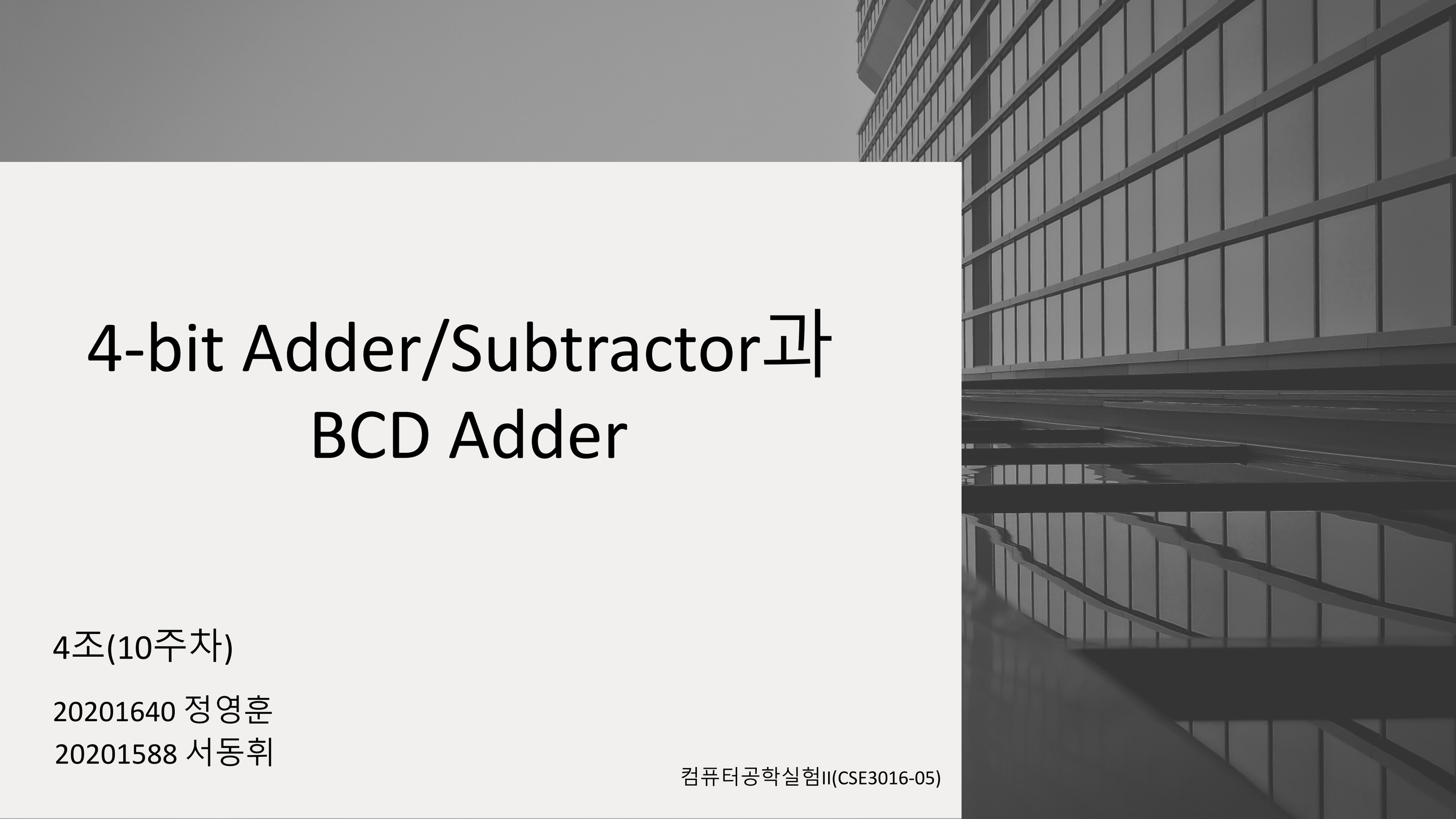


4-bit Adder/Subtractor과 BCD Adder

4조(10주차)

20201640 정영훈
20201588 서동휘

컴퓨터공학실험II(CSE3016-05)

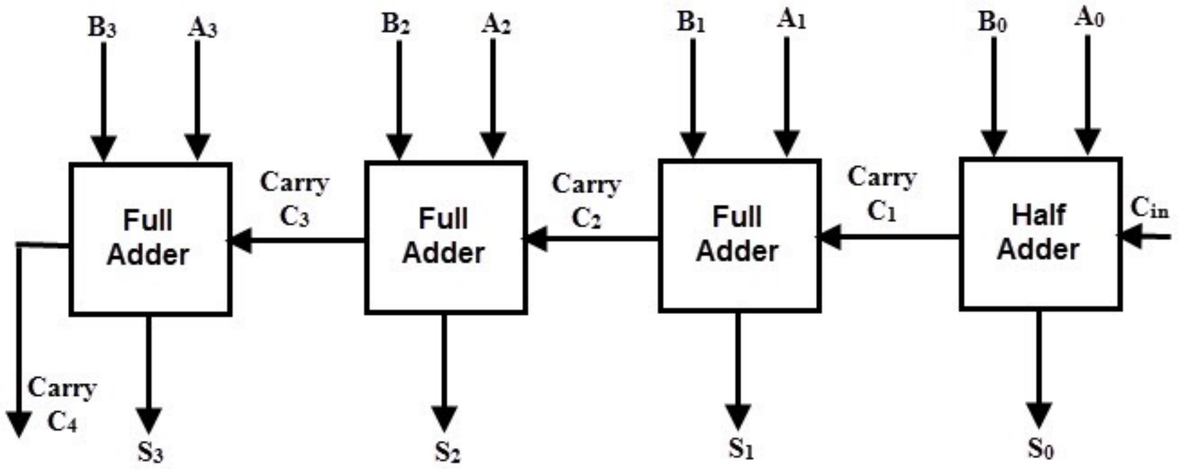
Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와 Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2’s complement
- BCD code의 성질
- BCD code - Adder
- BCD code –Subtractor(심화)

Index

- **4-bit adder, subtractor의 개요**
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- BCD code - Subtractor (심화)

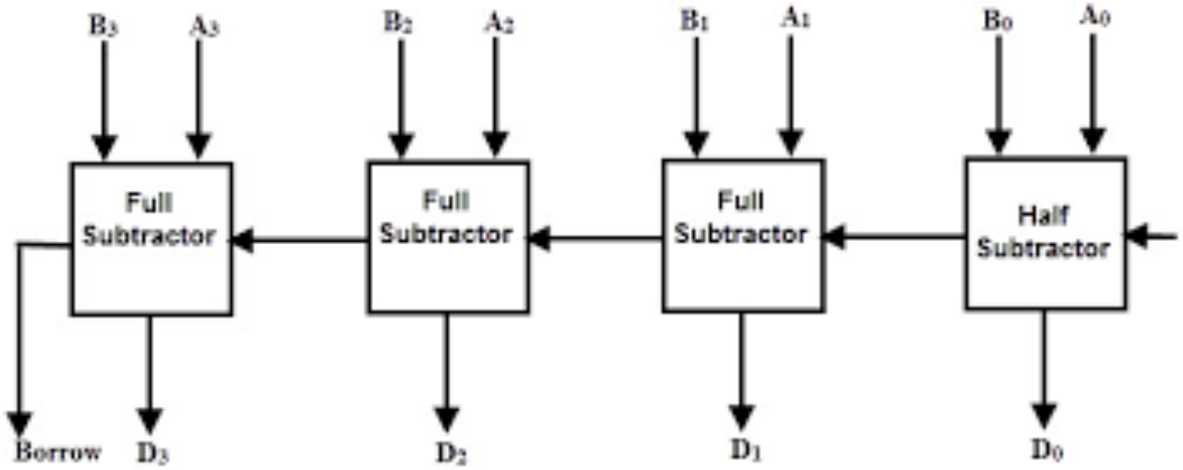
4-bit adder



- 4개의 bit로 이루어져 있는 2개의 2진수 숫자열의 덧셈을 해 주는 회로이다
- 동일한 위치에 있는 bit들의 값의 덧셈을 여러 번 반복하는 식으로 구현한다.

