6주차 예비보고서

전공: 수학과 학년: 4학년 학번: 20171273 이름: 심현우

1. **전 가산기 및 반 가산기에 대해 조사하시오(예시 포함).**

우선 가산기란 현재 다루고 있는 bit를 더하기 연산하는 것에 대한 결과를 도출해주는 기계이다. Bit는 0과 1로 이루어진 2진수이므로 이에 대한 가산은 0+0=0, 0+1=1, 1+1=10이 된다. 이렇게 자릿수가 한자리인 연산에 대한 가산기를 반 가산기라 부른다. 반 가산기에서 주목해야할 점은 1+1의 상황인데 이때 결과가 10으로 한자릿수가 아닌 두자리로 나타나는 것을 알 수 있다. 이때, 한자리를 넘어가 자릿수가 올라간 부분을 carry라고 부른다. 따라서 반 가산기에 대한 출력은 carry와 원래 자릿수인 sum으로 이루어져 있다. 반 가산기의 연산 과정을 진리표로 작성하면 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| In A | In B | Out C | Out S |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

반 가산기에 대한 회로 구성은 XOR게이트와 AND게이트이다. Sum에 대한 값은 XOR게이트를 통해 S=(~A&B) | (A&~B)로 input값이 서로 다를 때, 1을 출력하도록 한다. Carry는 AND게이트를 이용하여 C=(A&B)로 A, B모두 1일 때, 1을 출력하도록 한다.

전 가산기는 이러한 반 가산기를 확장하여 한자리가 아닌 임의의 자릿수의 bit를 계산하기 위한 기계이다. 임의의 자릿수를 계산하기 위해서는 하위 자릿수의 계산 결과에서 carry가 발생하였는지를 알아야 한다. 따라서 전 가산기의 input으로는 A, B와 Carry가 있어야 하고 출력으로는 반 가산기와 마찬가지로 carry와 sum이 있다. 전 가산기도 진리표로 작성해서 보면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | Out C | Out S |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

전 가산기의 회로 구성은 반 가산기 두개와 OR게이트로 이루어져 있다.

1. **전 감산기 및 반 감산기에 대해 조사하시오(예시 포함).**

감산기는 이름 그대로 감산, 즉 빼는 연산에 대한 계산 결과를 도출해주는 기계이다. 가산기와 반대로 작용하며 가산기와 똑같이 bit에 대한 감산을 다룬다. 0과 1로 이루어진 bit에 대해 한자릿수 감산 연산을 진행해보면 0-0=0, 1-0=1, 1-1=0이다. 특별히 0-1 연산에서는 가산기와 반대로 burrow가 필요하다. Burrow는 상위 자릿수에서 1bit를 빌려오는 것을 의미한다. 따라서 감산기의 출력은 burrow와 자릿수의 감산 연산인 difference이며 반 감산기는 한 자릿수에 대한 감산 연산을 진행한다. 반 감산기의 진리표로 작성해보자.

|  |  |  |  |
| --- | --- | --- | --- |
| In A | In B | Out B | Out D |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |

반 감산기의 회로 구성은 가산기와 마찬가지로 XOR게이트와 AND게이트를 이용한다. 우선 burrow의 출력을 얻기 위해서 AND게이트를 이용한다. A가 0 B가 1인 경우에만 1을 출력해야 하므로, B=~A&B를 이용하여 회로를 구성한다. Difference는 input 값이 서로 다른 경우 1, 같은 경우 0을 출력해야 하므로 XOR게이트를 이용하여 D=((~A&B) | (A&~B))를 통해 구현한다.

전 감산기는 이러한 반 감산기를 확장하여 한자리가 아닌 임의의 자릿수의 bit를 계산하기 위한 기계이다. 임의의 자릿수를 계산하기 위해서는 하위 자릿수의 계산 결과에서 burrow가 발생하였는지를 알아야 한다. 따라서 전 가산기의 input으로는 A, B와 burrow가 있어야 하고 출력으로는 반 가산기와 마찬가지로 burrow와 sum이 있다. 전 가산기도 진리표로 작성해서 보면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In Br | Out B | Out D |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

전 감산기 또한 반 감산기 두개와 OR게이트로 회로가 구성되어 있다.

