บทที่ 2
Number Systems and Arithmetic
ระบบจำนวนและการคำนวณทางคณิตศาสตร์

เค้าโครง

- Binary Number Systems
- Hexadecimal Code
- Binary Codes
- Binary Arithmetic
- Arithmetic Using BCD and Binary Codes

2.1 ระบบเลขฐานสอง (Bianry Number System)

ระบบจำนวนที่ใช้แทนค่าระดับแรงดันของระบบดิจิตอล ระบบเลขฐานสองและเลขฐานสิบ เป็นระบบตัวเลขที่จะนำมาศึกษาในการนับของสัญญาณดิจิตอล ใช้แทนรูปคลื่นของสัญญาณดิจิตอล และการเปลี่ยนแปลงเลขฐานสองกับเลขฐานสิบ

รูปคลื่นของสัญญาณดิจิตอลเมื่อเทียบกับเลขฐานสอง คือเป็นสัญญาณดรีสครีต (Discrete) ระดับแรงดันที่อ้างถึงในการแทนค่าของเลขฐานสองคือ HIGH หรือ LOW

ระบบจำนวนเลขฐานสอง ที่ใช้แทนค่าของระดับแรงดันในรูปของรูปคลื่นไบนารี การใช้ระบบจำนวนเลขฐานสองสามารถทำได้ง่ายเมื่อเทียบกับระดับของกรงดัน ที่แทนด้วยรูปคลื่น ระบบจำนวนไบนารีจะมีน้ำหนักของฐานมีค่าเป็น 2 ค่าของเลขฐานสองมีเพียง 2 ตัว คือ 0 และ 1 ตัวเลขฐานสองเราเรียกว่า บิต (BITs) ที่ใช้แทนลอจิก LOW หรือ 0 และลอจิก HIGH หรือ 1 เมื่อนำไปเปรียบเทียบกับระบบเลขฐานสิบ ซึ่งมีน้ำหนักของฐานเป็น 10 จะมีตัวเลขประกอบด้วยค่า 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9

การนับเลขฐานสอง ซึ่งประกอบไปด้วยบิตที่มีค่า 0 และ 1 บิตแรกของเลขฐานสองหรือบิตที่มีค่าต่ำสุด เราเรียกว่า LSB (Least Significant Bit) บิตที่มีค่าสูงสุดของระบบเลขฐานสองเราเรียกว่า Most Significant Bit (MSB). ดังรูปแบบที่แสดงในตาราง 2-1

Binary Bit Weights

....	2^3	2^3	2^3	2^3	2^3	2^3	2^3	2^3	.....
.	8	4	2	1	1/2	1/4	1/8	1/16	.....

Decimal Digit Weights

.	10^3	10^2	10'	10^0	10^{-1}	10^{-2}	10^{-3}	10^{-4}
	1000	100	10	1	1/10	1/100	1/1000	1/10000

รูปที่ 2-l Binary and Decimal Digit Weights

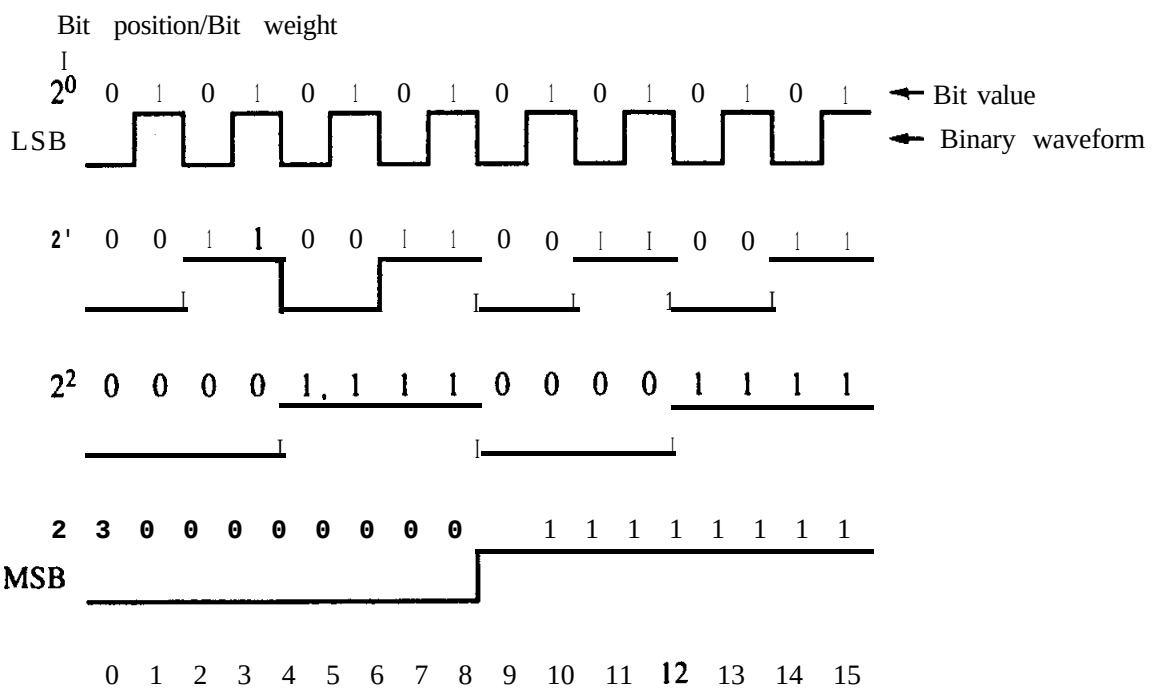
จากตาราง 2-1 เราสามารถเปรียบเทียบค่าเลขฐานสองกับค่าของเลขฐานสิบ ถ้าค่าของเลขฐานสอง n bit เราสามารถแทนค่าได้ 2^n ค่า อยู่ในช่วง 2^n - 1 ดังตัวอย่างแสดงเลขฐานสอง 4 บิต ที่สามารถแทนได้ 16 ค่า คือ ค่า 0 ถึง 15 ในระบบเลขฐานสอง

ตาราง 2-l Binary and Decimal Codes

Binary (base 2) Bit Weights	Decimal (base 10) Digit Weights
2^3 2^2 2' 2^0	10' 10^0
0	0
1	1
1 0	2
1 1	3
1 0 0	4
1 0 1	**5**
1 1 0	6
1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 10	1 0
1 0 1 1	1 1
1 1 0 0	1 2
1 1 0 1	1 3
1 1 1 0	1 4
1 1 1 1	1 **5**

2.1.1 รูปคลื่นกับเลขฐานสอง

ดิจิตอลอีเลกทรอนิกส์และการประยุกต์ใช้งานคอมพิวเตอร์ การทำงานจะแทนด้วยระบบเลขฐานสอง สำหรับการแทนค่าของแรงดันไฟฟ้า ระดับของแรงดันต่างๆเหล่านี้ จะแทนค่าด้วยกราฟในรูปของสัญญาณดิจิตอล ระบบจำนวนเลขฐานสองเป็นเพียงระบบเดียวทางคณิตศาสตร์ ที่สามารถแทนค่าของสัญญาณดิจิตอล เพราะเราสามารถเปลี่ยนแปลงสัญญาณดิจิตอลกับระบบเลขฐานสองได้ง่าย เราสมมุติลอจิกบวก (Positive Logic) ค่า 1 แทนค่าด้วยระดับแรงดันสูง และค่า 0 แทนค่าด้วยระดับแรงดันต่ำ ถ้าเราต้องการใช้เลขฐานสองนับค่า 0 ถึง 15 เราก็สารมารถใช้เลขฐานสอง 4 บิต ในการกำหนดสภาวะ 16 สภาวะ ในการนับเรียงลำดับ รูปที่ 2-2 แสดงรูปคลื่นในการนับ 0 - 15 สังเกตแต่ละรูปคลื่น จะมีการหาร 2 ความถี่ของรูปคลื่น หรือรูปคลื่นลดลงครึ่งหนึ่ง



