컴퓨터공학실험2 7주차 예비 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20201635 이름: 전찬

**0. 목차**

1. Parity Bit 생성기에 대해 조사한다.

2. Parity Bit 검사기에 대해 조사한다.(검사 부호 종류 포함)

3. Parity Bit 검사기 외의 다른 오류 검출기 및 오류 정정기.

4. N bit 비교기에 대해 조사한다

5. IC 7485 비교기에 대해 조사한다.

**1. Parity Bit 생성기**

parity bit란, 어떤 data가 존재할 때, data에 오류가 존재하는지 하지 않는지를 판단할 수 있는 data에 추가되는 1 bit이다. 2 종류의 parity bit가 존재할 수 있는데, even parity bit, odd parity bit 이다. even parity bit란, 추가되는 1bit가 전체 data와 함께 총 짝수 개의 1을 구성하도록 만드는 bit이며, odd parity bit란 even parity bit와 반대로 추가되는 parity bit와 data에서 총 홀수 개의 1이 존재하도록 만들어주는 bit 이다. 이렇게 parity bit를 설정했을 때, 단순하게 생각한다면 홀수/짝수 인가에 따라 총 1/2 확률로 오류가 존재하는지를 판단해낼 수 있다.

parity bit 생성기란, 위에서 설명한 even/odd 의 상황에 맞추어 알맞은 parity 생성해주는 회로를 의미한다. input으로는 data, output으로는 parity bit를 갖는 형태인데, data에서 1의 개수가 몇개인지는 XOR gate를 활용해서 간단하게 파악해낼 수 있다. input A, B, C에 대해서 X = A^B^C 라는 회로를 제작한다면, A, B, C에서 홀수 개의 1이 존재할 때, X의 output 값이 1이 될 것이다. 따라서 even parity bit 생성기는 XOR gate만으로 만들어낼 수 있다. odd parity bit 생성기라면, 앞 형태에서 NOT gate를 추가해주면 3-input even parity bit 생성기를 만들어낼 수 있다.

**2. Parity Bit 검사기**

위에서 XOR, NOT gate만으로 parity bit 생성기를 구현할 수 있다는 것을 파악했다. 이와 비슷하게 Parity Bit 검사기 또한 XOR gate를 이용해서 만들어낼 수 있다. 위에서는 even parity bit 생성기를 만들 때 XOR gate만으로 만들수 있음을 보였는데, 검사기의 경우에는 XOR gate를 통해서 odd parity bit 검사기를 만들 수 있다. input data의 총 1의 개수가 홀수 개라면 output으로 1을 가지는 것이다. 또한 even parity bit 검사기는 간단하게 위 형태에서 NOT gate를 추가한 형태로 구현해줄 수 있다.

**3. Parity bit 이외의 오류 검출기 및 정정기**

parity bit는 구현, 사용하기에 쉬운 오류 검사기 이지만, 위에서 말한 것처럼 총 오류의 50%만 파악해낼 수 있다는 단점이 존재한다. 이를 보완해낸 형태가 검사바이트(checkbytes)라고 할 수 있다. 이 형태는 1bit의 parity bit가 아닌, parity bytes 형태로 존재하는데, 많은 parity bit를 정해진 알고리즘을 이용해서 만들어내며(data에서 일정 byte마다 끊어서 1개의 parity bit를 주거나, 일정 간격마다 존재하는 bit들에 대한 parity bit를 만드는 형태) 오류 판단의 정확도를 올릴 수 있다. 이를 더욱 변형한 형태도 존재하는데, 검사합(checksum), CRC(Cyclic Redundancy Check) 등이 존재한다.

parity bit의 또 다른 단점으로는 오류를 고칠 수 없다는 것이다. 만약 오류를 고칠 수 없다면 실제로 data가 저장되는 하드 디스크, SSD 등에서도 1bit 단위의 오류가 발생할 수 있는데, 1bit의 오류만으로 몇 byte, …, 몇 메가바이트 단위 정도의 큰 data 를 전부 사용할 수 없게 된다. 따라서 오류 정정 또한 컴퓨터 공학 분야에서 중요하다. 실제로 오류를 정정할 수 있는 방법으로는 해밍 거리가 존재한다. 해밍 거리란 data input을 이미 존재하는 table의 각각의 data에 대해서 비교하며 거리가 가장 가까운 것으로 판별하는 것이다. 예를 들면 A = 000000 / B = 001111 이라고 table에 이미 존재하고 있을 때, 001000 이라는 input이 들어온다면, A와의 거리는 1, B와의 거리는 3임을 통해서 거리가 가장 가까운 A로 정정해주는 형태이다. 이를 통해서 오류를 정정할 수 있으며, 더 나아가 table에서의 각 symbol의 bit 수가 길어질수록 오류를 검출해낼 수 있는 경우도 늘어날 수 있다. 하지만 해밍 거리의 단점 또한 존재하는데, 위에서 설명한 것처럼 A, B를 판단하기 위해서도 많은 수의 bit 가 사용되는 것을 볼 수 있다. 따라서 오류를 정정하기 위해 사용되는 bit의 양이 훨씬 많다는 단점이 존재하기도 한다.

**4. N bit 비교기**

이전 실습에서 1 bit 비교기가 실제로 어떻게 작용하는지 verilog를 통해 구현해 보았다. 이를 활용해서 n bit 비교기를 만들어낼 수 있는데, 이는 아래 형태와 같다.

(1) 두 input의 MSB에 대해 1bit comparator을 적용한다.

(2) 만약 MSB가 동일하다면(=), 그 다음 MSB에 대해 1bit comparator을 적용한다. 동일하지 않은 경우에는 두 input의 대소관계를 파악할 수 있다.

(3) LSB까지 모든 bit가 동일하다면, 두 input은 동일하다고 할 수 있다.

위 형태로 진행하며 n bit 비교기를 구현해낼 수 있다.

**5. IC 7485 비교기**

IC 7485 비교기는 fixed function logic gate의 한 종류이다. 이름처럼 비교(input A, B에 대해서 A>B, A=B, A<B인지 판단)를 수행하는데, 4 bit comparator의 역할을 수행할 수 있다. input으로는 A3, A2, A1, A0, B3, B2, B1, B0, A<B, A=B, A>B가 존재하며, output으로는 A<B, A=B, A>B 가 존재한다. 여기에서 두 4 bit input인 A3A2A1A0 / B3B2B1B0 를 비교해서 output으로 올바를 값을 전달해주는 역할을 수행한다. 추가적인 input(A<B, A=B, A>B)은 종속적인 input으로 output과 동일한 형태를 가지게 된다.

**6. 출처**

1. J.Glenn Brookshear Dennis Brylow, Person, 컴퓨터 과학 총론 13th Edition, 78p