7주차 결과보고서

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

**1.**

텍스트, 영수증, 폰트, 스크린샷이(가) 표시된 사진

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

자동 생성된 설명

스크린샷, 디스플레이, 사각형, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 도표, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

위의 사진들은 차례대로 even parity bit generator의 test bench, design source, simulation, schematic diagram의 결과이다. 이를 바탕으로 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **E** |
| 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 |

위의 진리표를 바탕으로 카르노맵을 작성하면 다음과 같다.

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

E = A’B’CD’ + A’B’C’D + A’BC’D’ + A’BCD + ABCD’ + ABC’D + AB’CD + AB’C’D’

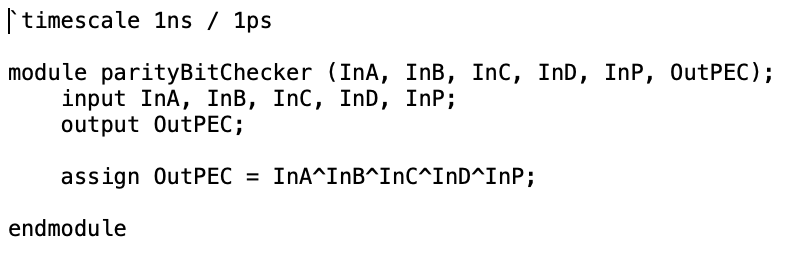
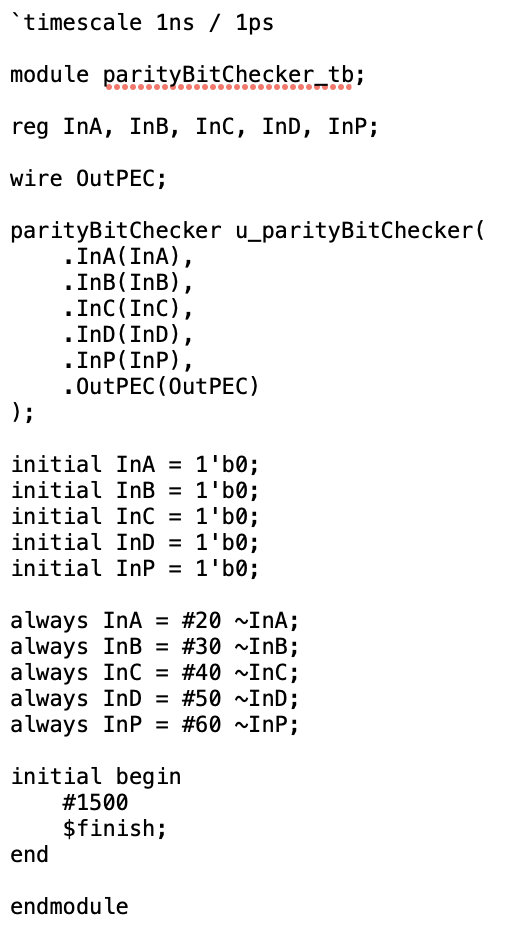
= A’B’(C’D+CD’) + A’B(C’D’+CD) + AB(C’D+CD’) + AB’(C’D’+CD)

= A’B’(CD) + A’B(CD)’ + AB(CD) + AB’(CD)’

= (CD)(AB)’+(CD)’(AB)

= ABCD

4bit의 데이터를 전송할 때 even parity bit generator을 사용하면 입력 값이 1인 입력 값의 개수가 만약 짝수개라면 even parity bit가 0이 되고, 홀수개라면 even parity bit가 1이 되므로 even parity bit generator를 구현할 때는 XOR gate를 이용하여 even parity bit의 값을 결정하였다. 이는 simulation결과와 카르노맵의 결과를 통하여 올바르게 verilog를 작성했다는 것을 알 수 있다.



