실습 2. 덧셈기와 뺄셈기

■ 실습목표

- 산술연산을 수행하는 논리회로를 설계한다.
- 모듈 개념에 기반한 계층설계 방법을 이해한다.
- 덧셈기를 활용하여 뺄셈기를 구현한다.

■ 사전지식

- 논리회로를 게이트로 구현하는 방법
- CircuitVerse로 모듈을 구현하는 방법 (별도 CircuitVerse 모듈 생성 문서 참조)

■ 예습문제

1. 다음 회로의 입력과 출력의 수를 결정하시오.

(a) 1-bit adder

half adder의 경우 입력은 2개, 출력은 2개.
full adder의 경우 입력은 3개, 출력은 2개 입니다.

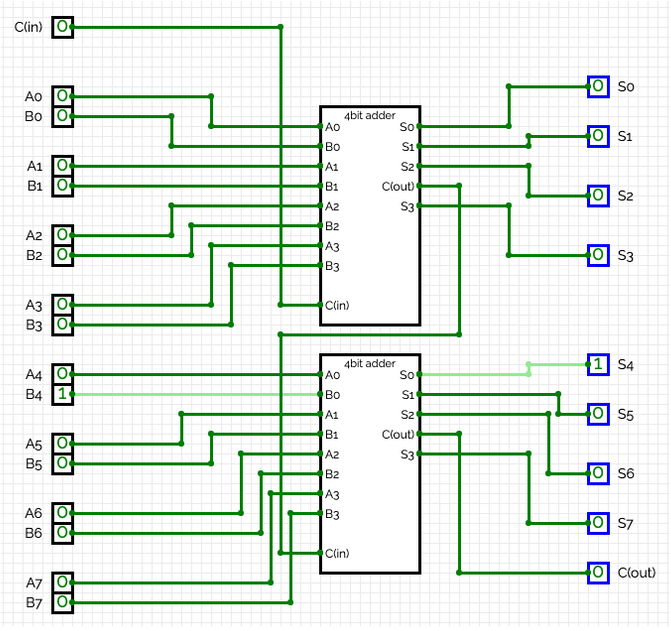
(b) 4-bit adder

9개의 입력, 5개의 출력으로 이루어집니다.

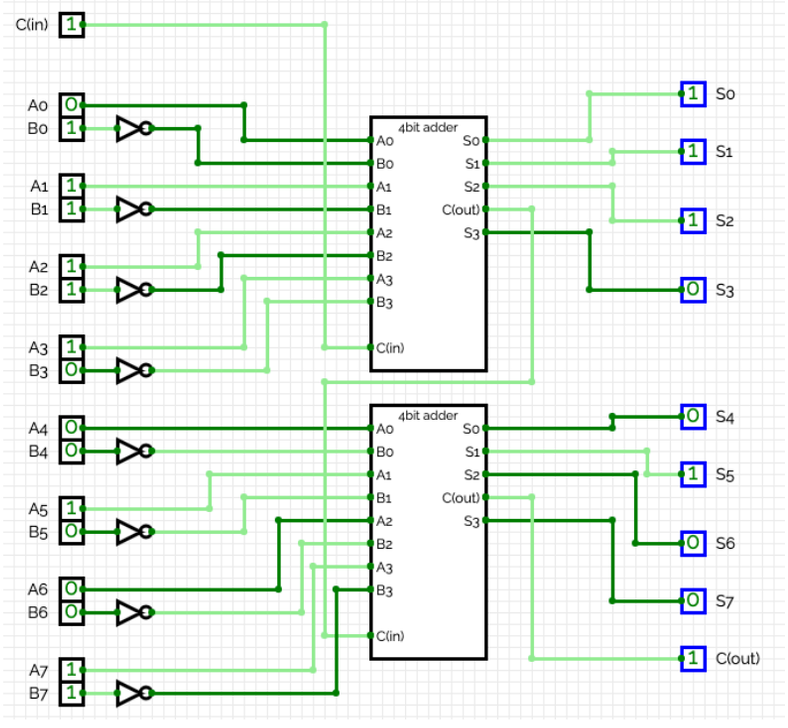
(c) 8-bit adder

17개의 입력, 9개의 출력으로 이루어집니다.

2. 슬라이드 4-16에 1-bit adder가 4개 주어질 때 4-bit adder를 설계하는 방법이 제시되어 있다. 이제 슬라이드 4-15와 같은 4-bit adder가 2개 주어질 때. 8-bit adder를 설계하시오.



