7주차 결과보고서

전공 : 컴퓨터공학과 학년 : 3학년 학번 : 20211558 이름 : 윤준서

**1. Even Parity Bit Generator / Checker**

**Even Parity Bit Generator**의 논리식은 다음과 같다.

**e = ~(a ^ b ^ c ^ d)**

이를 통해 출력한 Schematic, Simulation, 카르노 맵, 진리표는 다음과 같다.

도표, 라인, 그래프, 폰트이(가) 표시된 사진

자동 생성된 설명

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a | b | c | d | e |
| 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 |

**Even Parity Bit Checker**의 논리식은 다음과 같다.

**f = a ^ b ^ c ^ d ^ e**

이를 통해 출력한 Schematic, Simulation, 카르노 맵, 진리표는 다음과 같다.

도표, 라인, 그래프, 폰트이(가) 표시된 사진

자동 생성된 설명

스크린샷, 사각형, 디자인이(가) 표시된 사진

자동 생성된 설명

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

**e (Parity Bit) = 0**  **e (Parity Bit) = 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f |
| 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 |

**2. Odd Parity Bit Generator / Checker**

**Odd Parity Bit Generator**의 논리식은 다음과 같다.

**e = a ^ b ^ c ^ d**

이를 통해 출력한 Schematic, Simulation, 카르노 맵, 진리표는 다음과 같다.

도표, 라인, 평면도, 그래프이(가) 표시된 사진

자동 생성된 설명

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |

**Odd Parity Bit Checker**의 논리식은 다음과 같다.

**f = a ^ b ^ c ^ d ^ e**

이를 통해 출력한 Schematic, Simulation, 카르노 맵, 진리표는 다음과 같다.

도표, 라인, 평면도, 그래프이(가) 표시된 사진

자동 생성된 설명

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

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

**e (Parity Bit) = 0**  **e (Parity Bit) = 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f |
| 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 |

**3. 2-Bit Binary Comparator**

**2-Bit Binary Comparator**의 논리식은 다음과 같다.

**c = (a1 & (~b1)) || (a1 & a2 & (~b2)) || (a2 & (~b1) & (~b2)) (A > B)**

**d = ~(a1 ^ b1) & ~(a2 ^ b2) (A = B)**

**e = ((~a1) & b1) || ((~a2) & b1 & b2) || ((~a1) & (~a2) & b2) (A < B)**

이를 통해 출력한 Schematic, Simulation, 카르노 맵, 진리표는 다음과 같다.

도표, 평면도, 라인이(가) 표시된 사진

자동 생성된 설명

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a1a2  b1b2 | 00 | 01 | 11 | 10 | a1a2  b1b2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 1 | 00 | 1 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 1 | 01 | 0 | 1 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 1 | 0 | 10 | 0 | 0 | 0 | 1 |

**c** **d**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a1a2  b1b2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 1 | 0 | 0 | 0 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 1 | 0 | 0 |

**e**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| a1 | a2 | b1 | b2 | c | d | e |
| 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 |

**4. 결과 검토 및 논의 사항**

Even Parity Bit Generator는 값이 1인 입력의 개수가 짝수면 1을, 홀수면 0을 출력한다.

Even Parity Bit Checker는 Parity Bit를 포함하여 값이 1인 입력의 개수가 짝수면 1을, 홀수면 0을 출력한다.

Odd Parity Bit Generator 는 값이 1인 입력의 개수가 짝수면 0을, 홀수면 1을 출력한다.

Odd Parity Bit Checker는 Parity Bit를 포함하여 값이 1인 입력의 개수가 짝수면 1을, 홀수면 1을 출력한다.

Parity Bit Generator는 결과 값을 함께 목적지로 전달함으로써 전달하는 데이터의 오류를 사전에 보수하는 데 사용된다. Parity Bit Checker는 받은 데이터를 검사함으로써 오류의 유무를 판단하는 데 사용된다.

5-input의 논리식을 카르노 맵으로 표현하기 위해선 4-input 카르노 맵을 두 개 그려야 하는 번거로움이 있다. 이를 해결하기 위해 5-input 이상에서 유용한 맥클러스키 알고리즘 등을 사용할 수 있다.

2-Bit Binary Comparator는 두 입력 값을 서로 비교해서 크기 차이를 출력한다. a1, a2 가 하나의 2비트 수를(A), b1, b2가 하나의 2비트 수(B)를 나타낸다. 이 때 a1과 b1이 각 수의 MSB다. A가 B보다 큰 경우 c = 1, A와 B가 같은 경우 d = 1, B가 A보다 큰 경우 e = 1을 출력한다.

**5. 추가 이론**

Parity Bit Generator / Checker는 입력 값에 오류가 존재하는 지 확인할 때 쓰인다. 하지만 입력 값의 1의 개수가 짝수인지 홀수인지 만으로 판별하기에, 정확히 몇 개의 비트에서 오류가 발생하였는지는 알 수 없다. 이를 해결하기 위해 Checksum이란 방법을 사용하는데, 각 비트를 모두 합한 값을 통해 그 합이 짝수인지 홀수인지에 따라 오류를 판별할 수 있다. 이 때 합의 크기가 크거나 작으면 1개 이상의 비트에 오류가 발생한 것을 알 수 있다.