9주차 결과보고서

전공: 컴퓨터공학과/생명과학과 학년: 9학기 학번: 20181435 이름: 박다희

**1.**

**텍스트, 영수증, 폰트, 대수학이(가) 표시된 사진

자동 생성된 설명텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명**

**스크린샷, 디스플레이, 멀티미디어 소프트웨어, 텍스트이(가) 표시된 사진

자동 생성된 설명도표, 평면도, 텍스트이(가) 표시된 사진

자동 생성된 설명**

위의 사진은 차례대로 2 to 4 decoder(active high)의 test bench, design source, simulation, schematic diagram의 결과이다. 2 to 4 decoder(active high)의 진리표는 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | | **Output** | | | |
| **A** | **B** | **D0** | **D1** | **D2** | **D3** |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

진리표에 대한 카르노 맵 결과는 다음과 같다.

D0의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 1 | 0 |
| **1** | 0 | 0 |

D0 = A’B’

D1의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 0 | 1 |
| **1** | 0 | 0 |

D1 = A’B

D2의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 0 | 0 |
| **1** | 1 | 0 |

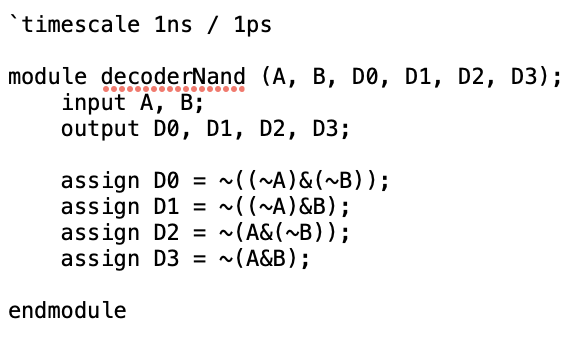
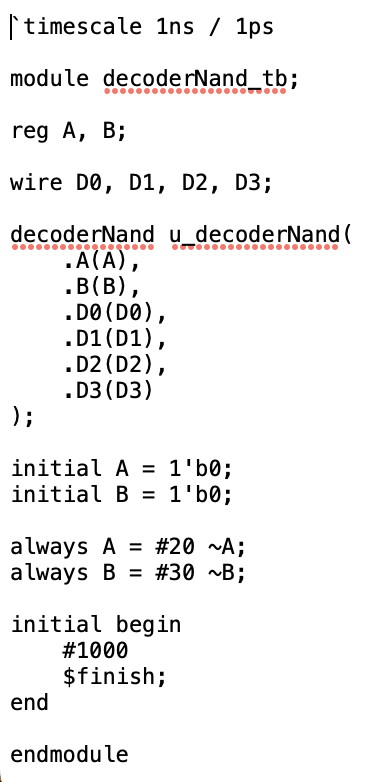
D2 = AB’

D3의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 0 | 0 |
| **1** | 0 | 1 |

D3 = AB

2 to 4 decoder의 active high는 활성화 된 것을 1(high)로 나타내는 decoder로 카르노 맵을 통한 결과를 살펴보면 알 수 있듯이 A’B’, A’B, AB’, AB라는 논리식으로 각각의 출력 값을 구할 수 있으므로 Verilog 코드를 짤 때도 AND연산과 NOT연산만을 이용하여 짰다. Schematic diagram을 보면 알 수 있다시피 AND gate 4개와 NOT gate 2개를 이용하여 구현했고 입력 값에 대한 출력 값을 simulation결과를 통해 확인해보면 잘 나오는 것을 알 수 있다.

****

**스크린샷, 컴퓨터, 전자 기기, 전자제품이(가) 표시된 사진

자동 생성된 설명도표, 텍스트, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명**

위의 사진은 차례대로 2 to 4 decoder(active low)의 test bench, design source, simulation, schematic diagram의 결과이다. 2 to 4 decoder(active low)의 진리표는 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | | **Output** | | | |
| **A** | **B** | **D0** | **D1** | **D2** | **D3** |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

진리표에 대한 카르노 맵 결과는 다음과 같다.

