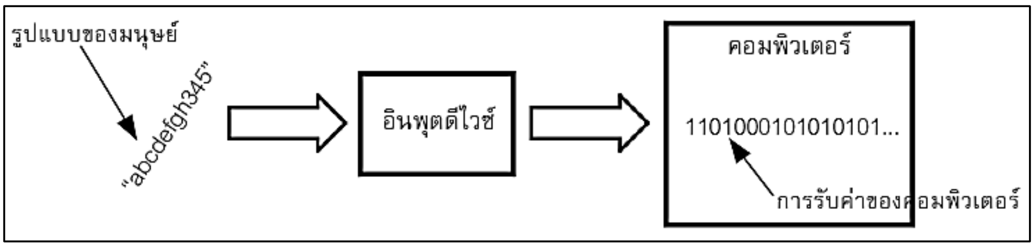
บทที่ 2
การแทนข้อมูล และการคำนวณของคอมพิวเตอร์

2.1 พื้นฐานข้อมูล

การทำงานของคอมพิวเตอร์มี 3 ขั้นตอนหลักคือ การรับข้อมูลเข้า (Input) การดำเนินการ (Process) และการนำข้อมูลออก (Output)

ภาพที่ 2.1 การทำงานของคอมพิวเตอร์

ข้อมูลนั้นมีทั้ง คาแรกเตอร์ (Character) รูปภาพ (Picture) เสียง (Sound) หรือข้อมูลในแบบต่าง ๆ จะต้องสามารถนำเข้าสู่คอมพิวเตอร์ และแปลงให้อยู่ในรูปแบบที่เหมาะสมด้วยอุปกรณ์อินพุตที่ให้คอมพิวเตอร์สามารถดำเนินการจัดเก็บ หรือนำไปใช้ในระบบคอมพิวเตอร์ได้



ภาพที่ 2.2 การแปลงข้อมูลของคอมพิวเตอร์

2.2 รูปแบบมาตรฐานของข้อมูล

2.2.1 คาแรกเตอร์ (Character)

1) ASCII (แอสกี : ASCII : American Standard Code for Information Interchange)

เป็นรหัสมาตรฐานของสหรัฐอเมริกาเพื่อการแลกเปลี่ยนสารสนเทศ เป็นรหัสอักขระที่ประกอบด้วยอักษรละติน เลขอารบิก เครื่องหมายวรรคตอน และสัญลักษณ์ต่างๆ โดยแต่ละรหัสจะแทนด้วยตัวอักขระหนึ่งตัว เช่น รหัส 65 (เลขฐานสิบ) ใช้แทนอักษรเอ (A) พิมพ์ใหญ่ เป็นต้น

รหัสแอสกีมีใช้ในระบบคอมพิวเตอร์ และเครื่องมือสื่อสารแบบดิจิทัลต่าง ๆ พัฒนาขึ้นโดยคณะกรรมการ X3 ซึ่งอยู่ภายใต้การดูแลของสมาคมมาตรฐานอเมริกา (American Standards Association) ภายหลังกลายเป็นสถาบันมาตรฐานแห่งชาติอเมริกา (American National Standard Institute : ANSI) ในปี ค.ศ. 1969 โดยเริ่มต้นใช้ครั้งแรกในปี ค.ศ. 1967 ซึ่งมีอักขระทั้งหมด 128 ตัว (7 บิต) โดยจะมี 33 ตัวที่ไม่แสดงผล (unprintable/control character) ซึ่งใช้สำหรับควบคุมการทำงานของคอมพิวเตอร์บางประการ เช่น การขึ้นย่อหน้าใหม่สำหรับการพิมพ์ (CR & LF - carriage return and line feed) การสิ้นสุดการประมวลผลข้อมูลตัวอักษร (ETX - end of text) เป็นต้น และอีก 95 ตัวที่แสดงผลได้ (printable character)

รหัสแอสกีได้รับการปรับปรุงล่าสุดเมื่อ ค.ศ.1986 ให้มีอักขระทั้งหมด 256 ตัว (8 บิต) และเรียกใหม่ว่าแอสกีแบบขยาย อักขระที่เพิ่มมา 128 ตัวใช้สำหรับแสดงอักขระเพิ่มเติมในภาษาของแต่ละท้องถิ่นที่ใช้ เช่น ภาษาเยอรมัน ภาษารัสเซีย ฯลฯ โดยจะมีผังอักขระที่แตกต่างกันไปในแต่ละภาษาซึ่งเรียกว่า โคดเพจ (codepage) โดยอักขระ 128 ตัวแรกส่วนใหญ่จะยังคงเหมือนกันแทบทุกโคดเพจ มีส่วนน้อยที่เปลี่ยนแค่บางอักขระ

ตารางที่ 2.1 รูปแบบมาตรฐานของข้อมูล

รูปแบบข้อมูล	มาตรฐาน
คาแรกเตอร์	UNICODE, ASCII, EBCDIC
รูปภาพ (บิตแมพ)	GIF (Graphical Image Format), TIFF (Tagged Image File Format), PNG (Portable Network Graphics)
รูปภาพ (อ็อบเจ็กต์)	PostScript, JPEG (Joint Photographic Experts Group), SWF (Macromedia Flash), SVG (Scalable Vector Graphics)
กราฟิกและฟอนท์	PostScript, TrueType
เสียง	WAV (Waveform Audio), AVI (Audio Visual Interleaved), MP3 MPEG (Moving Picture Experts Group) (audio layer) 3, MIDI (Musical Instrument Digital Interface), WMA
เพจ	PDF (Acrobat Portable Document Format), HTML, XML (eXtension Markup Language)
ภาพเคลื่อนไหว	Quicktime, MPEG-2 (Moving Picture Experts Group –2), RealVideo, WMV

- ผังอักขระแอสกีที่ไม่แสดงผล

อักขระที่ไม่แสดงผลเหล่านี้ถูกใช้เป็นรหัสควบคุมการพิมพ์บนเครื่องพิมพ์ หรือใช้เป็นตัวแบ่งข้อมูลในสื่อบันทึกข้อมูลบางชนิด (เช่น เทป) อักขระตัวแทนที่ปรากฏในตารางเป็นเพียงการแสดงว่า ณ ตำแหน่งนั้นมีรหัสดังกล่าวอยู่ ไม่ใช่สัญลักษณ์ที่จะนำมาแสดงผลเป็นหลัก

ฐานสอง	ฐานสิบ	ฐานสิบหก	อักขระ	ความหมาย
0000 0000	0	00	(ว่าง)	NUL - null character
0000 0001	1	01	☺	SOH - start of heading
0000 0010	2	02	☻	STX - start of text
0000 0011	3	03	♥	ETX - end of text
0000 0100	4	04	♦	EOT - end of transmission
0000 0101	5	05	♣	ENQ - enquiry
0000 0110	6	06	♠	ACK - acknowledge
0000 0111	7	07	•	BEL - bell
0000 1000	8	08	◘	BS - backspace
0000 1001	9	09	○	HT - horizontal tabulation
0000 1010	10	0A	◙	LF - line feed
0000 1011	11	0B	♂	VT - vertical tabulation
0000 1100	12	0C	♀	FF - form feed
0000 1101	13	0D	♪	CR - carriage return
0000 1110	14	0E	♫	SO - shift out
0000 1111	15	0F	☼	SI - shift in

ภาพที่ 2.3 ผังอักขระแอสกีที่ไม่แสดงผล

ฐานสอง	ฐานสิบ	ฐานสิบหก	อักขระ	ความหมาย
0001 0000	16	10	►	DLE - data link escape
0001 0001	17	11	◄	DC1 - device control one
0001 0010	18	12	↕	DC2 - device control two
0001 0011	19	13	‼	DC3 - device control three
0001 0100	20	14	¶	DC4 - device control four
0001 0101	21	15	§	NAK - negative acknowledge
0001 0110	22	16	▬	SYN - synchronous idle
0001 0111	23	17	↨	ETB - end of transmission block
0001 1000	24	18	↑	CAN - cancel
0001 1001	25	19	↓	EM - end of medium
0001 1010	26	1A	→	SUB - substitute
0001 1011	27	1B	←	ESC - escape
0001 1100	28	1C	∟	FS - file separator
0001 1101	29	1D	↔	GS - group separator
0001 1110	30	1E	▲	RS - record separator
0001 1111	31	1F	▼	US - unit separator
0111 1111	127	7F	⌂	DEL - delete

