10주차 MSI / LSI 연산회로

20221596 이성진
20202103 맹재영
20191264 윤성민

목차

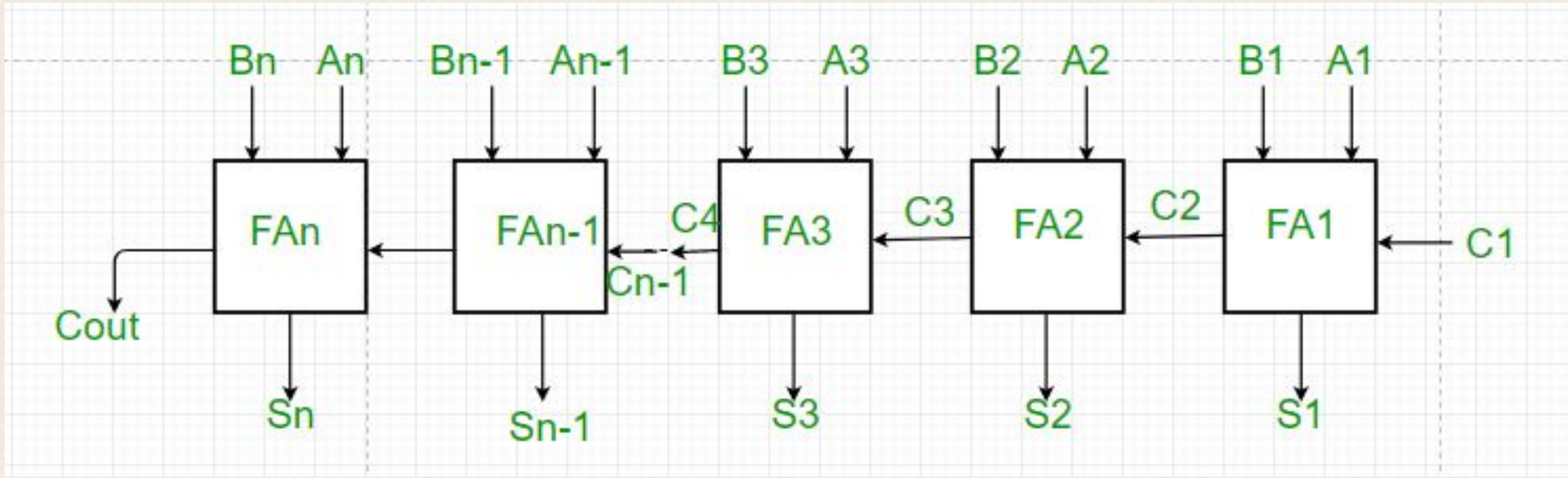
- ☑ MSI / LSI
- ☑ 4-Bit Binary Parallel Adder & Subtractor
- ☑ 2's Complement Using XOR
- ☑ 4-bit Carry Look Ahead Adder(CLA)
- ☑ BCD Adder
- ☑ ALU

MSI / LSI 연산 회로

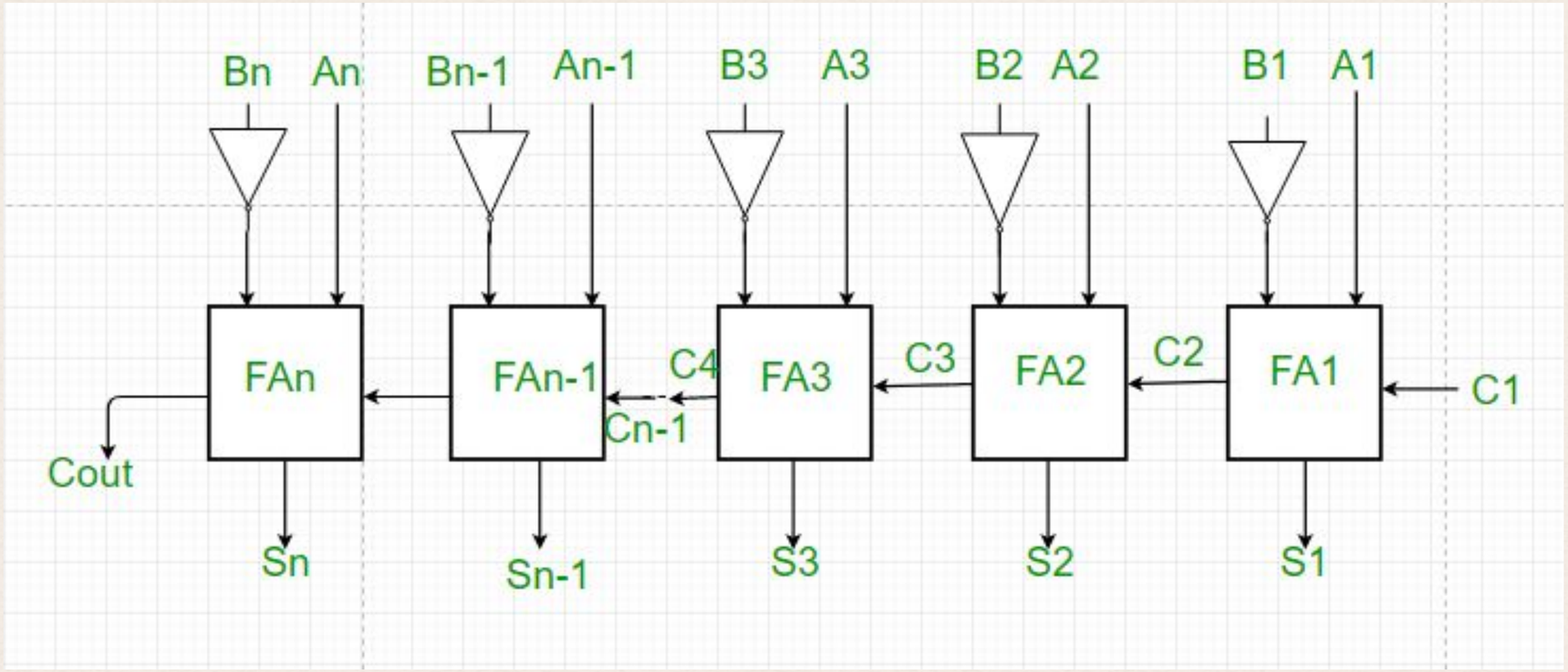
- MSI(Medium Scale Integrated Circuit)
  - 하나의 칩에 100~1000 개의 gate를 구현한 집적 회로(Integrated Circuit)
  - adder, subtractor, comparator, decoder, encoder, multiplexer, demultiplexer, ROM, PLA