รูปที่ 2.2 ระบบจำนวนและรูปคลื่นที่ใช้ในการนับ 0 - 15

2.1.2 การเปลี่ยนเลขฐานสองเป็นเลขฐานสิบ

ระบบเลขฐานสองเราสามารถเปลี่ยนเป็นเลขฐานสิบ โดยการคูณค่าของเลขฐานสองแต่ละบิตตามตำแหน่งน้ำหนักของบิต และนำมารวมกันเป็นผลลัพธ์ เทคนิคแบบนี้จะอ้างถึง Bit Weight expression ใช้ในการเปลี่ยนเลขฐานสองจำนวนเต็มเป็นเลขฐานสิบจำนวนเต็ม และเปลี่ยนเลขฐานสองเศษส่วนเป็นเลขฐานสิบเศษส่วน

ตัวอย่าง 2.1 การเปลี่ยนเลขฐานสองเป็นฐานสิบ

Bit Weights

$2^4\ 2^3\ 2^2\ 2^1\ 2^0 \cdot 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}$

$(1\ 0\ 1\ 1\ 0\ .)_2$

$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 =$
$2^4 + 2^2 + 2^1 = (22)_{10}$

ตัวอย่าง 2.2 การเปลี่ยนเลขฐานสองเป็นฐานสิบ

$(0\ 1\ 1\ 0\ 1\ .)_2$

$0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$
$2^3 + 2^2 + 1 = (13)_{10}$

ตัวอย่าง 2.3 การเปลี่ยนเลขฐานสองเป็นฐานสิบ

$(.1\ 0\ 1)_2$

$1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = 2^{-1} + 2^{-3} = (.625)_{10}$

ตัวอย่าง 2.4 การเปลี่ยนเลขฐานสองเป็นฐานสิบ

$(1\ 0\ 1\ 0\ 0\ .\ 1\ 1)_2$

$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 +$
$1 \times 2^{-1} + 1 \times 2^{-2} = 2^4 + 2^2 + 2^{-1} + 2^{-2} = (20.75)_{10}$

2.1.3 การเปลี่ยนเลขฐานสิบเป็นเลขฐานสอง

การเปลี่ยนเลขฐานสิบเป็นเลขฐานสอง จะมีวิธีการอยู่ 2 วิธี คือเลขจำนวนเต็มและเลขเศษส่วน ค่าของเลขฐานสิบทั้งจำนวนเต็มและเลขเศษส่วนที่จะเปลี่ยนเป็นเลขฐานสอง จะใช้เทคนิคในการเปลี่ยนที่แตกต่างกัน ทุกค่าของเลขจำนวนเต็มใช้เทคนิคหนึ่ง ส่วนเลขเศษส่วนใช้อีกเทคนิคหนึ่ง การเปลี่ยนเลขฐานสิบเลขจำนวนเต็มให้เป็นเลขฐานสองโดยใช้วิธีการที่เอา 2 มาหารซ้ำกันโดยเศษที่จะเกิดขึ้นจะเหลือค่า 1 หรือ 0 เท่านั้น ซึ่งเศษเหล่านี้คือค่าของเลขไบนารี สำหรับค่าของบิตที่เป็นที่มีค่าต่ำสุด (LSB) คือผลลัพธ์ที่ได้จากการหารครั้งแรก และเมื่อการหารสิ้นสุดลงเศษตัวสุดท้ายจะเป็นค่าบิตที่มีค่าสูงสุด (MSB) ขบวนการหารจะสิ้นสุดก็ต่อเมื่อผลหารเป็น 0

ตัวอย่าง 2.5 การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสอง

วิธีการแก้ปัญหา หารด้วย 2 เศษนั้นคือค่าของเลขฐานสอง

$(53)_{10} = (110101)_2$

DIVISION	REMAINDER	
2 | 53	1	LSB position
2 | 26	0	
2 | 13	1	
2 | 6	0	
2 | 3	1	
2 | 1	1	MSB position
0		

$$(110101)_2 = 2^5 + 2^4 + 2^2 + 2^0$$
$$= 32 + 16 + 4 + 1 = (53)_{10}$$

ตัวอย่าง 2.6 การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสอง

วิธีการแก้ปัญหา หารด้วย 2 เศษนั้นคือค่าของเลขฐานสอง

$(29)_{10} = (11101)_2$

DIVISION	REMAINDER	
2 | 29	1	LSB position
2 | 14	0	
2 | 7	1	
2 | 3	1	
2 | 1	1	MSB position
0		

ตัวอย่าง 2.7 การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสอง

วิธีการแก้ปัญหา การคูณเศษด้วย 2 ตัวทดออกคือค่าบิตเลขฐานสอง

$(.75)_{10} =$ $(.1\ 1)_2$

MULTIPLICATION	CARRY-OUT	
.75		
× 2		
1.50	1	MSB
.5		
× 2		
1.0	1	LSB

ตัวอย่าง 2.8 การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสอง

วิธีการแก้ปัญหา การคูณเศษด้วย 2 ตัวทดออกคือค่าบิตเลขฐานสอง

$(.375)_{10} =$ $(.0\ 1\ 1)_2$

MULTIPLICATION	CARRY-OUT	
.375		
x 2		
.750	0	MSB
× 2		
1.5	1	
.5		
× 2		
1.0	1	LSB

ตัวอย่าง 2.9 การเปลี่ยนเลขฐานสิบเป็นฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบเป็นเลขฐานสองทั้งจำนวนเต็มและเลขเศษส่วน

วิธีการแก้ปัญหา การหาผลลัพธ์โดยใช้วิธีการแยกออกจากกัน

$(50.14)_{10} = (50)_{10} + (.14)_{10}$

Integer:

$(50)_{10}$ = (110010)~

DIVISION		REMAINDER	
2	50	0	LSB position
2	25	1	
2	12	0	
2	6	0	
2	3	1	
2	1	1	MSB position
	0		

Fraction:

$(.14)_{10} = (.001000111101)_2$

2.2 รหัสเลขฐานสิบหก (Hexadecimal)

ระบบเลขฐานสิบหก นำหนักของของฐานคือ 16 เลขฐานสิบหก 1 หลักจะแทนด้วยเลขฐานสอง 4 บิต ระบบเลขฐานสิบหกนี้จะใช้มากในดิจิตอลอีเลกทรอนิกส์ ไมโครโปรเซสเซอร์ คอมพิวเตอร์ และการสื่อสารข้อมูล การประยุกต์ใช้งานจะใช้แทนค่าของเลขฐานสอง

ระบบเลขฐานสิบหก หรือ Hex code มีตัวเลข 16 ตัว คือเลข 0 ถึง 9 และตัวอักษร 6 ตัว คือ A , B , C , D , E , F ค่าของเลขฐานสอง 4 บิต เราสามารถแทนเลขฐานสิบหกได้ 1 ตัว

16^3 16^2 16^1 16^0 16^{-1} 16^{-2} 16^{-3}.....