스크린샷, 디스플레이, 사각형, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 도표, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

위의 사진들은 차례대로 even parity bit checker의 test bench, design source, simulation, schematic diagram의 결과이다. 이를 바탕으로 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **P** | **PEC** |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

위의 진리표를 바탕으로 먼저 A가 0일 때의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BC / DP** | **00** | **01** | **11** | **10** |
| **00** | 0 | 1 | 0 | 1 |
| **01** | 1 | 0 | 1 | 0 |
| **11** | 0 | 1 | 0 | 1 |
| **10** | 1 | 0 | 1 | 0 |

그 다음으로 A가 1일 때의 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BC / DP** | **00** | **01** | **11** | **10** |
| **00** | 1 | 0 | 1 | 0 |
| **01** | 0 | 1 | 0 | 1 |
| **11** | 1 | 0 | 1 | 0 |
| **10** | 0 | 1 | 0 | 1 |

PEC = A’B’C’D’P + A’B’C’DP’ + A’B’CD’P’ + A’B’CDP + A’BCD’P + A’BCDP’ + A’BC’D’P’ + A’BC’DP

= ABCDP

4bit의 데이터를 전송할 때 even parity bit도 전송하면 even parity bit checker는 모든 입력 값들 중에서 1을 가지는 입력 값의 개수가 홀수개라면 오류를 의미하는 1를, 짝수개라면 오류가 없음을 의미하는 0을 출력 값으로 가진다. 따라서 이를 코딩할 때도 앞서 even parity bit generator와 마찬가지로 XOR gate를 사용하여 구현하였고, 이는 simulation결과와 카르노 맵의 결과와 일치함을 알 수 있다.

**2.**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

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

자동 생성된 설명**

소프트웨어, 스크린샷, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위의 사진들은 차례대로 odd parity bit generator의 test bench, design source, simulation 의 결과이다. 이를 바탕으로 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **P** |
| 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 |

이를 바탕으로 작성한 카르노맵은 다음과 같다.

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

P = A’B’C’D’ + A’BC’D + A’B’CD + A’BCD’ + ABC’D’ + AB’C’D + ABCD + AB’CD’

= (ABCD)’

4bit의 데이터를 전송할 때 odd parity bit generator을 사용하면 입력 값이 1인 입력 값의 개수가 만약 짝수개라면 odd parity bit가 1이 되고, 홀수개라면 odd parity bit가 0이 되므로 odd parity bit generator를 구현할 때는 XOR gate와 NOT gate를 이용하여 odd parity bit의 값을 결정하였다. 이는 simulation결과와 카르노맵의 결과를 통하여 올바르게 verilog를 작성했다는 것을 알 수 있다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

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

자동 생성된 설명

소프트웨어, 멀티미디어 소프트웨어, 그래픽 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

위의 사진들은 차례대로 odd parity bit checker의 test bench, design source, simulation 의 결과이다. 이를 바탕으로 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **P** | **PEC** |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

이를 바탕으로 먼저 A가 0일 때의 카르노맵을 위한 표를 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BC / DP** | **00** | **01** | **11** | **10** |
| **00** | 1 | 0 | 1 | 0 |
| **01** | 0 | 1 | 0 | 1 |
| **11** | 1 | 0 | 1 | 0 |
| **10** | 0 | 1 | 0 | 1 |

그 다음으로 A가 1일 때의 표를 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BC / DP** | **00** | **01** | **11** | **10** |
| **00** | 0 | 1 | 0 | 1 |
| **01** | 1 | 0 | 1 | 0 |
| **11** | 0 | 1 | 0 | 1 |
| **10** | 1 | 0 | 1 | 0 |

PEC = A’B’C’D’P’ + A’B’C’DP + A’B’CD’P + A’B’CDP’ + A’BCD’P’ + A’BCDP + A’BC’D’P + A’BC’DP’

= (ABCDP)’

4bit의 데이터를 전송할 때 odd parity bit도 전송하면 odd parity bit checker는 모든 입력 값들 중에서 1을 가지는 입력 값의 개수가 짝수개라면 오류를 의미하는 1를, 홀수개라면 오류가 없음을 의미하는 0을 출력 값으로 가진다. 따라서 이를 코딩할 때도 앞서 odd parity bit generator와 마찬가지로 XOR gate와 NOT gate를 사용하여 구현하였고, 이는 simulation결과와 카르노 맵의 결과와 일치함을 알 수 있다.