- LSI(Large Scale Integrated Circuit)
  - 하나의 칩에 수천~10만 개의 gate를 구현한 집적 회로(Integrated Circuit)
  - 마이크로프로세서 등

4-Bit Binary Parallel Adder



4-Bit Binary Parallel Subtractor



2's Complement Using XOR

Binary representation of 5 is: 0 1 0 1

1's Complement of 5 is: 1 0 1 0

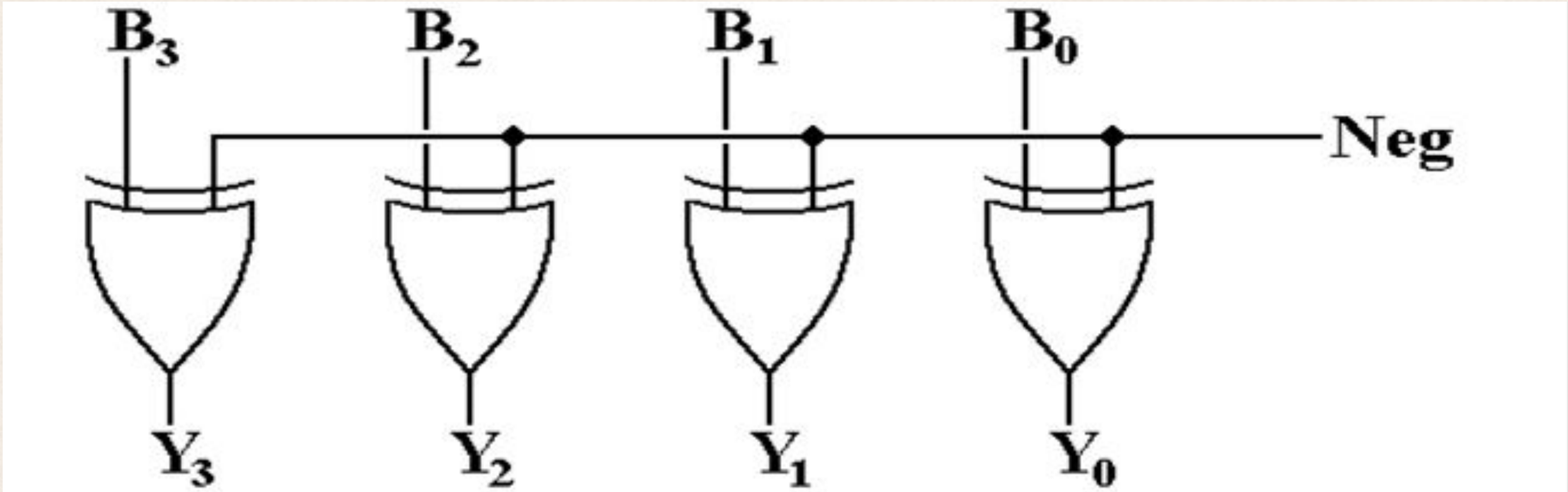
2's Complement of 5 is: (1's Complement + 1) i.e.

1 0 1 0 (1's Compliment)

\+ 1

1 0 1 1 (2's Complement i.e. -5)