D0의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 0 | 1 |
| **1** | 1 | 1 |

D0 = A + B = (A’B’)’

D1의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 1 | 0 |
| **1** | 1 | 1 |

D1 = A + B’ = (A’B)’

D2의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 1 | 1 |
| **1** | 0 | 1 |

D2 = A’ + B = (AB’)’

D3의 카르노 맵은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **A/B** | **0** | **1** |
| **0** | 1 | 1 |
| **1** | 1 | 0 |

D3 = A’ + B’ = (AB)’

2 to 4 decoder의 active low는 활성화 된 것은 0(low)로 나타내는 decoder로 카르노 맵을 통한 결과를 살펴보면 알 수 있듯이 (A’B’)’, (A’B)’, (AB’)’, (AB)’라는 논리식으로 각각의 출력 값을 구할 수 있으므로 Verilog 코드를 짤 때 NAND 연산을 이용하여 짰다. Schematic diagram을 보면 알 수 있다시피 NAND gate을 이용하여 구현했고 입력 값에 대한 출력 값을 simulation결과를 통해 확인해보면 잘 나오는 것을 알 수 있다.

**2.**

**텍스트, 폰트, 스크린샷, 영수증이(가) 표시된 사진

자동 생성된 설명텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명**

**스크린샷, 컴퓨터, 다채로움이(가) 표시된 사진

자동 생성된 설명도표, 평면도이(가) 표시된 사진

자동 생성된 설명**

위의 사진은 차례대로 4 to 2 encoder의 test bench, design source, simulation, schematic diagram의 결과이다. 4 to 2 encoder 의 진리표는 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | | | | **Output** | |
| A | B | C | D | E0 | E1 |
| 0 | 0 | 0 | 0 | x | x |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | x | x |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | x | x |
| 0 | 1 | 1 | 0 | x | x |
| 0 | 1 | 1 | 1 | x | x |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | x | x |
| 1 | 0 | 1 | 0 | x | x |
| 1 | 0 | 1 | 1 | x | x |
| 1 | 1 | 0 | 0 | x | x |
| 1 | 1 | 0 | 1 | x | x |
| 1 | 1 | 1 | 0 | x | x |
| 1 | 1 | 1 | 1 | x | x |

진리표에 대한 카르노 맵 결과는 다음과 같다.

E0의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AB/CD** | **00** | **01** | **11** | **10** |
| **00** | x | 0 | x | 0 |
| **01** | 1 | x | x | x |
| **11** | x | x | x | x |
| **10** | 1 | x | x | x |

E0 = A + B

E1의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AB/CD** | **00** | **01** | **11** | **10** |
| **00** | x | 0 | x | 1 |
| **01** | 0 | x | x | x |
| **11** | x | x | x | x |
| **10** | 1 | x | x | x |

E1 = A + C

앞서 구현하였던 decoder의 출력 값을 입력 값으로 받고, decoder에서의 입력 값을 출력 값으로 내보내는 4 to 2 encoder로, 입력 값 A, B, C, D는 앞선 decoder의 출력 값들과 같다고 가정하고, 출력 값인 E0과 E1은 MSB를 E0로 하는 2-bit의 이진수라고 가정하였다. 이에 따라 ABCD의 값이 0001, 0010, 0100, 1000일 때 출력 값인 E0E1은 각각 00, 01, 10, 11의 값을 갖도록 E0은 A와 B의 논리 합 연산, E1은 A와 C의 논리 합 연산으로 코딩하였다. Simulation의 결과를 확인해보면 입력 값이 0001, 0010, 0100, 1000일 때의 출력 값이 00, 01, 10, 11로 잘 나온다는 것을 확인할 수 있고, 4개의 입력 값 외에도 다른 입력 값들에 대한 출력 값들이 존재한다는 것도 알 수 있다.