ภาพที่ 2.3 ผังอักขระแอสกีที่ไม่แสดงผล (ต่อ)

- ผังอักขระแอสกีที่แสดงผล

ฐานสอง	ฐานสิบ	ฐานสิบหก	อักขระ	ฐานสอง	ฐานสิบ	ฐานสิบหก	อักขระ	ฐานสอง	ฐานสิบ	ฐานสิบหก	อักขระ
0010 0000	32	20	(ช่องว่าง)	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(	0100 1000	72	48	H	0110 1000	104	68	h
0010 1001	41	29	)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	,	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u
0011 0110	54	36	6	0101 0110	86	56	V	0111 0110	118	76	v
0011 0111	55	37	7	0101 0111	87	57	W	0111 0111	119	77	w
0011 1000	56	38	8	0101 1000	88	58	X	0111 1000	120	78	x
0011 1001	57	39	9	0101 1001	89	59	Y	0111 1001	121	79	y
0011 1010	58	3A	:	0101 1010	90	5A	Z	0111 1010	122	7A	z
0011 1011	59	3B	;	0101 1011	91	5B	[	0111 1011	123	7B	{
0011 1100	60	3C	<	0101 1100	92	5C	\	0111 1100	124	7C	|
0011 1101	61	3D	=	0101 1101	93	5D	]	0111 1101	125	7D	}
0011 1110	62	3E	>	0101 1110	94	5E	^	0111 1110	126	7E	~
0011 1111	63	3F	?	0101 1111	95	5F	_				

ภาพที่ 2.4 ผังอักขระแอสกีที่แสดงผล

2) EBCDIC (เอบซีดิก : EBCDIC : Extended Binary Coded Decimal Interchange Code)

เป็นรหัสสับเปลี่ยนเลขฐานสิบเข้ารหัสฐานสองแบบขยาย เป็นรหัสอักขระ 8 บิตที่พัฒนาโดยบริษัท IBM ซึ่งพัฒนาสำหรับระบบปฏิบัติการขนาดใหญ่ โดยเป็นรหัสสำหรับไฟล์ข้อความที่ใช้กับระบบปฏิบัติการ IBM OS-390 สำหรับเครื่องแม่ข่าย S/390 และบริษัทจำนวนมากใช้กับโปรแกรมประยุกต์ legacy application และฐานข้อมูล ในไฟล์เอบซีดิก ตัวอักษรพยัญชนะและตัวเลขได้รับการนำเสนอเป็นเลขฐานสอง 8 บิต (8 ตัวอักษรของ 0 และ 1) ทำให้สามารถสร้างรหัสได้ 256 รหัส (2 ยกกำลัง 8) ได้แก่ ตัวพยัญชนะ ตัวเลข และเครื่องหมายพิเศษ แต่ในปัจจุบันนิยมใช้รหัสแอสกี (ASCII) มากกว่า

3) UNICODE (ยูนิโค้ด)

ยูนิโค้ด คือมาตรฐานอุตสาหกรรมที่ช่วยให้คอมพิวเตอร์แสดงผลและจัดการข้อความธรรมดาที่ใช้ในระบบการเขียนของภาษาส่วนใหญ่ในโลกได้อย่างสอดคล้องกัน ยูนิโค้ดประกอบด้วยรายการอักขระที่แสดงผลได้มากกว่า 100,000 ตัว พัฒนาต่อยอดมาจากมาตรฐานชุดอักขระสากล (Universal Character Set: UCS) และมีการตีพิมพ์ลงในหนังสือ The Unicode Standard เป็นแผนผังรหัสเพื่อใช้เป็นรายการอ้างอิง นอกจากนั้นยังมีการอธิบายวิธีการที่ใช้เข้ารหัสและการนำเสนอมาตรฐานของการเข้ารหัสอักขระอีกจำนวนหนึ่ง การเรียงลำดับอักษร กฎเกณฑ์ของการรวมและการแยกอักขระ รวมไปถึงลำดับการแสดงผลของอักขระสองทิศทาง (เช่นอักษรอาหรับหรืออักษรฮีบรูที่เขียนจากขวาไปซ้าย)

ยูนิโค้ดคอนซอร์เทียม (Unicode Consortium) ซึ่งเป็นองค์กรไม่แสวงหาผลกำไร เป็นผู้รับผิดชอบในการพัฒนายูนิโค้ด องค์กรนี้มีจุดมุ่งหมายเกี่ยวกับการแทนที่การเข้ารหัสอักขระที่มีอยู่ด้วยยูนิโค้ดและมาตรฐานรูปแบบการแปลงยูนิโค้ด (Unicode Transformation Format: UTF) แต่ก็เป็นที่ยุ่งยากเนื่องจากแผนการที่มีอยู่ถูกจำกัดไว้ด้วยขนาดและขอบเขต ซึ่งอาจไม่รองรับกับสภาพแวดล้อมหลายภาษาในคอมพิวเตอร์

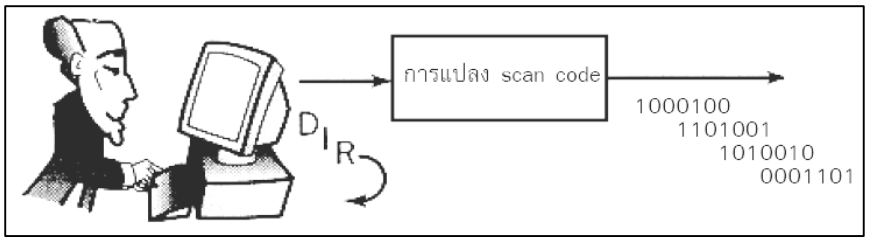
ความสำเร็จของยูนิโค้ดคือการรวมรหัสอักขระหลายชนิดให้เป็นหนึ่งเดียว นำไปสู่การใช้งานอย่างกว้างขวางและมีอิทธิพลต่อการแปลภาษาของซอฟต์แวร์คอมพิวเตอร์ นั่นคือโปรแกรมจะสามารถใช้ได้หลายภาษา มาตรฐานนี้มีการนำไปใช้เป็นเทคโนโลยีหลักหลายอย่าง อาทิ เอกซ์เอ็มแอล ภาษาจาวา ดอตเน็ตเฟรมเวิร์ก และระบบปฏิบัติการสมัยใหม่

ยูนิโค้ดสามารถนำไปใช้งานได้ด้วยชุดอักขระแบบต่าง ๆ ชุดอักขระที่เป็นที่รู้จักมากที่สุดคือ UTF-8 (ใช้ 1 ไบต์สำหรับอักขระทุกตัวในรหัสแอสกีและมีค่ารหัสเหมือนกับมาตรฐานแอสกี หรือมากกว่านั้นจนถึง 4 ไบต์สำหรับอักขระแบบอื่น) UCS-2 ซึ่งปัจจุบันเลิกใช้แล้ว (ใช้ 2 ไบต์สำหรับอักขระทุกตัว แต่ไม่ครอบคลุมอักขระทั้งหมดในยูนิโค้ด) และ UTF-16 (เป็นส่วนขยายจาก UCS-2 โดยใช้ 4 ไบต์สำหรับแทนรหัสอักขระที่ขาดไปของ UCS-2)