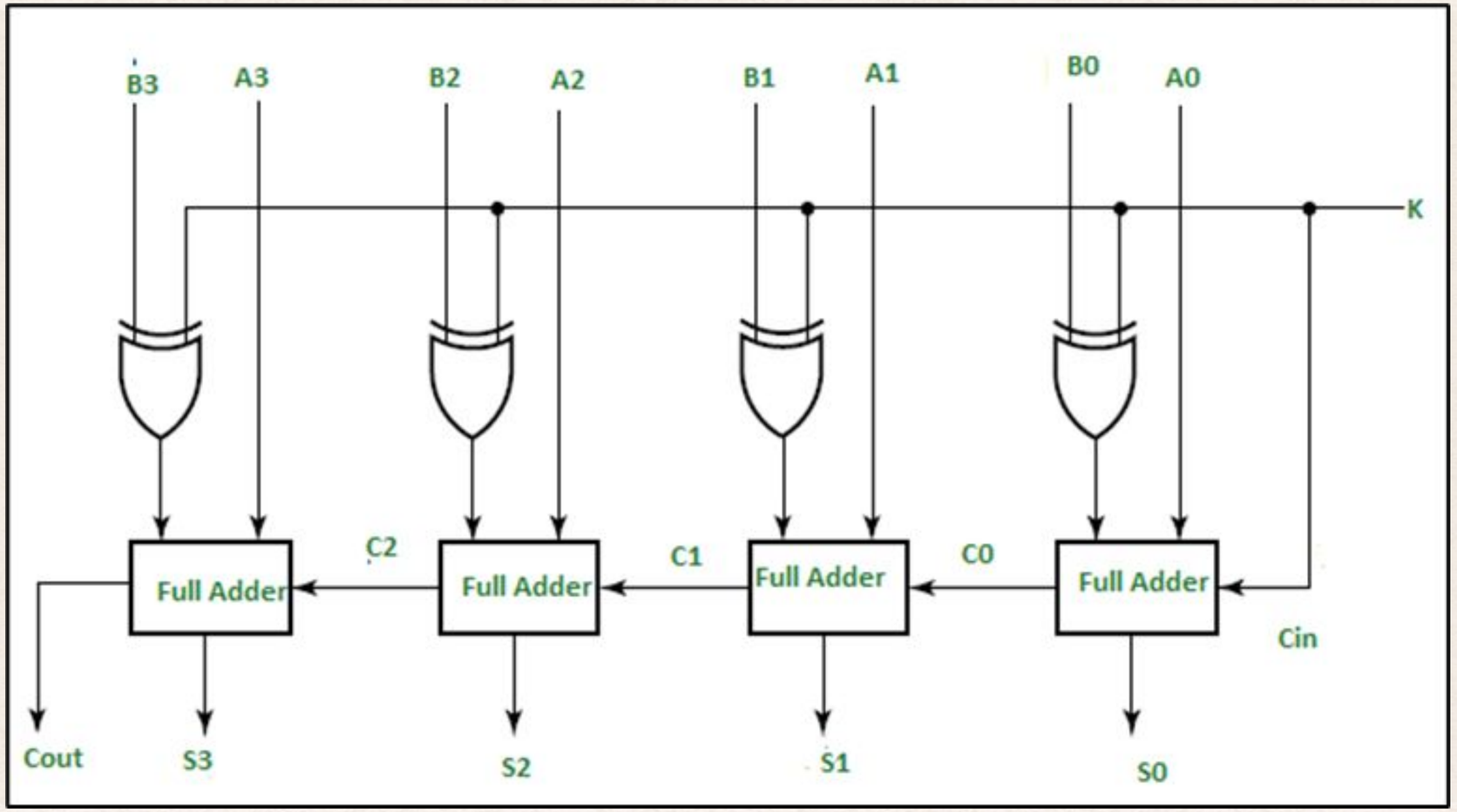
2's Complement Using XOR



If Neg = 0 Then $Y_3 = B_3, Y_2 = B_2, Y_1 = B_1$, and $Y_0 = B_0$

If Neg = 1 Then $Y_3 = \overline{B}_3, Y_2 = \overline{B}_2, Y_1 = \overline{B}_1$, and $Y_0 = \overline{B}_0$

2's Complement Using XOR

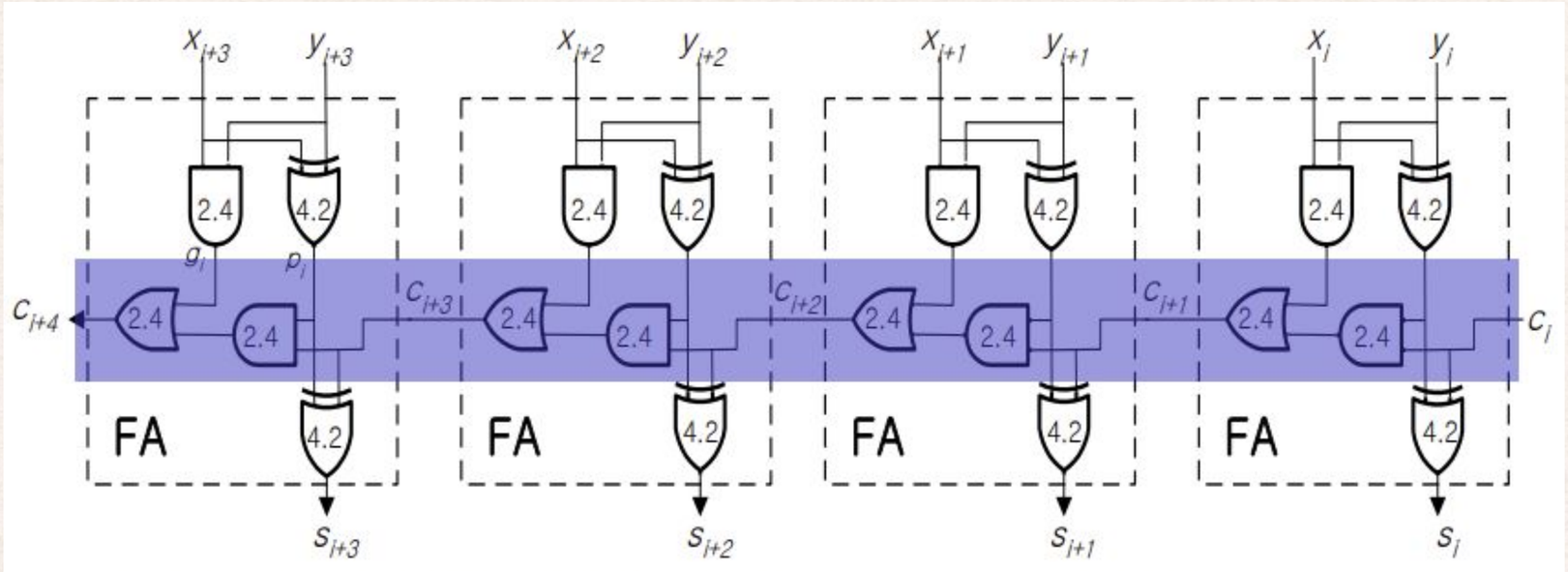


4-Bit Carry Look Ahead Adder (CLA)

