**9주차 예비보고서**

**전공: 경영학과 학년: 3학년 학번: 20190963 이름: 한다현**

**1.**

**디코더는 n개의 입력을 2n개의 서로 다른 정보로 변환시켜주는 조합 회로이다. 디코더의 인풋이 n개라면 디코더의 아웃풋은 2n개가 된다. 즉 디코더의 인풋이 2개라면 아웃풋은 4개가 되고, 인풋이 3개라면 아웃풋은 8개가 된다. 인풋이 a, b로 2개이고 아웃풋이 c, d, e, f로 4개인 디코더를 예로 들어보면, 두 개의 인풋인 a, b와 각각의 부정인 a’, b’을 조합하여 c, d, e, f를 구성한다. 따라서 2\*4 디코더는 총 4개의 AND 게이트로 구성되는데, 각각은 ab, a’b, ab’, a’b’이다. 즉 a와 b가 모두 1이면 c가 출력되고, 모두 0이면 f가 출력되는 형식이다. 이와 비슷하게, 3\*8 디코더는 인풋이 3개이고, 출력이 8개인 디코더를 의미하며, 총 8개의 AND 게이트로 구성되어 입력에 따라 특정 아웃풋을 출력할 수 있다. 디코더에는 앞서 나온 인풋과 아웃풋 외에 enable이라는 입력이 존재한다. Enable은 0 또는 1의 값을 가지며 디코더의 실행 여부를 결정하는 역할을 한다. 즉, Enable이 1이라면 디코더를 실행하여 특정 아웃풋을 출력하지만 enable이 0이라면 디코더를 실행하지 않는다. 이러한 enable은 하나의 디코더 내에서도 사용되지만, 여러 개의 작은 디코더를 조합해 큰 디코더를 설계할 때도 유용하게 사용된다. 4\*16 디코더를 설계하려고 할 때, 16개의 AND 게이트로 구성된 하나의 디코더로 설계할 수도 있지만 2\*4 디코더 5개를 조합하여 설계하는 것도 가능한데, 이 경우에 enable이 중요한 역할을 한다. 2\*4 디코더를 활용한 4\*16 디코더의 구성은 다음과 같다. 처음 2\*4 디코더의 출력은 a~d로 나머지 4개의 디코더의 enable 값을 의미하고, 나머지 네 개의 디코더는 각각 0~3, 4~7, 8~11, 12~15의 출력 값을 갖는다. 만약 첫번째 디코더의 값이 a라면 0~4를 출력하는 디코더의 enable이 1이 된다는 것을 의미하고 이는 0~4를 출력하는 디코더가 실행되도록 한다. 첫번째 디코더 값이 c라면, 8~11을 출력하는 디코더의 enable이 1이 된다는 것을 의미하며 이는 8~11을 출력하는 디코더가 실행되어 8~11 중 하나의 값이 출력되도록 작동한다. 이러한 방식으로 enable을 이용해 다양한 방법으로 디코더를 설계할 수 있다.**

**2.**

**인코더는 디코더의 기능과 반대의 동작을 수행하는 회로이며, 2n개의 입력과 n개의 출력을 갖는다. 즉 디코더와 반대로 8개의 인풋이 있으면 3개의 아웃풋을 출력하고, 4개의 인풋이 있으면 2개의 아웃풋을 출력한다. 인코더는 2n개의 입력 중 활성화된 입력에 따라 출력 값이 정해진다. 인풋이 4개이고 아웃풋이 2개인 4\*2 인코더를 예로 들면, 입력 a, b, c, d가 있고 출력 x, y가 있을 때, a의 값이 1이 되고 나머지가 0일 때는 x, y이 값이 0,0이 되고, b가 1이고 나머지가 0일 때는 x, y 값이 0,1이 되고, c가 1이고 나머지가 0일 때는 x, y 값이 1,0이 되고, d가 1이고 나머지가 0일 때는 x, y 값이 1,1이 된다. 이렇듯 4\*2 인코더에서는 4개의 인풋 중 하나가 활성화됨에 따라 그에 맞는 x, y 값이 결정된다. 이 때 2개의 출력 값을 결정해야 하므로 2개의 OR 게이트가 사용된다. 4개의 인풋 중 하나가 활성화되어야 출력 값이 결정되기 때문에 AND 게이트가 아닌 OR 게이트를 사용한다. 이러한 방식으로 8\*3 인코더를 간단하게 살펴보면, 인코더는 D0부터 D7까지 8개의 입력과 x, y, z 3개의 출력으로 구성되어 있고 8개의 입력 중 하나의 입력이 활성화되면 활성화된 입력이 무엇이냐에 따라 출력 값이 결정된다. 이 경우 3개의 출력이 있기 때문에 3개의 OR 게이트가 사용된다. 이러한 인코더는 10진수를 2진수로 변환하는 장치, 정보 전송을 일정한 규칙에 따라 암호로 변환하는 장치 등에 유용하게 사용된다.**

