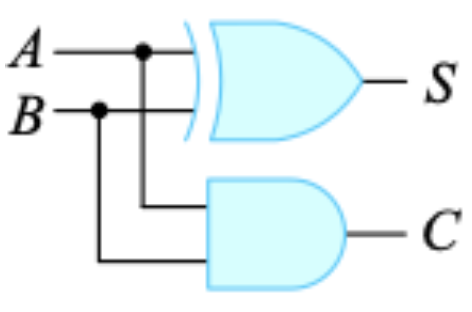
6주차 예비보고서

전공: 컴퓨터공학과/생명과학과 학년: 9학기 학번: 20181435 이름: 박다희

**1.**

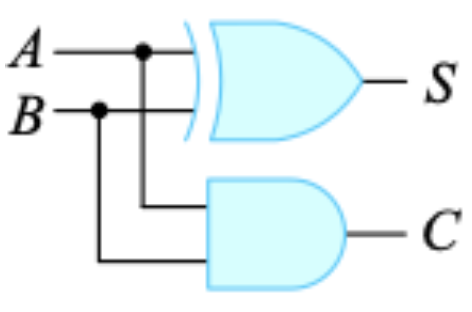
전가산기와 반가산기는 디지털 논리 회로 설계에서 이진 수를 더할 때 사용되는 논리 게이트의 조합으로, 이전의 출력 값과는 상관 없이 현재의 입력 값들의 조합으로만 출력 값이 결정되는 조합 논리 회로이다. 먼저 전가산기는 sum을 의미하는 S와 carry를 의미하는 C로 출력 값이 이루어지며 하위 자릿수에서 올라와 현재 자릿수의 덧셈 연산에 영향을 끼치고 포함 시키는 carry를 carry in(), 덧셈 연산에서 상위 자릿수에 의해 발생하는 carry를 carry out()이라고 한다. S는 으로 연산 결과 값이며, 은 으로 3개의 입력 값에 대하여 2개 이상이 참인 경우 은 1이 되고, 3개의 입력 값에 대하여 짝수 개의 입력 값이 참이라면 s는 0, 홀수 개의 입력 값이 참이라면 s는 1을 갖는다. 이를 바탕으로 한 진리표와 logic schematic의 예시는 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B |  | s |  |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |



반가산기는 2 개의 1-bit number인 입력 값과 carry(C)와 sum(S) 2개의 출력 값을 가진다. 2개의 입력 값을 더했을 때 1bit로 표현할 수 있는 범위를 넘는다면 carry가 발생하는데 이 carry 값은 전가산기와는 다르게 상위 자릿수 bit에 영향을 주지 않는다. 따라서 2개의 입력 값이 모두 1일 때만 C는 1을 갖고, S은 두 입력 값이 서로 다를 때만 1을 갖는다. S는 AND gate를 사용하여 라고 표현할 수 있고, C는 XOR gate를 사용하여 AB라고 표현할 수 있으며 이를 바탕으로 한 진리표와 logic schematic의 예시는 다음과 같다.

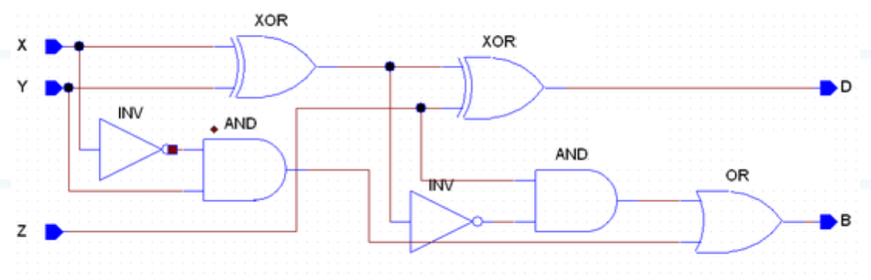
|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | S |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |



**2.**

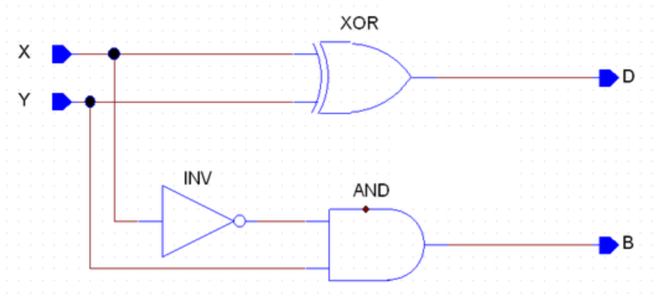
전감산기는 뺄셈 연산에 사용되는 논리 회로로, 2개의 입력 값을 가지고 빌림 수인 Borrow(B)와 두 입력 값의 차이인 Difference(D)이렇게 2개의 출력 값을 가진다. B의 발생 여부를 계산에 포함시키기 때문에 완전한 감산기의 기능을 갖추고 있고 이를 바탕으로 D는 , B는 로 나타낼 수 있다. 전감산기의 진리표와 그 logic schematic 예시는 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | D |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |



반감산기는 전감산기와는 다르게 상위 자릿수에서 B를 가져와 2개의 입력 값으로만 bit 자릿수에 맞춰서 뺄셈을 수행하기 때문에 불완전한 감산기이다. 논리 함수로 나타내면 D는 , B는 AB이고 이를 바탕으로 한 진리표와 logic schematic 에시는 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | B | D |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |



**3.**

BCD가산기는 BCD코드로 표현되어 있는 2개의 십진수를 입력 값으로 받아서 더하고 그 결과 값을 BCD코드로 내보내는 회로이다. BCD코드는 Binary Coded Decimal Code로 십진수를 자리 단위로 하여 이진수로 변환한 것이다. BCD코드는 0에서 9까지의 수로 표현하므로 일반적인 2진수 계산 후에 보정이 따로 필요하다.