☑ 4-bit Binary Parallel Adder

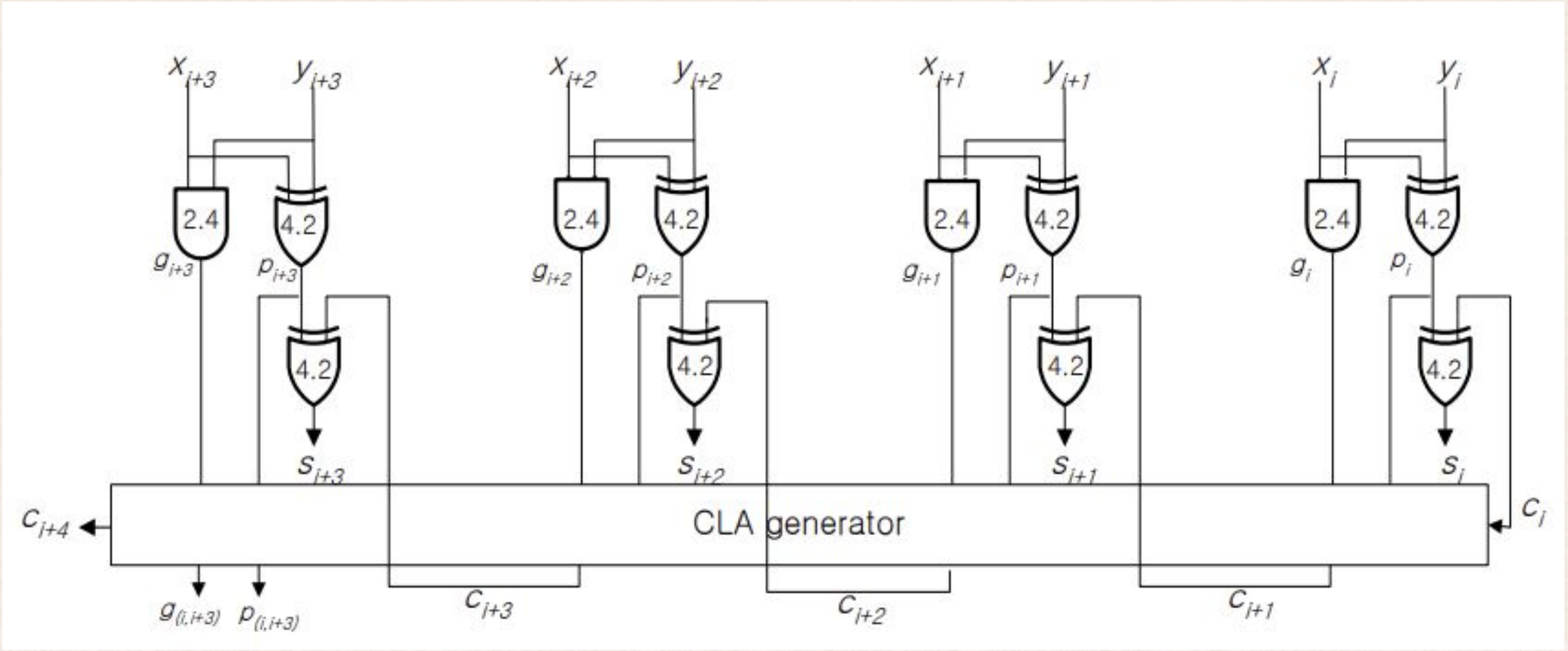
↳FA를 구성하고 있는 각각의 게이트들에는 gate delay가 존재

↳carry값을 구하기 위해서 이전의 carry값들의 연산 과정을 모두 기다려야 함 ⇒ 시간이 오래 걸림



4-Bit Carry Look Ahead Adder (CLA)

☑ 속도가 느린 4-bit Binary Parallel Adder의 단점을 해결하기 위한 CLA



4-Bit Carry Look Ahead Adder (CLA)

- Carry-Generate Function : $G_i = X_i \cdot Y_i$
- Carry-Propogate Function : $P_i = X_i \oplus Y_i$
- $S_i = X_i \oplus Y_i \oplus C_i \Rightarrow S_i = P_i \oplus C_i$
- $C_{i+1} = X_iY_i + (X_i \oplus Y_i)C_i \Rightarrow C_{i+1} = G_i + P_iC_i$

4-Bit Carry Look Ahead Adder (CLA)

☑ $C_{i+1} = G_i + P_i \cdot C_i$

↳ $C_{i+1} = G_i + P_i \cdot C_i$

↳ $C_{i+2} = G_{i+1} + P_{i+1} \cdot C_{i+1} = G_{i+1} + P_{i+1} \cdot G_i + P_{i+1} \cdot P_i \cdot C_i$

↳ $C_{i+3} = G_{i+2} + P_{i+2} \cdot C_{i+2} = G_{i+2} + P_{i+2} \cdot G_{i+1} + P_{i+2} \cdot P_{i+1} \cdot G_i + P_{i+2} \cdot P_{i+1} \cdot P_i \cdot C_i$

↳ $C_{i+4} = G_{i+3} + P_{i+3} \cdot C_{i+3} = G_{i+3} + P_{i+3} \cdot G_{i+2} + P_{i+3} \cdot P_{i+2} \cdot G_{i+1} + P_{i+3} \cdot P_{i+2} \cdot P_{i+1} \cdot G_i + P_{i+3} \cdot P_{i+2} \cdot P_{i+1} \cdot P_i \cdot C_i$

BCD Adder

☑ BCD 코드

Decimal	Binay (BCD) 8 4 2 1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

☑ BCD 연산

ex) 6 + 3 = 9

```
 0110
+0011
------
 1001
```

ex) 7 + 5 = 12

```
 0111
+0101
------
 1100
```

-> BCD 코드로 표현 X

BCD Adder

☑ BCD 연산

ex) 7 + 5 = 12

```
  0111
+ 0101
------
  1100
```

-> BCD 코드로 표현 X

```
   1100
+  0110
-------
0001 0010
   ↓  ↓
   1  2
```

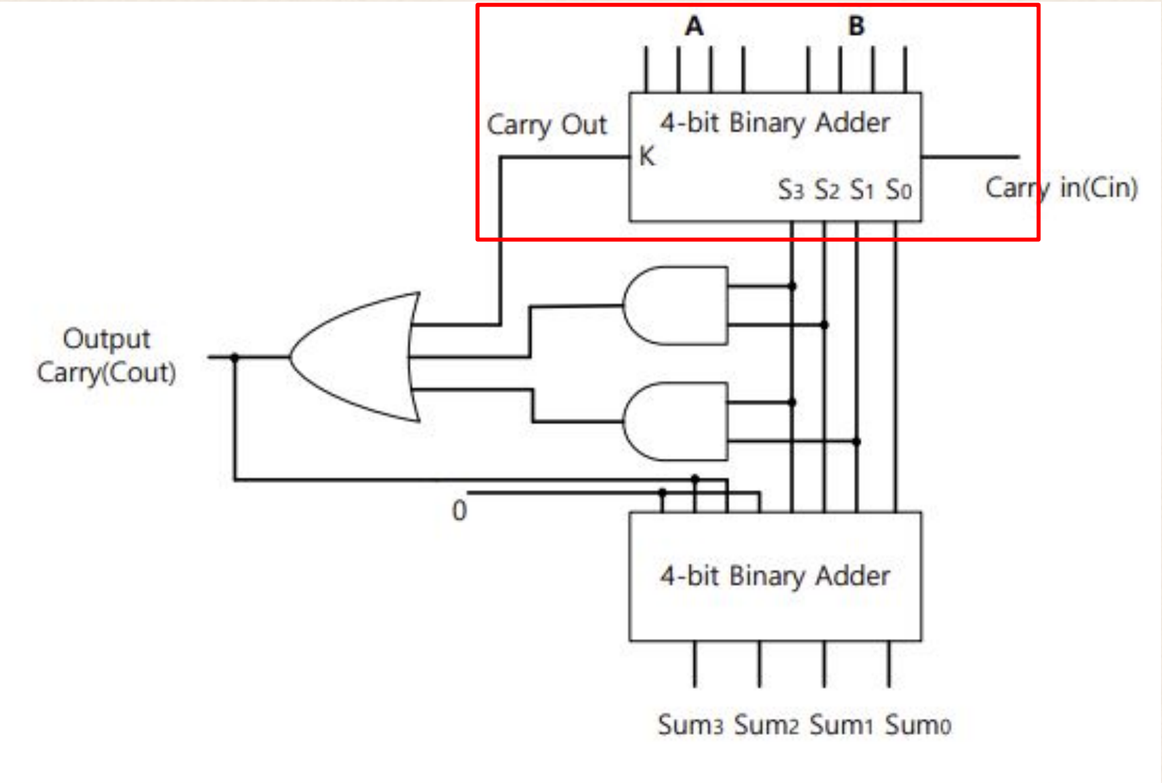
Decimal	Binary	BCD
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
14	1110	0001 0100
15	1111	0001 0101

