제 10장 Part-1

덧셈과 뺄셈 (Addition and Substraction)

- **기본 컴퓨터의 데이터 형식**
  - 부호절대값으로 표시된 고정소수점 이진 데이터 (34: 0 0100010)
  - 부호화된 2의 보수로 표현되는 고정소수점 이진 데이터 (-34: 1 1011110)
  - 부동소수점 이진 데이터 (0.34*10^2: 0 0100010 00000010)
  - 이진화된 십진수(BCD) 데이터 (1264: 0001 0010 0110 1000)
- **부호절대값 데이터를 이용한 덧셈과 뺄셈**
  - TABLE 10-1 Addition and Subtraction of Signed-Magnitude Numbers

Operation	Add Magnitudes	Subtract Magnitudes		
		When $A > B$	When $A < B$	When $A = B$
$(+A) + (+B)$	$+(A + B)$			
$(+A) + (-B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(-A) + (+B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$
$(-A) + (-B)$	$-(A + B)$			
$(+A) - (+B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(+A) - (-B)$	$+(A + B)$			
$(-A) - (+B)$	$-(A + B)$			
$(-A) - (-B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$

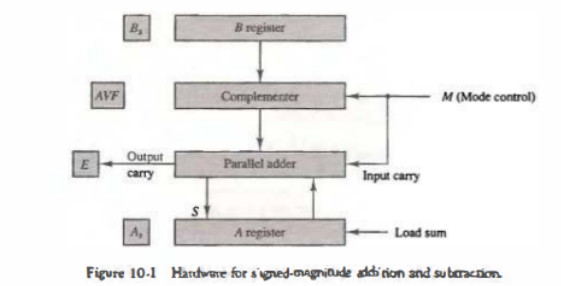
  - 

Figure 10-1 Hardware for signed-magnitude addition and subtraction.
  - **하드웨어 구성**
    - 덧셈 알고리즘
    - 뺄셈 알고리즘

    - 부호의 비교
        - XOR에 의해 비교
            - 0 : 동일 부호
            - 1 : 다른 부호
        - 오버플로우의 처리 필요
            - 같은 부호 연산 : YES
            - 다른 부호 연산 : NO
    - 절대값의 비교
        - A < B, A > B 인 경우의 처리
    -

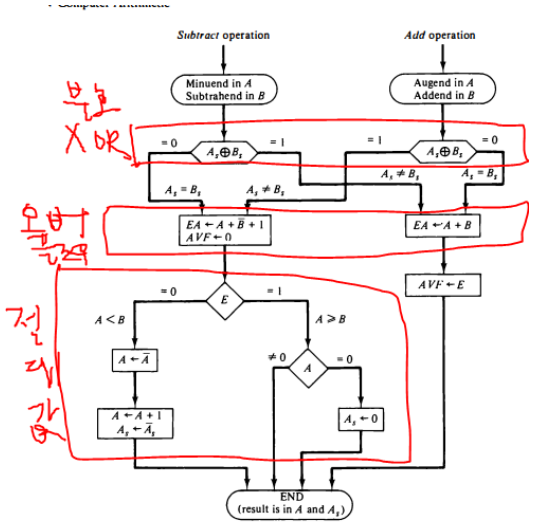
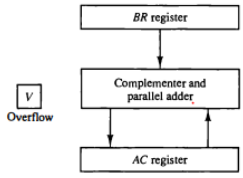


Figure 10-2 Flowchart for add and subtract operations.

- 부호가 있는 2의 보수 데이터를 이용한 덧셈과 뺄셈
    - 하드웨어 구조
        - A -> AC reg, B -> BR reg.
    -

Figure 10-3 Hardware for signed-2's complement addition and subtraction.



    -

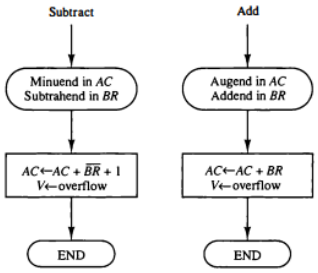


Figure 10-4 Algorithm for adding and subtracting numbers in signed-2's complement representation.