Index

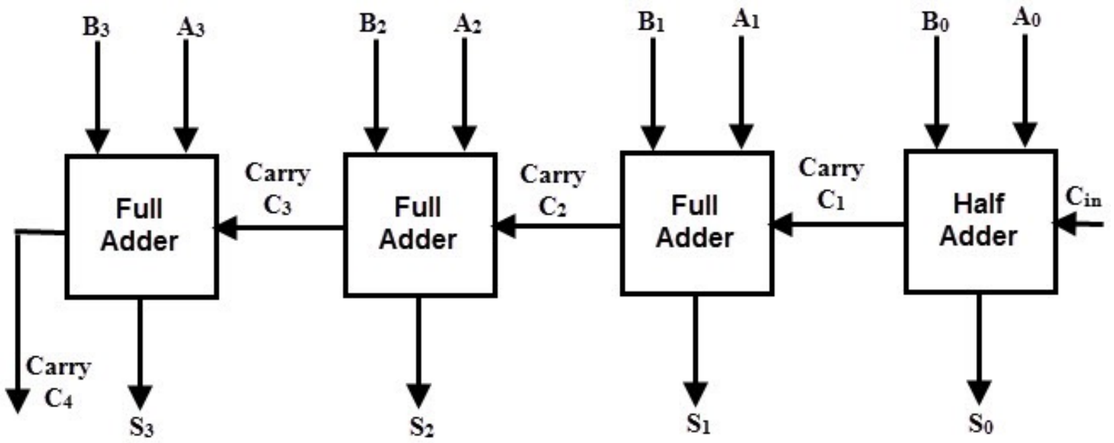
4-bit subtractor



- 4개의 bit로 이루어져 있는 2개의 2진수 숫자열의 뺄셈을 해 주는 회로
- 동일한 위치에 있는 bit들의 값의 뺄셈을 여러 번 반복하는 식으로 구현
- 뺄셈을 하는 방식에 따라 full subtractor, full adder로 구현할 수 있다.

Index

Ripple Carry Adder의 성질과 문제점



- 각 자리 연산을 수행하기 위해서는 앞 자리의 연산결과가 필요함
- 자릿수가 많아질 경우 delay가 생김
- 문제를 해결하기 위해 Look Ahead Carry 방식을 사용

서강대학교 SOGANG UNIVERSITY

Index

Look Ahead Carry

$$g_i = x_i y_i, \qquad p_i = x_i \oplus y_i$$

$$c_{i+1} = g_i + p_i c_i$$

$$c_{i+2} = g_{i+1} + p_{i+1} g_i + p_{i+1} p_i c_i$$

$$c_{i+3} = g_{i+2} + p_{i+2} g_{i+1} + p_{i+2} p_{i+1} g_i + p_{i+2} p_{i+1} p_i c_i$$

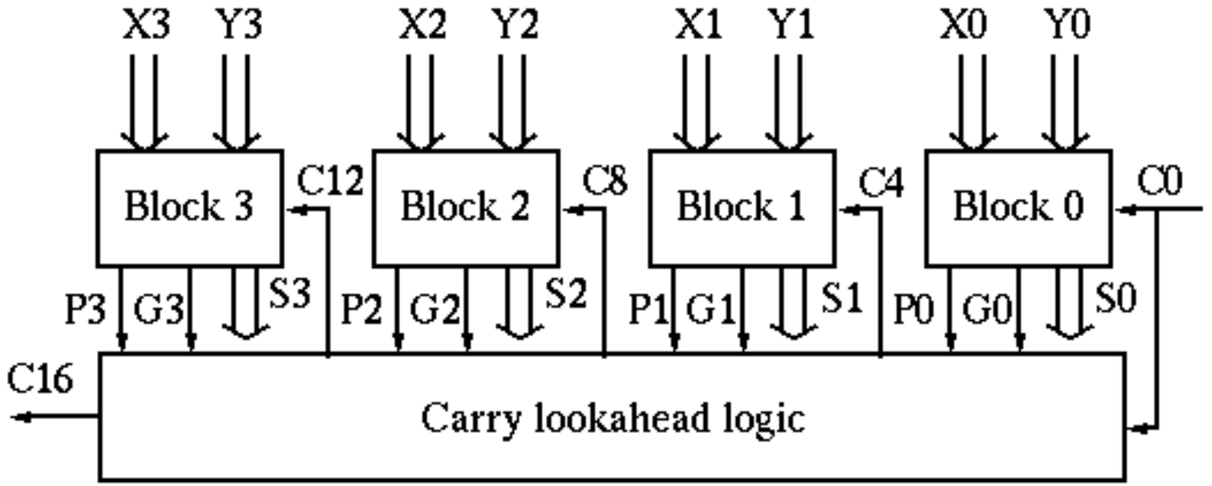
$$c_{i+4} = g_{i+3} + p_{i+3} g_{i+2} + p_{i+3} p_{i+2} g_{i+1} + p_{i+3} p_{i+2} p_{i+1} g_i + p_{i+3} p_{i+2} p_{i+1} p_i c_i$$

- 앞 자리 bit의 연산을 통해 Carry값을 구하는 것이 아닌,
  최초의 Carry input을 이용해 Carry 값를 계산하는 방식

Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- **n-bit adder(심화)**
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- BCD code - Subtractor (심화)

n - bit adder

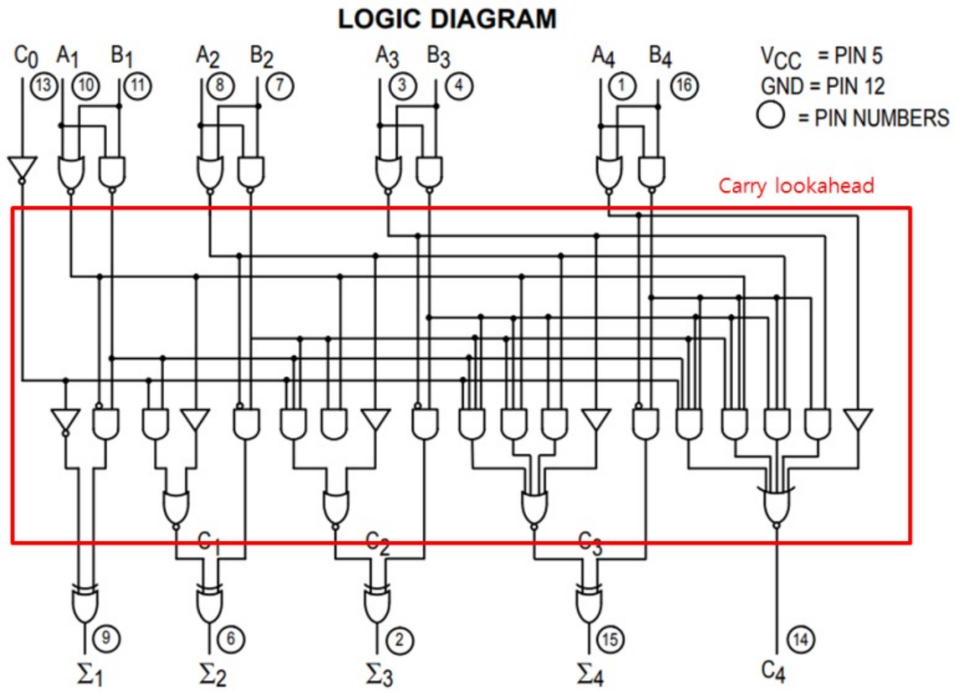


4-bit Adder 이용해 구현!!!

