Lab4. 이진수 연산

학번 : 20210273 이름 : 하태혁

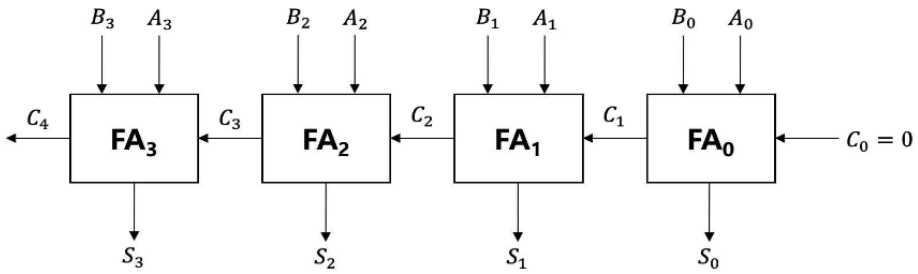
1. 개요

이진수 연산에 필요한 회로를 구현한다. carry-in, carry-out값에 따라 Adder의 종류를 다르게 해야 하므로 먼저, 1-bit adder를 구현해 본다. 이후에는 이를 확장하여 N-bit에 대한 연산도 가능한 Ripple adder를 구현해본다. Ripple adder에서 특정 operator를 더하고 수정하면 Ripple subtractor를 구현할 수 있고 Ripple adder를 확장하여 Binary multiplier까지 구현해 볼 수 있다.

2. 이론적 배경

1-bit adder의 종류로는 Half-adder와 Full-adder가 있다. Half adder는 두개의 input Bit를 더해서 나오는 값을 계산하여 해당 bit에 남는 sum값과 overflow되는 C값을 출력으로 갖는다. Full-adder는 carry-in된 값도 input으로 받아서 계산할 수 있는 adder로 HA 두 개로 구현할 수 있다.

Ripple adder는 여러개의 full-adder로 구현할 수 있으며 아래 그림과 같이 4bit 수를 더해야 할 때는 4개의 FA를 연결해서 만든다. 이때 subtractor의 경우에는 B대신 B의 2의 보수를 더하면 되므로 각각의 B bit에 not gate를 연결하고 C_0에 1을 넣어서 해결한다.



Binary multiplier는 각 자리의 부분곱은 and gate를 사용하여 연산하고 연산한 각 bit의 자리수를 계산할 때 값들을 모두 더해야 하므로 앞에서 사용한 Ripple adder를 사용한다.

3. 실험 준비

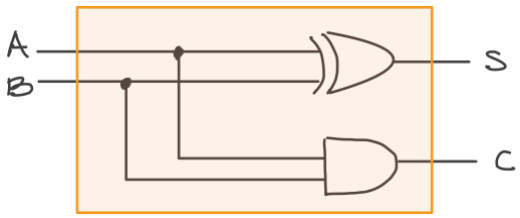
1) Lab4_1

반가산기(Half-adder)의 진리표는 다음과 같다.

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Output인 C와 S에 대한 Boolean expression을 작성하고 회로로 나타내면,

$S = A'B + AB' = A \oplus B$, $C = AB$



2) Lab4_2

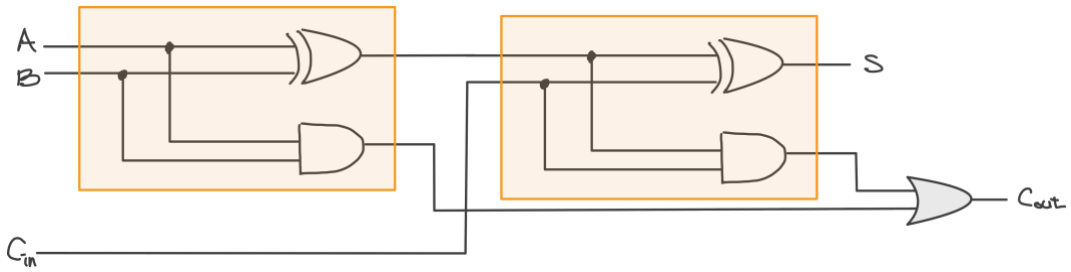
전가산기(Full-adder)의 진리표는 다음과 같다.