- 덧셈/뺄셈 알고리즘
  - 오버플로우의 조사 V <- Overflow

제 10장 Part-2

- **설명을 잘하는 인도유튜버**
  https://www.youtube.com/watch?v=B2bKdGf1Qoc&list=PLgBlB7BVKQmg__8dTZOkHBSj4WDRgZ4vX&index=17

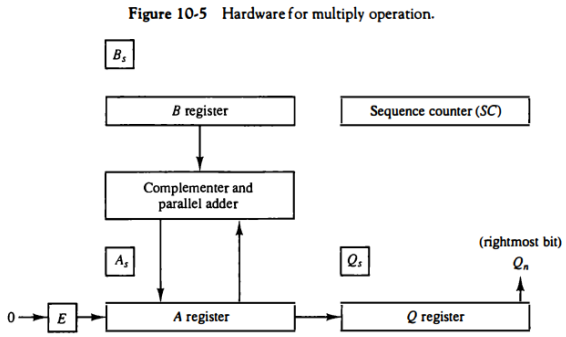
곱셈 알고리즘 (Multiplication Algorithm)

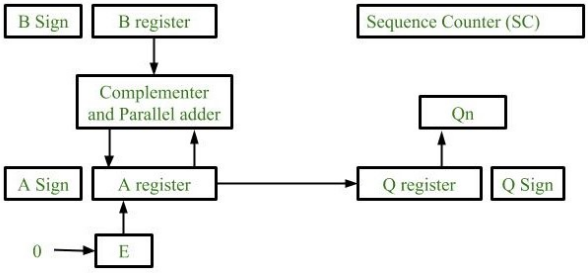
- 곱셈의 원리
  - 연속적인 시프트와 덧셈으로 구성
  -

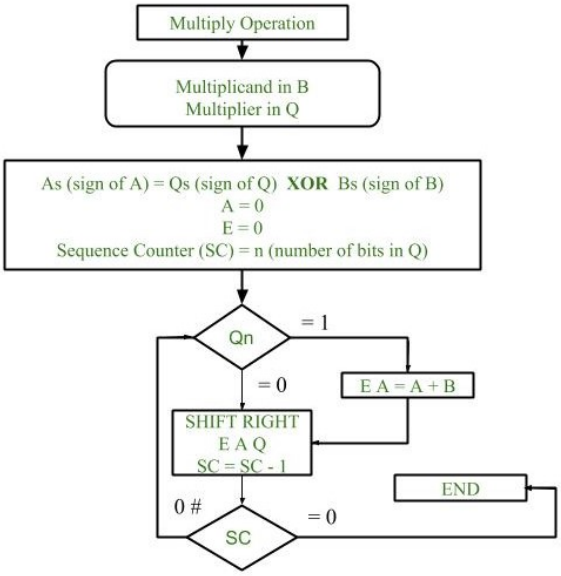
```
23      10111    Multiplicand
19    × 10011    Multiplier
        10111
       10111
      00000   +
     00000
    10111
437 110110101    Product
```

- 하드웨어 구성

- A -> AC (계산값) B -> 피승수
- Q -> 승수
- SC -> 시퀀스 카운터(5번 쉬프트하면서 덧셈해야 하니까)
- A_s, B_s, Q_s -> 부호 비트
- EAQ -> 결과값