**3.**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

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

자동 생성된 설명

스크린샷, 디스플레이, 사각형, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 도표, 평면도, 라인이(가) 표시된 사진

자동 생성된 설명

위의 사진들은 차례대로 2-bit binary comparator의 test bench, design source, simulation, schematic diagram의 결과이다. 이를 바탕으로 진리표를 작성하면 다음과 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **OutF1** | **OutF2** | **OutF3** |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 |

이를 바탕으로 카르노 맵을 작성하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CD\AB** | **00** | **01** | **11** | **10** |
| **00** | 0 | 1 | 1 | 1 |
| **01** | 0 | 0 | 1 | 1 |
| **11** | 0 | 0 | 0 | 0 |
| **10** | 0 | 0 | 1 | 0 |

위의 표는 OutF1의 진리표이다.

OutF1 = A(C)’ + B(C)’(D)’+AB(D)’

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CD\AB** | **00** | **01** | **11** | **10** |
| **00** | 1 | 0 | 0 | 0 |
| **01** | 0 | 1 | 0 | 0 |
| **11** | 0 | 0 | 1 | 0 |
| **10** | 0 | 0 | 0 | 1 |

위의 표는 OutF2의 진리표이다.

OutF2 = ((A^C)(B^D))’

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CD\AB** | **00** | **01** | **11** | **10** |
| **00** | 0 | 0 | 0 | 0 |
| **01** | 1 | 0 | 0 | 0 |
| **11** | 1 | 1 | 0 | 1 |
| **10** | 1 | 1 | 0 | 0 |

위의 표는 OutF3의 진리표이다.

outF3=(A)’C+(A)’(B)’D+(B)’CD

OutF1, OutF2, OutF3는 각각 큰 값을 내보내는 연산자, 등위 연산자, 작은 값을 내보내는 연산자이다. 작성한 Verilog 코드로 FPGA의 결과를 확인하였고 두 입력 값이 같을 때는 OutF2에 해당하는 LED에 불빛이 들어왔고, 대소비교는 그 결과에 따라 AB가 CD보다 크면 OutF1에 불빛이, AB가 CD보다 작으면 OutF3에 해당하는 LED에 불빛이 들어옴을 확인할 수 있었다.

**4.**

이번 실험에서는 even parity bit generator와 checker, odd parity bit generator와 checker 그리고 2-bit binary comparator를 카르노맵을 통해 식을 간소화한 뒤 Verilog로 구현하고 simulation으로 결과를 확인하였다. 실습에서는 even parity bit generator와 checker을 구현하였고, odd parity bit generator와 checker는 따로 구현해봤는데 even parity bit generator와 checker에 NOT을 취하면 된다는 것을 카르노맵을 통해 알 수 있었다. Odd parity bit generator와 checker가 even parity bit generator와 checker에 비해 NOT gate를 추가적으로 더 사용하므로 특별한 상황이 아니면 even parity bit generator와 checker를 사용하는 것이 더 효율적이다.

**5.**

한 개의 parity bit로는 오류가 발생했는지 그 여부만 확인할 수 있고, 올바르게 수정하는 것은 불가능하다. 하지만 여러 개의 parity bit를 사용한다면 오류 수정도 가능한데 그것이 바로 parrel parity이다. Parrel parity는 parity bit를 가로와 세로로 확인하여 나타내기 때문에 가로줄과 세로줄에서 겹치는 부분이 오류가 생긴 곳임을 알 수 있다.

또한 한 개의 parity bit만을 사용하면 2개의 bit에서 오류가 생겼을 때 검출이 불가능하다. 이럴 때는 checksum을 사용하면 이 단점을 보완할 수 있는데, checksum 방법은 모든 bit 수를 더하여 그 합산 값이 짝수인지 홀수인지 판별하는 방법이다. 총 합을 알고 있기 때문에 여러 개의 오류가 발생해도 검출이 가능하다.