3. 문제 2의 8-bit adder에 약간의 NOT gate를 추가하여 8-bit subtractor를 설계하시오.



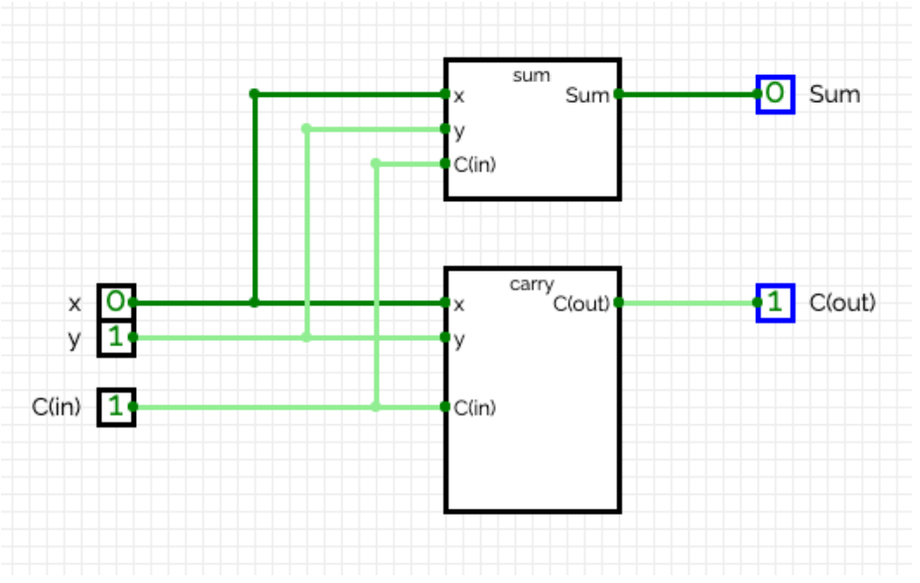
$A = 10101110_{(2)}$
$= 174$

$B = 10000111_{(2)}$
$= 135$

$A - B = 39$
$= 00100111_{(2)}$

■ 실습과정

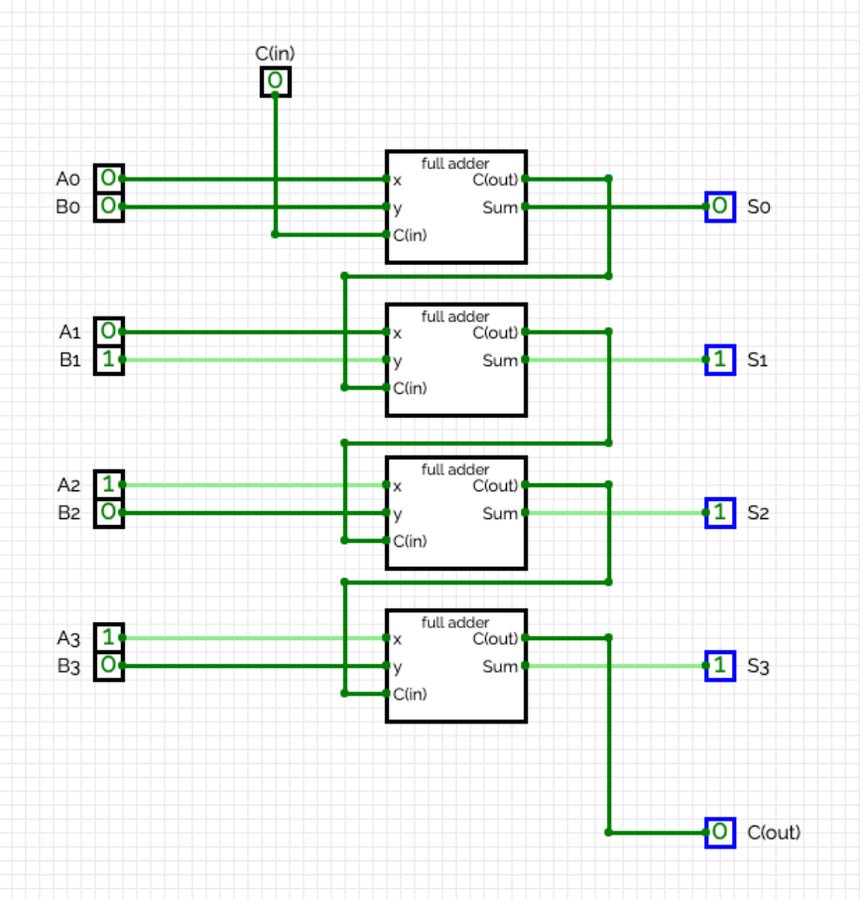
1. 슬라이드 4-18을 참조하여 1-bit adder를 설계하시오. 입력이 3개, 출력이 2개임에 주의하시오. 회로가 올바르게 동작함을 확인하고, 캡쳐하여 보고서에 포함하시오.



x 0
y 1
+. C.n. 1
10

Cout = 1, Sum = 0 정상 작동합니다.