- 

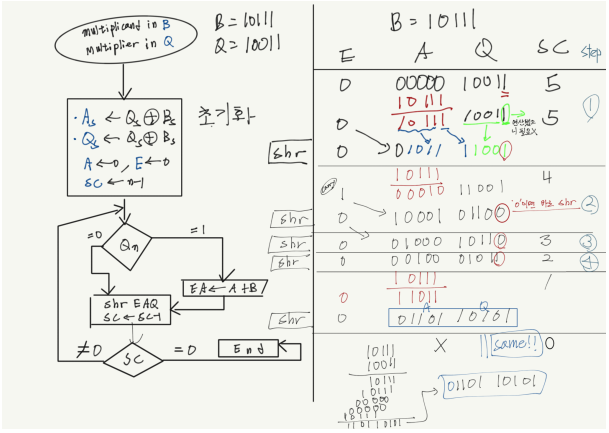
- 

- 

-

Multiplicand B = 10111	E	A	Q	SC
Multiplier in Q Qn = 1; add B First partial product Shift right EAQ	0 0 0	00000 10111 10111 01011	10011 11001	101 100
Qn = 1; add B Second partial product Shift right EAQ	1 0	10111 00010 10001	 01100	 011
Qn = 0; shift right EAQ	0	01000	10110	010
Qn = 0; shift right EAQ	0	00100	01011	001
Qn = 1; add B Fifth partial product Shift right EAQ	 0 0	10111 11011 01101	 10101	 000

Final product in AQ
0110110101

- 

- **Booth의 곱셈 알고리즘**

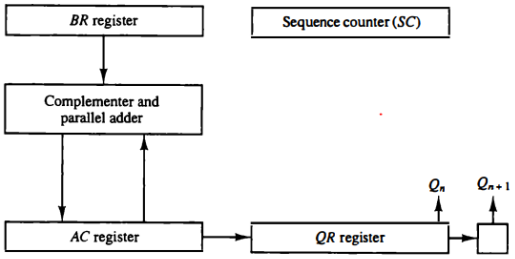
  - 부호가 있는 2의 보수로 표현된 정수에 대한 곱셈 수행
  - 승수값이 0인 경우 -> 시프트만 수행
  - 2^k ~ 2^M 까지의 값이 1인 경우 -> 2^(k+1) - 2^m 과 동등하게 취급
    - (14: 001110) =
    - $2^{k+1} - 2^m = 2^4 - 2^1 = 14$
    - $M \times 14 = M \times 2^4 - M \times 2^1$
      = 피승수 M을 왼쪽으로 4번 이동한 값 - 1번 이동한 값

- **하드웨어 구성**

  - Q_(n+1) : 승수의 두 비트 비교

- 승수 비트가 1인 경우
- 승수 비트가 0인 경우

- Figure 10-7 Hardware for Booth algorithm.



- Booth의 곱셈 알고리즘 수행 예

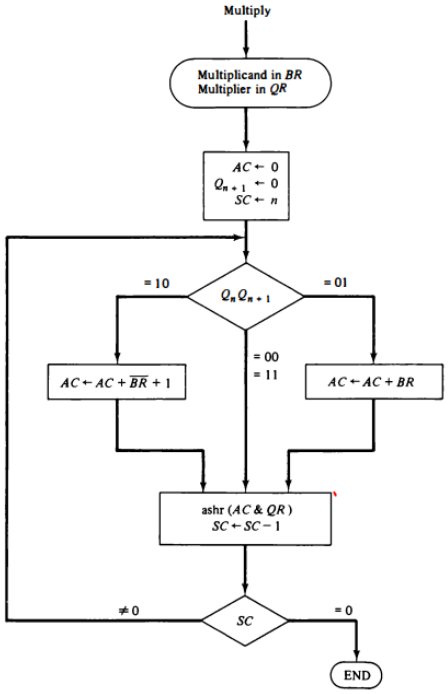
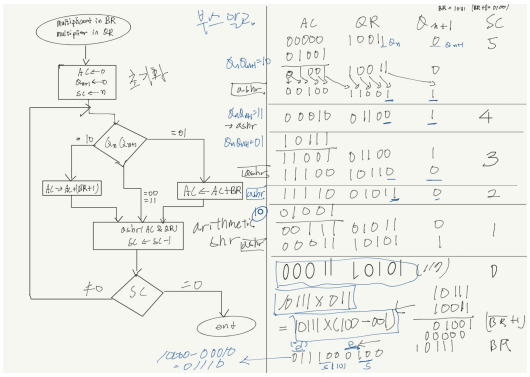


Figure 10-8 Booth algorithm for multiplication of signed-2's complement numbers.