รูปที่ 2.3 ตำแหน่งน้ำหนักของเลขฐานสิบหกแต่ละตำแหน่ง

ตาราง 2.2 การเปรียบเทียบระหว่างเลขฐานสิบหก ฐานสิบ ฐานสอง

Decimal (base 10)	Binary (base 2)	Hexadecimal (base 16)
0	0000 0000	0 0
1	00000001	0 1
2	0000 0010	0 2
3	0000 0011	0 3
4	0000 0100	0 4
5	0000 0101	0 5
6	0000 0110	0 6
7	00000111	0 7
8	00001000	0 8
9	0000 1001	0 9
10	0000 1010	O A
11	0000 1011	OB
12	0000 1100	o c
13	oooo 1101	O D
14	0000 1110	OE
15	0000 1111	O F
16	00010000	1 0
17	00010001	11
18	00010010	12
19	0001 0011	1 3
20	00010100	1 4

2.2.1 การเปลี่ยนเลขฐานสิบหกเป็นเลขฐานสอง

การเปลี่ยนเลขฐานสิบหกเป็นเลขฐานสอง ด้วยเลขฐานสิบหกแต่ละตัวจะแทนด้วยเลขฐานสอง 4 บิต

ตัวอย่าง 2.10 การเปลี่ยนเลขฐานสิบหกเป็นฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบหกเป็นฐานสอง

วิธีการแก้ปัญหา แทนค่าเลขฐานสิบหก 1 ตัวด้วยฐานสอง 4 บิต

$(A B C . D)_{16} = (1010\ 1011\ 1100\ .\ 1101)_2$

ตัวอย่าง 2.11 การเปลี่ยนเลขฐานสิบหกเป็นฐานสอง

$(F 1_5 6)_{16} = (111100010101\ \ 0110)_2$

2.2.2 การเปลี่ยนเลขฐานสองเป็นฐานสิบหก

การเปลี่ยนเลขฐานสองเป็นเลขฐานสิบหก ก็สามารถทำได้โดยการจับกลุ่มของเลขฐานสอง 4 บิตโดยเริ่มต้นที่จุดของไบนารีและเลื่อนไปทางซ้ายหรือขวาครั้ง 4 บิตไม่ว่าจะเป็นเลขจำนวนเต็มหรือเลขเศษส่วนก็ตาม หลังจากนั้นก็แทนค่าเลขฐานสอง 4 บิตด้วยค่าของเลขฐานสิบหก 1 ตัว

ตัวอย่าง 2.12 การเปลี่ยนเลขฐานสองเป็นเลขฐานสิบหก

ปัญหา การเปลี่ยนเลขฐานสองเป็นฐานสิบหก

วิธีการแก้ปัญหา เขียนกลุ่มของเลขฐานสองออกเป็น 4 บิต เริ่มจากจุดของเลขฐานสอง ถ้าเป็นเลขจำนวนเต็มกลุ่มสุดท้ายไม่ครบ 4 บิตให้เติม 0 ด้านหน้า ส่วนเลขเศษส่วนกลุ่มสุดท้ายไม่ครบ 4 บิต ให้เติม 0 ต่อท้าย หลังจากนั้นแทนค่าแต่ละกลุ่มด้วยเลขฐานสิบหก ดังนี้

$(1101011.1001111\ \ )_2 =$

$(110,1011.1001,111)_2 =$

$(110,1011.1001,1110)_2 =$

$(6B . 9E)_{16}$

ตัวอย่าง 2.13 การเปลี่ยนเลขฐานสองเป็นฐานสิบหก

$(11110111.010001)_2$

$(1111,0111.0100,01)_2$

$(1111,0111.0100,0100)_2$

$(F7.44)_{16}$

2.2.3 การเปลี่ยนเลขฐานสิบหกเป็นฐานสิบ

ระบบเลขฐานสิบหกเราสามารถเปลี่ยนเป็นเลขฐานสิบโดยการขยายเลขฐานสิบหกที่น้ำหนักแต่ละตัว เหมือนกับการเปลี่ยนเลขฐานสองเป็นเลขฐานสิบ การเปลี่ยนเลขฐานสิบหกเป็นเลขฐานสิบ เลขฐานสิบหกแต่ละตัวจะคูณด้วยน้ำหนักของแต่ละตำแหน่งและนำมารวมกันเป็นผลลัพธ์ ค่าของเลขฐานสิบหกจะเท่ากับเลขฐานสิบสำหรับตัวเลข 0 - 9 และถ้าเป็นตัวอักษร A , B , C , D , E , F จะมีค่าเท่ากับ 10 , 11 , 12 , 13 , 14 , 15

ตัวอย่าง 2.14 การเปลี่ยนเลขฐานสิบหกเป็นเลขฐานสิบ

ปัญหา การเปลี่ยนเลขฐานสิบหกเป็นเลขฐานสิบ

วิธีการแก้ปัญหา โดยการคูณเลขฐานสิบหกแต่ละตัวด้วยค่านำหนักและนำมารวมกันเป็นผลลัพธ์

$$(156.32)_{16} =$$

$$1 \times 16^2 + 5 \times 16^1 + 6 \times 16^0 + 3 \times 16^{-1} + 2 \times 16^{-2}$$

$$= 256 + 80 + 6 + \frac{3}{16} + \frac{2}{256}$$

$$= (342.1953)_{10}$$

ตัวอย่าง 2.15 การเปลี่ยนเลขฐานสิบหกเป็นฐานสิบ

$$(ABC.D)_{16} =$$

$$10 \times 16^2 + 11 \times 16^1 + 12 \times 16^0 + 13 \times 16^{-1}$$

$$= 2560 + 176 + 12 + \frac{13}{16}$$

$$= (2748.812)_{10}$$

2.2.4 การเปลี่ยนเลขฐานสิบเป็นฐานสิบหก

การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสิบหกจำนวนเต็ม โดยการหารค่าของเลขฐานสิบด้วย 16 เศษที่เหลือของการหารแต่ละครั้งจะเป็นเลขฐานสิบหก เริ่มต้นจากค่า (LSB) ค่าที่มีน้ำหนักน้อยที่สุดของเลขจำนวนเต็ม การหารจะหารซ้ำไปเรื่อยๆ จนหารไม่ได้ เศษของตัวหารครั้งสุดท้ายจะเป็นค่า (MSB) ค่าที่มีน้ำหนักสูงสุดของเลขฐานสิบหก วิธีการนี้เหมือนกับการเปลี่ยนเลขฐานสิบเป็นเลขฐานสอง

การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสิบหกเศษส่วน โดยการคูณเศษด้วยค่า 16 ตัวทดออกของเศษคือค่าของเลขฐานสิบหก ค่าแรกเป็นค่า (MSB) ของเศษ ผลลัพธ์ของเศษจากการคูณตัวแรกจะเป็นเศษที่ต่อจากเลขจำนวนเต็ม

