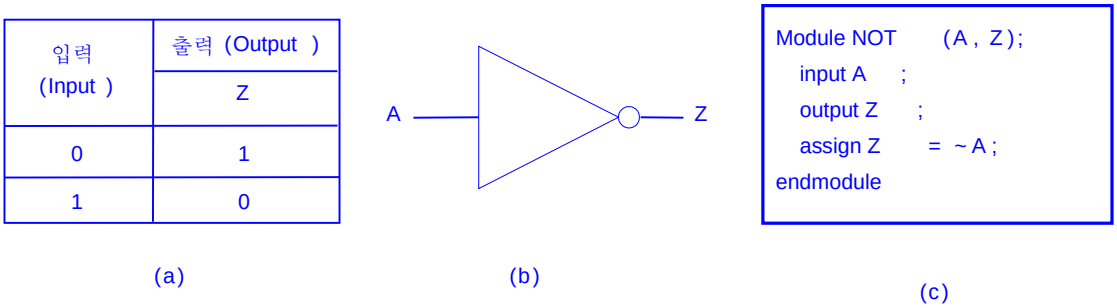


그림 1-5. NOT 게이트 (a) 진리표, (b) NOT 게이트 기호, (c) HDL 표현.

```
module gate(AND_A, AND_B, AND_Z, OR_A, OR_B, OR_Z, NOT_A, NOT_Z);
        input AND_A, AND_B, OR_A, OR_B, NOT_A;
        output AND_Z, OR_Z, NOT_Z;
        assign AND_Z = AND_A & AND_B;
        assign OR_Z = OR_A | OR_B;
        assign NOT_Z = ~NOT_A;
endmodule
```

그림 1-6. 기본 논리 게이트 가)~다)의 Verilog HDL 코드.

라) NAND 게이트

NAND 게이트는 AND 게이트에 NOT 게이트를 결합한 것이다. 모든 입력이 '1'인 경우에만 출력이 '0'이 된다. 또한 NAND 게이트는 자체의 조합만으로도 AND, OR, NOT 등의 연산을 수행할 수 있는 매우 유용한 범용 논리 소자이다. 그림 1-6 은 NAND 게이트의 기호와 진리표를 나타낸다. NAND 게이트 함수를 논리식으로 표현하면 Z = (A . B)'으로 표현된다. 그림 1-6(c)에서 Verilog HDL 로 NAND 게이트를 기술하였다.

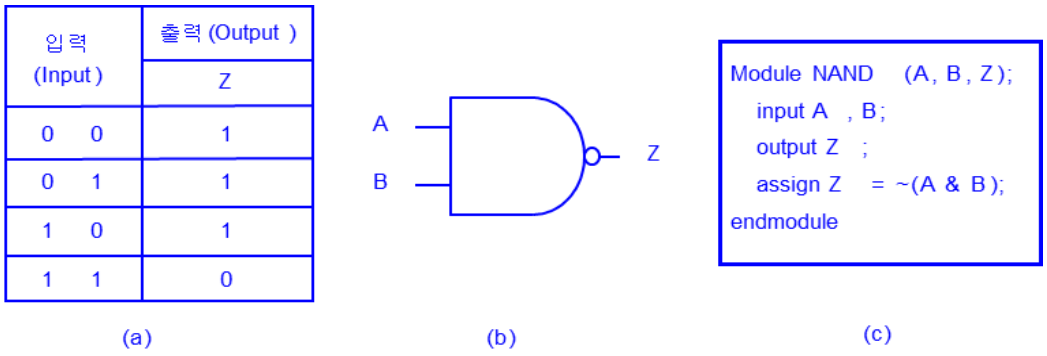


그림 1-6. NAND 게이트 (a) 진리표, (b) NAND 게이트 기호, (c) HDL 코드.

그림 1-6 (c)에서는 NAND 게이트를 구현하는 것으로 모듈명을 NAND 로 하고 입력신호와 출력신호 리스트를 괄호 안에 기술한다. 모듈명과 함께 기술된 신호리스트는 모듈 본체에서

입력신호와 출력신호로 명확하게 구분되어 기술된다. 모듈 내부의 신호의 연결은 할당문을 이용하여 데이터 플로우 모델링 기법으로 기술되어 있다.

마) NOR 게이트

NOR 게이트 역시 범용 게이트로 사용할 수 있는 매우 유용한 논리소자이다. 즉, NOR 게이트의 조합만으로도 AND, OR, NOT 등의 연산을 수행할 수 있다. NOR 게이트는 모든 입력이 '0'인 경우에만 '1'이 출력된다. 그림 2-2 (c)에서 Verilog HDL 로 NOR 게이트를 기술하였다.