**3.**

Encoder에서는 입력되는 여러 bit들 중에서 한 개의 bit만 1의 값을 가지고 남은 bit들은 0의 값을 가지는 형태를 띠거나 또는 한 개의 bit만 0의 값을 가지고 나머지 bit들은 1의 값을 가져야 한다. 이러한 상태가 아닌 값들이 질문에서 언급하는 나머지 입력 형태이다. 그래서 2번 질문에서의 진리표를 보면 입력 값 중에서 1개만 1의 값을 가지고, 모두 0을 나타낼 때만 0의 값을 가지며 나머지 값들은 don’t care로 나타내었다. 나머지 입력 형태와 같은 경우는 데이터를 전송할 때에 오류로 발생할 수 있으나 priority encoder을 사용한다면 이러한 입력 값에 대해서도 출력 값을 받을 수 있다.

**4.**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명**

소프트웨어, 멀티미디어 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷, 텍스트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위의 사진은 차례대로 priority encoder의 test bench, design source, simulation, schematic diagram의 결과이다 A, B, C, D을 차례대로 우선순위가 높은 순으로 고려하여 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Input** | | | | **Output** | | |
| **A** | **B** | **C** | **D** | **X** | **Y** | **V** |
| 0 | 0 | 0 | 0 | X | X | 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 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 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 |

우선순위가 더 높다는 것은 만약 ABCD가 1010의 입력 값으로 들어왔을 때 A와 C의 값이 1로 들어왔지만 A의 우선순위가 C보다 더 크므로 1000으로 처리하겠다는 의미이다. 이렇게 우선순위를 두고 처리하면 입력되는 bit가 한 개의 값만이 1 또는 0을 갖지 않더라도 오류 없이 출력 값을 나타낼 수 있다. 출력 값 중에서 V는 입력 값의 유효성을 판단하는데, 입력 값들이 모두 0이면 0을 가지고 그 외의 경우에는 1을 가진다. 따라서 V값이 0을 가질 때는 X, Y의 값은 don’t care가 된다.

**5.**

**텍스트, 패턴이(가) 표시된 사진

자동 생성된 설명텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명**

**스크린샷, 컴퓨터이(가) 표시된 사진

자동 생성된 설명텍스트, 도표, 라인이(가) 표시된 사진

자동 생성된 설명**

위의 사진은 차례대로 BCD to decimal decoder의 test bench, design source, simulation, schematic diagram의 결과이다. BCD to decimal decoder의 진리표는 다음과 같다.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A0 | A1 | A2 | A3 | O1 | O2 | O3 | O4 | O5 | O6 | O7 | O8 | O9 |
| 0 | 0 | 0 | 0 | X | X | X | X | X | X | X | X | X |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | X | X | X | X | X | X | X | X | X |
| 1 | 0 | 1 | 1 | X | X | X | X | X | X | X | X | X |
| 1 | 1 | 0 | 0 | X | X | X | X | X | X | X | X | X |
| 1 | 1 | 0 | 1 | X | X | X | X | X | X | X | X | X |
| 1 | 1 | 1 | 0 | X | X | X | X | X | X | X | X | X |
| 1 | 1 | 1 | 1 | X | X | X | X | X | X | X | X | X |

위의 진리표를 바탕으로 한 카르노 맵의 결과는 다음과 같다.

O1의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 1 | 0 | 0 |
| **01** | 0 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O2의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 1 |
| **01** | 0 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O3의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 1 | 0 |
| **01** | 0 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O4의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 1 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O5의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 0 | 1 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O6의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 0 | 0 | 0 | 1 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O7의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 0 | 0 | 1 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 0 | X | X |

O8의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 0 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 1 | 0 | X | X |

O9의 카르노 맵은 다음과 같다.