**3.**

**멀티플렉싱이란 여러 개의 데이터를 소수의 채널을 통해서 선택적으로 전송하는 것을 의미한다. Multiplexer란 여러 개의 입력선 중 하나를 선택하여 출력선에 연결하는 조합논리회로이다. 멀티플렉서는 일반적으로 2n개의 입력선과 n개의 선택선으로 구성되어 있다. 즉 입력선이 2개면 선택선은 하나이고, 입력선이 4개면 선택선은 2개이며, 입력선이 8개면 선택선은 3개가 필요하다. 이 때 n개의 선택선의 조합에 따라 입력 중 하나를 선택하여 출력선에 전달하여 출력한다. 이렇듯 여러 개의 입력 중 하나를 선택하여 출력선에 전달하기 때문에 data selector라고도 불린다. 4\*1 멀티플렉서를 예로 들어보자. 이 멀티플렉서에는 D0~D3의 4개의 입력선과 2개의 선택선, 하나의 출력선이 존재한다. 선택선 2개의 조합은 (0,0), (0,1), (1,0), (1,1)로 총 4개의 조합이 존재하며 이 조합에 따라 각각 D0부터 D3 중 하나의 입력이 선택되어 출력된다. 마찬가지로 8\*1 멀티플렉서에는 D0~D7까지 8개의 입력선과 3개의 선택선, 그리고 하나의 출력선이 존재한다. 선택선 3개의 조합은 (0, 0, 0)부터 (1, 1, 1)까지 총 8개의 조합이 존재하며 각각의 조합에 따라 D0부터 D7까지의 입력선 중 하나가 선택되어 출력선에 전달된다.**

**4.**

**Priority encoder는 인코더의 한 종류로 인코더의 인풋에 우선 순위가 정해져 있는 인코더를 의미한다. 앞서 조사했던 것처럼 인코더는 N개의 입력 중 하나의 입력이 활성화되면 결과가 출력된다. 하지만 N개의 입력 중 하나가 아닌 2개 이상의 입력이 동시에 활성화되면 에러가 발생한다. 이러한 에러를 해결하기 위해 입력에 우선 순위를 정하여 여러 개의 입력이 동시에 활성화되었을 때 우선 순위가 높은 입력 값에 해당하는 아웃풋을 출력하는 회로가 Priority encoder이다. Priority encoder에는 don’t care가 사용되는데, 그 이유는 예를 통해 알아보고자 한다. 4\*2 priority encoder를 살펴보면, 인코더의 인풋은 D3~D0까지 4개가 있고, 출력은 x, y가 있다. D3~D0의 우선순위는 D3가 가장 높고 D0이 가장 낮다고 가정한다. 우선순위가 가장 낮은 D0가 활성화됨에 따른 결과를 출력하기 위해서는 D0를 제외한 나머지 입력이 모두 0이 되어야 한다. 반면 우선 순위가 가장 높은 D3가 활성화됨에 따른 결과를 출력하기 위해서는 D3보다 우선순위가 낮은 입력의 값이 0이 되든 1이 되든 그 값은 출력과 상관이 없다. 다른 모든 입력이 모두 1이 되더라도 D3의 우선 순위가 가장 높기 때문에 결과적으로 D3에 해당하는 아웃풋이 출력될 것이다. 따라서 D3의 값이 1일 때는 나머지 3개의 입력의 값은 don’t care가 된다. 마찬가지로 D2가 활성화됨에 따른 결과를 출력하기 위해서는 D2보다 우선순위가 높은 D3의 값은 반드시 0이 되어야한다. D3가 활성화된다면 인코더는 D3에 해당하는 결과를 출력하기 때문이다. 반면 D2보다 우선순위가 낮은 D1과 D0의 값은 두 값이 모두 1이 되더라도 우선 순위가 더 높은 D3에 해당하는 결과가 출력되기 때문에 두 입력의 활성화 여부는 중요하지 않다. 따라서 D1과 D0의 값은 don’t care가 된다. 이렇듯 priority encoder는 기존 인코더가 갖는 여러 개의 입력이 동시에 활성화될 때 발생하는 에러를 방지할 수 있는 인코더이다.**