A	B	C_{in}	C_{out}	S
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

Output의 Boolean expression을 작성하고 회로로 나타내면

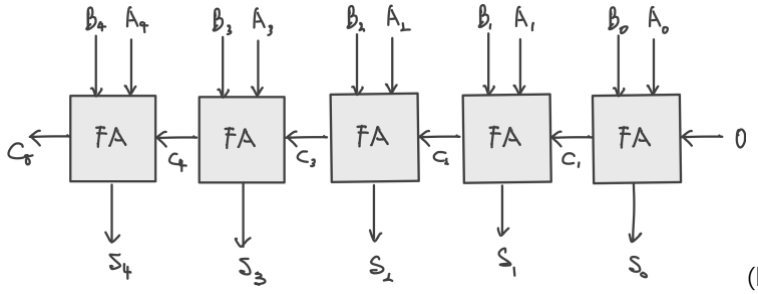
$C_{out} = A'BC_{in} + AB'C_{in} + ABC_{in}' + ABC_{in} = AB + (A \oplus B)C_{in}$

$S = A'B'C_{in} + A'BC_{in}' + AB'C_{in}' + ABC_{in} = A \oplus B \oplus C_{in}$

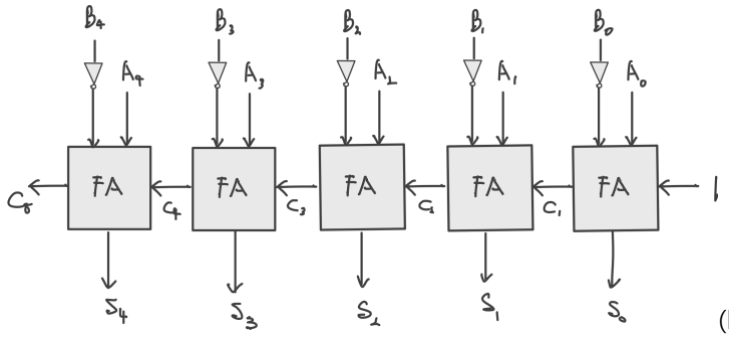


3) Lab4_3

5-bit 리플 가산기(ripple adder)와 리플 감산기(ripple subtractor)의 회로는 다음과 같다.

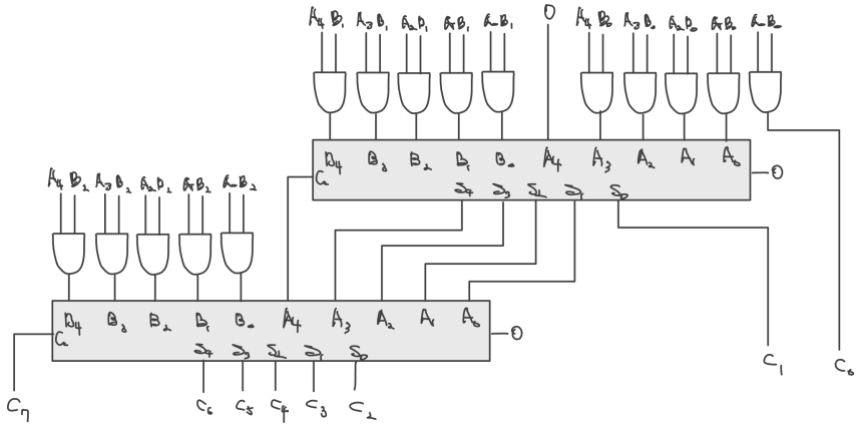


(Ripple adder)



(Ripple subtractor)