2.3 การเก็บข้อมูลในคอมพิวเตอร์

2.3.1 การใส่ตัวอักษรจากคีย์บอร์ด

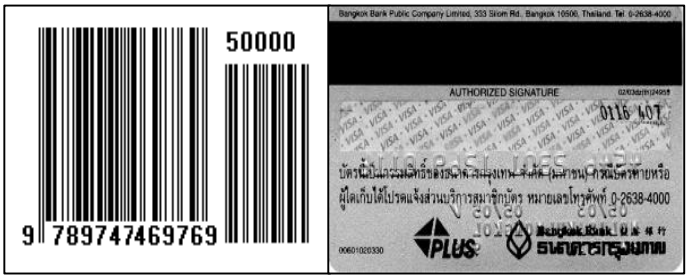
ตัวอักษรที่ใส่ผ่านคีย์บอร์ดจะถูกแปลง scan code แล้วส่งเข้าไป สมมุติว่าผู้ใช้งานคีย์ตัวอักษร 3 ตัว “D”, “I”, “R” แล้วกดคีย์ Enter คอมพิวเตอร์จะแปลง scan code 4 ตัวเป็นโค้ด ASCII ฐานสองเป็น 1000100, 1001001, 1010010, 0001101



ภาพที่ 2.5 การแปลง Scan Code

2.3.2 การใส่ตัวอักษรจากแหล่งอื่น

ได้แก่ OCR (Optical Character Recognition), Barcode Reader, Magnetic Stripe Reader, Voice Input



ภาพที่ 2.6 Barcode Reader และ Magnetic Stripe Reader
ที่มา (น.ท.ไพศาล โมลิสกุลมงคล และคณะ สถาปัตยกรรมคอมพิวเตอร์, 2547)

2.3.3 เลขจำนวนเต็ม

ในระบบเลขฐานสอง สามารถแสดงค่าโดยใช้เลขเพียง 2 ตัวคือ 0 และ 1 โดยอาจจะใช้เครื่องหมาย + และ - รวมทั้งแทนจำนวนทศนิยมได้ เช่น

1000011 = 35 หรือ -1011.0110 = -11.375

แต่ในคอมพิวเตอร์จะรู้จักเฉพาะเลข 0 และ 1 ไม่สามารถใช้ประโยชน์จากจุดได้ แต่สามารถใช้เครื่องหมายลบโดยใช้เงื่อนไขเพิ่มเติม การเก็บข้อมูลจะเป็นการเก็บค่าในรีจิสเตอร์ เช่น ถ้าเป็นขนาด 8 บิต จะได้

00000000 = 0 00000001 = 1

01000001 = 65 10000001 = 129

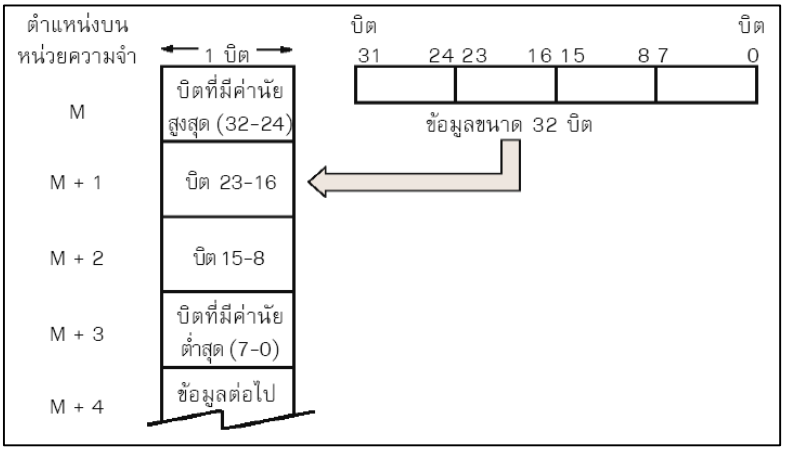
10000011 = 131 11111111 = 255

1) เลขจำนวนเต็มไม่มีเครื่องหมาย

การเก็บข้อมูลในคอมพิวเตอร์ที่เป็นเลขจำนวนเต็มไม่มีเครื่องหมายจะเก็บเลขฐานสอง ตามจำนวนบิตที่กำหนด เช่น

8 บิต = 256 ค่า (0-255) 16 บิต = 65,536 (0–65,535)

16 บิต = 4,294,967,296 (0–4,294,967,295)



ภาพที่ 2.7 การเก็บข้อมูลเลขจำนวนเต็มไม่มีเครื่องหมาย
ที่มา (น.ท.ไพศาล โมลิสกุลมงคล และคณะ สถาปัตยกรรมคอมพิวเตอร์, 2547)

2) เลขจำนวนเต็มที่มีเครื่องหมาย

Sign-and-magnitude : คล้ายเลขจำนวนเต็มทั่วไป แต่บิตซ้ายสุดเป็นบิตเครื่องหมาย 0 หมายถึงจำนวนเต็มบวก และ 1 เป็นจำนวนเต็มลบ

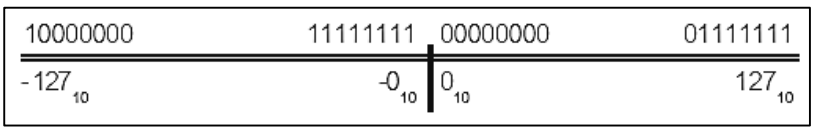
00110101 = +53

10110101 = -53

1’s complement : เกิดจากนำเลขจำนวนเต็มฐานสองจำนวนนั้นลบออกจากค่าที่เป็น 1 ทุกบิต เช่น 1’s complement ของ 10110101 คือ

11111111 – 10110101 = 01001010 หรือ สลับค่าทุกบิตเป็นค่าตรงข้าม

10110101 ---> 01001010

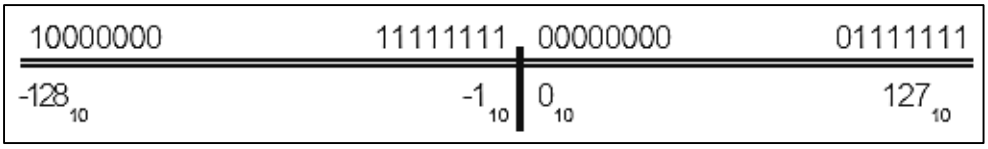


ภาพที่ 2.8 การเก็บข้อมูลเลขจำนวนเต็มที่มีเครื่องหมาย แบบ 1’s complement

2’s complement : เกิดจากบวก 1 เข้ากับบิตขวาสุดของ 1’s complement เช่น หาค่า 2’s complement ของ 10110100 คือ

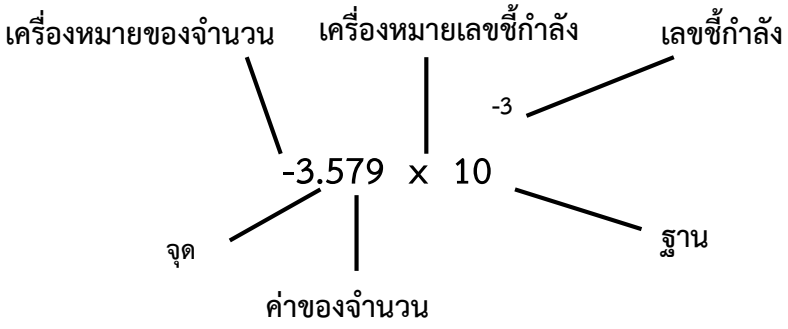
ค่า 1’s complement ของ 10110100 = 01001011

ค่า 2’s complement ของ 10110100 = 01001011 + 1 = 01001100



ภาพที่ 2.9 การเก็บข้อมูลเลขจำนวนเต็มที่มีเครื่องหมาย แบบ 2’s complement

2.3.4 เลขทศนิยม



เลขทศนิยม : จำนวนเต็มที่เป็นมาตรฐาน 1 word ประกอบด้วยเลข 7 ตัวและเครื่องหมายอีก 1 ตัว รูปแบบจะเป็น SMMMMMMM