Index

- 4-bit adder, subtractor의 개요
- **구현 방식 –**
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- BCD code - Subtractor (심화)

Look Ahead Carry



Index

복습: 2진수의 보수를 이용한 뺄셈

0100(4) – 0111(7)↵

0111 → 1000 (1의 보수)↵

1000 + 1 = 1001↵

0111 → 1001 (2의 보수)↵

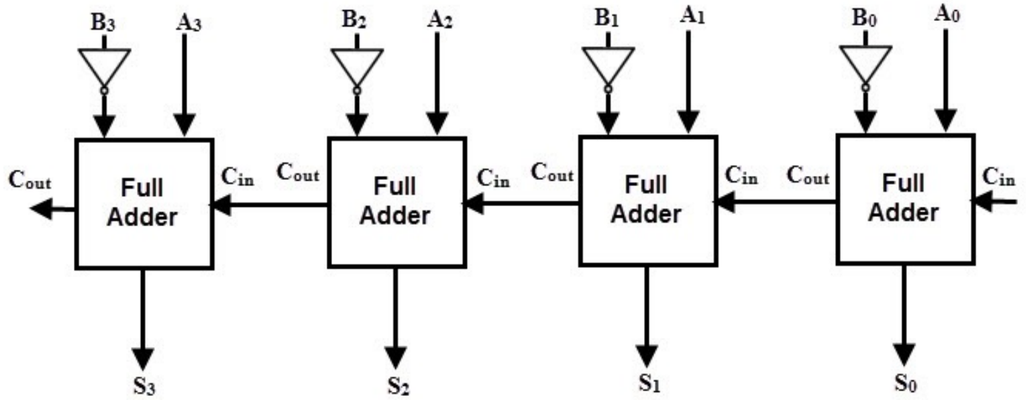
0100 + 1001 = 1101↵

1(부호가 음수) 101(carry가 없음으로 2의 보수를 취해야 함)↵

0100 – 0111 = -011(-3)↵

Index

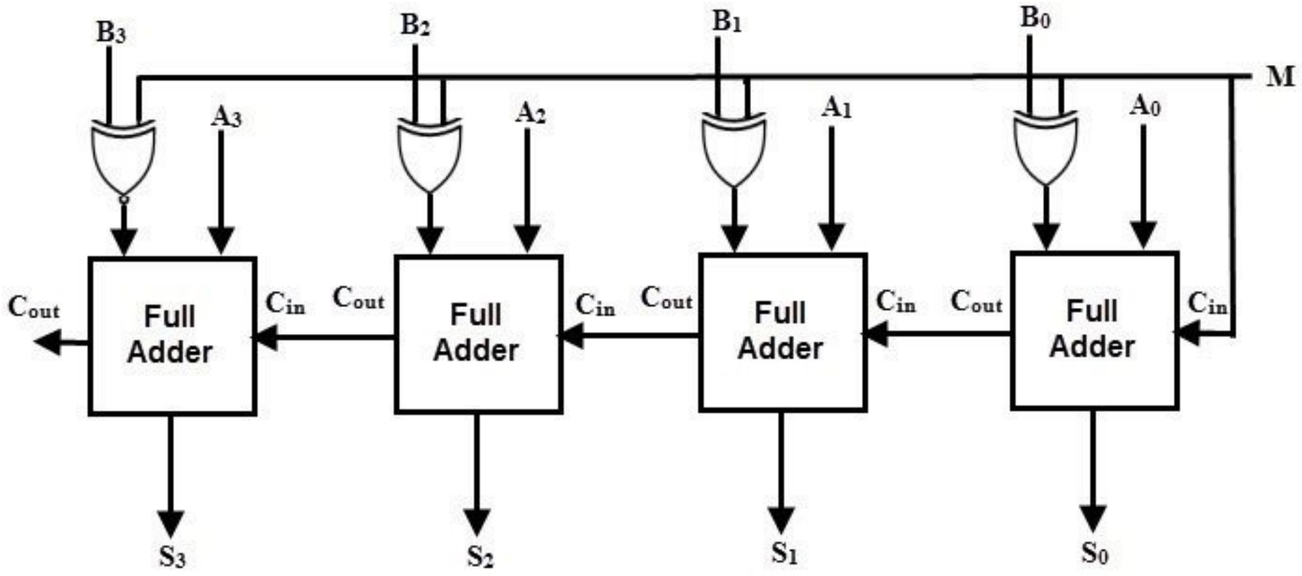
2's complement 방식을 이용하여 구현한 4-bit subtractor



- B의 bit열에 not을 취해 주어 1의 보수를 취해준다.
- c_{in}의 값에 1을 입력하여 2의 보수를 취해준다.

Index

4-bit 가감산기

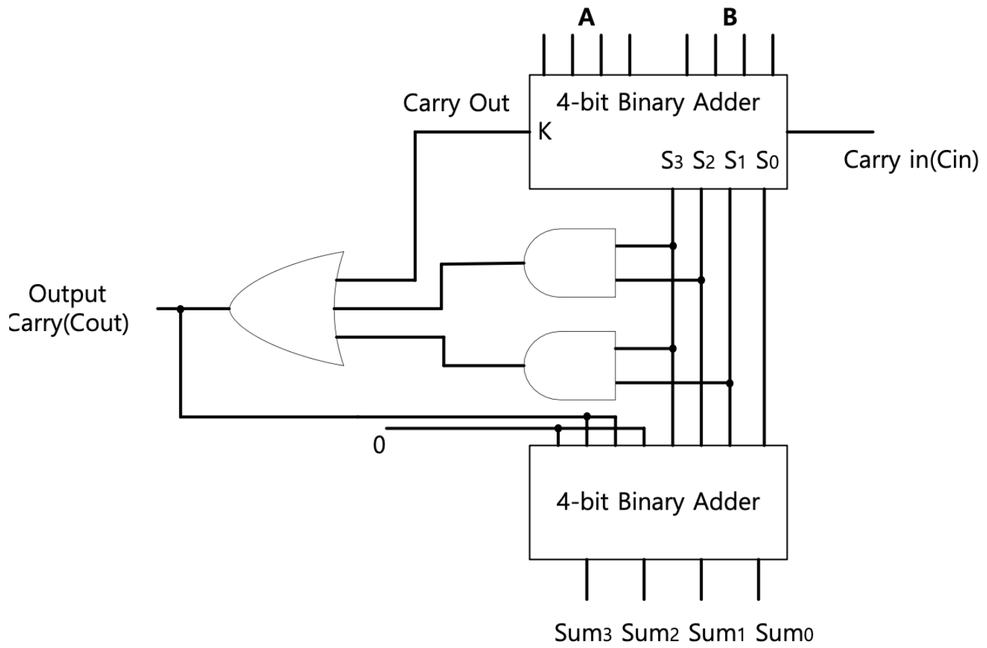


- M이라는 input값을 통해 가산인지 감산인지를 선택
- 계산되는 bit 열에 2의 보수를 취하기 위해 XOR과 C_0의 값을 이용

Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- **BCD code의 성질**
- BCD code - Adder
- BCD code - Subtractor (심화)

BCD Adder (Binary-Coded Decimal)



서강대학교 SOGANG UNIVERSITY

Index

BCD code

10진수(0~9)를 이진수(0,1)로 어떻게 표현할까?
0~9까지의 Decimal을 -> 4비트의 Binary로

Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- **BCD code의 성질**
- BCD code - Adder
- BCD code - Subtractor (심화)

Binary code VS BCD code

8	0000 1000	0000 1000
9	0000 1001	0000 1001
10	0000 1010	0001 0000
11	0000 1011	0001 0001
12	0000 1100	0001 0010
13	0000 1101	0001 0011

Index

129

Binary Code : 1000 0001

BCD Code : 0001 0010 1001

서강대학교 SOGANG UNIVERSITY

Index

BCD 코드의 덧셈

8 + 4 = 12

1000 + 0100 = 1100 (x)

= 0001 0010 (o)

= 1100에 **0110**을 더해주기 + Carry 발생!