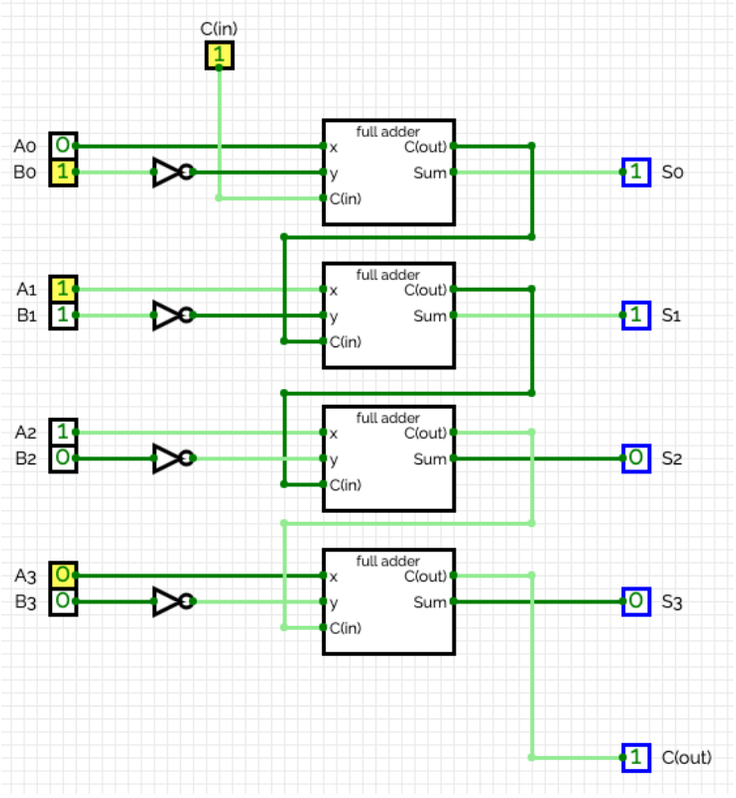
2. 동일한 1-bit adder를 4개 활용하여 4-bit adder를 설계하려고 한다. 별도로 제시된 모듈 생성 방법을 숙지하고, 우선 1-bit adder를 모듈로 정의하되, 모듈의 입력이 3개, 출력이 2개임에 주의하시오. 4개의 1-bit adder 모듈을 활용하여 4-bit adder를 설계하시오. 임의의 4-bit 수가 2개 입력될 때 회로가 올바르게 동작함을 확인하고, 캡쳐하여 보고서에 포함하시오.



$0011_{(2)} + 0100_{(2)} = 0111_{(2)}$

정상 작동합니다.

3. 슬라이드 4-20을 참조하여 실습과정 2에서의 4-bit subtractor를 설계하시오. 실습 과정 2에서와 같이 1-bit adder 모듈을 활용하여야 한다. 임의의 4-bit 수가 2개 입력될 때 회로가 올바르게 동작함을 확인하고, 캡쳐하여 보고서에 포함하시오.