โดยที่ S คือ Sign ซึ่งเป็นบิตที่เก็บเครื่องหมาย

M คือ Mantissa ซึ่งเป็นบิตค่าของจำนวนเต็ม

รูปแบบนี้สามารถเก็บค่าจำนวนเต็มในช่วง –9,999,999 < I < 99,999,999

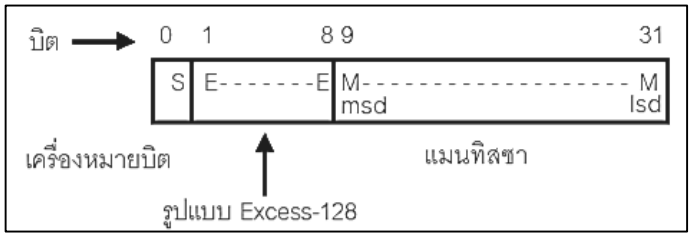
เลขทศนิยมที่มีเครื่องหมาย : จะมีบิตเครื่องหมาย 2 บิต รูปแบบจะเป็น SEEMMMMM

โดยที่ S คือ Sign ซึ่งเป็นบิตที่เก็บเครื่องหมาย

E คือ Exponent ซึ่งเป็นบิตที่เก็บเลขชี้กำลัง

M คือ Mantissa ซึ่งเป็นบิตค่าของจำนวนทศนิยม

เลขทศนิยมในคอมพิวเตอร์ : นำรูปแบบเลขทศนิยมที่มีเครื่องหมายมาประยุกต์ใช้งาน ถ้าใช้ขนาด 32 บิต จะใช้บิตเครื่องหมาย 1 บิต, เลขชี้กำลัง 8 บิต (ใช้ excess-128 ในช่วง 10^{-128} ถึง 10^{+127}) และค่าของจำนวนใช้ 23 บิต (ระหว่าง 10^{-38} ถึง 10^{+38}) รูปแบบจะเป็น



ภาพที่ 2.10 การเก็บข้อมูลเลขทศนิยมในคอมพิวเตอร์

2.4 ระบบเลขฐาน

ระบบตัวเลขเกิดจากการนับจำนวนสิ่งของต่าง ๆ และมีการพัฒนาโดยใช้เครื่องหมายหรือสัญลักษณ์แทนจำนวนที่นับได้ โดยทั่วไปใช้ระบบตัวเลขฐานสิบ โดยมีตัวเลข 10 ตัว (0-9) หลักการพื้นฐานนี้เองสามารถประยุกต์ใช้กับเลขฐานสอง (Binary) ฐานแปด (Octal) และฐานสิบหก (Hexadecimal) ได้อย่างง่ายดาย

2.4.1 ระบบเลขฐานสิบ (Decimal Number)

ประกอบด้วยสัญลักษณ์ 10 สัญลักษณ์ คือ เลข 0 ถึง 9 ที่เรียกว่า ตัวเลขฐานสิบ (Decimal Digit) ลำดับของตัวเลขแต่ละตัวจะมีค่าน้ำหนักที่แน่นอนเป็นตัวคูณ ซึ่งตัวคูณนี้จะเป็นลักษณะเลขยกกำลังของ 10 เช่น หลักหน่วยจะมีตัวคูณเป็น 10^0 หลักสิบมีตัวคูณเป็น 10^1 หลักร้อยมีตัวคูณเป็น 10^2 เช่นนี้เรื่อยไป ดังนั้นเราสามารถกระจายจำนวนเลขที่เป็นฐานสิบได้ เช่น

$$\begin{aligned} 4512 &= (4 \times 10^3) + (5 \times 10^2) + (1 \times 10^1) + (2 \times 10^0) \\ &= (4 \times 1000) + (5 \times 100) + (1 \times 10) + (2 \times 1) \\ &= 4000 + 500 + 10 + 2 \\ &= 4512 \end{aligned}$$

จะเห็นว่าค่าตัวคูณที่เป็นเลขยกกำลังของ 10 นั้น จากซ้ายไปขวาค่าจะลดลงจากกำลังสูงสุดไปจนถึง 0 ซึ่งเป็นค่าของหลักหน่วยนั่นเอง ในทำนองเดียวกันกับเลขที่เป็นจำนวนทศนิยมค่าตัวคูณจะลดลงจากซ้ายไปขวาจาก 10^{-1}, 10^{-2}, 10^{-3} เรื่อยไป เช่น

$$\begin{aligned} 4512.642 &= (4 \times 10^3) + (5 \times 10^2) + (1 \times 10^1) + (2 \times 10^0) + (6 \times 10^{-1}) + (4 \times 10^{-2}) + (2 \times 10^{-3}) \\ &= (4 \times 1000) + (5\times 100) + (1 \times 10) + (2 \times 1) + (6 \times 0.1) + (4 \times 0.01) + (2 \times 0.001) \\ &= 4000 + 500 + 10 + 2 + 0.6 + 0.04 + 0.002 \\ &= 4512.642 \end{aligned}$$

2.4.2 ระบบเลขฐานสอง (Binary Number)

เลขฐานสองเป็นระบบที่คอมพิวเตอร์รู้จักดี เนื่องจากในสภาวะทั่วไปทางไฟฟ้าหรือทางตรรกศาสตร์จะมีสภาวะมาตรฐานสองสภาวะคือ High (สูง) กับ Low (ต่ำ) หรือ On (เปิด) กับ Off (ปิด) หรือ True (จริง) กับ False (เท็จ) ทำให้สัญลักษณ์ที่ใช้กับเลขฐานสองมีสองตัวคือ 0 และ 1

ตัวอย่างที่ 2.1 จงแปลง 10011001_2 ให้เป็นเลขฐานสิบ

$$10011001_2 = (1\times 2^7) + (0 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$$

$$= 128 + 0 + 0 + 16 + 8 + 0 + 0 + 1$$

$$= 153_{10}$$

$$\therefore\ 10011001_2 = 153_{10}$$

สำหรับเลขฐานสองที่เป็นเลขทศนิยม ค่าตัวคูณก็เริ่มจาก 2^{-1}, 2^{-2}, 2^{-3},.... จากซ้ายไปขวาเริ่มจากบิตที่อยู่ถัดจากจุดทศนิยม

ตัวอย่างที่ 2.2 จงแปลง 1101.101_2 ให้เป็นเลขฐานสิบ

$$1101.101_2 = (1\times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$$

$$= (1\times 8) + (1 \times 4) + (0 \times 2) + (1 \times 1) + (1 \times \frac{1}{2}) + (0 \times \frac{1}{4}) + (1 \times \frac{1}{8})$$

$$= 8 + 4 + 0 + 1 + 0.5 + 0 + 0.125$$

$$= 13.625_{10}$$

$$\therefore\ 1101.101_2 = 13.625_{10}$$

1) การบวกเลขฐานสอง

การบวกเลขฐานสองก็คล้ายกับการบวกเลขฐานสิบ แต่จะมีค่า 0 กับ 1 เท่านั้น หลักการบวกเป็นดังนี้

ตารางที่ 2.2 หลักการบวกเลขฐานสอง

ตัวตั้ง + ตัวบวก	ผลลัพธ์
0 + 0	0
0 + 1	1
1 + 0	1
1 + 1	1 0 (ผลลัพธ์ = 0 , ตัวทด = 1)
1 + 1 + 1	1 + 1 = 1 0 , 1 + 1 0 = 1 1 (ผลลัพธ์ = 1 , ตัวทด = 1)

ตัวอย่างที่ 2.3 $1011001_2 + 1101001_2 = ?$

$1\ 0\ 1\ 1\ 0\ 0\ 1_2$ ซึ่งมีค่าเป็น 89_{10}

\+ \+

$1\ 1\ 0\ 1\ 0\ 0\ 1_2$ 105_{10}

$1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1_2$ 94_{10}

$\therefore\ 1011001_2 + 1101001_2 = 94_{10}$

ตัวอย่างที่ 2.4 $1010.0_2 + 0011.1_2 = ?$

$1\ 0\ 1\ 0\ .\ 0_2$ ซึ่งมีค่าเป็น 10_{10}

\+ \+

$0\ 0\ 1\ 1\ .\ 1_2$ $3\ .\ 5_{10}$

$1\ 1\ 0\ 1\ .\ 1_2$ $13\ .\ 5_{10}$

$\therefore\ 1010.0_2 + 0011.1_2 = 13.5_{10}$

2) การลบเลขฐานสอง

การลบเลขฐานสองก็คล้ายกับการลบเลขฐานสิบ แต่จะมีค่า 0 กับ 1 เท่านั้น หลักการลบเป็นดังนี้