서강대학교 SOGANG UNIVERSITY

Index

- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder

- n-bit adder(심화)

- 구현 방식 – 2's complement

- BCD code의 성질

- **BCD code - Adder**

- BCD code - Subtractor (심화)

BCD 코드의 덧셈

1. 그냥 이진수 덧셈
2. 0000~1001 범위 넘어가는가??
3. 넘어가면 Carry 발생시키고, 0110 더하기

서강대학교 SOGANG UNIVERSITY

Index

BCD 코드의 덧셈

35 + 49

0011 0101 + 0100 1001

일의 자리

1. 그냥 이진수 덧셈 : 0101 + 1001 = 1110

2. 1001보다 큰가? : 크다 (1110 > 1001)

3. 0110 더해주고 Carry 발생 : 1110 + **0110** = **0100** + Carry

십의 자리

1. 그냥 이진수 덧셈 : 0011 + 0100 + Carry = 1000

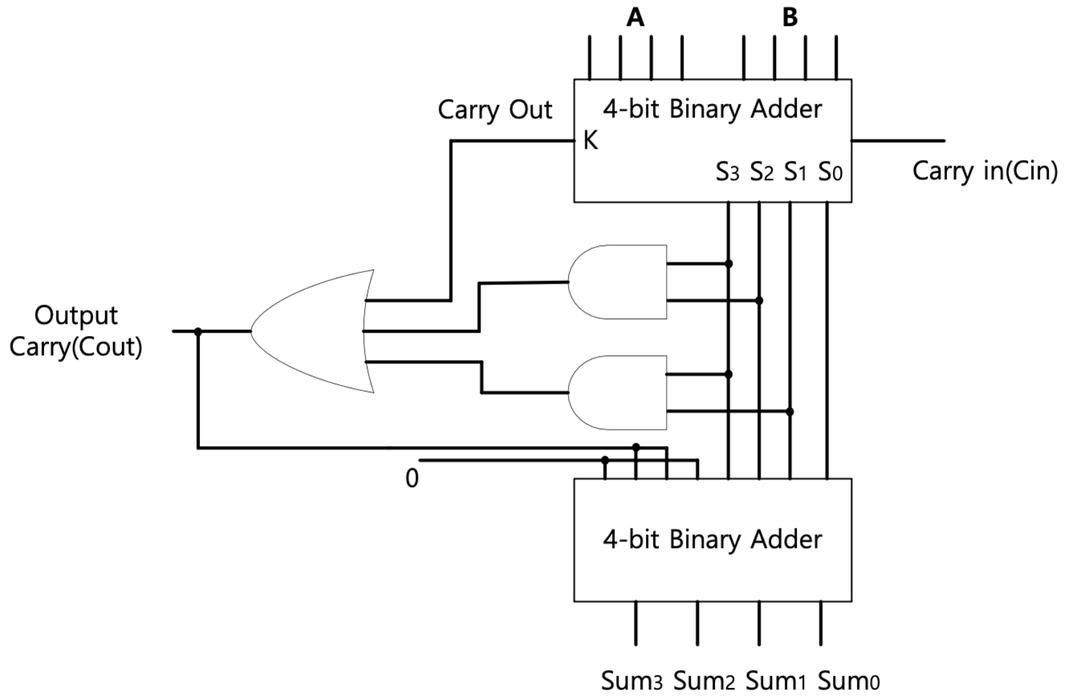
2. 1001보다 큰가? : 작다 (1000 < 1001)

→ **1000 0100** (84)

Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2’s complement
- BCD code의 성질
- **BCD code - Adder**
- BCD code - Subtractor (심화)

BCD Adder 회로 분석



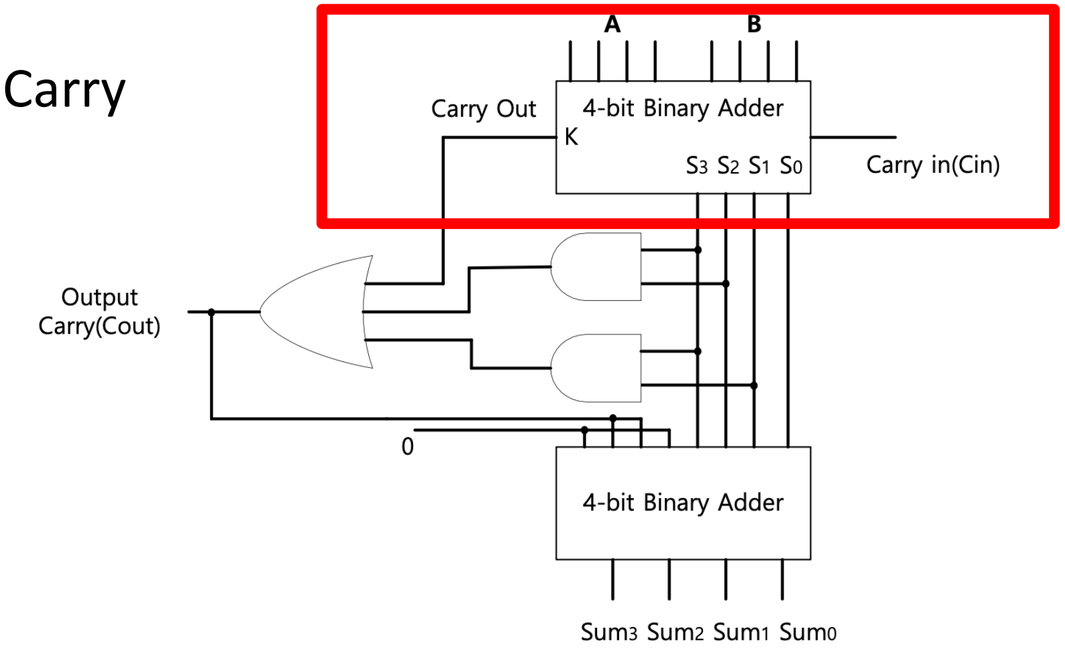
BCD Adder

Index

- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- **BCD code - Adder**
- BCD code - Subtractor (심화)

1. 그냥 이진수 덧셈

A와 B의 2진수 덧셈 + Carry



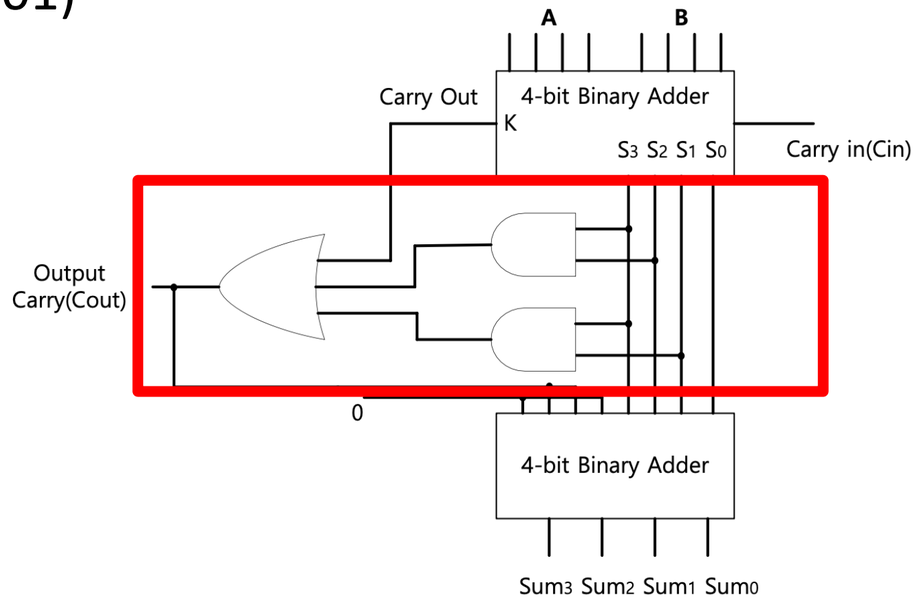
Index

- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- **BCD code - Adder**
- BCD code - Subtractor (심화)

2. 1001보다 큰가?