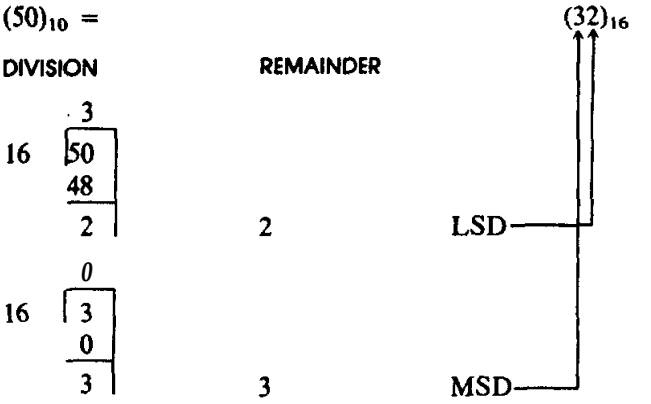
กระบวนที่จะเปลี่ยนเลขฐานสิบเป็นสิบหกเราสามารถใช้เลขฐานสองเข้ามาช่วยได้โดยการเปลี่ยนเลขฐานสิบเป็นเลขฐานสองก่อน แล้วจึงมาเปลี่ยนเป็นเลขฐานสิบหกอีกครั้งหนึ่งได้

ตัวอย่าง 2.16 การเปลี่ยนเลขฐานสิบเป็นฐานสิบหก

ปัญหา *การเปลี่ยนเลขฐานสิบจำนวนเต็มเป็นเลขฐานสิบหกจำนวนเต็ม*

วิธีการแก้ปัญหา โดยการหารเลขฐานสิบด้วย 16 เศษที่เหลือจะเป็นเลขฐานสิบหก เริ่มต้นจากค่าที่น้อยที่สุด ทำการหารซ้ำๆกัน จนได้ค่าเลขฐานสิบหกทั้งหมด

ตัวอย่าง 2.17 การเปลี่ยนเลขฐานสิบเศษส่วนเป็นเลขฐานสิบหกเศษส่วน



ตัวอย่าง 2.18 การเปลี่ยนเลขฐานสิบเป็นเลขฐานสิบหกผ่านเลขฐานสอง

ปัญหา การเปลี่ยนเลขฐานสิบจำนวนเต็ม และเลขเศษส่วนเป็นเลขฐานสิบหกผ่านเลขฐานสอง

วิธีการแก้ปัญหา เปลี่ยนเลขฐานสิบเป็นฐานสอง และเปลี่ยนเลขฐานสองเป็นฐานสิบหก

$(SO.14)_{10}$ =

$(50)_{10}$ = (1100010)~

$(.14)_{10} = (.00100111101)_2$

Binary Result:

$(50.14)_{10} = (1100010.001000001)_2$

$(11\ 0010.0010\ 0011\ 1101)_2$ =

$(0011\ 0010.0010\ 0011\ 1101)_2$ =

$(32.23D)_{16}$

Hex Result:

$(50.14)_{10} = (32.23D)_{16}$

2.3 รหัสเลขฐานสอง

รหัสเลขฐานสองเราสามารถออกแบบแทนค่าของเลขฐานสิบได้และแทนค่าตัวอักขระต่างๆได้ รหัสเหล่านี้จะใช้ในดิจิตอลอีเลคโทรนิและการประยุกต์ใช้งานด้านคอมพิวเตอร์ และระบบสื่อสาร

2.3.1 รหัส BCD 8421

รหัส 8421 Binary Coded decimal ที่เรานำมาใช้แทนตัวเลข 0 - 9 ด้วยค่าของเลขฐานสอง 4 บิต เลขฐานสิบสามารถเปลี่ยนเป็นเลขฐานสองได้ง่าย โดยการแทนค่าของเลขฐานสิบ 1 ตัวด้วยค่าเลขฐานสอง 4 บิต รูปแบบโดยไม่ต้องมีการคูณหรือการหาร การเปลี่ยนใช้น้ำหนักของแต่ละตำแหน่งคือ 8421 แทนค่าเลขฐานสิบในรูปของไบนารี

รหัส 8421 หรือ BCD Code เรารู้จักโดยทั่วไปว่าจะมีเลขฐานสองเพียง 4 บิตเท่านั้น ในการแทนค่าเลขฐานสิบ 1 ตัว คือ 0 -9 ส่วนอีก 6 ค่าของเลขฐานสองนั้นเราไม่ใช้เพราะว่าไม่ใช่ส่วนของรหัส BCD ดังแสดงในตาราง 2.3 8421 BCD Code ส่วนตาราง 2.4 เป็นการเปรียบเทียบเลขฐานสิบ ฐานสิบหก ฐานสอง และรหัส BCD

ตาราง 2.3 8421 BCD Code

Decimal	8421 B C D
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
Invalid 8421 BCD Codes	
	1010
	1011
	1100
	1101
	1110
	1111

ตาราง 2.4 **Decimal, BCD , Binary and Hex Numbers**

Decimal	BCD	Binary	Hex
0	0000	00	0
1	0001	01	1
2	0010	10	2
3	0011	11	3
4	0100	100	4
5	0101	101	5
6	0110	110	6
7	0111	111	7
8	1000	1000	8
9	1001	1001	9
10	00010000	1010	A
11	0001 0001	1011	B
12	0001 0010	1100	C
13	0001 0011	1101	D
14	0001 0100	1110	E
15	0001 0101	1111	F
16	0001 0110	10000	10
17	0001 0111	10001	11
18	0001 1000	10010	12
19	0001 1001	10011	13
20	0010 0000	10100	14

ตัวอย่าง 2.19 การเปลี่ยนเลขฐานสิบเป็น BCD

$(1990)_{10}$ =

$(0001\ \ 1001\ \ 1001\ \ 0000)_{BCD}$

ตัวอย่าง 2.20 Decimal to BCD Conversion

$(\ 20.57\ )_{10}$ = (0010 0000.0101 0111)$_{BCD}$

ตัวอย่าง 2.21 BCD to Decimal Conversion

$(0010\ 1001\ \ 0001.\ \ 0110)_{BCD} = (291.6)_{10}$

ตัวอย่าง 2.22 BCD to Decimal Conversion

$(1001\ \ 0010.\ \ 1000\ 0111)_{BCD} = (92.87)_{10}$

2.3.2 รหัสค่าเกิน3กับรหัส BCD

รหัสค่าเกิน 3 คือรหัสไบนารีที่ใช้แทนค่าของเลขฐานสิบแต่ละตัวด้วยค่าของเลขฐานสอง 4 บิต รหัสค่าเกิน 3 หมายถึงรหัส 8421 BCD แล้วค่าอีก 3 $(0011)_2$ ของค่า BCD แต่ละตัว

ตาราง 2.5 Excess - 3

Decimal	Excess-3
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100
Invalid Excess-3 Codes	
0000	
0001	
0010	
1101	
1110	
1111	

ตาราง 2.6 Decimal, 8421 BCD , and Excess-3

Decimal	8421 BCD	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

รหัสค่าเกิน 3 ที่แสดงในตาราง 2.5 เหมือนกับรหัส 8421 BCD คือรหัสค่าเกิน 3 จะมีค่าไบนารี 6 ค่าที่ไม่ใช้ ส่วนรหัสค่าของไบนารีที่ไม่ใช้แสดงในตาราง 2.5 ค่าเลขฐานสิบสามารถเปลี่ยนเป็นรหัสค่าเกิน 3 โดยอยู่ในรูปแบบของ 8421 BCD แล้วบวกเพิ่มอีก 3 ในแต่ละตัว ตาราง 2.6 แสดงความสัมพันธ์ระหว่างเลขฐานสิบ รหัส 8421 BCD และรหัสค่าเกิน 3

ตัวอย่าง 2.23 Decimal to Excess - 3 Conversion

$$(1990)_{10} = (0100\ 1100\ 1100\ 0011)_{x-3}$$

ตัวอย่าง 2.24 Decimal to Excess - 3 Conversion

$$(20.57)_{10} = (01000011.\ 1000\ 1010)_{x-3}$$

ตัวอย่าง 2.25 Excess - 3 to **Decimsl** Comversion

$$(0101\ 1100\ 0100\ .\ 1001)_{x-3} = (291.6)_{10}$$

2.3.3 รหัสเกย์ (Gray)

รหัสเกรย์ เป็นรหัสไบนารีพิเศษที่พิจารณาตามบิตที่มีการเปลี่ยนแปลง ดูรายละเอียดจากตาราง 2.7 .