ตารางที่ 2.3 หลักการลบเลขฐานสอง

ตัวตั้ง - ตัวลบ	ผลลัพธ์
0 - 0	0
0 - 1	1 (เกิดจากการขอยืมจากบิตที่มีค่านัยสำคัญสูงกว่ามา 1 ค่าที่ยืมมาจะเป็น 2)
1 - 0	1
1 - 1	0

ตัวอย่างที่ 2.5 $1011_2 - 101_2 = ?$

ตัวยืม 2

$1\ 0\ 1\ 1_2$ ซึ่งมีค่าเป็น 11_{10}

– –

$0\ 1\ 0\ 1_2$ 5_{10}

$0\ 1\ 1\ 0_2$ 6_{10}

$\therefore\ 1011_2 - 101_2 = 6_{10}$

ตัวอย่างที่ 2.6 $111.01_2 - 100.10_2 = ?$

ตัวยืม 2

$1\ 1\ 1\ .\ 0\ 1_2$ ซึ่งมีค่าเป็น $7\ .\ 25_{10}$

– –

$1\ 0\ 0\ .\ 1\ 0_2$ $4\ .\ 5_{10}$

$0\ 1\ 0\ .\ 1\ 1_2$ $2\ .\ 75_{10}$

$\therefore\ 111.01_2 - 100.10_2 = 2.75_{10}$

2.4.3 ระบบเลขฐานแปด (Octal Number)

ระบบเลขฐานแปดจะประกอบด้วยตัวเลข 0 , 1 , 2 , 3 , 4 , 5 , 6 และ 7 เลขฐานแปดมีโครงสร้างเช่นเดียวกับเลขฐานสิบและเลขฐานสอง คือ ตัวเลขแต่ละตัวจะมีค่าตัวคูณในลักษณะเลขยกกำลังของเลข 8 เช่น 8^0 , 8^1 , 8^2 , ... ส่วนค่าของเลขฐานแปดที่เป็นทศนิยมจะมีค่าตัวคูณเป็นเลขยกกำลังของค่าติดลบเริ่มจาก 8^{-1} , 8^{-2} , ... ค่าที่ออกมาเมื่อนำมารวมกันจะเป็นค่าของเลขฐานสิบ

ตัวอย่างที่ 2.7 จงแปลง 1234.56_8 ให้เป็นเลขฐานสิบ

$1234.56_8 = (1 \times 8^3) + (2 \times 8^2) + (3 \times 8^1) + (4 \times 8^0) + (5 \times 8^{-1}) + (6 \times 8^{-2})$

$= 512 + 128 + 24 + 4 + 0.625 + 0.09375$

$= 668.71875_{10}$

$\therefore 1234.56_8 = 668.71875_{10}$

1) การบวกเลขฐานแปด

การบวกเลขฐานแปดคล้ายกับการบวกเลขฐานสิบ ในกรณีนี้ถ้าผลบวกไม่เกิน 7 ให้ใส่ค่านั้นได้ทันที แต่ถ้าเกิน 7 ให้ลบออกด้วย 8 (ค่าฐาน) ก่อน แล้วค่าที่เหลือใส่ได้ทันที แล้วทดไป 1 (ถ้าบวกกันหลายจำนวน จำนวนครั้งที่ลบด้วย 8 จะได้ 1 ถ้าลบหลายครั้งก็จะทดได้ตามจำนวนนั้น เช่น ถ้าบวกกันได้ 19 เมื่อลบ 8 สองครั้งจะเหลือเศษ 3 ดังนั้นจึงใส่ 3 ที่ตำแหน่งนั้น แล้วเหลือทดไป 2 นั่นเอง)

ตัวอย่างที่ 2.8 $235_8 + 76_8 = ?$

ตัวทด 1 1

235_8 (5+6 = 11 ; 11-8 = 3 ; ใส่ 3 ทด 1)

\+ (3+7 = 10+ทด 1 = 11 ; 11-8 =3 ; ใส่ 3 ทด 1)

76_8 (2+ทด 1 = 3)

333_8

$\therefore 235_8 + 76_8 = 333_8$

2) การลบเลขฐานแปด

การลบเลขฐานแปดก็เช่นเดียวกับเลขฐานสิบ ถ้าตัวตั้งมีค่าน้อยกว่าตัวลบก็ต้องยืมมาจากตัวเลขข้างหน้า การยืมมา 1 จะมีค่าเท่ากับ 8 ในหลักนั้น รวมกับที่มีของเดิมก่อนแล้วลบออกมา

ตัวอย่างที่ 2.9 $235_8 - 76_8 = ?$

ตัวยืม 8

235_8 (5-6 ไม่ได้ 5 ไปยืม 3 มาได้ 8 ; 5+8 = 13 ; 13-6 = 7)

\- (3 ถูก 5 ยืมไป เหลือ 2 ; 2-7 ไม่ได้ ไปยืม 2 มาได้ 8 ; 2+8 = 10 10-7 = 3)

76_8

137_8 (2 ถูก 3 ยืมไป เหลือ 1)

$\therefore\ 235_8 - 76_8 = 137_8$

2.4.4 ระบบเลขฐานสิบหก (Hexadecimal Number)

เลขฐานสิบหกนำมาใช้งานเพื่อลดข้อยุ่งยากสำหรับเลขฐานสอง ทำให้สะดวกในการใช้งาน เลขฐานสิบหกประกอบด้วย 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , A , B , C , D , E , F

โครงสร้างเลขฐานสิบหก คือ ตัวเลขแต่ละตัวจะมีค่าตัวคูณในลักษณะเลขยกกำลังของเลข 16 เช่น 16^0 , 16^1 , 16^2 , ... ส่วนค่าของเลขฐานสิบหกที่เป็นทศนิยมจะมีค่าตัวคูณเป็นเลขยกกำลังของค่าติดลบ เริ่มจาก 16^{-1} , 16^{-2} , ... ค่าที่ออกมาเมื่อนำมารวมกันจะเป็นค่าของเลขฐานสิบ

<u>ตัวอย่างที่ 2.10</u> จงแปลง 1234.56_{16} ให้เป็นเลขฐานสิบ

$$1234.56_{16} = (1 \times 16^3) + (2 \times 16^2) + (3 \times 16^1) + (4 \times 16^0) + (5 \times 16^{-1}) + (6 \times 16^{-2})$$

$$= 4096 + 512 + 48 + 4 + 0.3125 + 0.0234375$$

$$= 4660.3359375_{10}$$

$\therefore\ 1234.56_{16} = 4660.3359375_{10}$

1) การบวกเลขฐานสิบหก

ถ้าการบวกได้ค่าเกิน 15 ให้เอาเลขฐาน (16) ลบออก แล้วได้ผลลัพธ์เป็นเศษที่เหลือ พร้อมกับได้ตัวทดเป็น 1 (ถ้าลบด้วย 16 หลายครั้งก็จะได้ตัวทดตามจำนวนครั้งที่ลบนั่นเอง) ถ้าเลขฐานสิบหกนั้นเป็นจุดทศนิยมก็ตั้งตำแหน่งจุดให้ตรงกันก่อน

<u>ตัวอย่างที่ 2.11</u> $7A35.6F_{16} + 36B3.36_{16} = ?$

ตัวทด 1 1

7 A 3 5 . 6 F$_{16}$ (F=15 ; 15+6=21 ; 21-16=5 ; ใส่ 5 ทด 1)

\+ (6+3=9 + ทด1 = 10 ; ใส่ A) (5+3=8)