BCD 코드 범위 (0000 ~ 1001)

1010
1011
1100
1101
1110
1111



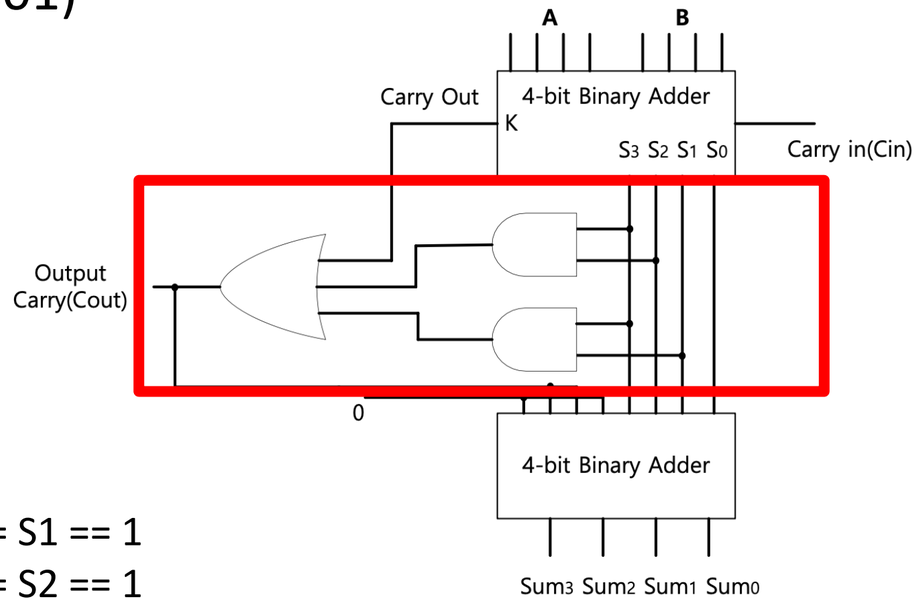
Index

- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- **BCD code - Adder**
- BCD code - Subtractor (심화)

2. 1001보다 큰가?

BCD 코드 범위 (0000 ~ 1001)

1010
1011
1100
1101
1110
1111

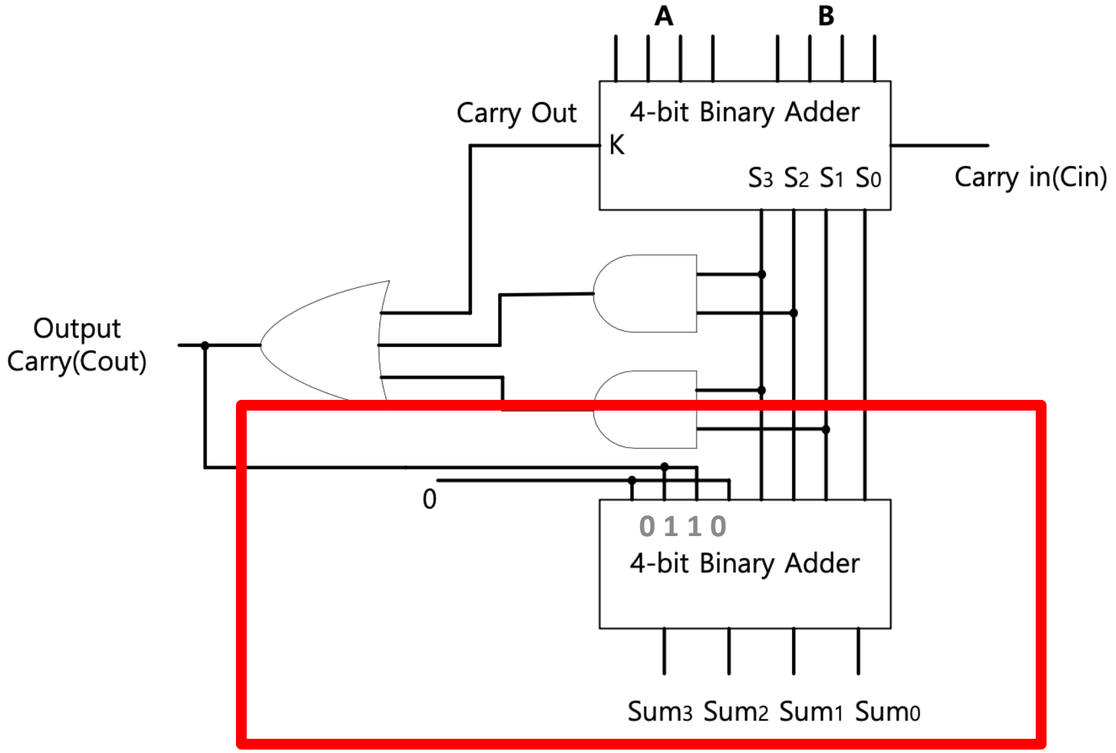


1. S3 == S1 == 1
2. S3 == S2 == 1
3. A + B > 1111

Index

- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- **BCD code - Adder**
- BCD code - Subtractor (심화)

3. 크다면 0110 더하고 Carry 발생



Index

심화 - BCD Subtraction

원리

A – B

A + B의 9’s complement + 1

Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

BCD Subtraction

예시

1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호

	Regular Subtraction	9's Complement Subtraction	
(a)	8	8	
	− 2	+ 7	9's complement of 2
	6	① 5	
		+ 1	Add carry to result
		6	
(b)	28	28	
	− 13	+ 86	9' complement of 13
	15	① 14	
		+ 1	Add carry to the result
		15	
(c)	18	18	
	− 24	+ 75	9' complement of 24
	−6	93	9's complement of result (No carry indicates that the answer is negative and in complement form)
		−06	

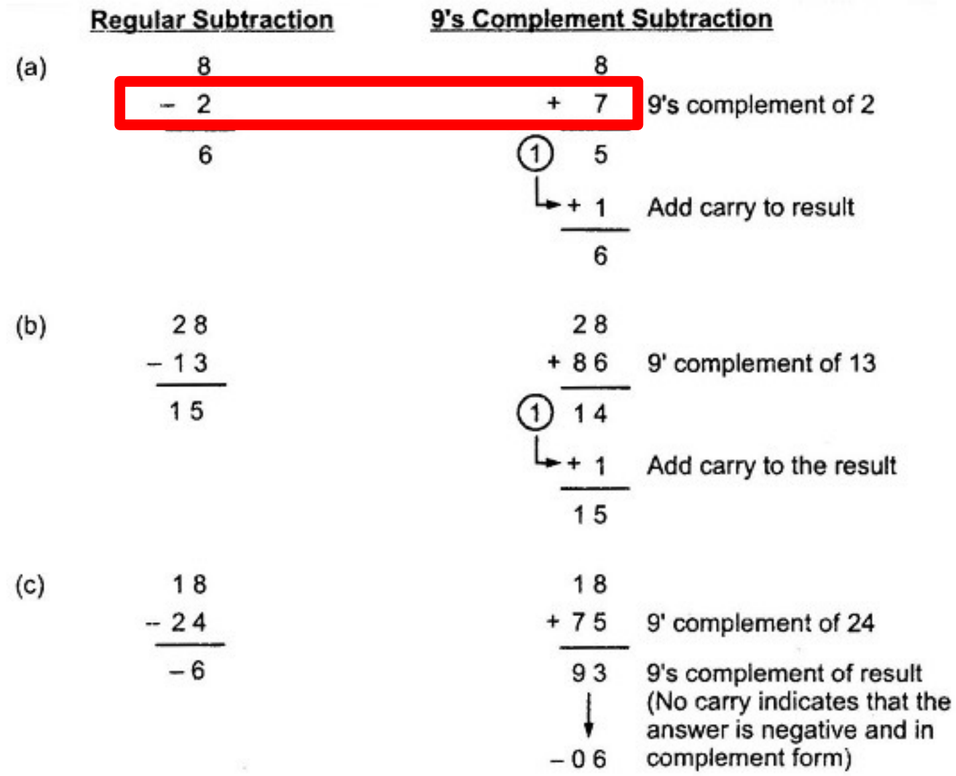
Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