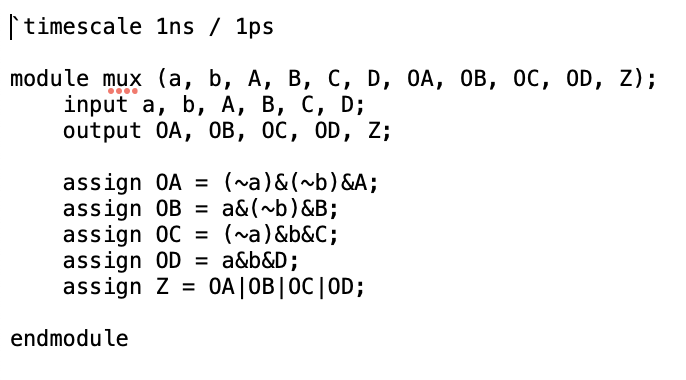
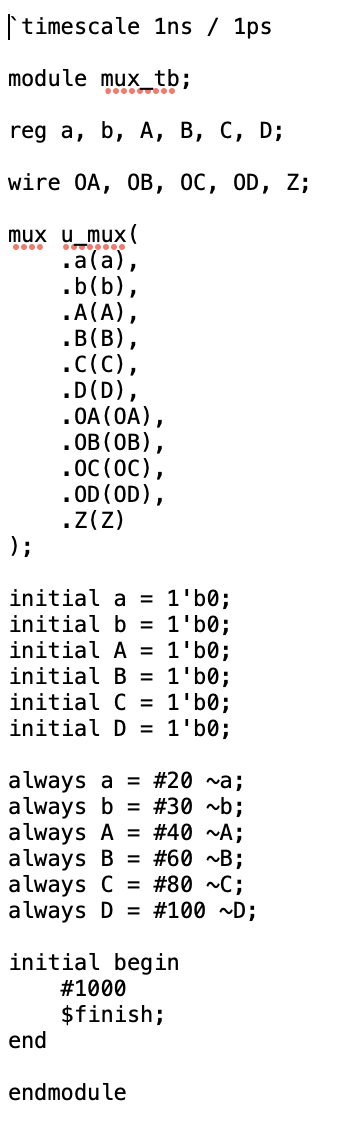
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A3A2/A1A0** | **00** | **01** | **10** | **11** |
| **00** | X | 0 | 0 | 0 |
| **01** | 0 | 0 | 0 | 0 |
| **10** | X | X | X | X |
| **11** | 0 | 1 | X | X |

이는 BCD to Decimal Decoder는 입력 값으로 8421 BCD code을 받아서 이에 해당하는 출력 값을 내보내는 것으로 입력 값 A0, A1, A2, A3는 각각 1, 2, 4, 8을 의미한다. 만약 입력 값으로 0110이 들어온다면 이는 2, 4에 해당하는 bit가 설정된 것이므로 6을 의미하고, 출력 값으로는 000001000이 된다. 이에 따라 각각의 출력 값에 알맞게 논리 식을 구현하였고 이는 simulation의 결과를 통해서도 확인할 수 있다.

**6.**

Encoder와 decoder의 주요 응용은 여러 곳에서 발견할 수 있는데 크게 동영상 파일 압축, 디지털화, 7-segment, 통신 보안 등이 있다. 먼저 동영상 파일 압축에서는 압축되지 않은 파일을 encoder가 변환 알고리즘을 통해 압축할 수 있고, 이는 다시 decoder가 변환 알고리즘을 통해 압축을 풀 수 있다. 디지털화는 encoder가 디지털 회로를 통해 아날로그 신호를 디지털 신호로 변환할 수 있고, decoder도 마찬가지로 디지털 회로를 통해 디지털 신호를 다시 아날로그 신호로 변환할 수 있다. 7-segment에서는 encoder가 디지털 회로를 통해 7-segment 숫자를 input code로 변환시킬 수 있으며 반대로 decoder가 디지털 회로를 통해 input code을 다시 7-segment 숫자로 변환시킬 수 있다. 마지막으로 통신 보안에서는 encoder가 암호화를 통해 신호를 암호화할 수 있고, decoder을 통해 암호화된 신호를 암호화 되지 않은 신호로 변환시킬 수 있다.