<u>3 6 B 3 . 3 6$_{16}$</u> (3+B=3+11=14 ; ใส่ E)

<u>B 0 E 8 . A 5$_{16}$</u> (A=10;10+6=16;16-16=0;ใส่ 0 ทด 1) (7+3=10+ทด1=B)

$\therefore\ 7A35.6F_{16} + 36B3.36_{16} = B0E8.A5_{16}$

2) การลบเลขฐานสิบหก

ถ้าตัวตั้งน้อยกว่าตัวลบก็ขอยืมจากหลักข้างหน้า หากยืมมา 1 จะได้ค่าเป็น 16 บวกเข้ากับค่าที่ตำแหน่งเดิม แล้วจึงลบ และถ้าเลขฐานสิบหกนั้นเป็นจำนวนทศนิยมก็ต้องตั้งตำแหน่งจุดทศนิยมให้ตรงกัน

ตัวอย่างที่ 2.12 $853D.53_{16} - 6F18.A2_{16} = ?$

ตัวยืม 16 16

8 5 3 D . 5 3 (3-2=1) (5-A ไม่ได้ ยืม D มาได้ 16 ; 16+5=21 ; 21-A=B)

- (D ถูก 5 ยืมไปเหลือ C ; C-8=4) (3-1=2)

6 F 1 8 . A 2 (5-F ไม่ได้ ยืม 8 มาได้ 16 ; 16+5=21 ; 21-F=6)

1 6 2 4 . B 1 (8 ถูก 5 ยืมไปเหลือ 7 ; 7-6=1)

$\therefore 853D.53_{16} - 6F18.A2_{16} = 1624.B1_{16}$

2.4.5 การแปลงระหว่างเลขฐาน

1) การแปลงระหว่างเลขฐานสองเป็นเลขฐานสิบ

ทำได้โดยการใช้ค่าฐานสองยกกำลังตำแหน่งน้ำหนักแต่ละหลัก แล้วนำมารวมกัน

ตัวอย่างที่ 2.13 จงแปลง 110011.1011_2 ให้เป็นเลขฐานสิบ

$110011 = (1 \times 2^5) + (1 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$

$= (1 \times 32) + (1 \times 16) + (0 \times 8) + (0 \times 4) + (1 \times 2) + (1 \times 1)$

$= 32 + 16 + 0 + 0 + 2 + 1 = 51$

$0.1011 = (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) + (1 \times 2^{-4})$

$= (1 \times 1/2) + (0 \times 1/4) + (1 \times 1/8) + (1 \times 1/16)$

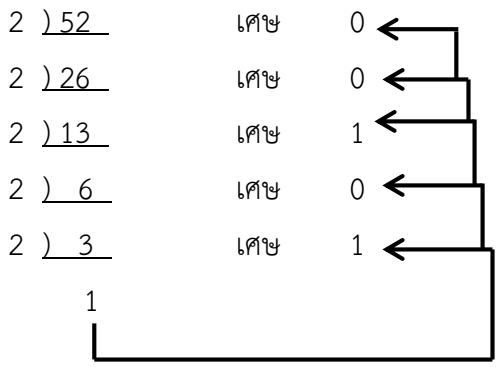
$= 0.5 + 0 + 0.125 + 0.0625 = 0.6875$

$\therefore 110011.1011_2 = 51.6875_{10}$

2) การแปลงระหว่างเลขฐานสิบเป็นเลขฐานสอง

ทำได้โดยการหารเลขฐานสิบด้วย 2 ของเลขฐานสิบ และนำผลลัพธ์ที่เป็นเศษมาจัดเรียงตำแหน่งเป็นเลขฐานสอง

ตัวอย่างที่ 2.14 จงแปลง 52_{10} ให้แปลงเป็นเลขฐานสอง



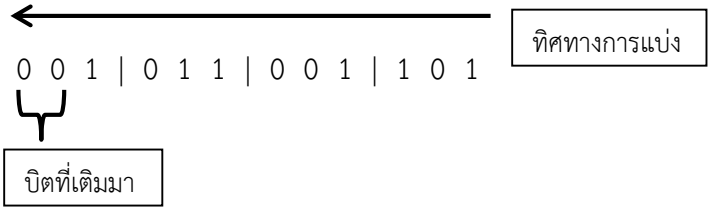
$\therefore\ 52_{10} = 110100_{2}$

3) การแปลงเลขฐานสองเป็นเลขฐานแปดหรือฐานสิบหก

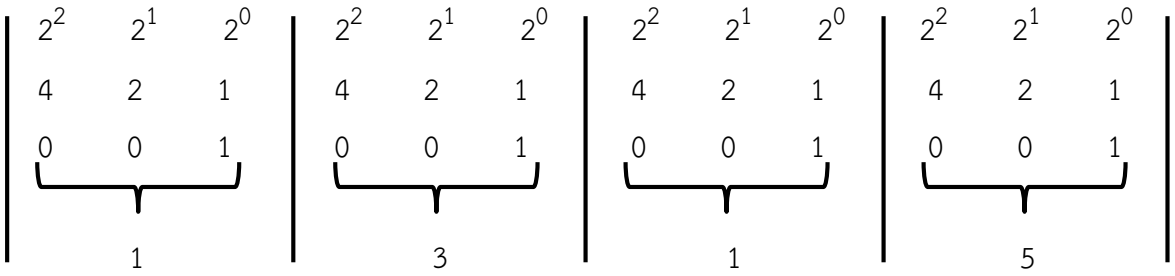
เนื่องจากเลขฐานแปดหรือเลขฐานสิบหกเป็นชุดของเลขฐานสอง กล่าวคือ เลขฐานแปดจะประกอบด้วยเลขฐานสอง 3 ตัว ($8 = 2^3 = 3$ บิต) และเลขฐานสิบหกประกอบด้วยเลขฐานสอง 4 ตัว ($16 = 2^4 = 4$ บิต) โดยนับจากตัวที่มีค่านัยสำคัญต่ำสุด (หรือจากจุดทศนิยม) เลื่อนไปทางซ้ายหรือขวา

ตัวอย่างที่ 2.15 จงแปลง 1011001101_{2} เป็นเลขฐานแปด

ขั้นที่ 1 แบ่งเลขฐานสองออกเป็นกลุ่ม กลุ่มละ 3 บิต ถ้าไม่ครบ 3 บิต สามารถเติมเลขศูนย์ให้ครบ 3 บิตได้