BCD Subtraction

예시

1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호



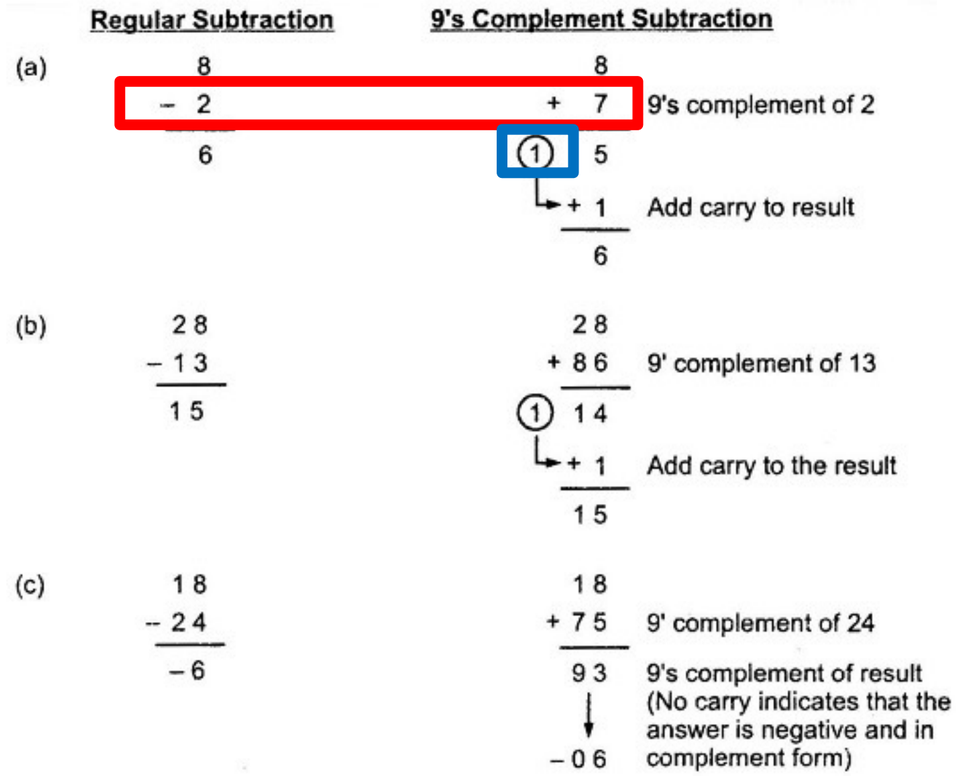
Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

BCD Subtraction

예시

1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호



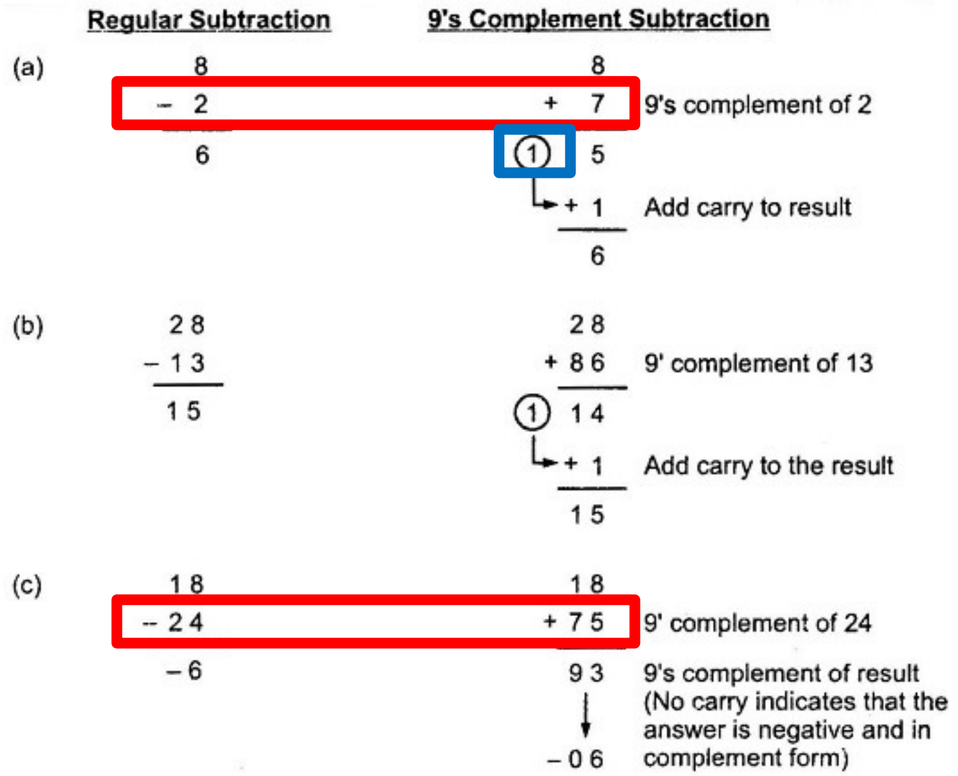
Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

BCD Subtraction

예시

1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호



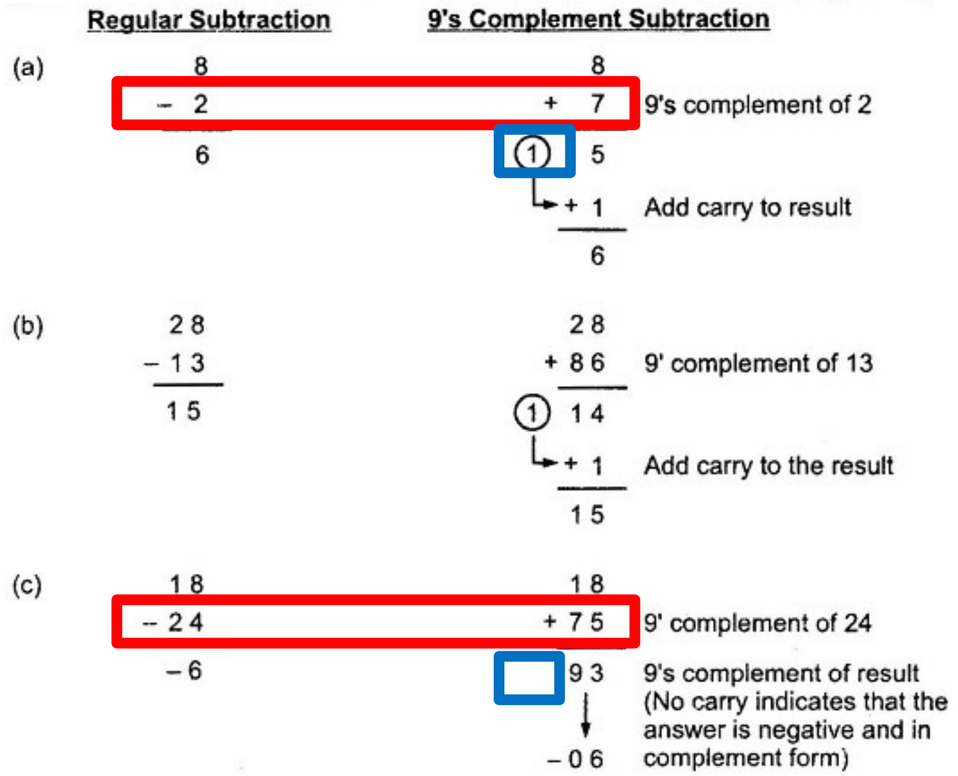
Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

