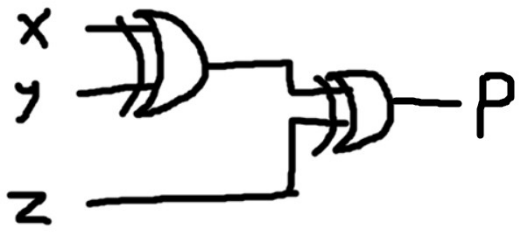
7주차 예비보고서

전공: 경영학과 학년: 4학년 학번: 20190808 이름: 방지혁

1.
우선 Parity Bit의 정의부터 먼저 설명하겠습니다. Parity bit란 네트워크 통신 같은 정보 전송을 할 경우 error 검출을 위하여 사용하는 비트입니다. 짝수 parity bit과 홀수 parity bit 2가지의 경우가 존재합니다. 예를 들어, 짝수 parity bit는 만약 data 내 1의 개수가 짝수 개라면 parity bit는 0이고, 1의 개수가 홀수 개라면 parity bit는 비트열의 1의 개수가 짝수 개가 되어야 하기 때문에 1이 됩니다.

이러한 parity bit를 추가하는 것이 parity bit 생성기입니다. 이를 위해 xor 게이트를 사용합니다. 이전의 실험들에서 알다시피 xor 게이트는 1의 개수가 홀수 개면 1을 출력합니다. 반면, 짝수면 0을 출력합니다. 짝수 parity bit을 구현하기 위해서는 xor 게이트를 사용하면 되고, 홀수의 경우에는 xnor게이트를 사용하면 됩니다.

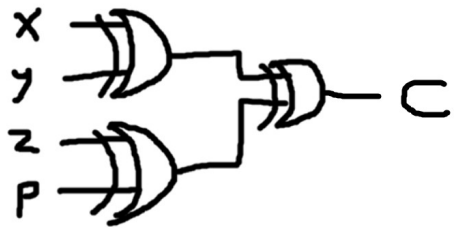
예시) <짝수 parity bit 생성기>
dataword 'xyz'에 대한 parity bit: P
식으로 나타낸다면 패리티 비트는 P(Parity bit) = x^y^z 가 됩니다.



2.
Parity Bit 검사기는 사전에 약속된 규칙 하에 전송 받은 데이터에 대하여 오류 발생 여부를 체크하는 것으로 짝수와 홀수 검사기가 존재합니다. 이를 위해 마찬가지로 xor 게이트를 응용합니다.

짝수 parity bit 검사기는 결과 값이 1일 때 오류가 발생했다는 것을 의미합니다. 홀수 parity bit 검사기는 이와 반대로 0일 때 오류가 발생했다는 것을 의미합니다.

예시) <짝수 parity bit 검사기>
dataword 'xyz' 및 parity bit P에 대한 오류 발생 여부: C
식으로 나타낸다면 C(Check) = (x^y)^(z^P)입니다.



3.

CRC(Cyclic Redundancy Checking)

순환 중복 검사로 원래 전송할 데이터에 대하여 미리 정의된 divisor로 xor 연산하여 나누고, divisor보다 1비트 자리 수가 적은 나머지를 데이터의 뒤에 덧붙여 보내는 오류 검출 방법입니다. 수신하고 나서 동일한 divisor로 연산을 하였을 때, 나머지가 0이면 오류가 없고, 그 외의 경우에는 오류가 발생한 것입니다. 하드웨어로 구현하기 쉽고, 능력이 매우 우수하다는 장점이 있습니다.

Hamming Code

해밍코드는 block coding을 통해 오류 검출 및 수정할 수 있는 방식입니다. d비트의 dataword에 p비트의 parity 비트를 추가합니다. $2^p \geq d+p+1$ 이 식을 만족하는 최소의 p값을 구합니다. 그리고 2^{k-1}번째 자리마다 패리티 비트를 추가합니다. 이 데이터를 수신할 때에는 패리티 비트의 비트 각각마다 오류를 찾고 기록합니다.

예를 들어 4비트를 전송해야 한다면 패리티 비트 3개를 추가하여, 1번째 비트: P1 (패리티 비트), 2번째 비트: P2 (패리티 비트), 3번째 비트: D1 (데이터 비트), 4번째 비트: P3 (패리티 비트), 5번째 비트: D2 (데이터 비트), 6번째 비트: D3 (데이터 비트), 7번째 비트: D4 (데이터 비트)가 전송됩니다. P1은 1, 3, 5, 7번째 비트들을 검사하여 패리티를 결정하고, P2는 2, 3, 6, 7번째 비트를 검사하고, P3는 4, 5, 6, 7번째 비트를 검사합니다. 각 패리티 비트는 그 비트들의 XOR 결과로 결정합니다.