ตาราง 2.7 Decimal, Gray Code, Binary Values

Decimal	Gray	Binary
0	0000	0
1	0001	1
2	0011	10
3	0010	11
4	0110	100
5	0111	101
6	0101	110
7	0100	111
8	1100	1000
9	1101	1001
10	1111	1010
11	1110	1011
12	1010	1100
13	1011	1101
14	1001	1110
15	1000	1111
16	11000	10000

2.3.4 รหัสไบนารีอัลฟานิวเมอริค

เป็นรหัสไบนารีของตัวอักขระที่เป็นตัวเลขและตัวอักษร จุดประสงค์ของรหัสชนิดนี้ เพื่อใช้ในการส่งข้อมูลของคอมพิวเตอร์ แต่ละคีย์ของคีย์บอร์ดจะถูกกำหนดค่าของไบนารี ดังนั้น คอมพิวเตอร์สามารถกำหนดค่าในแต่ละตัวได้ ตัวอักขระเหล่านี้รวมถึง ตัวอักษร ตัวเลข เครื่องหมายพิเศษ เครื่องหมายควบคุม และตัวอักขระกำหนดรูปแบบของหน้า (Page format) เช่น Space , backspace and Line feed เป็นต้น

รหัส Alphanumeric ที่ใช้ทั่วๆไปมี 2 ชนิด คือรหัส ASCII และ EBCDIC รหัส ASCII คือ American Standard Code for Information Interchange เป็นรหัสไบนารีขนาด 7 บิต สามารถแทนตัวอักขระได้ 128 ตัว แสดงในตาราง 2.8 สังเกตว่ารหัสจะแทนด้วยค่าเลขฐานสิบหก สามารถเปลี่ยนค่าเป็นเลขฐานสองได้ง่ายถ้าจำเป็น

ตาราง 2.8 ASCII CODE

HEX	Character	HEX	Character	HEX	Character
0	NUL	12	DC2	24	$
1	SOH	13	DC3	25	%
2	STX	14	DC4	26	&
3	ETX	15	NAK	27	
4	EOT	16	SYN	28	(
5	ENQ	17	ETB	29	)
6	ACK	18	CAN	2A	*
7	BEL	19	EM	2B	+
8	BS	1A	SUB	2c	
9	HT	1B	ESC	2D	—
A	LF	1C	FS	2E	.
B	VT	1D	GS	2F	/
C	FF	1E	RS	30	0
D	CR	1F	us	31	1
E	so	20	SP	32	2
F	S1	21	!	33	3
10	DLE	22	"	34	4
11	DC1	23	#	35	5

HEX	Character	HEX	Character	HEX	Character
36	6	54	T	72	r
37	7	55	U	73	s
38	8	56	V	74	t
39	9	57	W	75	u
3A		58	X	76	v
3B		59	Y	77	w
3c	<	5A	Z	78	x
3 D	=	5B	[	79	Y
3E	>	5C		7A	z
3F	?	5D	]	7B	{
40	a	5E	^	7c	|
41	A	5F		7 D	}
42	B	60		7E	
43	C	61	a	7F	DEL
44	D	62	b		
45	E	63	c		
46	F	64	d		
47	G	65	e		
48	H	66	f		
49	I	67	g		
4A	J	68	h		
4B	K	69	i		
4c	L	6A	j		
4 D	M	6B	k		
4E	N	6 C	l		
4F	0	6 D	m		
50	P	6E	n		
51	Q	6F	o		
52	R	70	p		
53	s	71	q		

รหัส ASCII ขนาด 7 บิต นั้นบิตที่ 8 ใช้เป็นบิตตรวจสอบ (Parity bit) จัดเป็นบิตที่มีค่าสูงสุด บิตตรวจสอบใช้ในการป้องกันข้อผิดพลาดเกิดขึ้นเวลาการส่งข้อมูล

EBCDIC (Extended Binary Code Decimal Information Code) เป็นรหัสขนาด 8 บิต สามารถแทนตัวอักษรได้ 256 ตัว ดังแสดงในตาราง 2.9

ตาราง 2.9 EBCDIC CODE

HEX	Character	HEX	Character	HEX	Character
00	NUL	29		32	SYN
0i	SOH	2A	SM	33	
02	STX	2B		34	PN
03	ETX	2c		35	RS
04	PF	2D	ENQ	36	UC
05	PT	2E	ACK	37	EOT
06	LC	2F	BEL	38	
07	DEL	90		39	
08	VT	91	j	3A	
09	RLF	92	k	3B	NAK
OA	SMM	93	l	3c	DC4
OB		94	m	3D	NL
oc	FF	95	n	3E	
OD	CR	96	o	3F	SUB
OE	so	97	p	40	SP
OF	S1	98	q	41	
10	DLE	99	r	42	
11	DC1	9A		43	
12	DC2	9B		44	
13	DC3	9c		45	
14	RES	9D		46	
15		9E		47	
16	BS	9F		48	
17	L	A0		49	
18	CAN	Al		4A	¢
19	EM	A2	s	4B	.
1A	cc	A3	t	4c	<
1B		A4	u	4D	(
1C	FS	A5	v	4E	+
1D	GS	A6	w	4F	|
1E		A7	x	50	&
1F	us	A8	y	51	
20	DS	A9	z	52	
21	SOS	AA		53	
22	FS	AB		54	
23		AC		55	
24	BYP	AD		56	
25	LF	AE		57	
26	ETB	AF		58	
27	ESC	30		59	
28		31		5A	!

ตาราง 2.9 ต่อ

HEX	Character	HEX	Character	HEX	Character
5B	*	62		89	i
SC	)	64 63		8A	
SD				8B	
5E	¬	65		8C	
5F	{	6766		SD	
co				8E	
Cl	A	68		8F	
c2	B	69		EO	\
c3	C	6A		El	
c4	D	6B		E2	S
C5	E	6C	%	E3	T
C6	F	6D		E4	U
C7	G	6E	>	E5	V
C8	H	6F	?	E6	W
c9	I	70		E7	X
CA		71		E8	Y
CB		72		E9	Z
cc		73		EA	
CD		74		EB	
CE		75		EC	
CF		76		ED	
Do	}	77		EE	
D1	J	78		EF	
D2	K	79	A	FO	0
D3	L	7A	:	F1	1
D4	M	7B	#	F2	2
D5	N	7C	a	F3	3
D6	0	7D	A	F4	4
D7	P	7E	=	F5	5
D8	Q	7F	"	F6	6
D9	R	80		F7	7
DA		81	a	F8	8
DB'		82	b	F9	9
DC		83	c	FA	
DD		84	d	FB	
DE		85	e	FC	
DF		86	f	FD	
60		87	g	FE	
61	/	88	h	FF	