TABLE 10-3 Example of Multiplication with Booth Algorithm

$Q_n Q_{n+1}$	$BR = 10111$ $\overline{BR} + 1 = 01001$	AC	QR	Q_{n+1}	SC
	Initial	00000	10011	0	101
1 0	Subtract BR	01001			
		01001			
	ashr	00100	11001	1	100
1 1	ashr	00010	01100	1	011
0 1	Add BR	10111			
		11001			
	ashr	11100	10110	0	010
0 0	ashr	11110	01011	0	001
1 0	Subtract BR	01001			
		00111			
	ashr	00011	10101	1	000



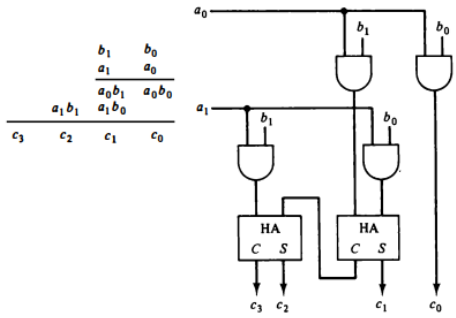
- 배열 승산기
  - 조합회로에 의한 논리곱 마이크로연산 수행
  - 2 * 2 배열 승산기
  - 

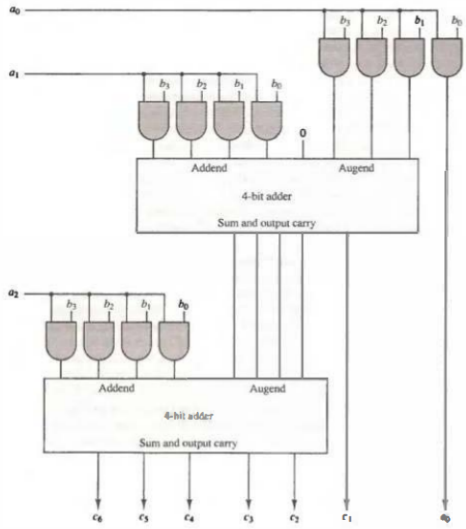
Figure 10-9 2-bit by 2-bit array multiplier.
  - 4 * 3 배열 승산기
    - 

Figure 10-10 4-bit by 3-bit array multiplier.

제 10장 Part-3

나눗셈 알고리즘 (Division Algorithm)

- 이진 나눗셈
  - 하드웨어는 곱셈과 동일
  - 나눗셈 오버플로우의 처리
- 나눗셈의 처리

  -

Figure 10-11 Example of binary division.

Divisor:		
B = 10001	11010	Quotient = Q
	)0111000000	Dividend = A
	01110	5 bits of $A < B$, quotient has 5 bits
	011100	6 bits of $A \geq B$
	-10001	Shift right B and subtract; enter 1 in Q
	-010110	7 bits of remainder $\geq B$
	--10001	Shift right B and subtract; enter 1 in Q
	--001010	Remainder $< B$; enter 0 in Q; shift right B
	---010100	Remainder $\geq B$
	----10001	Shift right B and subtract; enter 1 in Q
	----000110	Remainder $< B$; enter 0 in Q
	-----00110	Final remainder