$A = 0110_{(2)} = 6$

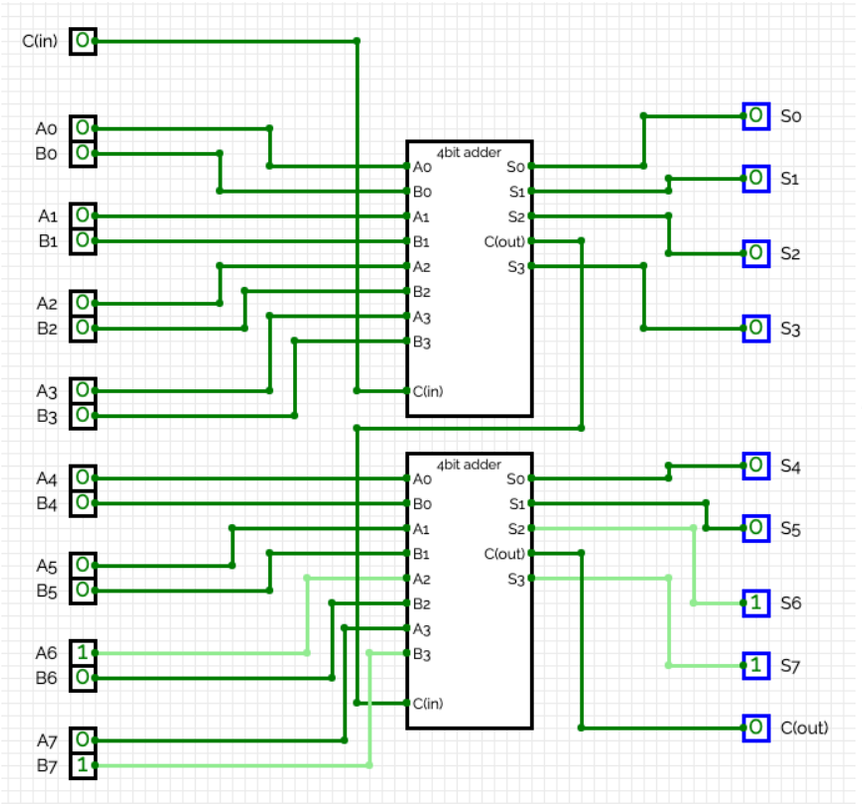
$B = 1100_{(2)} = 3$

$A - B = 3$

$= 0011_{(2)}$

정상작동합니다.

4. 실습과정 2에서의 4-bit adder를 모듈로 정의하시오. 두 개의 4-bit adder 모듈을 활용하여 8-bit adder를 설계하시오. 임의의 8-bit 수가 2개 입력될 때 회로가 올바르게 동작함을 확인하고, 캡쳐하여 보고서에 포함하시오.



A=64
= 01000000(2)
B=128
= 10000000(2)
A+B
= 192
= 11000000(2)
정상 작동합니다.

■ 정리

1. 1-bit full subtractor를 설계하려고 한다. Carry 대신 borrow가 주어진다. 슬라이드 1-18을 참조하되, X, Y, B_{in}이 입력으로 주어질 때, Diff, B_{out}을 출력하여야 한다. (B_{in}과 B_{out}은 각각 입력 빌림수와 출력 빌림수이다.)

   (a) 슬라이드 4-19를 참조하여 진리표를 보이시오.

X	Y	B_{in}	Diff	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

   (b) 간소화된 논리식을 유도하시오. 슬라이드 4-18과 같이 XOR 연산자를 사용할 수 있다.

Diff에 대한 카르노맵

X \ YB_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

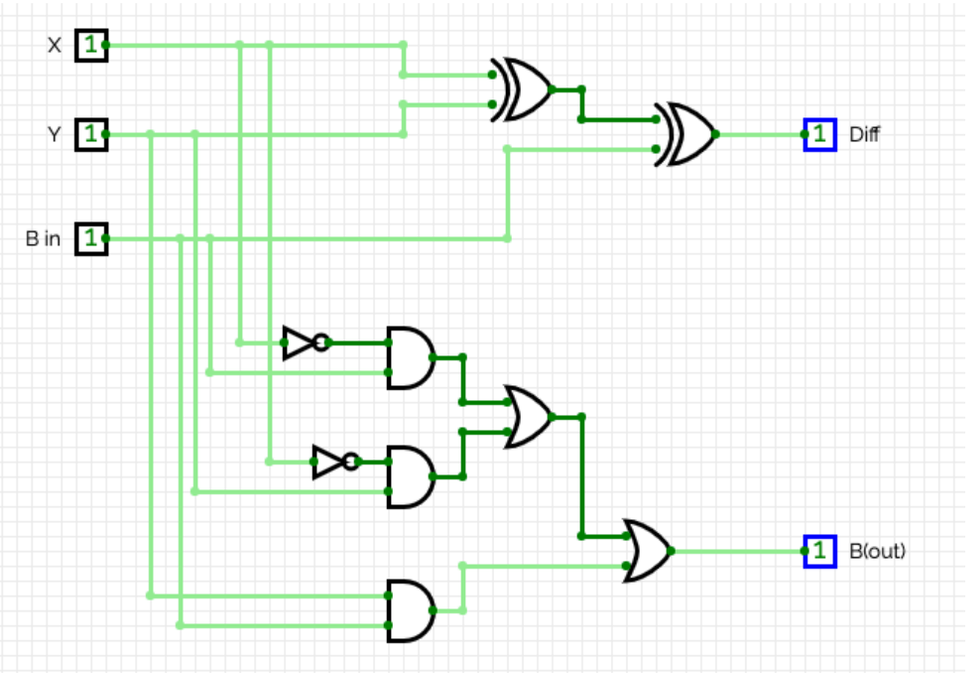
$$\begin{aligned} Diff &= XY'B_{in}' + X'Y'B_{in} + XYB_{in} + X'YB_{in}' \\ &= X(Y'B_{in}' + YB_{in}) + X'(Y'B_{in} + YB_{in}') \\ &= X(Y \oplus B_{in})' + X'(Y \oplus B_{in}) \\ &= X \oplus Y \oplus B_{in} \end{aligned}$$