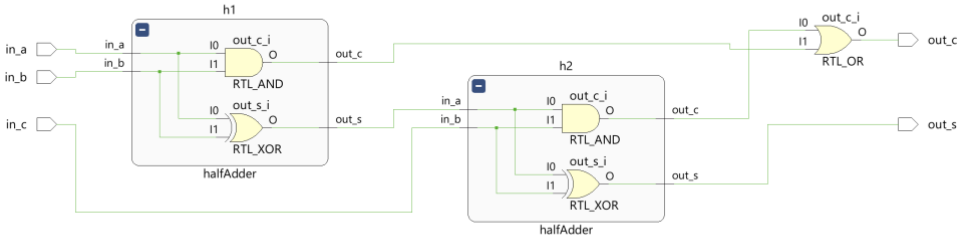
4) Lab4_4



5x3 이진 곱셈기의 회로는 다음과 같다.

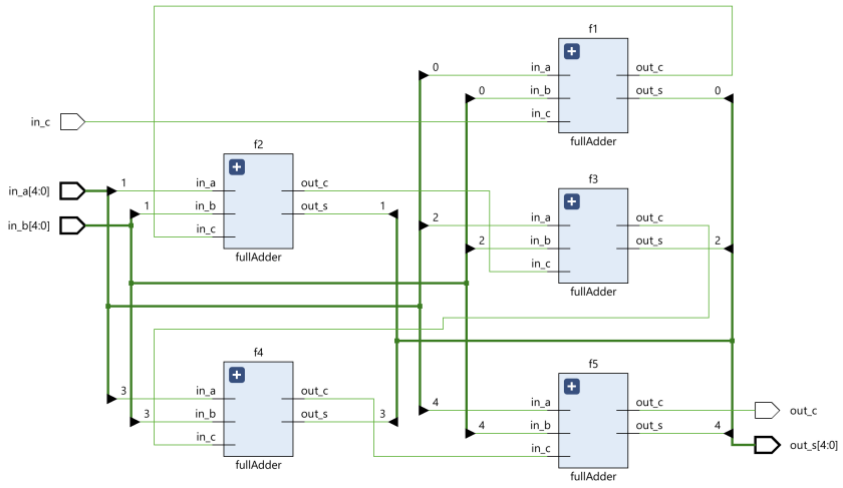
4. 결과

1) Lab4_1

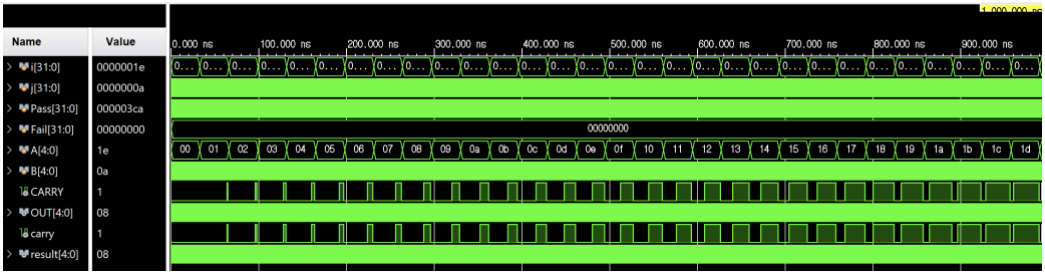


HA 2개를 써서 FA 한 개를 구현하였다. 실험 준비에서 예측했던 결과와 동일한 회로가 나왔음을 확인할 수 있다.