**5.**

**디멀티플렉서는 멀티플렉서와 반대의 기능을 수행한다. 즉, 하나의 입력을 전달받아 여러 개여 출력으로 분배하는 조합논리회로이다. 디멀티플렉서는 하나의 입력선과 여러 개의 출력선을 갖는데, 출력선의 개수는 디멀티플렉서의 선택선의 개수에 따라 결정된다. 선택선의 수가 1개이면 출력선의 수는 2개이고, 선택선의 수가 2개이면 출력선의 수는 4개이며, 선택선의 수가 3개이면 출력선의 수는 8개가 되는 방식이다. 디멀티플렉서는 선택선을 조합한 값에 따라 출력될 하나의 출력선을 결정한다. 1\*4 디멀티플렉서를 예로 들어보면, 4\*1 디멀티플렉서는 전달받은 1개의 입력 D와 두 개의 선택선, 그리고 4개의 출력선으로 구성되어 있다. 선택선의 조합은 (0,0), (0,1), (1,0), (1,1)로 총 4개가 있고, 각각은 출력선 Y0~Y3와 연결되어 있다. 따라서 선택선의 조합에 따라 전달받은 데이터 D를 4개의 출력선 중 하나를 통해 출력할 수 있다. 이는 앞서 조사했던 디코더와 유사한 기능을 수행하는 것을 알 수 있다. 하지만 디코더와 디멀티플렉서는 약간의 차이점이 있는데, 2\*4 디코더와 1\*4 디멀티플렉서를 비교해보면, 디코더는 2개의 입력과 4개의 출력으로 구성되어 있고 2개의 입력의 조합에 따라 그 조합에 해당하는 출력이 결정된다. 반면 디멀티플렉서는 입력 받는 데이터는 하나이다. 출력을 결정하는 것은 입력 받은 데이터가 아니라, 디멀티플렉서 내에 있는 2개의 선택선이다. 디멀티플렉서는 두 개의 선택선의 조합에 따라 입력 받은 데이터를 그에 해당하는 출력선에 전달하는 기능을 하기 때문에 출력을 결정하는 부분에서 디코더와 디멀티플렉서는 차이점이 존재한다고 할 수 있다. 또한 디멀티플렉서에는 정보 입력선이 존재하는 반면 디코더에는 정보 입력선이 존재하지 않기 때문에 차이점이 있다. 하지만 앞서 조사했던 바와 같이, 디코더에 enable이 사용되기도 하는데, enable이 있는 디코더의 경우 enable을 디멀티플렉서의 정보 입력선이라고 본다면 enable이 있는 디코더는 디멀티플렉서라고 볼 수 있다.**

**6.**

**디코더는 ROM을 설계할 때 사용되기도 한다. ROM이란 read only memory의 약자로, 저장된 데이터를 읽을 수는 있지만 특별한 장치 없이는 데이터를 기록하거나 변경할 수 없는 장치이다. ROM은 디코더와 OR 게이트를 연결시킴으로써 설계할 수 있다. 예를 들어 16\*4 ROM을 살펴보면, 이 ROM은 4\*16 디코더와 4개의 OR 게이트로 구성되어 있다. 4개의 입력선의 조합을 메모리의 주소라고 부르고 이 주소에 대응하는 데이터를 워드라고 부른다. 디코더의 출력선이 16개이므로 총 16개의 워드가 ROM에 저장되어 있다. 만약 입력 주소가(입력선의 조합) 0000이라면 0으로 주소가 지정된 워드가 선택되어 출력될 것이고, 입력 주소가 1111이면 주소가 15로 지정된 워드가 선택되어 출력될 것이다. 이 때 ROM이 4개의 OR 게이트로 이루어져 있기 때문에 워드는 4비트의 데이터이고, 만약 OR 게이트의 개수가 4개가 아니라 8개라면 이 ROM의 워드는 8비트의 데이터일 것이다. ROM을 표현할 때는 워드의 개수와 워드의 크기로 표시하는데, 앞서 예로 들었던 16\*4 ROM은 16개의 워드가 저장되어 있고 그 워드의 크기가 4비트라는 뜻이며, 이는 16\*4 ROM 대신 64 ROM이라고 표시하기도 한다.**

**참조:**

**디지털 논리회로(임석구, 홍경호)**

**디지털 공학(조현철)**

**디지털 논리회로 설계(이상범)**