  -

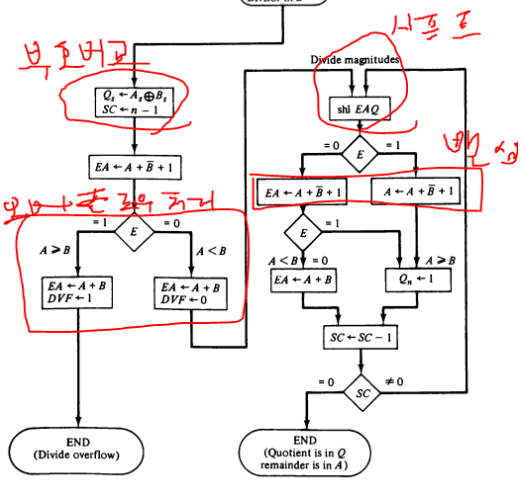
Divisor B = 10001, $\overline{B} + 1$ = 01111

	E	A	Q	SC
Dividend:		01110	00000	5
shl EAQ	0	11100	00000	
add $\overline{B} + 1$		01111		
E = 1	1	01011		
Set Q_n = 1	1	01011	00001	4
shl EAQ	0	10110	00010	
Add $\overline{B} + 1$		01111		
E = 1	1	00101		
Set Q_n = 1	1	00101	00011	3
shl EAQ	0	01010	00110	
Add $\overline{B} + 1$		01111		
E = 0; leave Q_n = 0	0	11001	00110	
Add B		10001		2
Restore remainder	1	01010		
shl EAQ	0	10100	01100	
Add $\overline{B} + 1$		01111		
E = 1	1	00011		
Set Q_n = 1	1	00011	01101	1
shl EAQ	0	00110	11010	
Add $\overline{B} + 1$		01111		
E = 0; leave Q_n = 0	0	10101	11010	
Add B		10001		
Restore remainder	1	00110	11010	0
Neglect E				
Remainder in A:		00110		
Quotient in Q:			11010	

Figure 10-12 Example of binary division with digital hardware.

  - 하드웨어 알고리즘
    - ○

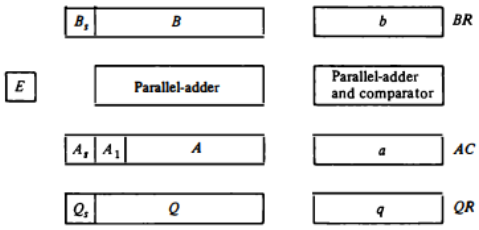
Figure 10-13 Flowchart for divide operation.



부동 소수점 산술 연산 (Floating-Point Arthmetic Operations)

- 숫자의 표현
  - M x r^n / r: Radix
- 레지스터 구성
  -

Figure 10-14 Registers for floating-point arithmetic operations.



- 덧셈/뺄셈 알고리즘
  - 1. 0인지의 여부 조사
  - 2. 가수의 위치 조정
  - 3. 가수의 덧셈/뺄셈
  - 4. 결과의 정규화
-