BCD Adder

10진수	BCD 코드	10진수	BCD 코드	10진수	BCD 코드
0	0000	10	0001 0000	20	0010 0000
1	0001	11	0001 0001	31	0011 0001
2	0010	12	0001 0010	42	0100 0010
3	0011	13	0001 0011	53	0101 0011
4	0100	14	0001 0100	64	0110 0100
5	0101	15	0001 0101	75	0111 0101
6	0110	16	0001 0110	86	1000 0110
7	0111	17	0001 0111	97	1001 0111
8	1000	18	0001 1000	196	0001 1001 0110
9	1001	19	0001 1001	237	0010 0011 0111

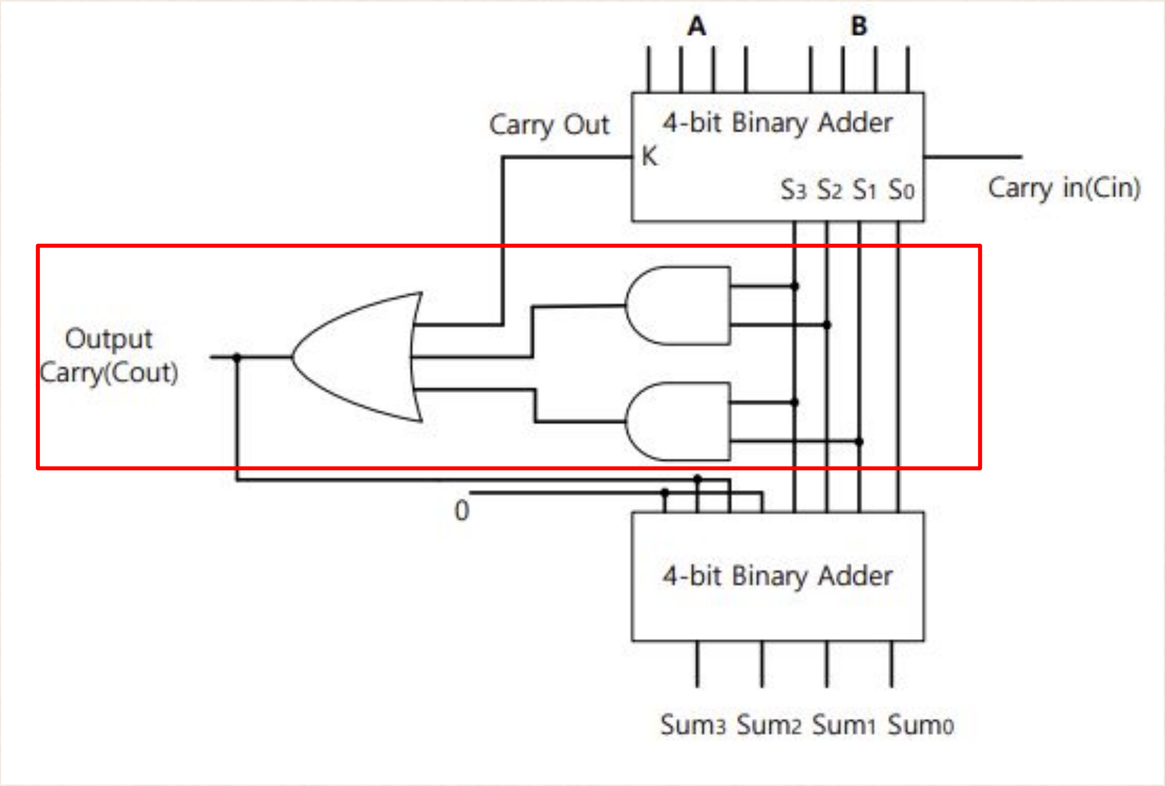
BCD Adder

☑ BCD Adder의 회로도



BCD Adder

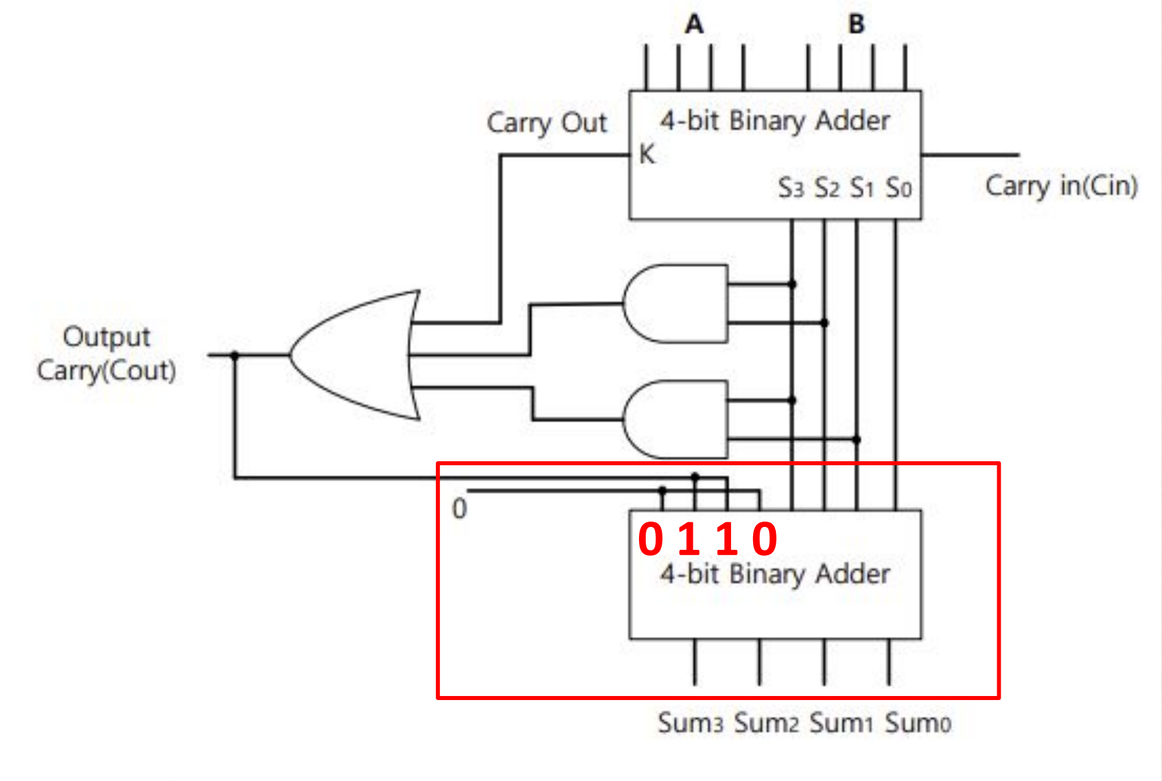
☑ BCD Adder의 회로도



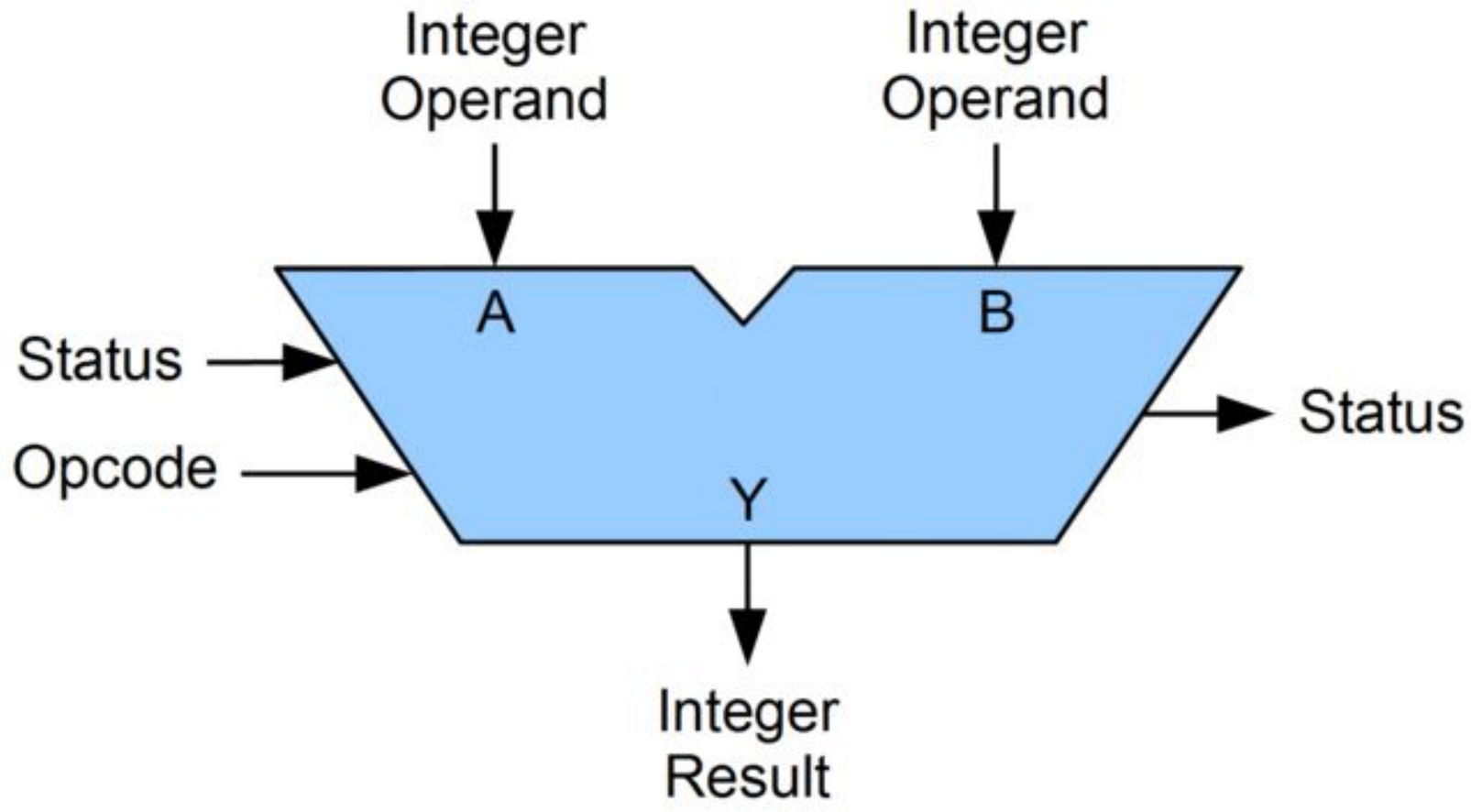
Decimal	Binary ($S_3S_2S_1S_0$)	BCD
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
14	1110	0001 0100
15	1111	0001 0101