그림 1-7 은 NOR 게이트의 기호와 진리표를 나타낸다. NOR 게이트 함수를 논리식으로 표현하면 Z = (A + B)'으로 표현된다. 그림 1-7(c)에서는 NOR 게이트의 구현으로 모듈명은 NOR 로 하고 입력신호와 출력신호 리스트를 괄호 안에 순서대로 기술한다. 신호리스트는 모듈 본체에서 입력신호와 출력신호로 명확하게 구분되어 기술된다. 모듈 내부의 신호의 연결은 할당문을 이용하여 데이터 플로우 모델링기법으로 기술되어 있다.

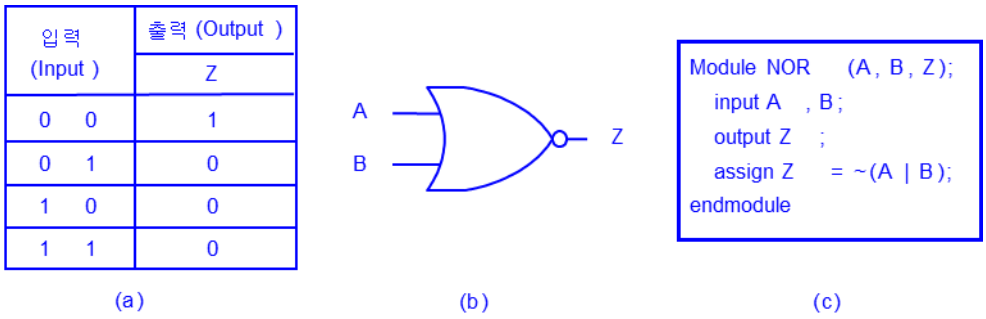


그림 1-7. NOR 게이트 (a) 진리표, (b) NOR 게이트 기호, (c) HDL 코드.

```
module gate(NAND_A, NAND_B, NAND_Z, NOR_A, NOR_B, NOR_Z);
        input NAND_A, NAND_B, NOR_A, NOR_B;
        output NAND_Z, NOR_Z;
        assign NAND_Z = ~(NAND_A & NAND_B);
        assign NOR_Z = ~(NOR_A | NOR_B);
endmodule
```

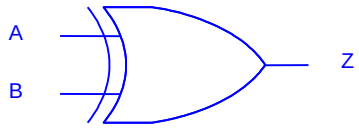
그림 1-8 기본 논리 게이트 라)~마)의 Verilog HDL 코드

바) XOR / XNOR 게이트

XOR 게이트는 두 입력 값이 같으면 '0' 서로 다르면 '1'이 되는 논리 함수이다. 즉, 입력 X 와 Y 에 대하여, X XOR Y = X ・ Y' + X'・ Y 로 기본 AND, OR 게이트의 조합으로 구성된다. 그러나 다양한 응용분야에서 사용 빈도수가 높고, 중요한 역할을 하기 때문에 이 게이트들도 고유 기호를 가진 기본 논리 게이트로 정의하여 사용한다. XOR 게이트의 출력은 두 입력레벨이 서로 반대일 경우에만 '1'이다. 그리고 XNOR 게이트의 출력은 두 입력레벨이 같은 경우에만 '1'을 가진다. 그림 1-9 은 XOR 게이트의 기호와 진리표를 나타낸다. 그림 1-9(c)에서 Verilog HDL 로 XOR 게이트를 기술하였다.

입력 (Input)		출력 (Output)
		Z
0	0	0
0	1	1
1	0	1
1	1	0

(a)



(b)

```
Module XOR(A, B, Z);
   input A, B;
   output Z;
   assign Z = A ^ B;
endmodule
```

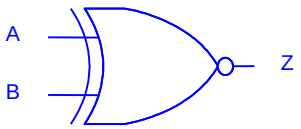
(c)

그림 1-9. XOR 게이트 (a) 진리표, (b) XOR 기호, (c) HDL 코드.

XNOR 게이트는 두 입력 값이 같으면 '1' 서로 다르면 '0'이 되는 논리 함수이다. 즉, 입력 X 와 Y 에 대하여, X XNOR Y = X ・ Y + X'・ Y'로 기본 AND, OR 게이트의 조합으로 구성된다. 그러나 다양한 응용분야에서 사용 빈도수가 높고, 중요한 역할을 하기 때문에 이 게이트들도 고유 기호를 가진 기본 논리 게이트로 정의하여 사용한다 그림 1-10 은 XNOR 게이트의 기호와 진리표를 나타낸다. 그림 1-10(c)에서 Verilog HDL 로 XOR 게이트를 기술하였다.