BCD Subtraction

예시

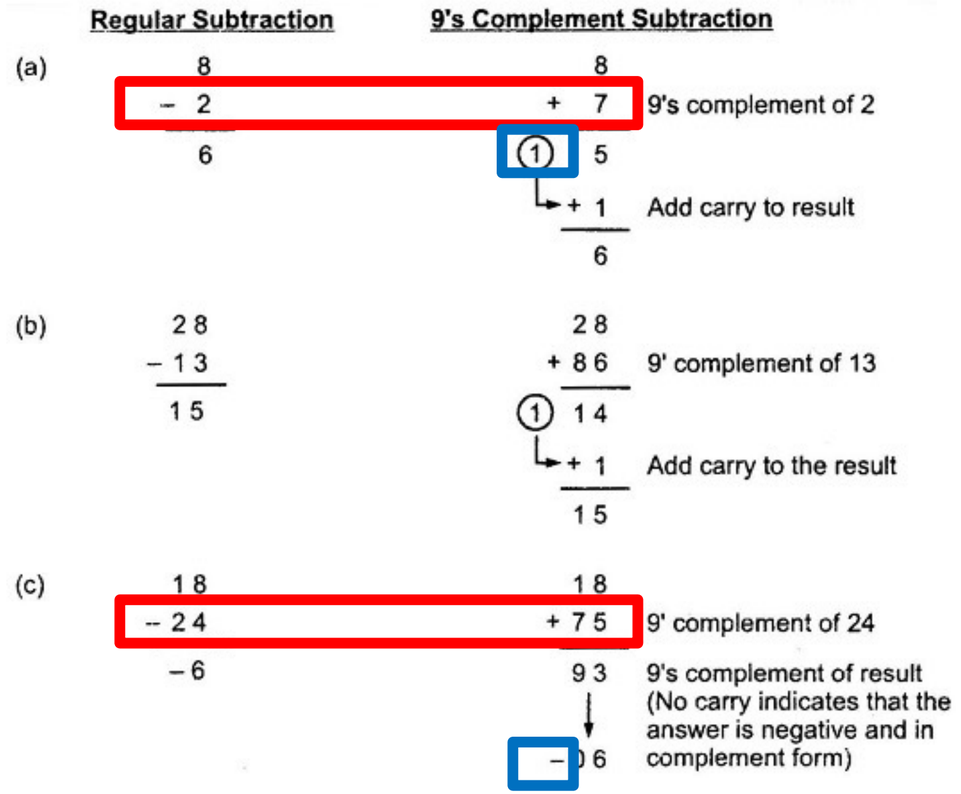
1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호



Index

BCD Subtraction

1. 빼는 수 -> 9's complement로 바꿔주기
2. BCD Adding
3. Carry 있다면 – 1 더해주기
   없다면 – 9's complement and 부호 음수



Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

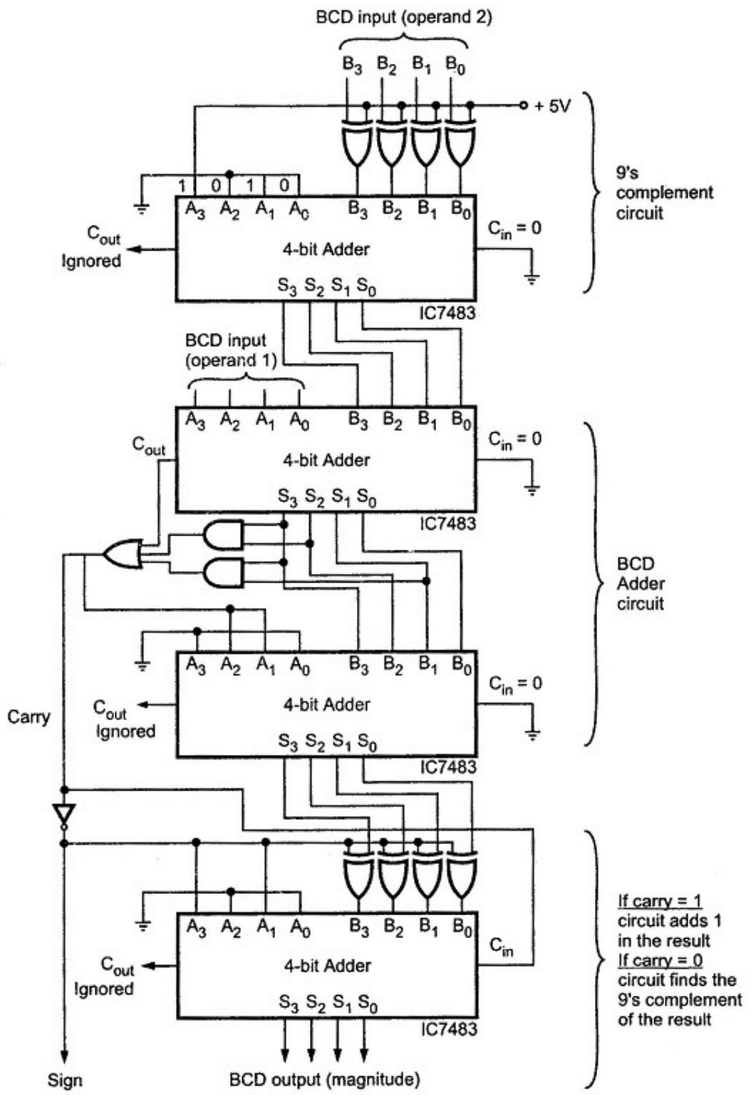


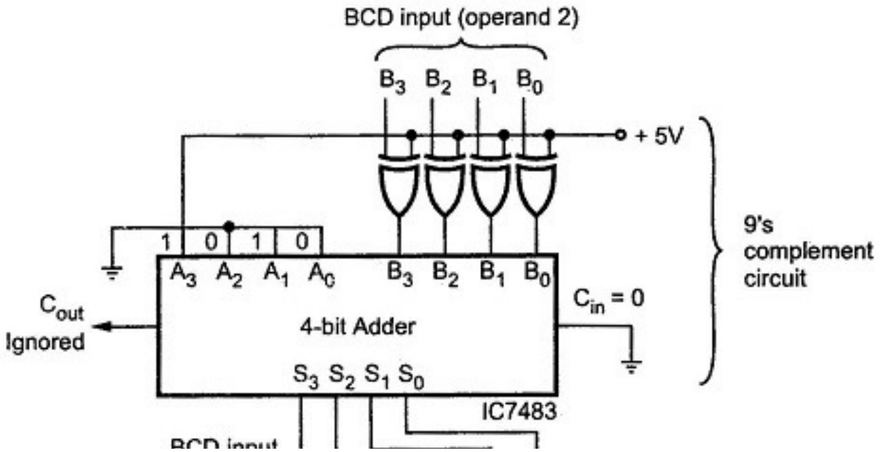
Fig. 3.34 4-bit BCD subtractor using 9's complement method