2) Lab4_2

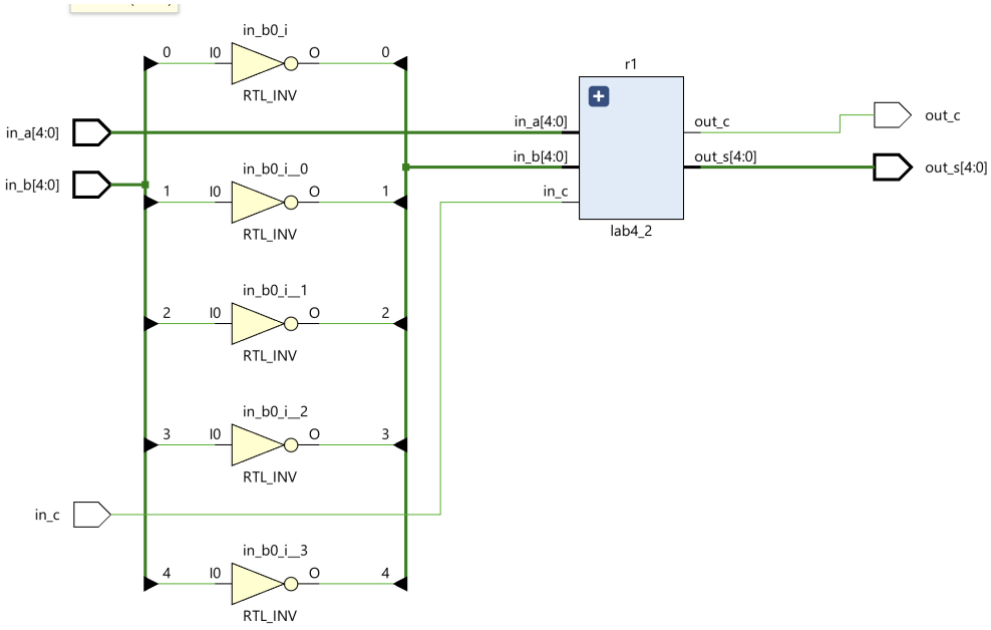


Full adder 5개를 연결해 5bit ripple adder를 구현하였다. 실험준비에서 예상한 회로와 동일함을 알 수 있다.

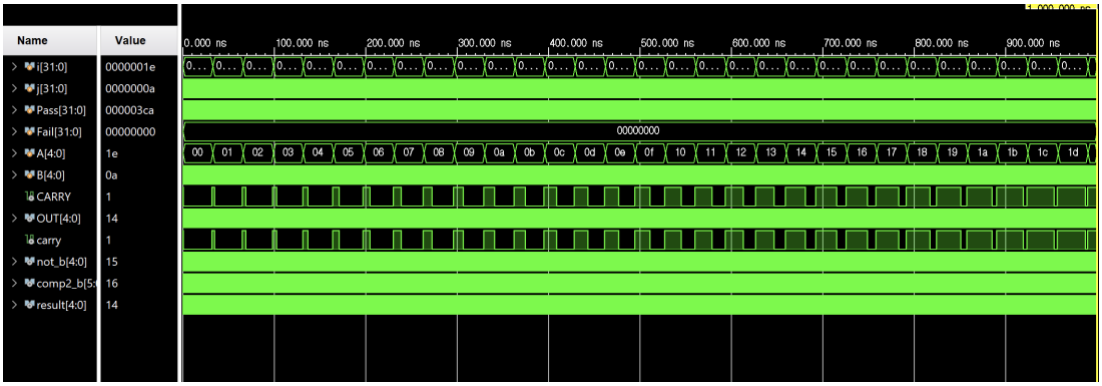


주어진 테스트벤치로 시뮬레이션을 돌렸을 때의 결과는 위와 같다.