ตัวอย่าง 2.26 รหัส ASCII and EBCDIC Coding of Alphanumeric Information

รหัส ASCII ในรูปเลขฐานสิบหก

22 54 68 65 20 77 65 65 6B 65 6E 64 20
“ T h e space w e e k e n d space

69 73 20 68 65 72 65 20 61 74 20
i s space h e r e space a t space

6C 61 73 74 21 22
l a s t ! “

รหัส EBCDIC ในรูปเลขฐานสิบหก

7F E3 88 85 40 A6 85 85 92 85 95 84 40
“ T h e space w e e k e n d space

89 A2 40 88 85 99 85 40 81 A3 40
i s space h e r e space a t space

93 81 A2 A3 5A 7F
l a s t ! “

2.4 การคำนวณไบนารี

การคำนวณทางคณิตศาสตร์ การบวก การลบ การคูณ การหาร การทำงานจะอยู่ในรูปแบบของไบนารี่ การกำหนดวิธีการคำนวณเหล่านี้ เหมือนกับกฎเกณฑ์ของการคำนวณเลขฐานสิบ ดังแสดงวิธีการดังต่อไปนี้

2.4.1 การบวกเลขฐานสอง

การบวก คือ วิธีการคำนวณที่สำคัญที่สุดของระบบไบนารี เพราะสามารถใช้แทนการคำนวณทางคณิตศาสตร์อื่นๆได้ทั้งหมดไม่ว่าจะเป็นการลบ การคูณ การหาร เช่นการลบเราสามารถใช้วิธีการบวกช่วยในการคำนวณได้ การคูณก็ใช้การบวกซ้ำๆกัน ส่วนการหารก็ใช้วิธีการลบซ้ำๆกัน แต่การลบเราใช้วิธีการบวกแทนได้ ฉะนั้นท่านจะต้องทำความเข้าใจการบวกเลขฐานสองหรือไบนารี

การบวกเลขฐานสองมีกฎเกณฑ์ดังต่อไปนี้ แสดงในรูป 2.4

0 + 0 = 0	0 + 1 = 1
1 + 0 = 1	1 + 1 = 0 ทด 1

รูป 2.4 กฎเกณฑ์ของการบวก

ตัวอย่าง 2.27 Binary Addition

	1	**Carry-out**
1001010	**1001010**	
+ 10011	**+ 10011**	
	1011101	

ตัวอย่าง 2.28 Binary Addition

	111111	Carry-out
1111111	1111111	
+1110101	+ 1110101	
	11110100	

ตัวอย่าง 2.29 Binary Addition

	1	Carry-out
1000000	**1000000**	
+ 1100011	+ llooolI	
	10100011	

2.4.2 การลบเลขฐานสองไม่คิดเครื่องหมาย

การลบเลขโดยไม่คิดเครื่องหมายมีกฎเกณฑ์พื้นฐานดังต่อไปนี้ แสดงในรูป 2.5

				Borrow in
Values to be subtracted:	0	1	1	1 0
	- 0	- 0	- 1	1
Result:	0	1	0	1

1 0 0	**1 Borrow in**
- 1	**1 0 0**
	- 1
	1 1

รูป 2.5 กฎเกณฑ์การลบเลขฐานสอง

ตัวอย่าง 2.30 การลบเลขฐานสอง

```
  10110010        0 1          1      Borrow
- 1010001       X 0 1 1 0 0 X 0
                - 1 0 1 0 0 0 1
                  1 1 0 0 0 0 1
```

ตัวอย่าง 2.31 การลบเลขฐานสอง

```
  11100000          01          Borrow
- 11100          1 1 X Ø A 00
                - 1 1 1 0 0
                  1 1 0 0 0 1 0 0
```

ตัวอย่าง 2.32 การลบเลขฐานสอง

```
  1000000          0 1 1 1 1          Borrow
- 101011         X Ø Ø Ø Ø A
                - 1 0 1 0 1 1
                    1 0 1 0 1
```

2.4.3 การคูณเลขฐานสอง

การคูณเลขฐานสองเหมือนกับการคูณเลขฐานสิบ มีกฎการคูณดังต่อไปนี้

0 x 0 = o

o x 1 = 0

1 x 0 = 0

l x 1 = 1

รูป 2.6 กฎการคูณเลขฐานสอง

ตัวอย่าง 2.33 การคูณเลขฐานสอง

```
  100011
x    10
-------
 1000110
```

ตัวอย่าง 2.34 การคูณเลขฐานสอง

```
   101010
   x  11
   101010
 +101010
  1111110
```

ตัวอย่าง 2.35 การคูณเลขฐานสอง

```
   100100
   x 1011
   100100
  100100
 +100100
 110001100
```

ตัวอย่าง 2.36 การคูณเลขฐานสองโดยวิธีบวก

```
   101010          101010
   x  11     +     101010
   101010    +     101010
  101010           1111110
  1111110
```

2.4.4 การหารเลขฐานสอง

กฎการหารเลขฐานสองเหมือนกับการหารเลขฐานสิบ

ตัวอย่าง 2.37 การหารเลขฐานสอง

```
110 ÷ 10 = 10/110
             11
            10
            10
            00
```

ตัวอย่าง 2.38 การหารเลขฐานสอง

$$10100 \div 101 = 101\overline{)\,10100}\;\begin{array}{r}100\\ \underline{101}\\ 00\end{array}$$

ตัวอย่าง 2.39 การหารเลขฐานสอง

$$1001 \div 10 = 10\overline{)\,1001.0}\;\begin{array}{r}100.1\\ \underline{10}\\ 010\\ \underline{10}\\ 00\end{array}$$

2.4.5 เลขฐานสองคิดเครื่องหมาย

ระบบจำนวนเราสามารถกำหนดค่าเป็นเลขที่เป็นค่าบวกและค่าลบ เพื่อกำหนดค่าในการคำนวณทางคณิตศาสตร์ การแทนค่าที่เป็นเครื่องหมายของระบบจำนวน จะต้องกำหนดเครื่องหมายและขนาด หรือค่าที่อยู่ในรูปคอมพลีเม้นต์ อาจจะเป็น 1 คอมพลีเม้นต์ หรือ 2 คอมพลีเม้นต์

Sign and magnitude (SAM) ที่แทนในระบบเลขฐานสอง จะต้องมีบิตเครื่องหมาย ที่เพิ่มเข้ามาเป็นบิตที่มีค่าสูงสุด (MSB) ตำแหน่งของบิตนี้จะใช้แทนเครื่องหมาย ถ้าเป็นค่า 0 ใช้แทนเครื่องหมายบวก แต่ถ้าเป็น 1 ใช้แทนเครื่องหมายลบ ส่วนบิตที่เหลือเป็นขนาดของเลขฐานสอง