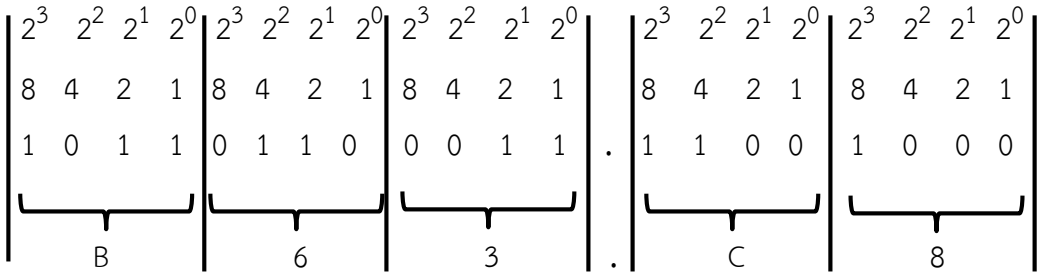


ขั้นที่ 2 แทนค่าเลขฐานแปด



$\therefore\ 1011001101_{2} = 1315_{8}$

ตัวอย่างที่ 2.16 จงแปลง 101101100011.11001000_2 เป็นเลขฐานสิบหก

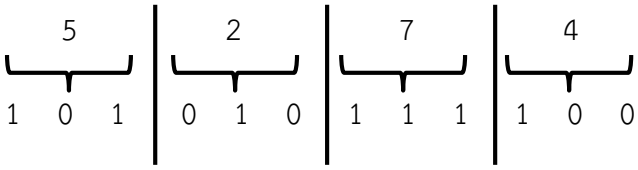


$\therefore$ $101101100011.11001000_2 = B63.C8_{16}$

4) การแปลงเลขฐานแปดหรือเลขฐานสิบหกเป็นเลขฐานสอง

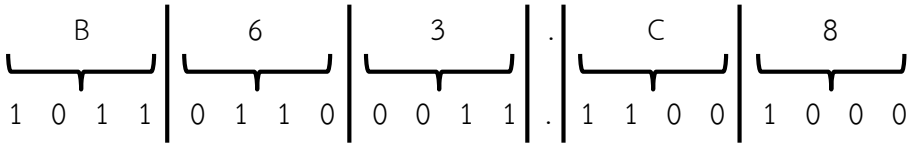
ถ้าต้องการแปลงเลขฐานแปดหรือเลขฐานสิบหกเป็นเลขฐานสอง ให้กระจายเลขฐานแปดหรือเลขฐานสิบหกเป็นเลขฐานสอง โดยจะต้องใส่ให้ครบจำนวนบิตของเลขฐานที่เราต้องการ โดยเลขฐานแปดจะประกอบด้วยเลขฐานสอง 3 ตัว ($8 = 2^3 = 3$ บิต) และเลขฐานสิบหกประกอบด้วยเลขฐานสอง 4 ตัว ($16 = 2^4 = 4$ บิต)

ตัวอย่างที่ 2.17 จงแปลง 5274_8 เป็นเลขฐานสอง



$\therefore$ $5274_8 = 101010111100_2$

ตัวอย่างที่ 2.18 จงแปลง $B63.C8_{16}$ เป็นเลขฐานสอง



$\therefore$ $B63.C8_{16} = 101101100011.11001000_2$

5) การแปลงเลขฐานแปดเป็นเลขฐานสิบ

ใช้หลักการ คือ ใช้การกระจายที่มีตัวคูณในลักษณะของเลขยกกำลังของ 8 คูณด้วยตัวเลขของหลักนั้น แล้วนำค่ามารวมกัน

ตัวอย่างที่ 2.19 จงแปลง 1234.56_8 แปลงเป็นเลขฐานสิบ

$$1234.56_8 = (1 \times 8^3) + (2 \times 8^2) + (3 \times 8^1) + (4 \times 8^0) + (5 \times 8^{-1}) + (6 \times 8^{-2})$$

$$= (1 \times 512) + (2 \times 64) + (3 \times 8) + (4 \times 1) + (5 \times \frac{1}{8}) + (6 \times \frac{1}{64})$$

$$= 512 + 128 + 24 + 4 + 0.625 + 0.09375$$

$$= 668.71875_{10}$$

$\therefore$ $1234.56_8 = 668.71875_{10}$

6) การแปลงเลขฐานสิบเป็นเลขฐานแปด

โดยใช้เลขฐานสิบเป็นตัวตั้งหารแล้วใช้ 8 เป็นตัวหาร ค่าเศษที่เกิดขึ้นจะเป็นค่าบิตที่มีนัยสำคัญต่ำสุด แล้วนำค่าผลลัพธ์ไปเป็นตัวตั้งหารต่อไป ค่าเศษที่เกิดขึ้นจะมีค่านัยสำคัญเพิ่มขึ้น ทำเช่นนี้เรื่อยไปจนกว่าจะหมด ค่านัยสำคัญจะเป็นค่าสูงสุด

ตัวอย่างที่ 2.20 จงแปลง 668_{10} ให้เป็นเลขฐานแปด

8) 668 เศษ 4

8) 83 เศษ 3

8) 10 เศษ 2

1

$\therefore$ $668_{10} = 1234_8$

7) การแปลงเลขฐานสิบหกเป็นเลขฐานสิบ

สามารถทำได้โดยการกระจายที่มีตัวคูณในลักษณะเลขยกกำลังของ 16

ตัวอย่างที่ 2.21 จงแปลง 1234.56_{16} แปลงเป็นเลขฐานสิบ

$$1234.56_{16} = (1 \times 16^3) + (2 \times 16^2) + (3 \times 16^1) + (4 \times 16^0) + (5 \times 16^{-1}) + (6 \times 16^{-2})$$

$$= (1 \times 4096) + (2 \times 256) + (3 \times 16) + (4 \times 1) + (5 \times \frac{1}{16}) + (6 \times \frac{1}{256})$$

$$= 4096 + 512 + 48 + 4 + 0.3125 + 0.0234375$$

$$= 4660.3359375_{10}$$

$\therefore$ $1234.56_{16} = 4660.3359375_{10}$

8) การแปลงเลขฐานสิบเป็นเลขฐานสิบหก

สามารถใช้หลักการ โดยการตั้งหารที่มีตัวหารเป็น 16 ค่าเศษที่เกิดขึ้นจะเป็นค่าบิตที่มีนัยสำคัญต่ำสุด แล้วนำค่าผลลัพธ์ไปเป็นตัวตั้งหารต่อไป ค่าเศษที่เกิดขึ้นจะมีค่านัยสำคัญเพิ่มขึ้น ทำเช่นนี้เรื่อยไปจนกว่าจะหมด ค่านัยสำคัญจะเป็นค่าสูงสุด

ตัวอย่างที่ 2.22 จงแปลง 45661_{10} เป็นเลขฐานสิบหก

16) 45661 เศษ D

16) 2853 เศษ 5

16) 178 เศษ 2

16) 11 เศษ B

0

∴ $45661_{10} = B25D_{16}$

9) การแปลงเลขฐานแปดเป็นเลขฐานสิบหก

เนื่องจากเลขฐานแปดและเลขฐานสิบหกเป็นเลขที่เป็นชุดของเลขฐานสอง (เลขฐานแปดประกอบด้วยเลขฐานสอง 3 ตัว ส่วนเลขฐานสิบหกประกอบด้วยเลขฐานสอง 4 ตัว) ดังนั้นการแปลงเลขฐานแปดให้กลายเป็นเลขฐานสิบหก จะต้องแปลงให้เป็นเลขฐานสองก่อน หลังจากนั้นจัดกลุ่มใหม่เป็นกลุ่มละ 4 ตัว (บิต) เพื่อแปลงเป็นเลขฐานสิบหก

ตัวอย่าง 2.23 จงแปลง 5371_8 ให้เป็นเลขฐานสิบหก

ขั้นที่ 1 แปลงเลขฐานแปดให้เป็นเลขฐานสอง

5	3	7	1
1 0 1	0 1 1	1 1 1	0 0 1

ขั้นที่ 2 ทำการจัดกลุ่มใหม่เพื่อแปลงเป็นเลขฐานสิบหก

1 0 1 0	1 1 1 1	1 0 0 1
A	F	9

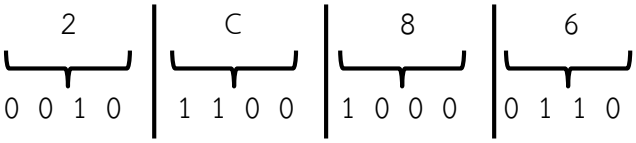
∴ $5371_8 = AF9_{16}$

10) การแปลงเลขฐานสิบหกเป็นเลขฐานแปด

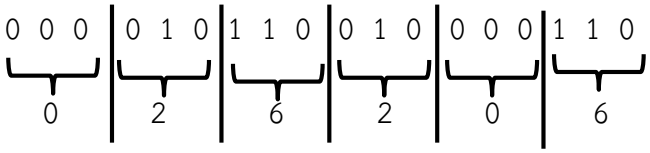
สามารถทำได้โดยแปลงเป็นเลขฐานสองก่อน (เลขฐานสิบหก 1 ตัว จะได้เลขฐานสอง 4 ตัว) หลังจากนั้นจัดกลุ่มใหม่เป็นกลุ่มละ 3 เพื่อแปลงเป็นเลขฐานแปด

