3주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20231561 이름: 심소현

**1.**

.................

대부분의 FPGA는 4입력 논리 함수를 구현할 수 있는 대조표를 사용한다. 대조표의 출력은 논리 블록 출력 중 하나와 멀티플렉서 입력 중 하나와 연결된다. 여기서 멀티플렉서는 여러 개의 입력 중 하나를 선택하여 단일 출력을 하도록 만드는 로직으로 MUX라고 칭하기도 한다. 선택 논리 회로 MUX를 거친 출력은 레지스터 입력으로 이어진다. 레지스터는 High 또는 Low로 활성화할 수 있다. 이때 논리 블록의 출력이 다른 논리 블록의 입력에 연결되도록 프로그래밍하며 이를 조정하며 디바이스 출력을 제어할 수 있다.

................

**2.**

.......................

3-input AND gate의 진리표는 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | Out D | Out E |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

3-input AND gate는 input A와 input B가 AND gate를 지나 output D를 출력한다. 이후 output D가 입력이 되어 input C와 AND gate를 지나 output E를 출력한다. 이때 AND gate는 두 입력이 모두 1인 경우에만 1을 출력하기 때문에 모든 input이 1이어야만 최종 출력인 output E가 1이 될 수 있다.

.........................

**3.**

.......................

4-input AND gate의 진리표는 다음과 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out E | Out F | Out G |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

4-input AND gate는 input A와 input B가 AND gate를 지나 output E를 출력한다. 이후 output E가 입력이 되어 input C와 AND gate를 지나 output F를 출력한다. output F는 input D와 만나 AND gate를 지나 최종 출력인 output G를 만들어낸다. 이때 중간 출력인 output E와 F가 1이 되는 경우들은 존재하지만 이후 만나는 또다른 입력도 1이어야 다음 gate를 지나서도 1이 유지된다. 따라서 AND gate는 두 입력이 모두 1인 경우에만 1을 출력하기 때문에 모든 input이 1이어야만 최종 출력인 output G가 1이 될 수 있다.

.........................

**4.**

.......................

3-input OR gate의 진리표는 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | Out D | Out E |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

3-input OR gate는 input A와 input B가 OR gate를 지나 output D를 출력한다. 이후 output D가 입력이 되어 input C와 OR gate를 지나 output E를 출력한다. OR gate는 입력에 둘 중 하나라도 1이 존재한다면 1을 출력한다. 따라서 모든 입력이 0인 경우를 제외하면 최종 출력인 output E는 항상 1이다.

.........................

**5.**

.......................

4-input OR gate의 진리표는 다음과 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out E | Out F | Out G |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

4-input OR gate는 input A와 input B가 OR gate를 지나 output E를 출력한다. 이후 output E가 입력이 되어 input C와 OR gate를 지나 output F를 출력한다. output F는 input D와 만나 OR gate를 지나 최종 출력인 output G를 만들어낸다. OR gate는 두 입력 중 1이 하나라도 있으면 1을 출력하기 때문에 중간 OR gate의 출력 중 하나가 0이 되더라도 다음에 지나는 OR gate에서 만나는 입력이 1이라면 다음 출력은 1이 된다. 따라서 모든 input이 0인 경우만 제외한다면 최종 출력인 output G는 항상 1이다.

.........................

**6.**

.......................

결과적으로 AND gate의 경우에서 최종 출력을 1로 만들기 위해서는 모든 입력이 1이어야 한다. OR gate에서는 최종 출력을 0으로 만들기 위해서는 모든 입력이 0이어야 한다. 두 gate 모두 앞의 상황을 제외하면 최종 출력은 AND gate에서는 0, OR gate에서는 1만 출력된다.

.........................

**7.**

.......................

추가적으로 NOT gate에서는 입력값을 반전시켜 출력한다. 단순 값 반전 로직이기 때문에 하나의 입력만 받는다. 입력이 1이면 0을 출력하고, 입력이 0이면 1을 출력한다고 볼 수 있다.

.........................