ตัวอย่าง 2.40 Sign and Magnitude

$(5)_{10} = (101)_2$ SAM + 5 = 0101

− 5 = 1101

ตัวอย่าง 2.41 Sign and Magnitude Notation

$(12)_{10} = (1100)_2$ SAM + 12 = 01100

− 12 = 11100

ตัวอย่าง 2.42 Sign and Magnitude Notation

$(31)_{10} = (11111)_2$ SAM + 31 = 011111

− 31 = 111111

2.4.6 ค่าของ 1's และ 2's คอมพลีเม้นต์

คำว่า คอมพลีเม้นต์ การเปลี่ยนเครื่องหมายของระบบจำนวนสำหรับการลบ 1 คอมพลีเม้นต์หรือ 2 คอมพลีเม้นต์ จะใช้ในการคำนวณของระบบไมโครโปรเซสเซอร์ หลักการลบจะใช้การลบแบบคอมพลีเม้นต์ คือ การลบโดยวิธีการบวก หลักการลบโดยวิธีการบวกนั้น เราสามารถใช้ได้ทั้ง 1 คอมพลีเม้นต์และ 2 คอมพลีเม้นต์

ตาราง 2.1 Folded Binary Code - SAM Notation

Decimal	Binary
+5	0101
+4	**0100**
+3	**0011**
+2	**0010**
+1	**0001**
+o	**0000**
- 0	**1000**
- 1	**1001**
- 2	**1010**
- 3	**1011**
- 4	**1100**
- 5	1101

1 คอมพลีเม้นต์ คือการกลับค่าของเลขฐานสองนั้นเป็นตรงกันข้าม หรือการลบแต่ละบิตด้วยค่า 1 ถ้าเป็นการเปลี่ยนค่าของบิตนั้นเป็นตรงกันข้ามคือ 1 เปลี่ยนเป็น 0 หรือ ถ้ามีค่าเป็น 0 ก็จะเปลี่ยนค่าเป็น 1 ส่วนในกรณีของ 2 คอมพลีเม้นต์ ให้ทำเหมือนกับ 1 คอมพลีเม้นต์และบวก 1 เข้ากับบิตที่มีค่าต่ำสุด (LSB)

คอมพลีเม้นต์ของเลขจำนวนเต็มบวก บิตเครื่องหมายจะมีค่าเป็น 0 ส่วนบิตเครื่องหมายของค่าลบจะเป็น 1 จากตาราง 2.11 เป็นการสรุปขั้นตอนของการทำ 1 คอมพลีเม้นต์และ 2 คอมพลีเม้นต์ วิธีที่สั้นที่สุดในการทำ 2 คอมพลีเม้นต์ คือใช้หลักการของตาราง 2.11 การตรวจสอบสามารถทำเริ่มต้นจาก LSB และเลื่อนต่อไปยัง MSB ถ้าพบบิตที่เป็นค่า 1 ค่าแรกจะไม่มีการเปลี่ยนแปลง แต่บิตต่อไปจากบิต 1 ที่เป็นค่าแรกจะเปลี่ยนค่าเป็นตรงกันข้ามจนถึงบิต MSB

ตัวอย่าง 2.43 One's Complement Notation

$(5)_{10} = (101)_2$

ค่า 1 คอมพลีเม้นต์อยู่ในรูป 8 **บิต**

+5 = 00000101

- 5 = 11111010

SAM (8 bit form)

+ 5 = 0000101

▪ 5 = 1000101

ตัวอย่าง 2.44 One's Complement Notation

$(12)_{10} = (1100)_2$

One's complement (8-bit form):

+12 = 00001100

-12 = 11110011

SAM (I-bit form):

+12 = 00001100

-12 = 10001100

ตัวอย่าง 2.45 One's Complement Notation

$(31)_{10} = (11111)_2$

One's complement (8-bit form):

+31 = ooollllll

-31 = llloooO

SAM (8-bit form):

+31 = 00011111

-31 = 10011111

ตัวอย่าง 2.46 Two's Complement Notation

$(12)_{10} = (1100)_2$

Two's complement (8-bit form):

+12 = 00001100

-12 = 11110100

One's complement (8-bit form):

+12 = 00001100

-12 = 11110011

SAM (S-bit form):

+12 = 00001100

-12 = 10001100

ตัวอย่าง 2.47 Two's Complement Notation

$(31)_{10} = (11111)_2$

Two's complement (8-bit form):

+31 = 00011111

-31 = 11100001

One's complement (8-bit form):

+31 = 00011111

-31 = 11100000

SAM (I-bit form):

+31 = 00011111

-31 = 10011111

ตัวอย่าง 2.48 Two's Cpmplement Notation

$(5)_{10} = (101)_2$

Two's complement (8-bit form):

+5 = 00000101

-5 = 11111011

One's complement (8-bit form):

+5 = 00000101

-5 = 11111010

SAM @-bit form):

+5 = 00000101

-5 = 10000101

2.4.7 การคำนวณเลขฐานสองคิดเครื่องหมาย

การทำงานในการคำนวณทางคณิตศาสตร์ ของเลขฐานสองเราสามารถทำงานคำนวณ ด้วยการบวก ซึ่งค่าคอมพลีเม้นต์ใช้แทนค่าลบ ค่าบวกจะอยู่ในรูปแบบของค่าจริง (True) คอมพลีเม้นต์จะใช้ได้ทั้งการคำนวนของ 1 คอมพลีเม้นต์และ 2 คอมพลีเม้นต์

การทำงานของการลบ คือจะใช้วิธีการลบโดยวิธีการบวกโดยอาศัยหลักการของคอมพลีเม้นต์ การลบที่เปลี่ยนเป็นการบวกโดยการเปลี่ยนเครื่องหมายเป็นลบ วิธีการนี้เทียบเท่ากับการทำคอมพลีเม้นต์ในการลบ

ไมโครโปรเซสเซอร์และระบบดิจิตอล จะทำงานรูปแบบของการบวก ส่วนการลบใช้หลักการของ 1 และ 2 คอมพลีเม้นต์ กฎการทำงานดูจากตาราง 2.12 ที่ใช้ในการคำนวณ

ตาราง 2.12 1 และ 2 การลบแบบคอมพลีเม้นต์

1 คอมพลีเม้นต์	**2 คอมพลีเม้นต์**
1. เปลี่ยนค่าตัวลบเป็น 1 คอมพลีเม้นต์	1. เปลี่ยนค่าตัวลบเป็น 2 คอมพลีเม้นต์
2. บวกเข้ากับตัวตั้ง	2. บวกเข้ากับตัวตั้ง
3. ถ้ามีตัวทดออกที่ MSB ให้นำมาบวกกับค่าบิตต่ำสุด LSB	3. ถ้ามีตัวทดออกที่ MSB ให้ตัดทิ้ง
4. ตรวจสอบบิตเครื่องหมายสำหรับการเกิด Overflow ถ้าเกิดขึ้นจากการบวกเลข 2 จำนวนที่เครื่องหมายเดียวกัน ผลเครื่องหมายจะตรงกันข้าม	4. ตรวจสอบบิตเครื่องหมายสำหรับการเกิด Overflow ถ้าเกิดขึ้นจากการบวกของเลข 2 จำนวนเครื่องหมายเดียวกัน ผลลัพธ์เครื่องหมายจะตรงกันข้าม
5. ผลลัพธ์ที่เป็นค่าบวกคือ ค่าจริง ผลลัพธ์ที่เป็นค่าลบจะต้องทำ 1 คอมพลีเม้นต์อีกครั้งหนึ่งจึงจะเป็นค่าจริง	5. ผลลัพธ์ที่เป็นค่าบวกคือ ค่าจริง ส่วนผลลัพธ์ที่เป็นค่าลบจะต้องเปลี่ยนค่าเป็นค่า 2 คอมพลีเม้นต์อีกครั้งหนึ่ง