3) Lab4_3

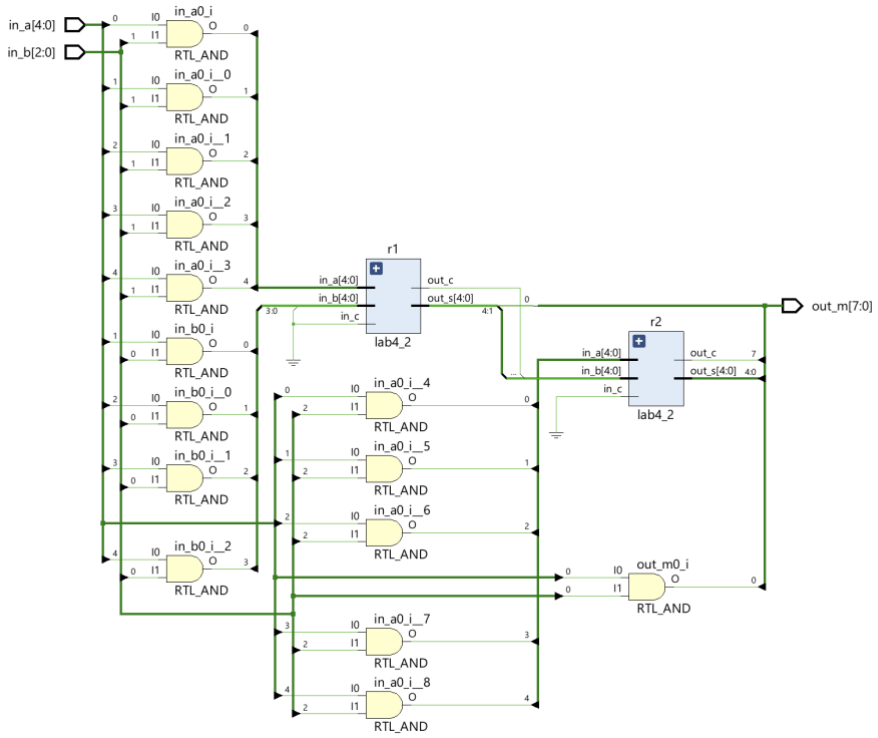


Ripple adder에서 빼는 값에 not gate를 연결한 뒤, 처음 in_c 값을 1로 하여 2의 보수를 계산한다. Ripple adder에 연결하여 값을 계산한다.

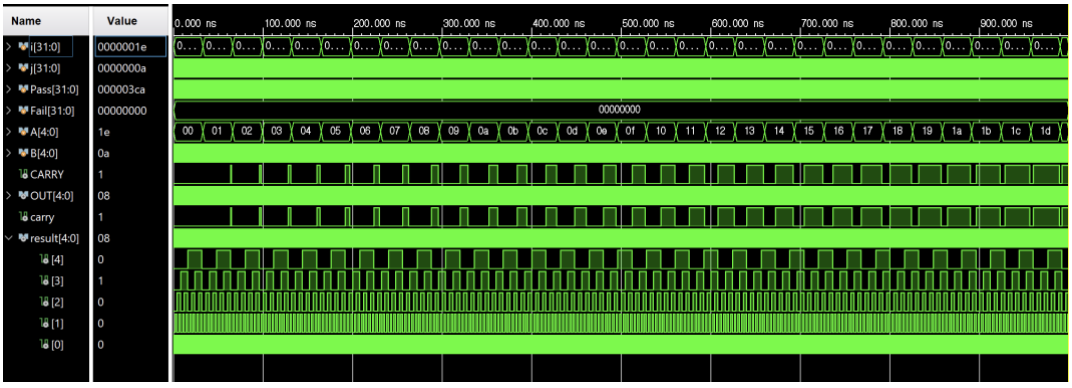


주어진 테스트벤치로 시뮬레이션을 돌렸을 때의 결과는 위와 같다.

4) Lab4_4



코드로 구현한 5x3 이진 곱셈기의 회로는 실험 준비의 결과와 같음을 확인할 수 있다.



주어진 테스트벤치로 시뮬레이션을 돌렸을 때의 결과는 위와 같다.

5. 논의

리플 가산기와 감산기를 구현할 때 각각 따로 구현하지 않고 아래의 그림과 같이 sub input값과 XOR gate를 사용하면 sub의 값에 따라 가산기와 감산기의 역할을 모두 수행할 수 있는 연산기를 제작할 수 있었을 것 같다.