입력 (Input)		출력 (Output)
		Z
0	0	1
0	1	0
1	0	0
1	1	1

(a)



(b)

```
Module XNOR(A, B, Z);
   input A, B;
   output Z;
   assign Z = A ^~ B;
endmodule
```

(c)

그림 1-10. XNOR 게이트 (a) 진리표, (b) XNOR 기호, (c) HDL 코드.

```
module gate(XOR_A, XOR_B, XOR_Z, XNOR_A, XNOR_B, XNOR_Z);
        input XOR_A, XOR_B, XNOR_A, XNOR_B;
        output XOR_Z, XNOR_Z;
        assign XOR_Z = XOR_A ^ XOR_B;
        assign XNOR_Z = XNOR_A ^~ XNOR_B;
endmodule
```

그림 1-11. 기본 논리 게이트 바)의 Verilog HDL 코드.

3. 예비보고서 작성

▣ 기본 게이트 (AND, OR, NOT, NAND, NOR, XOR, XNOR)의 내부회로 구성도(gate diagram)와 이에 대한 진리표 (truth table) 작성

▣ 각 게이트의 지연시간에 대한 자료를 조사하고, 이에 대해 서술하시오.

▣ AND, OR, NOT 기반 회로 구성과 NAND, NOR 회로 구성의 장 단점 조사 (특히, 이 조사와 관련하여, Universal Gate 가 어떤 의미를 가지는지 고찰하시오.)

▣ XOR, XNOR 게이트의 특성과 그 응용분야에 대해 조사 후, 이에 대해 서술하시오.

4. 실험에 필요한 기기

▣ 실험용 PC

▣ MAX PLUS II 소프트웨어

5. 실험 내용과 과정

① 1-6, 1-8, 1-11 에 해당 되는 기본 게이트 및 [6. 실험 결과 보고서]에서 진행할 ③, ④, ⑥ 실험의 논리 회로를 구상한다.

② MAX PLUS II 를 실행하고 각 모듈들에 해당되는 Verilog 코드 파일들 및 Project 들을 생성한다.

③ MAX PLUS II 의 Text Editor 를 통해 Verilog 코드를 작성한다.

④ MAX PLUS II 상단의 메뉴에서 Compiler 를 선택하여 Verilog 코드를 컴파일 한다.

⑤ MAX PLUS II 상단의 메뉴에서 Waveform Editor 를 선택하여, 입출력 Node 를 등록한 뒤 Timeline 에 맞추어 입력 값을 정한다.

⑥ MAX PLUS II 상단의 메뉴에서 Simulator 를 선택하여 수행한 후, 해당 Verilog 코드가 Waveform Editor 에서 입력한 신호에 맞추어 올바른 결과값을 보이는지 확인한다.

6. 실험 결과 보고서

① 그림 1-2 (a)와 (b) 같이 회로를 구성할 경우, 두 회로는 어떤 차이를 보이는지 분석하시오.

② 그림 1-4 (a)와 (b) 같이 회로를 구성할 경우 두 회로는 어떤 차이를 보이는지 분석하시오.

③ 그림 1-2 / 1-4 (a)와 (b) 같이 Verilog HDL 로 작성하여, Simulation 을 통해 동작을 검증하시오.

④ NAND, NOR 게이트를 조합하여 AND, OR, NOT 게이트의 연산을 수행하도록 회로를 그리고, Verilog HDL 코드를 통해 구현하고, Simulation 을 통해 동작을 검증하시오.

⑤ 2 input XOR 게이트를 이용해서 1bit buffer 를 구현할 수 있는지 설명하시오.

⑥ 예비 보고서 작성 및 실험 진행을 통해 알게 된 사항들을 바탕으로, 아래 그림 1-12 와 같이 A, B 각 1bit 의 입력이 주어질 시, 각 $\overline{A}$, $\overline{B}$ 를 출력하는 회로를 구현하고, 동작을 검증하시오. (단, NOT gate 는 한 번만 사용가능하며, AND, OR, XOR gate 의 사용 횟수에는 제한이 없음)

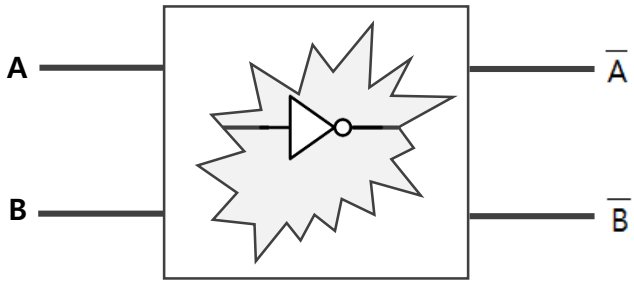


그림 1-12. NOT gate 를 1 개만 사용하는 2-input invertor logic 모듈.