B_{out}에 대한 카르노맵

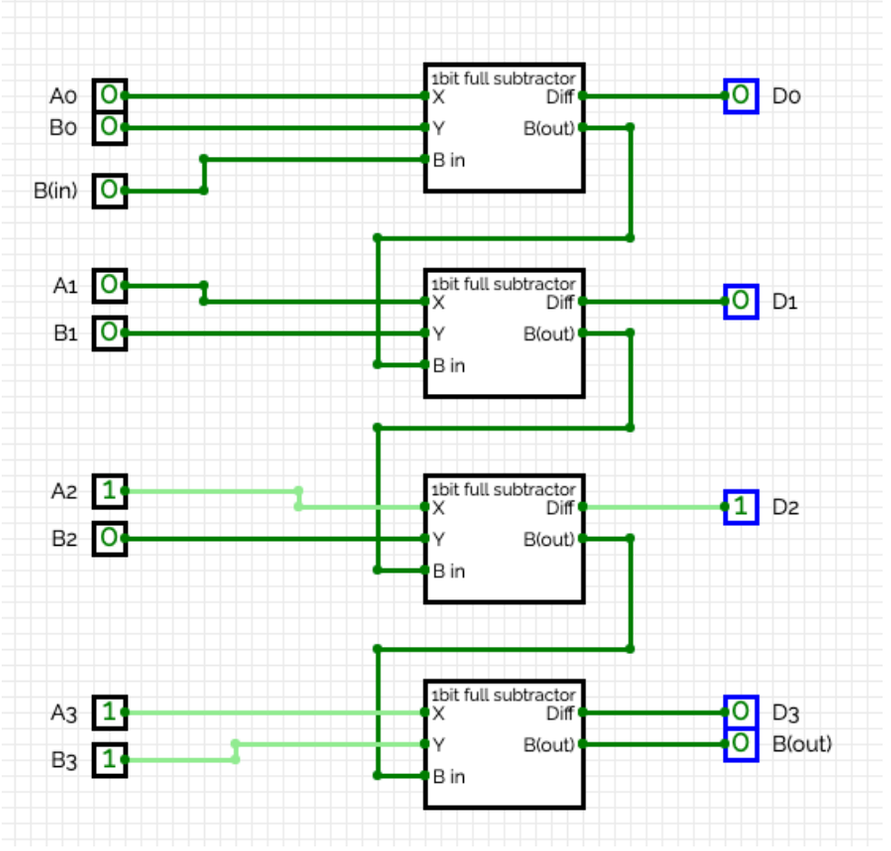
X \ YB_{in}	00	01	11	10
0	0	1	1	1
1	0	0	1	0

$$B_{out} = X'B_{in} + X'Y + YB_{in}$$

(c) 슬라이드 4-18과 같이 논리도로 표현하시오.



(d) 슬라이드 4-16과 같이 4개의 full subtractor를 활용하여 4-bit subtractor를 설계하시오.



2. 실습과정에서의 1-bit adder, 4-bit adder, 그리고 8-bit adder의 동작 시간을 추정하시오. 회로의 동작시간은 신호가 통과하는 게이트의 수에 비례한다고 가정한다.

1-bit adder의 경우 Sum에서 1개, Carry에서 4개 총 8개의 게이트를 사용하였습니다.

4-bit adder은 1-bit adder가 4개가 모인 것이므로, 1-bit adder의 4배가 소요됩니다.

8-bit adder은 4-bit adder가 2개 모인 것이므로, 1-bit adder의 8배가 소요됩니다.