**7.**

****

**스크린샷이(가) 표시된 사진

자동 생성된 설명텍스트, 도표, 평면도, 기술 도면이(가) 표시된 사진

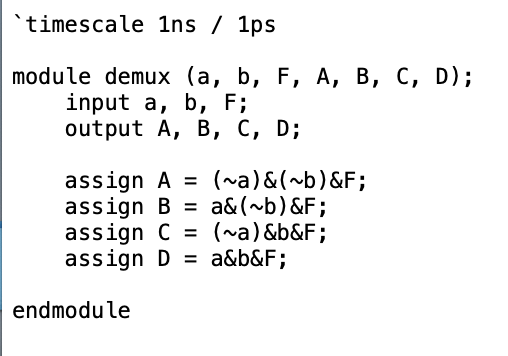
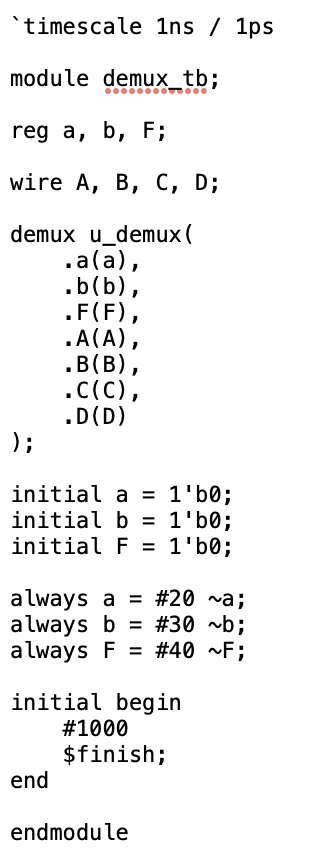
자동 생성된 설명**

위의 사진은 차례대로 4 to 1 line MUX의 test bench, design source, simulation, schematic diagram의 결과이다. 4 to 1 line MUX의 진리표는 다음과 같다.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | A | B | C | D | OA | OB | OC | OD | Z |
| 0 | 0 | 1 | X | X | X | 1 | X | X | X | 1 |
| 0 | 0 | 0 | X | X | X | 0 | X | X | X | 0 |
| 0 | 1 | X | 1 | X | X | X | 1 | X | X | 1 |
| 0 | 1 | X | 0 | X | X | X | 0 | X | X | 0 |
| 1 | 0 | X | X | 1 | X | X | X | 1 | X | 1 |
| 1 | 0 | X | X | 0 | X | X | X | 0 | X | 0 |
| 1 | 1 | X | X | X | 1 | X | X | X | 1 | 1 |
| 1 | 1 | X | X | X | 0 | X | X | X | 0 | 0 |

입력 값들 중에서 a, b는 select line을 의미하는 입력 값으로 ab의 값이 00이면 A, 01이면 B, 10이면 C, 11이면 D에 의해 출력 값이 결정된다. 이에 따라 각각의 입력 값에 해당하는 출력 값 OA는 a’b’A, OB는 a’bB, OC는 ab’C, OD는 abD고 표현하였고 따라서 최종 출력 값을 의미하는 Z는 OA+OB+OC+OD고 나타낼 수 있다. 진리표를 작성할 때도 모든 경우를 다 작성하면 너무 많기에 간단히 표현하기 위해서 don’t care을 의미하는 X을 사용하여 작성하였고 이에 따른 결과는 simulation의 결과로도 확인할 수 있다. 입력 값 a, b가 00일 때는 입력 값 A와 최종 출력 값인 Z가 같음을 확인할 수 있고, a, b가 01일 때는 입력 값 B와 Z가, a,b가 10일 때는 C와 Z가, a,b가 11일 때는 D와 Z가 같음을 확인할 수 있다.

