6주차 예비보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20211547 이름: 신지원

**1.**

가산기는 컴퓨터에서 쓰이는 연산 장치인 CPU 의 하나로 주로 제어 장치의 명령으로 2진수의 덧셈을 수행한다.

반 가산기는 영어로 Half Adder 로 표기할 수 있다. 2개의 입출력을 가지는데, 입력은 1 bit 로 구성된 2개의 input 으로, 출력은 sum 과 carry 라고 일컫는 2개의 output 으로 구성된다. XOR gate 하나와 AND gate 하나를 조립하여 설계한 논리회로며, Carry는 AND gate, Sum은 XOR gate와 결과가 같다. 논리식은 아래와 같다.

**Sum = A XOR B, Carry = A AND B**

논리식을 활용하여 회로를 나타내면 아래와 같다.

도표, 라인, 텍스트, 폰트이(가) 표시된 사진

자동 생성된 설명

전 가산기는 영어로 Full Adder 로 표기할 수 있다. carry를 더할 수 있는 가산기로 실질적인 연산회로로 사용되곤 한다. XOR gate 2개와, AND gate 2개, 그리고 OR gate 1개를 조합라여 설계한 논리 회로다. 한 자릿수 덧셈을 수행할 때 이전 자리의 연산 결과로 받은 Carry를 함께 연산하는 회로이며 두 개의 반가산기와 1개의 OR gate 로 구성할 수 있다. 논리식은 아래와 같다.

**Sum = (A XOR B) XOR C-in, C-out = (A AND B) OR (C-in AND (A XOR B))**

논리식을 활용하여 회로를 나타내면 아래와 같다.

도표, 라인, 평면도, 폰트이(가) 표시된 사진

자동 생성된 설명

**2.**

감산기는 두 개 이상의 입력에서 하나 입력으로부터 나머지 입력들을 뺄셈한 값을 출력하는 조합 논리다. 가산기에서 carry 가 발생하였다면 감산기에서는 borrow 가 발생한다.

반 감산기는 영어로 Half Subtractor 로 표기할 수 있다. 2개의 입출력을 가지는데, 입력은 1 bit의 길이를 가진 2개의 input 으로, 출력은 difference 과 borrow 라고 일컫는 2개의 output 으로 구성된다. XOR gate 하나와 (NOT을 포함한) AND gate 하나를 조립하여 설계한 논리회로며, Borrow는 ~A 와 B의 AND gate, difference 는 XOR gate와 결과가 같고. 논리식은 아래와 같다.

**Different= A XOR B, Borrow = ~A AND B**

논리식을 활용하여 회로를 나타내면 아래와 같다.

도표, 라인, 기술 도면, 평면도이(가) 표시된 사진

자동 생성된 설명

빌림수가 나타나는 과정은 아래와 같다.

직사각형, 도표, 디자인이(가) 표시된 사진

자동 생성된 설명

전 감산기는 영어로 Full Subtractor 로 표기할 수 있다. borrow를 뺄 수 있는 감산기로 실질적인 연산회로로 사용되곤 한다. XOR gate 2개와, (NOT을 포함한) AND gate 2개, 그리고 OR gate 1개를 조합하여 설계한 논리 회로다. 한 자릿수 뺄셈을 수행할 때 이전 자리의 연산 결과로 받은 Borrow를 함께 연산하는 회로이며 두 개의 반감산기와 1개의 OR gate 로 구성할 수 있다. 논리식은 아래와 같다.

**Different = (A XOR B) XOR (Br0), Br = (Br0 AND ~(A XOR B)) OR (~A AND B)**

논리식을 활용하여 회로를 나타내면 아래와 같다.