1. **BCD 가산기에 대해 조사하시오.**

BCD 가산기는 1번에서 조사한 반, 전 가산기에서 다루는 2진수인 bit와 다르게 10진수를 가산해주는 기계이다. 이 때, 10진수를 나타내는 방법은 bit를 4개, 즉 4bit를 사용하여 0000부터 1001까지의 10개를 0부터 9로 대응하여 계산한다. 이 때, 1001부터 1111까지 나머지 표현들은 사용하지 않는다. BCD는 전 가산기처럼 한자릿수가 아닌 임의의 자릿수를 계산할 수 있으며 전 가산기처럼 carry가 발생한다. 따라서 BCD 가산기의 input으로는 감산 계산을 할 4bit 두 개와 carry가 있고, output으로는 4bit 한 개와 carry가 있다. 4bit의 가산 계산은 전 가산기에서 4자릿수를 계산할 때와 동일한 방법으로 진행해주면 된다.

1. **병렬 가감산기에 대해 조사하시오.**

병렬 가감산기는 전 가산기의 회로를 병렬로 연결하여, 확장된 자릿수의 bit에 대한 가산 연산을 해주는 기계이다. 또한 input으로 sign이 더해져 sign값에 따라 가산 또는 감산 연산을 나눠 원하는 연산을 진행할 수 있다. 자릿수마다 전 가산기가 배정되어 전 가산기의 출력인 carry가 다음 가산기, 즉 상위 자릿수의 input carry로 들어간다.

1. **Carry Look-Ahead Adder을 Ripple Carry Adder와 비교하여 설명하시오.**

Ripple carry adder는 전 가산기를 병렬로 연결하여 임의의 자릿수에 대한 가산 연산을 한다. 이는 하위 자릿수부터 계산을 시작하여 carry를 구하고 상위 자릿수로 차례로 계산을 진행한다. 이는 회로는 간단한 반면에 상위 자릿수 계산을 위해서 하위 자릿수 계산을 기다려야 한다는 단점이 있어 계산 속도가 비교적 느린 편이다. Carry Look-Ahead Adder는 이러한 단점을 보완하기 위해 사용된다. 말 그대로 알 수 있듯이 carry look-ahead adder는 하위 자릿수에서 발생하는 carry를 예측하여 임의의 자릿수 가산 연산을 한번에 진행하도록 한다. Carry의 발생을 예측하는 원리는 carry의 generate와 propagate로 구성한다. Generate는 현재 input으로 들어오는 값들의 자리 bit 중에서 실제로 carry가 발생하는지 안하는지에 대해 본다. 가산 연산에서 carry는 1+1에서 발생하므로 이때 generate가 발생한다는 것을 알 수 있다. Propagate의 경우 하위 비트에서 carry가 발생 시 해당 상위 자릿수에서 carry가 발생할 수 있는지를 확인한다. 예를 들어 0+1이나 1+0인 경우 하위 자릿수에서 carry발생 시 해당 자릿수에서도 carry가 발생한다. 이때, propagate가 발생한다고 할 수 있다. 이 generate와 propagate를 통해 carry를 예측하여 한번에 계산하도록 한다.

1. **기타 이론.**

가산기와 감산기 이외에 곱셈기와 비교기가 존재한다. 비교기는 5주차의 실습에서 구현한 1bit 비교기이다. 곱셈기에 대해 알아보면 bit에 대한 곱셈 연산을 한다. 곱셈기는 각 자릿수의 곱셈을 나누어 AND게이트를 이용하여 진행하고, 각 자릿수에 대한 곱은 가산기를 통하여 자릿수를 맞추어 가산 연산을 진행해 결과를 도출한다. 이때, 자릿수가 커질수록 곱셈 연산이 많아지므로 AND게이트를 추가해야하며, 또한 곱셈에 대한 가산 연산할 자릿수도 많아지기 때문에 가산기도 추가해야한다.