**8.**

****

**스크린샷, 디스플레이, 텍스트, 디자인이(가) 표시된 사진

자동 생성된 설명텍스트, 도표, 라인이(가) 표시된 사진

자동 생성된 설명**

위의 사진은 차례대로 1 to 4 line deMUX의 test bench, design source, simulation, schematic diagram의 결과이다. 1 to 4 line deMUX의 진리표는 다음과 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| a | b | F | A | B | C | D |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |

1 to 4 line deMUX는 입력 값으로 a, b인 select signal과 data 값인 F를 가진다. 출력 값으로는 어떻게 select signal이 들어오는지에 따라 1개의 출력 값만이 1의 값을 갖게 되는데 입력 값인 F의 값이 1의 값을 가질 때에만 이 동작이 수행되고 0의 값을 가질 때에는 다른 입력 값들은 상관 없이 출력 값은 0000이 된다. 따라서 simulation의 결과와 같이 살펴보면 ab가 00이면 입력 값 F가 출력 값 A에 반영되고, ab가 01이면 F가 B에 반영되고, ab가 10이면 F가 C에 반영되고, ab가 11이면 F가 D에 반영된다는 것을 확인할 수 있다. 따라서 Verilog 코드로는 출력 값 A, B, C, D가 각각 a’b’F, a’bF, ab’F, abF로 구현될 수 있다.

**9.**

이번 실험에서는 2 to 4 Decoder(active high), 2 to 4 Decoder(active low), 4 to 2 Encoder, Priority Encoder, BCD to decimal decoder , 4 to 1 line MUX, 1 to 4 line deMUX에 대해 알아보았다. Mux, deMUX를 제외하고는 각각 진리표를 작성하고 카르노 맵을 통하여 식을 간소화시켰다. 2 to 4 decoder는 AND, NOT, NAND gate를 사용하여 두 종류의 decoder를 구현해봤고, 이 후 encoder도 구현해보면서 encoder와 decoder가 서로 반대됨을 직접 확인하였다. Priority encoder도 구현하면서 우선순위가 높은 입력에 출력 값이 결정되는 것을 확인하였고, BCD to decimal decoder에서는 9개의 출력 값들 중에서 입력 값에 따라 한 개의 출력 값만이 1의 값을 갖는 다는 것을 확인하였다. 입력 값이 4-bit면 총 16개의 값을 표현할 수 있으므로 출력 값도 최대 16개까지 만들 수 있다는 점도 알 수 있었다. 4 to 1 line MUX는 4개의 입력 값들 중에서 한 개만 선택하는 스위치 역할을 하는 것을 확인할 수 있었고, 반대로 1 to 4 line deMUX는 4개의 출력 값들 중에서 한 개의 출력 선만을 선택하여 입력 값을 전달할 수 있다는 것도 확인하였다.

**10.**

Coder-Decoder의 약어로 잘 알려져 있는 코덱은 영상이나 음성을 컴퓨터가 처리할 수 있는 디지털 신호로 변환시켜 주는 coder와 이를 다시 사람이 이해할 수 있는 아날로그 신호로 변환시켜 주는 decoder의 기능을 동시에 갖춘 기술이다. 본래 크기의 음성 매체나 영상들은 그 크기가 매우 크기 때문에 코덱을 통해 이 데이터들을 압축시켜서 파일의 크기를 줄여주거나 또는 압축을 풀어서 본래의 크기로 돌려놓고 재생할 수 있도록 해준다. 코덱에는 크게 2가지 종류가 있는데 하나는 오디오 코덱이고 나머지 하나는 비디오 코덱이다. 오디오 코덱의 대표적인 예시로는 MP3, AAC, AC3, FLAC, OGG가 있고 비디오 코덱의 대표적인 예시로는 DivX, MPEG, Xvid, ASF 등이 있다. 이 뿐만 아니라 코덱을 압축의 품질로 종류를 나눈다면 손실 코덱과 비손실 코덱으로로 나눌 수 있는데 먼저 비손실 코덱은 압축된 파일이 원래의 파일에 있는 모든 정보를 갖고 있는 코덱이고, 손실 코덱은 사람이 잘 알아낼 수 없는 부분을 먼저 손실시켜서 압축률을 높이는 코덱이다. 보통 오디오 코덱과 비디오 코덱은 손실 코덱을 사용하고 비손실 코덱은 빠른 처리가 필요할 때 사용한다.