Check Sum

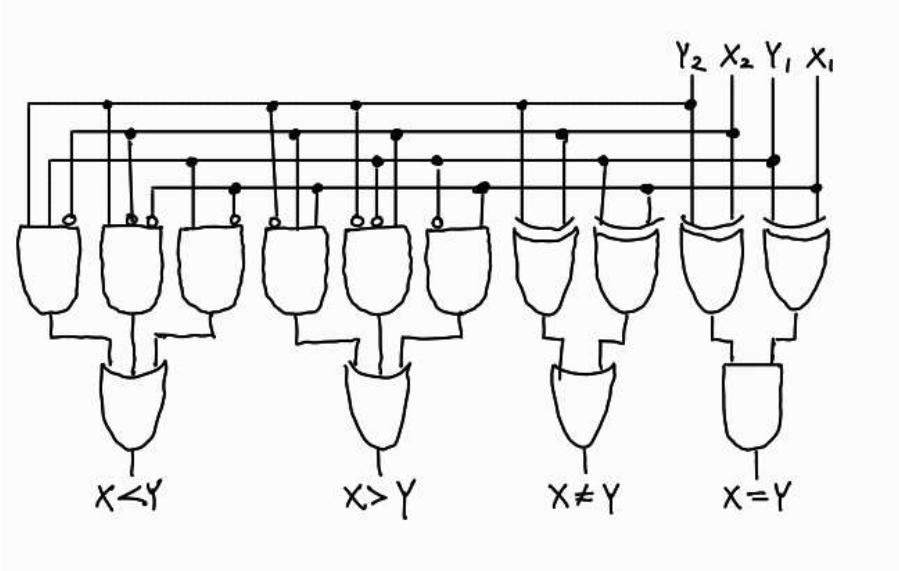
데이터를 보내는 측에서는 데이터를 일정한 크기의 블록으로 나눈 후, 1의 보수를 취한 후 각 블록의 값을 더하여 결과를 생성합니다. 받는 측에서는 수신한 데이터에서 동일한 연산을 수행해 체크섬을 계산한 후, 송신된 체크섬과 비교합니다. 만약 일치한다면, 데이터가 손상되지 않았다고 판단합니다.

4.

N bit 비교기는 N개의 자리수를 가진 2개의 이진수를 크기를 비교하는 비교기입니다. 각 자리의 XOR 연산을 통해 두 입력이 같은지 아닌지 확인할 수 있고, 만약 같지 않다면 MSB부터 LSB순으로 비교하여 대소 비교 확인이 가능합니다.

ex) 2 bit comparator

입력		출력			
X	Y	X=Y	X≠Y	X>Y	X<Y
X_1X_2	Y_1Y_2	F_1	F_2	F_3	F_4
00	00	1	0	0	0
	01	0	1	0	1
	10	0	1	0	1
	11	0	1	0	1
01	00	0	1	1	0
	01	1	0	0	0
	10	0	1	0	1
	11	0	1	0	1
10	00	0	1	1	0
	01	0	1	1	0
	10	1	0	0	0
	11	0	1	0	1
11	00	0	1	1	0
	01	0	1	1	0
	10	0	1	1	0
	11	1	0	0	0



5.

IC7485 비교기는 4-bit Comparator IC입니다. A3, A2, A1, A0과 B3, B2, B1, B0의 8개 입력이 존재합니다. A > B (GT), A < B (LT), A = B (EQ) 3개의 출력이 존재합니다.

우선, A와 B의 각 비트를 비교하여 동치 여부를 판단합니다. 만약, 모든 비트가 같음 A = B 출력이 1이 됩니다. 그렇지 않다면, MSB부터 우선적으로 비교를 시작하여 그 다음 비트로 비교가 이어집니다. 즉, 가장 상위 비트(A3, B3)부터 비교하여, 크기에 따라 A > B 또는 A < B 출력이 결정됩니다

$X_i = A_i \odot B_i$라고 한다면, 각 출력의 식은

$Result_{A>B} = A_3B_3' + X_3A_2B_2' + X_3X_2A_1B_1' + X_3X_2X_1A_0B_0'$,

$Result_{A<B} = A_3'B_3 + X_3A_2'B_2 + X_3X_2A_1'B_1 + X_3X_2X_1A_0'B_0$

$Result_{A=B} = X_3X_2X_1X_0$로 표현할 수 있습니다.

6.

Hamming Distance

두 개의 이진수 간에 다른 비트의 개수를 측정하는 것입니다. 이는 오류 검출 및 수정에서 매우 중요한 지표가 됩니다. d개의 error를 감지하기 위해서는 minimum hamming distance가 d + 1 이상이 되어야 합니다. 감지 뿐만 아니라 수정을 하기 위해서는 2d + 1 이상의 hamming distance가 요구됩니다.

Encoding & Decoding

인코딩 : 기존의 데이터를 어떠한 정해진 규칙에 의하여 저장 공간 절약, 보안, 속도 향상 및 오류 검출을 위해 다른 형태로 변형하는 것을 말합니다. 패리티 비트 추가, 해밍코드, CRC 모두 인코딩이라고 볼 수 있습니다.

디코딩 : 인코딩된 정보를 인코딩 이전으로 되돌리는 방식을 말합니다. 인코더와 디코더는 서로 반대로 동작합니다. 예를 들어 수신 측에서 해밍 코드를 통해 오류를 찾아내고 복원하는 과정이 디코딩입니다.