ตัวอย่าง 2.49 One's Complement Subtraction

	1111 11	carry-out
01011101	01011101	
−10010000	+01101111	
Result:	11001100	

ตัวอย่าง 2.50 One's Complement Subtraction

01100100	01100100
-00110111	+ 11001000
	100101100
End-around carry:	+ 1
Result:	00101101

ตัวอย่าง 2.51 Two's Complement Subtraction

10011110	10011110
-01011001	+ 10100111
Drop the carry-out:	11000101
Result:	11000101

ตัวอย่าง 2.52 Two's Complement Subtraction

11101001	11101001
−11000011	+ 00111100
Drop the **carry-out:**	00100101
Result:	00100101

2.5 การคำนวณใช้รหัส BCD และรหัสเลขฐานสอง

การคำนวณทางคณิตศาสตร์ในระบบดิจิตอลใช้หลักการของเลขฐานสอง แต่การแทนค่าระบบเลขฐานสอง ที่จะทำให้เข้าใจง่ายสำหรับการคำนวณในการบวก การลบ เราจัดให้อยู่ในรูปเลขฐานสิบหก 8421 BCD ดังต่อไปนี้

2.5.1 การคำนวณเลขฐานสิบหก

เลขฐานสิบหกสามารถเปลี่ยนกับเลขฐานสองได้ง่าย เราสามารถใช้เลขฐานสิบหกแทนเลขฐานสองเพื่อนำมาคำนวณได้สะดวก และทำการเปลี่ยนกับเป็นเลขฐานสองเพื่อให้ผลลัพธ์ ดังตัวอย่างต่อไปนี้

ตัวอย่าง **2.53** Hexadecimal Addition and Subtraction 11 carry-out

2F1	0010 1111 0001
+503	+0101 0000 0011
7F4	0111 1111 0100

ตัวอย่าง 2.54 Hexadecimal Addition and Subtraction

B17	1011	**0001**	0111
−F2C	-1111	0010	1100
Form two's	1011	**0001**	**0111**
complement:	+oooo	1101	0100
	1011	1110	1011

The result is a negative number in the two's complement form.

Convert result to hex: $(BEB)_{16}$

2.5.2 การคำนวณ 8421 BCD

การคำนวณทางคณิตศาสตร์สามารถใช้รูปแบบของ BCD และคอมพลีเม้นต์ ในการแทนค่าเลขฐานสิบ โดยไม่ต้องเปลี่ยนค่าเป็นเลขฐานสอง อย่างไรก็ตามการบวกค่าเลข BCD ต้องการกฎพิเศษเพื่อป้องกันผลลัพธ์ที่ผิดพลาด

8421 BCD ที่แทนเลขฐานสิบแต่ละตัว คือ 0 ถึง 9 แทนด้วยเลขฐานสอง 4 ตัว เลขฐานสอง 4 ตัวมารวมกันแล้วค่าที่มากกว่า 9 ถือว่าไม่ใช่เลข BCD เมื่อมีการบวกเลข BCD ถ้าผลลัพธ์มากกว่า 9 ให้บวกอีก 6 หรือ 0110 ในเลข BCD แต่ละกลุ่ม เพื่อคำนวณค่าของผลลัพธ์ที่มากกว่า 9 หรือค่าที่มีตัวทดไปยังตัวเลข BCD สูงถัดไปก็ให้บวกอีก 6 เช่นกัน ดังตัวอย่างต่อไปนี้

ตัวอย่าง 2.55 8421 BCD Arithmetic

$(146)_{10} + (259)_{10} = (405)_{10}$

	oool	0100	0110	Form BCD
+	**0010**	0101	1001	
	0011	1001	1111	
+			0110	Correction
	0011	1010	0101	
+		0110		Correction
	0100	0000	0101	= $(405)_{10}$

ตัวอย่าง 2.56 8421 BCD Arithmetic

$(52)_{10} + (199)_{10} = (251)_{10}$

	0000	0101	0010	Form BCD
+	**0001**	1001	1001	
	0001	1110	1011	
+		0110	0110	Correction
	0010	0101	oool	= $(251)_{10}$

ตัวอย่าง 2.57 8421 BCD Arithmetic

$(8)_{10} + (8)_{10} = (16)_{10}$

	1000	Form BCD
+	1000	
0001	oooo	
+	0110	Correction
0001	0110	= $(16)_{10}$

สรุป

ระบบเลขฐานสองเป็นระบบที่ใช้ในวงจรดิจิตอล ซึ่งค่าของเลขฐานสอง คือ 1 และ 0 เหมือนกับค่าของสัญญาณดิจิตอล ระบบเลขฐานสองมีฐานเป็นค่า 1 ถ้ามี n บิต สามารถกำหนดสภาวะได้ 2^n ค่า

ระบบเลขฐานสิบ สามารถแทนอยู่ในรูปของเลขฐานสอง สามารถเปลี่ยนระหว่างเลขฐานสิบกับเลขฐานสองได้

ระบบเลขฐานสิบหก คือเราสามารถนำมาแทนเลขฐานสองได้ โดยใช้เลขฐานสิบหก 1 ตัว แทนด้วยเลขฐานสอง 4 ตัว ฐานของเลขนี้คือ 16 สามารถเปลี่ยนให้อยู่ในเลขฐานสิบและฐานสองได้

รหัสไบนารีใช้แทนเลขฐานสิบ ที่อยู่ในรูปของ BCD และยังใช้กำหนดค่าของตัวอักขระ รหัส BCD ใช้แทนค่าเลขฐานสิบ 0 ถึง 9 ด้วยค่าไบนารี 4 บิต

รหัส ASCII , EBCDIC เป็นรหัสตัวอักษรที่ใช้แทนตัวอักขระมากที่สุด

แบบฝึกหัด

1. ระบบเลขฐานสองมีความสัมพันธ์อย่างไรกับระบบดิจิตอล
2. จงเปลี่ยนระบบเลขฐานสองต่อไปนี้ให้เป็นเลขฐานสิบ

   2..1 01100 2.2 10011 2.3 101101 2.4 1100111
3. จงเปลี่ยนระบบเลขฐานสิบต่อไปนี้ให้เป็นเลขฐานสอง

   3.1 186 3.2 245 2.3 98 3.4 129
4. จงเปลี่ยนระบบเลขฐานสองต่อไปนี้ให้เป็นเลขฐานสิบหก

   4.1 1100 1001 4.2 0111 1010 4.3 0011 1100 4.4 1010 0101 1101
5. จงเปลี่ยนระบบเลขฐานสิบหกต่อไปนี้ให้เป็นเลขฐานสอง

   5.1 0AC9 5.2 1DC6 5.3 5BF8 5.4 0C47
6. . จงเปลี่ยนระบบรหัส BCD ต่อไปนี้ให้เป็นเลขฐานสิบ

   6.1 $1001\ 0011_{BCD}$ 6.2 $0001\ 0011_{BCD}$ 6.3 $0101\ 0111_{BCD}$ 6.4 $0011\ 1001_{BCD}$
7. จงเปลี่ยนตัวอักขระรหัส ASCII ให้เป็นเลขฐานสอง

   7.1 A , G , Z 7.2 5 , 9 , 0 7.3 P , a , g 7.4 D , 6 , y
8. จงเขียนเลขฐานสองในการจัดกลุ่มที่ใช้แทนอักขระของรหัส ASCII กับ EBCDIC อักษร A - Z