도표, 텍스트, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

이러한 BCD코드에는 밑의 그림처럼 여러 종류의 코드들이 존재하고 밑의 예시들 외에도 십진수를 이진수로 변환하는 방법은 더 있다.

테이블이(가) 표시된 사진

자동 생성된 설명

**4.**

병렬 가산기는 앞서 설명한 전가산기를 병렬 연결로 여러 개 연결한 회로이다. 병렬 연결된 개수가 n개라면 n bit로 구성되어 있는 이진수 A, B의 덧셈 연산을 할 수 있고, 이 때 전파지연이 발생할 수 있는데 이를 줄이기 위해 carry look ahead adder가 사용되곤 한다.

도표, 라인, 스크린샷, 그래프이(가) 표시된 사진

자동 생성된 설명

병렬 가감산기는 병렬 가산기 회로에서의 입력을 부분적으로 바꾸어 뺄셈 뿐만 아니라 덧셈도 가능하도록 만든 회로이다. 병렬 가산기 회로의서의 입력을 sign값과 XOR하여 바꿔 주기 때문에 sign값이 0이라면 B값이 그대로 유지되고, sign값이 1이라면 B의 보수 값이 출력 값이 된다. 이 때의 출력 값은 또 다음 전 가산기의 입력 값으로 들어가게 된다.

도표, 라인, 그래프, 디자인이(가) 표시된 사진

자동 생성된 설명

**5.**

병렬 가산기는 ripple carry adder라고도 불리고 이는 병렬 가산기 하위 자릿수에서의 출력 값이 상위 자릿수에서의 계산에 사용되기 때문이다. 만약 하위 자릿수에서의 출력 값이 존재하지 않으면 이 전가산기의 출력 값은 유효하지 않다고 판단하고 상위 자릿수에 영향을 끼치기 때문에 하위 자릿수에서의 출력 지연은 문제가 될 수 있다. 만약 n bit의 ripple carry adder가 존재한다면 전체적인 시간 지연은 (2n+4)Δ라고 나타낼 수 있다. 이러한 시간지연을 줄이기 위해 carry look ahead adder를 사용할 수 있는데 carry look ahead adder는 입력 값 carry에 대한 식을 풀어 계산하여 모든 bit의 carry 값을 한 번에 계산하기 때문에 시간 지연을 없앨 수 있다. 입력 값 carry를 이고, 자리올림수 생성값인 () 그리고 자리올림수 전달값을 ()일 때, 각 비트 자릿수 I, i+1, i+2, i+3 의 각각의 은 다음과 같다. , , , 이고, 을 바탕으로 식을 정리하면 이고, 을 바탕으로 식을 정리하면 이고, 마찬가지로 을 바탕으로 식을 정리하면 와 같다. 이렇게 로만 모든 을 표현함으로 시간 지연을 줄일 수 있는 것이다.

**6.**

이진수를 사용하여 음수를 표현할 때에 컴퓨터는 부호를 사용하여 구분하지 않고, 다양한 방법으로 음수를 나타낸다. 그 중에서도 Sign-magnitude representation이란 방법이 있는데 이 방법은 이진수의 맨 앞자리 bit는 sign indicator로 사용하는 방법이다. 양수 5는 8421code로 0101이지만 음수 5를 표현할 때는 앞선 0101에서 맨 앞자리 수를 1로 바꾸어 1101로 표현한다. 그러나 이 방법은 이진수의 첫번째 bit를 부호를 나타내는 데에 사용하기 때문에 4bit 이진수는 +7부터 -7까지만 표현할 수 있고 arithmetic이 복잡하기 때문에 컴퓨터는 보통 2의 보수 방법을 사용하여 음수를 나타낸다. 2의 보수로 나타내기 위해서는 모든 자릿수의 비트를 반전시킨 후에 1을 더하면 된다.

각 논리 회로 별로 전파지연이 존재하는데 NOT gate는 1ns, buffer gate는 2ns, AND gate는 2.4ns, OR gate는 2.4ns, NAND gate와 NOR gate는 1.4ns, XOR gate는 4.2ns, XNOR gate는 3.2ns이다. 이러한 각 gate들의 전파지연을 최대한 줄이기 위해 다양한 감산기와 가산기를 사용하는 것이고, 이렇게 감산기와 가산기를 이용하면 비싼 값의 gate사용을 줄일 수 있기 때문에 보다 더 효율적인 회로를 만들 수 있다는 장점도 있다.