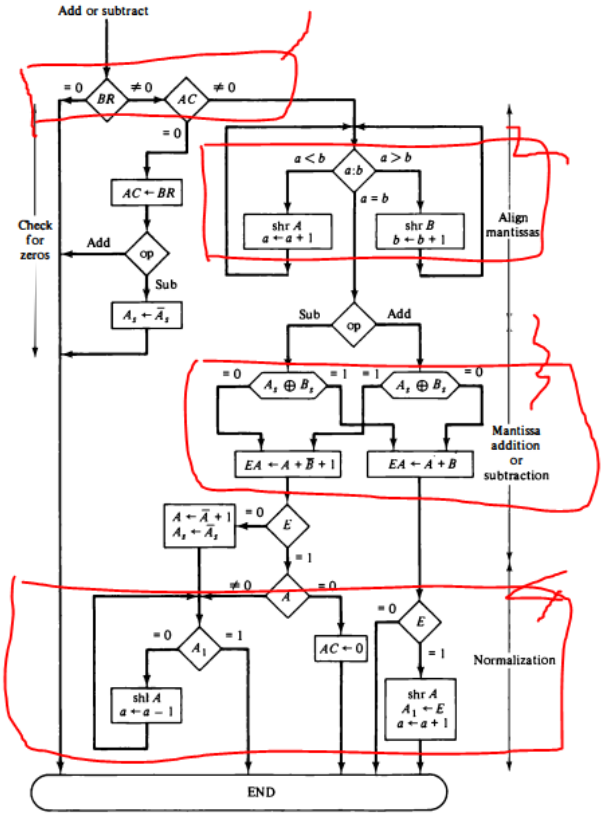
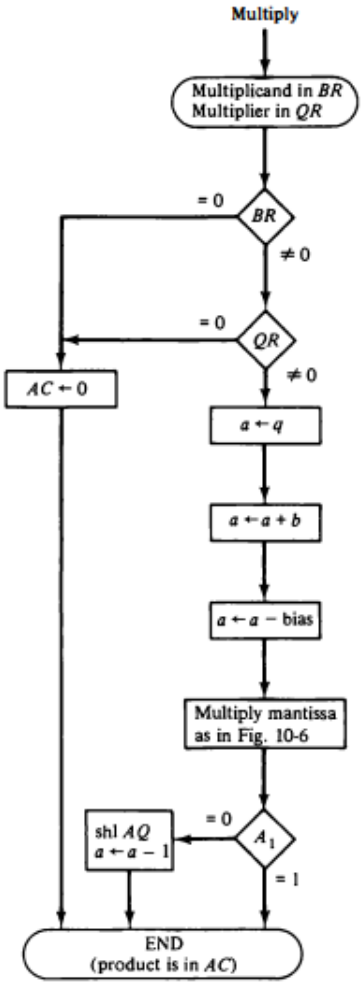


Figure 10-15 Addition and subtraction of floating-point numbers.

- 곱셈 알고리즘
  - 0의 확인
  - 지수의 덧셈
  - 가수의 곱셈
  - 결과의 정규화
  -

Figure 10-16 Multiplication of floating-point numbers.



- 나눗셈 알고리즘
  - 0 의 확인
  - 레지스터 초기화
  - 부호의 결정
  - 피젯수 위치 조정
  - 지수의 뺄셈
  - 가수의 나눗셈
  -

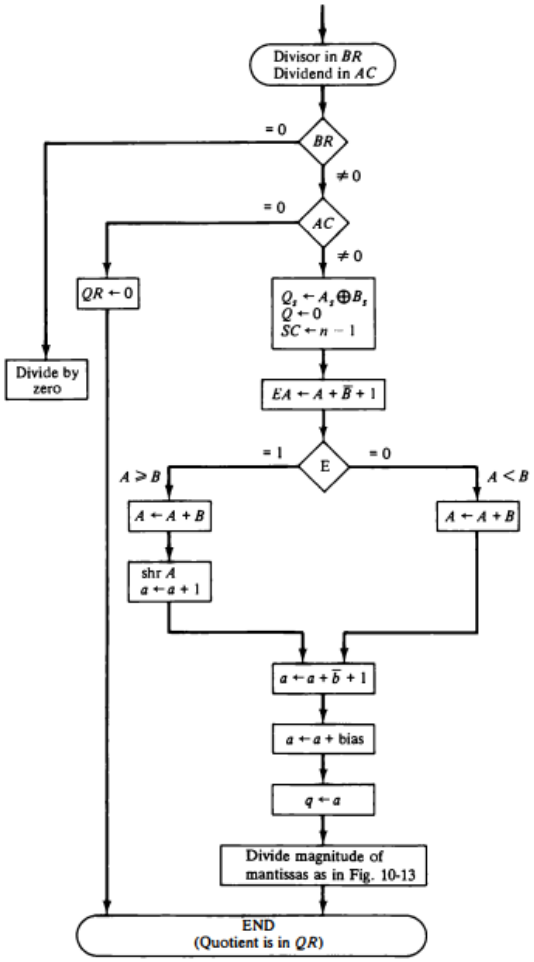


Figure 10-17 Division of floating-point numbers.