Index

BCD Subtractor : A-B 구하기

1. B의 9’s complement 구하기

B의 각 비트 1과 XOR + 1010

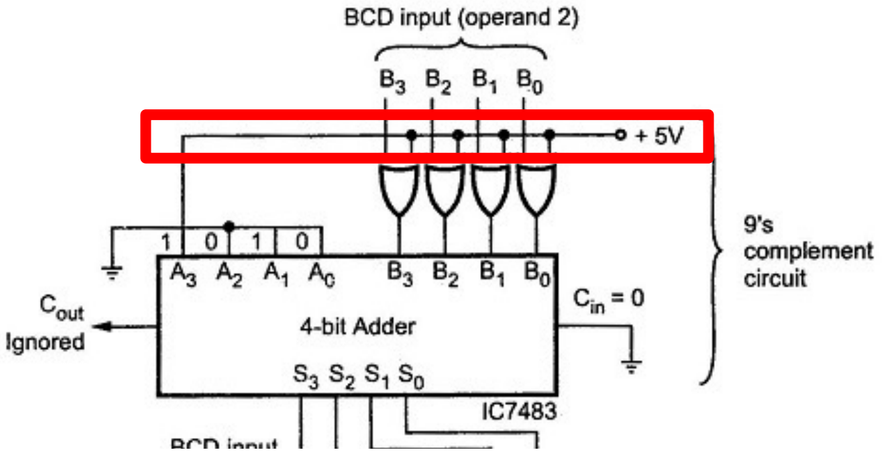


Index

BCD Subtractor : A-B 구하기

1. B의 9's complement 구하기

B의 각 비트 1과 XOR + 1010

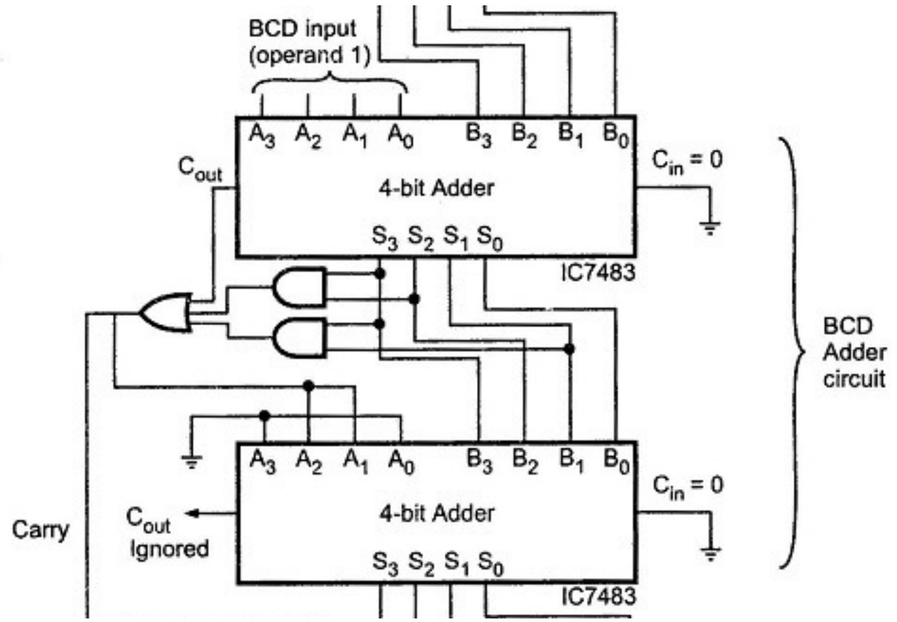


Index

BCD Subtractor : A-B 구하기

2. BCD Adding

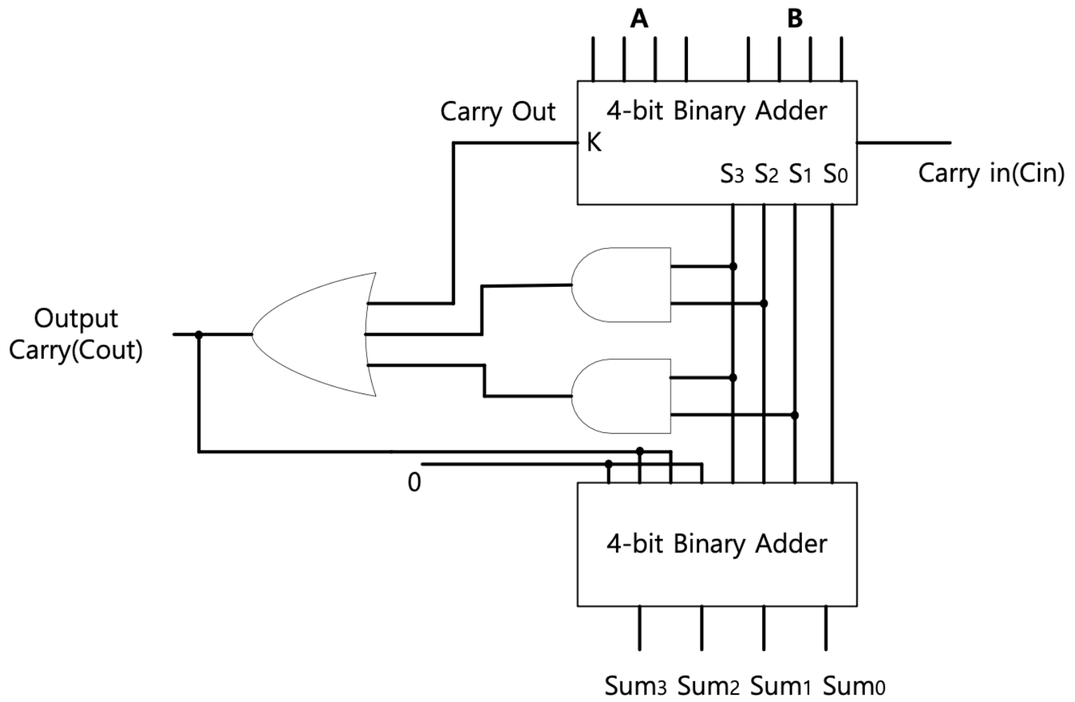
A + B의 9's complement



Index

- 4-bit adder, subtractor의 개요
- 구현 방식 –
  Look Ahead Carry와
  Ripple Carry Adder
- n-bit adder(심화)
- 구현 방식 – 2's complement
- BCD code의 성질
- BCD code - Adder
- **BCD code - Subtractor (심화)**

똑같다!



BCD Adder

Index

BCD Subtractor : A-B 구하기

3. Carry에 따라 다르게 처리

If (Carry == 1)
-> 0000 + S + Carry
-> **S + 0001**

Else (carry ==0)
-> S의 각 비트 XOR + 1010
= **S의 9's complement & Sign**

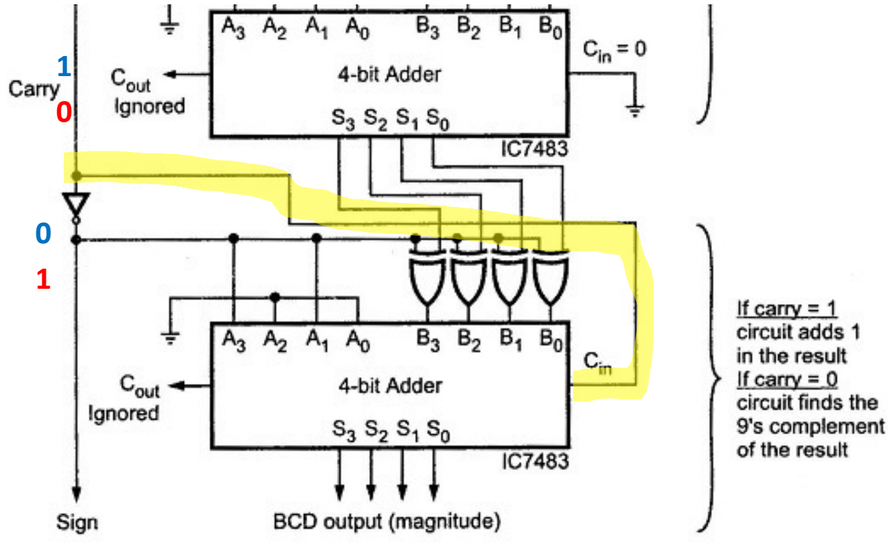


Fig. 3.34 4-bit BCD subtractor using 9's complement method

서강대학교 SOGANG UNIVERSITY

Index

<참고자료>

- 디지털 논리회로(4판), 한빛아카데미, 임석구, 홍경호(2015)
- 서강대학교 10주차 강의자료

- 4-bit Adder/subtractor 구조 – https://www.geeksforgeeks.org/4-bit-binary-adder-subtractor/
- Binary Code VS BCD Code 표 - https://jhnyang.tistory.com/232
- BCD Subtraction Circuit - https://www.eeeguide.com/bcd-subtraction/

<기여도>

서동휘 : 50%
정영훈 : 50%