BCD Adder

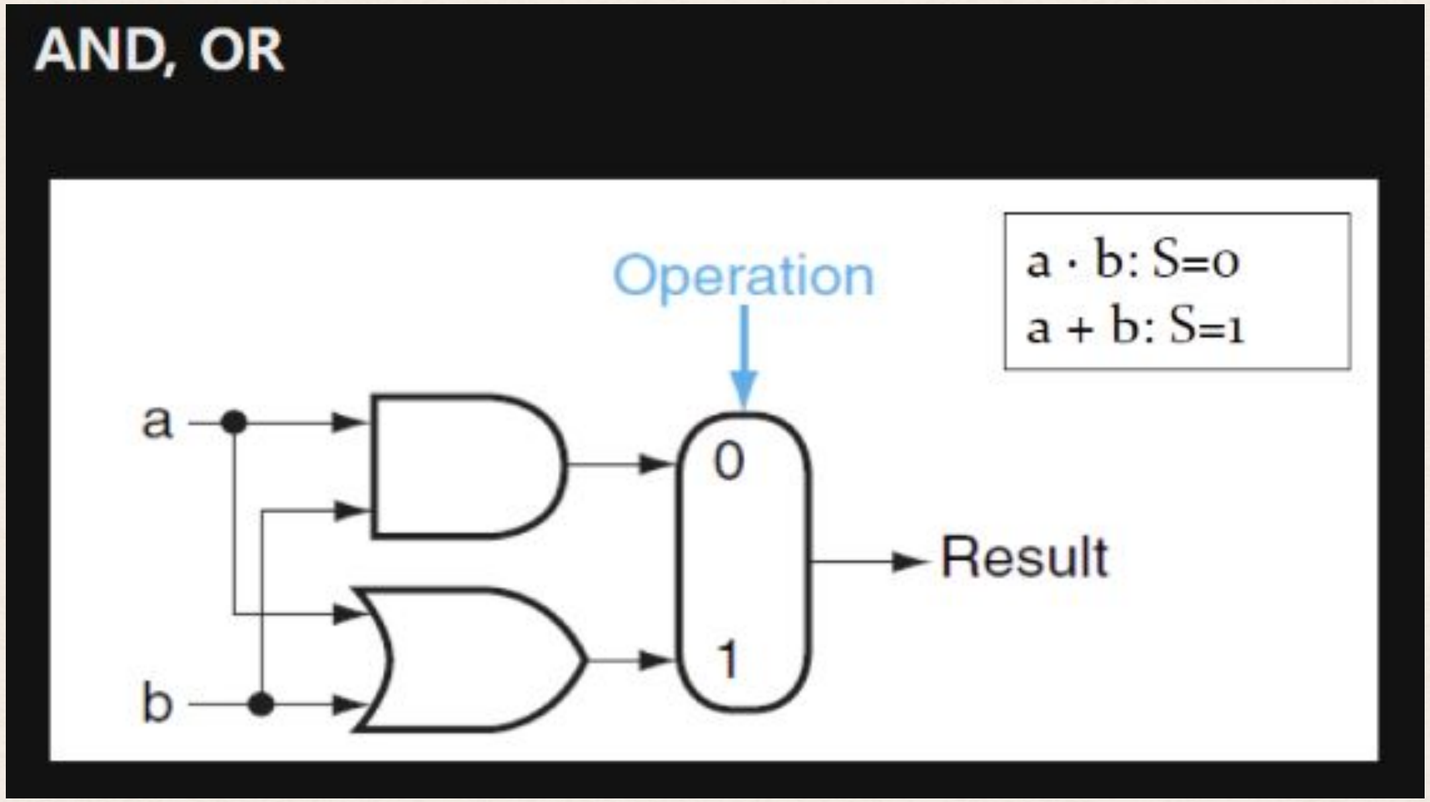
☑ BCD Adder의 회로도



ALU (Arithmetic and Logical Unit)



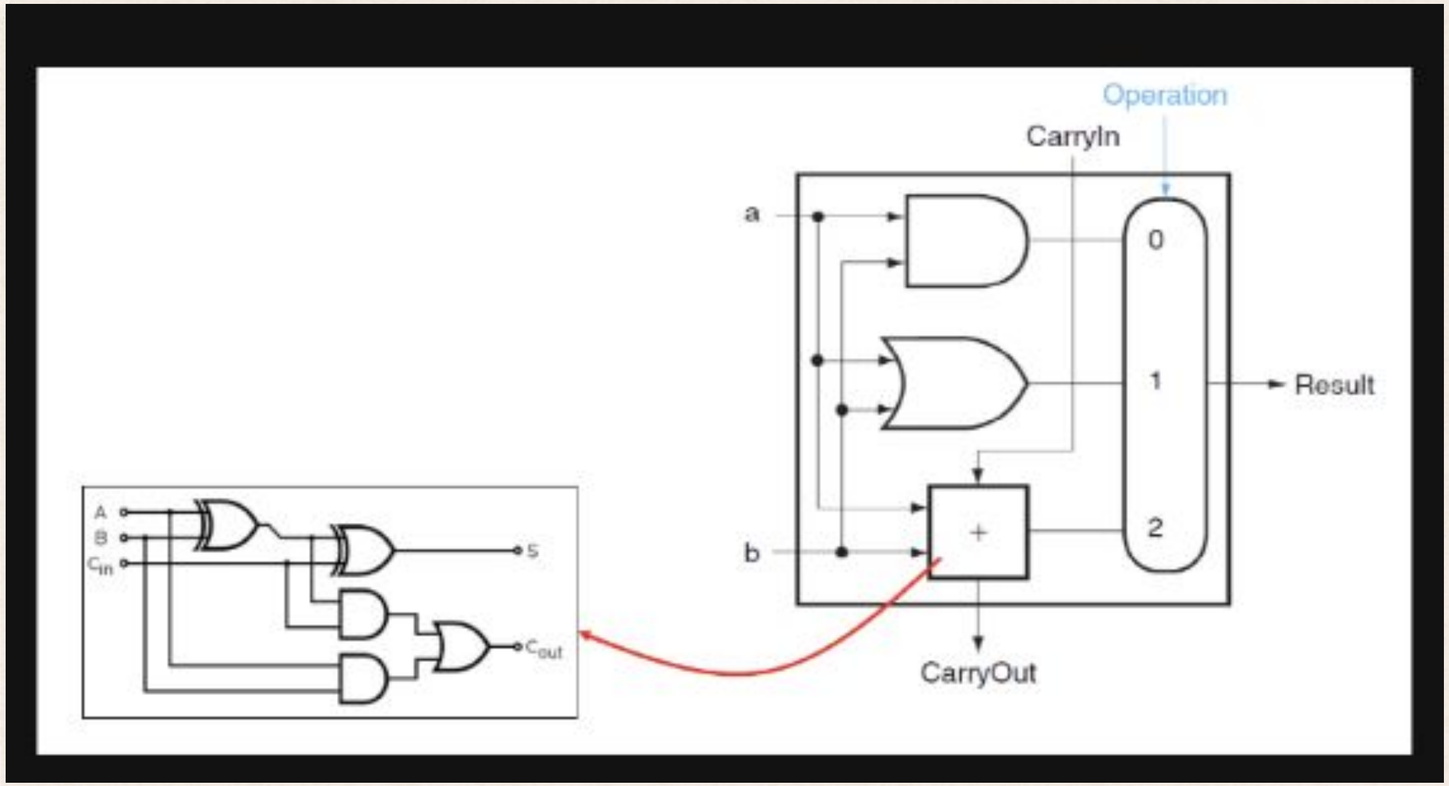
ALU (Arithmetic and Logical Unit)