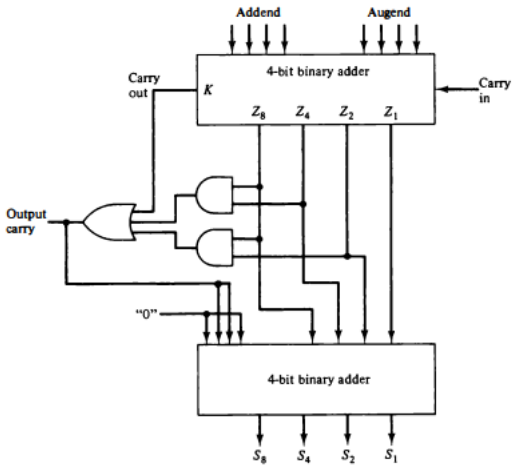
십진 산술 장치 (Decimal Arithmetic Unit)

- BCD 가산기

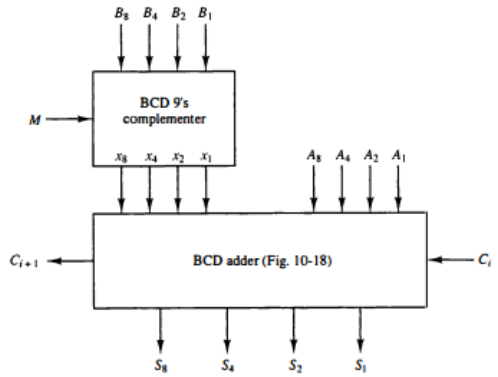
TABLE 10-4 Derivation of BCD Adder

Binary Sum					BCD Sum					
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	Decimal
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

Figure 10-18 Block diagram of BCD adder.



- BCD 감산기
  - M 비트에 의한 연산 전환
  - BCD 9의 보수 회로 구현
  - $$x_1 = B_1 M' + B_1' M$$
    $$x_2 = B_2$$
    $$x_4 = B_4 M' + (B_4' B_2 + B_4 B_2')M$$
    $$x_8 = B_8 M' + B_8' B_4' B_2' M$$
  - Figure 10-19 One stage of a decimal arithmetic unit.



십진 산술 연산 (Decimal Arthmetic Operations)

- 덧셈과 뺄셈
  - 십진 산술 마이크로 연산 기호

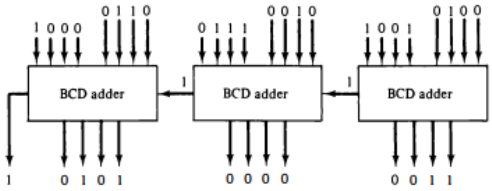
-

TABLE 10-5 Decimal Arithmetic Microoperation Symbols

Symbolic Designation	Description
$A \leftarrow A + B$	Add decimal numbers and transfer sum into A
$\overline{B}$	9's complement of B
$A \leftarrow A + \overline{B} + 1$	Content of A plus 10's complement of B into A
$Q_L \leftarrow Q_L + 1$	Increment BCD number in Q_L
dshr A	Decimal shift-right register A
dshl A	Decimal shift-left register A

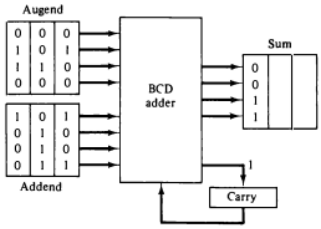
- 3가지 십진 연산 장치

-

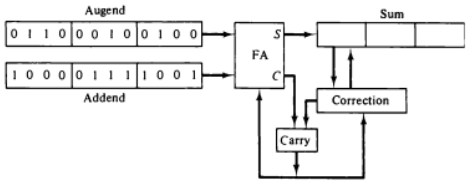


(a) Parallel decimal addition: 624 + 879 = 1503

-



(b) Digit-serial, bit-parallel decimal addition



(c) All serial decimal addition

Figure 10-20 Three ways of adding decimal numbers.