도표, 라인, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

**3.**

BCD 가산기란 BCD 코드로 표현된 10진수 숫자 2개를 입력 받아 덧셈을 거쳐 얻은 결과를 BCD 코드로 출력하는 회로를 말한다. 이는 십진가산기라고도 불린다. BCD 는 자리마다 4bit 로 구성되기 때문에 각각 0~15 중 하나의 값을 얻을 수 있다. 만약, BCD 가산기를 작동하는 중 덧셈의 결과가 10 미만일 경우, 별도의 과정이 필요없다. 하지만 10 이상일 경우엔 변환이 필요하다. BCD 가산기는 carry까지 포함하게 되면 총 input이 4+4+1=9개이고, output이 4+1=5개가 된다. 만약 input을 5개 이상으로 설정하고 싶다면 카노맵이 요구된다.

텍스트, 스크린샷, 번호, 라인이(가) 표시된 사진

자동 생성된 설명

위는 BCD 덧셈표다. 덧셈을 할 때 자릿수 별로 더하기를 하는데, 만약 더한 값이 9를 초과한다면 6을 더해준다. 따라서 10이상의 수부터 값을 보정하는 가산기라고 할 수 있다. 예를 들어 8 + 7 = 15 를 계산하고자 할 때 1000 + 0111 = 1111 + 0110 = 10101 로 표기할 수 있다. 따라서 위와 동일하다.

텍스트, 도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

BCD 가산기의 논리회로는 위와 같다.

**4.**

병렬 가산기란 n개의 전가산기로 이루어져 있다. 즉, 전가산기 여러 개를 병렬로 연결하여 2비트 이상의 덧셈을 수행하는 것을 의미한다. n bit의 2진수 2개의 input을 계산하는 논리회로이다. 최종 carry는 Cn+1이 되며 병렬 가산기의 y에 부호 S(sign)와 XOR 연산을 거치면 덧셈과 뺄셈 모두 가능하게 변환시킬 수 있다. 전파지연을 위해 Carry Look Ahead 를 이용한다.

도표, 라인, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

위는 병렬 가산기의 다이어그램이다. 이에 감산기를 추가하여 병렬가감산기를 만들기 위해서 부호를 XOR gate 를 추가하였다. S가 1이면 Y의 값은 반전이 되어 1의 보수가 입력되고 캐리의 값도 1이 입력되어 결과적으로 Y의 2의 보수가 만들어진다. 아래는 병렬가감산기의 다이어그램이다.

도표, 라인, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

**5.**

Ripple Carry Adder : carry 를 물결처럼 넘긴다는 데에서 이름이 붙여졌다. 입출력 비트 수가 제한되어 있는 Adder에 출력 값이 비트 수를 초과할 경우에 Carry를 사용한다. 이때 Carry 값까지 포함하여 계산한 뒤, 다른 하나의 출력을 생성하는 것이 Half Adder이다. 반면, 아래에서 올라온 Carry 값까지 구현한 것이 Full Adder이다. Full Adder를 결과를 출력하고 싶은 비트 수에 맞춰 구성하여 원하는 비트 크기에 맞는 덧셈 과정을 수행하는 논리회로를 구성할 수 있다.

Carry Look-Ahead Adder : 앞서 언급한 Ripple Carry Adder는 계산 과정에서 Carry값을 리플 해야하기 때문에 속도가 굉장히 느리다. 속도를 보완하기 위해 구상된 가산기가 바로, Carry Look-Ahead Adder이다. Adder 외에 Carry 값만을 계산하는 Carry Look Ahead Block을 추가적으로 가지고 있어 리플을 하는 과정 없이 미리 Carry를 계산할 수 있게 된다. 앞서 언급한 병렬 가산기에서 사용된다.

**6.**

다른 논리회로 변환기인 코드 변환기에 대하여 알아보고자 한다. 코드 변환기는 어떤 코드로 되어 있는 것을 다른 종류의 코드로 바꾸어 주는 논리회로다. 평소엔 10진수를 사용하지만 컴퓨터는 2진수를 사용하기 때문에, 이를 위해 변환기를 사용한다. 하지만 뺄셈 연산에 사용되는 보수는 변환이 되지 않는다는 한계점이 있다.