a and b
a or b

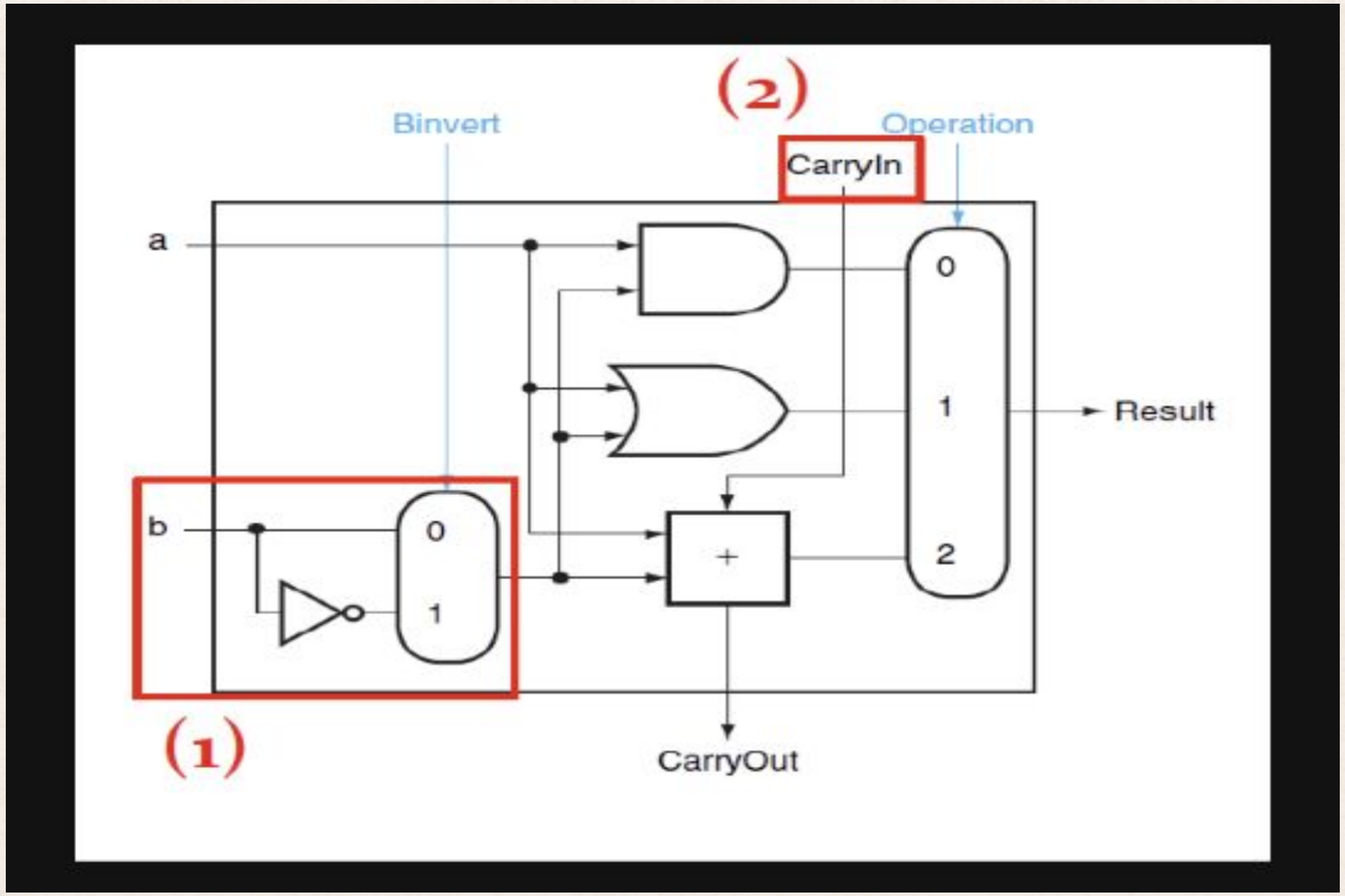
MUX를 통해 선택 전달

ALU (Arithmetic and Logical Unit)



adder회로 MUX에 추가

ALU (Arithmetic and Logical Unit)



2’s complement for subtractor function

ALU (Arithmetic and Logical Unit)

- N is a '1' if the last ALU operation produced a negative number;
- Z is a '1' if the last operation produced a '0' result;
- C is a '1' if the last operation produced a carry-out;
- V is a '1' if the last operation produced an overflow condition.

ex) 1010 = 10 (unsigned) range : 0 ~ 15
1010 = -6 (signed) range : -8 ~ 7

ALU (Arithmetic and Logical Unit)

Operation	Operand A	Operand B	Result indicating overflow
$A + B$	≥ 0	≥ 0	< 0
$A + B$	< 0	< 0	≥ 0
$A - B$	≥ 0	< 0	< 0
$A - B$	< 0	≥ 0	≥ 0

☑ 출처

-http://artoa.hanbat.ac.kr/lecture_data/digital_system/09.pdf
-https://www.geeksforgeeks.org/parallel-adder-and-parallel-subtractor/
-https://www.geeksforgeeks.org/2s-complement-for-a-given-string-using-xor/
-https://www.vedantu.com/maths/2s-complement-addition
-http://www.edwardbosworth.com/CPSC2105/Lectures/Slides_05/Chapter_03/BinaryAdders.htm
-https://en.wikipedia.org/wiki/Arithmetic_logic_unit
-https://aimpugn.blogspot.com/2017/10/2.html
-서강대학교 디지털 회로개론 chapter3 강의자료
-서강대학교 컴퓨터공학실험2 10주차 강의자료

☑ 기여도

-20221596 이성진 MSI/LSI & Binary Parallel Adder/Subtractor & 2’s complement (33.3%)
-20191264 윤성민 CLA & BCD (33.3%)
-20202103 맹재영 ALU & 심화 (33.3%)

감사합니다!!