<u>ตัวอย่างที่ 2.24</u> จงแปลง $2C86_{16}$ ให้เป็นเลขฐานแปด

ขั้นที่ 1 แปลงเลขฐานสิบหกให้เป็นเลขฐานสอง



ขั้นที่ 2 ทำการจัดกลุ่มใหม่เพื่อแปลงเป็นเลขฐานแปด



$\therefore\ 2C86_{16} = 26206_{8}$

2.4.6 เลขจำนวนเต็ม

ในระบบเลขฐานสอง สามารถแสดงค่าโดยใช้ตัวเลขเพียง 2 ตัว คือ 0 และ 1 โดยอาจจะมีเครื่องหมาย + และ - รวมทั้งมีจุด ตัวอย่างเช่น

$-1011.0110_{2} = -11.375_{10}$

แต่ในการเก็บข้อมูลและประมวลผลในคอมพิวเตอร์ ไม่สามารถใช้ประโยชน์จากจุดได้ แต่สามารถแสดงค่าติดลบได้โดยใช้วิธีการเพิ่มเติม ในการเก็บข้อมูลทั่วไปจะเป็นการเก็บค่า 0 และ 1 เช่น รีจิสเตอร์ที่ใช้เก็บข้อมูลขนาด 8 บิต สามารถเก็บค่าได้สูงสุด 256 ค่า เริ่มตั้งแต่ 0 - 255 เช่น

00000000 = 0

00000001 = 1

01000001 = 65

10000001 = 129

11111111 = 255

เราจะพิจารณาการเก็บข้อมูลเฉพาะค่า (Value) และเครื่องหมายในรูปแบบจำนวนเต็มที่ไม่มีเครื่องหมาย (Unsigned Integer), จำนวนเต็มที่มีเครื่องหมาย (Sign Integer) และ 2'Complement

1) การแสดงค่าจำนวนเต็มที่ไม่มีเครื่องหมาย (Unsigned Integer)

การแสดงค่าจำนวนเต็มแบบนี้เป็นการแสดงค่าเลขฐานสองโดยตรงเมื่อเปรียบเทียบกับเลขฐานสิบ ขอบเขตของจำนวนเต็มที่สามารถแสดงได้ก็ขึ้นอยู่กับจำนวนบิต เช่น รีจิสเตอร์ขนาด 8 บิตสามารถเก็บข้อมูลได้ 2^8 = 255 (ระหว่าง 0 - 255) เป็นต้น ถ้าต้องการขยายช่วงที่สามารถจัดเก็บข้อมูล ต้องขยายจำนวนบิต ซึ่งทำได้โดยการใช้ตำแหน่งเพื่อจัดเก็บข้อมูลที่มากกว่าเดิม

การใช้ตำแหน่งเพื่อจัดเก็บข้อมูลหลายตำแหน่งจะทำให้เกิดข้อยุ่งยากในการคำนวณ และจัดการกับข้อมูลเหล่านี้ เนื่องจากการคำนวณจะต้องดำเนินการให้เสร็จครั้งละ 1 ส่วน นอกจากนี้ยังอาจจะเกิดปัญหาตัวทด หรือ ตัวยืม ที่ทำการบวกหรือลบจำนวนเหล่านั้น ในคอมพิวเตอร์สมัยใหม่จะมีการนำคำสั่งเพื่อจัดการด้านนี้รวมมาให้ใช้งาน นอกจากนี้ยังมีซอฟท์แวร์ใหม่ที่ช่วยในการคำนวณในคอมพิวเตอร์ทำได้ง่าย และไม่เกิดข้อผิดพลาดขึ้นด้วย

2) การแสดงค่าจำนวนเต็มที่มีเครื่องหมาย (Signed Integer)

จำนวนเต็มที่ไม่มีเครื่องหมายสามารถแปลงเป็นเลขฐานสองได้โดยตรง แต่การเพิ่มเครื่องหมายให้กับเลขฐานสองเป็นเรื่องที่ยุ่งยาก เนื่องจากไม่มีวิธีการใดที่จะแสดงเลขฐานสองให้เป็นจำนวนลบได้ ในความเป็นจริงมี 2–3 วิธีที่จะแสดงค่าลบของเลขฐานสองได้ คือ วิธีการแสดงค่าแบบ Sign – and – Magnitude, วิธีการแสดงค่าแบบ 1’s Complement และ วิธีที่ใช้มากที่สุดคือ การแสดงด้วยค่า 2’s Complement

- การแสดงค่าแบบ Sign – and – Magnitude

สำหรับคอมพิวเตอร์ที่ถูกจำกัดให้เป็น 0 หรือ 1 เท่านั้น เราต้องพิจารณาให้ดีว่าจะเลือกบิตใดให้แสดงค่าบวก หรือลบตามต้องการ เช่น ถ้าเลือกบิตที่มีค่าบิตที่มีค่านัยสำคัญสูงสุด เป็นบิตเครื่องหมาย (บิตที่อยู่ซ้ายสุด) โดยกำหนดว่าถ้าบิตซ้ายสุดนี้มีค่าเป็นบวก บิตเครื่องหมายมีค่าเป็น 0 ถ้ามีค่าเป็นลบ บิตเครื่องหมายจะมีค่าเป็น 1 เช่น

	เลขฐานสอง (Binary)	เลขฐานสิบ (Decimal)
บิตเครื่องหมาย →	0 1011101	+ 93
	1 1011101	-93

ในการคำนวณเมื่อเราใช้ Sign – and – Magnitude จะเกิดข้อยุ่งยากเป็นอย่างมาก เนื่องจากมีบิตที่กำหนดให้เป็นบิตเครื่องหมาย และอัลกอริทึมในการบวกก็เป็นเรื่องยุ่งยากที่จะออกแบบฮาร์ดแวร์อีกด้วย นอกจากข้อยุ่งยากในการคำนวณแล้ว อีกข้อหนึ่งที่น่าพิจารณาก็คือค่า +0 และ –0

+0 = 00000000

–0 = 10000000

เมื่อคอมพิวเตอร์ทำการคำนวณเสร็จสิ้นแล้วจะต้องมีการตรวจสอบค่าผลลัพธ์ให้ชัดเจนว่ามี 0 เพียงค่าเดียว การแสดงค่า 0 เป็นเรื่องปกติ แต่ถ้าแสดงเป็น –0 จะทำให้ผู้ใช้งานเกิดความสับสนขึ้นได้ และด้วยเหตุที่เกิดข้อยุ่งยากทั้งสองประการนี้ ทำให้การแสดงค่าด้วย Sign – and – Magnitude ไม่นิยมนำมาใช้งาน

- เลขฐานสองกลับค่า (1’s Complement or Not)

การหา 1’s Complement ทำได้โดยเปลี่ยนแต่ละบิตเป็นตรงข้าม คือ ถ้าเป็นเลข 1 ให้กลับเป็น 0 และถ้าเป็นเลข 0 ให้กลับเป็น 1 ซึ่งในคอมพิวเตอร์ไม่ต้องใช้วิธีการลบตามปกติ ค่า 1's Complement ของเลขฐานสอง 8 บิต ค่าที่เป็นบวกเมื่อทำเป็น 1's Complement จะได้เป็นค่าลบ เช่น ค่า 9 คือ 00001001 ดังนั้น ค่า –9 คือ 11110110 ซึ่งเป็นค่า 1's Complement ของ 9 นั่นเอง

การพิจารณาว่าผลลัพธ์เป็นจำนวนบวกหรือลบใช้หลักการเดิมว่า ถ้าบิตซ้ายสุดเป็น 0 แสดงว่าเป็นค่าบวก แต่ถ้าบิตสุดท้ายเป็น 1 แสดงว่าเป็นค่าลบ เช่น 1's Complement ของ 1011101 คือ 0100010

- เลขฐานสองสมบูรณ์ (2’s Complement)

สำหรับ 2’s Complement ซึ่งนิยมใช้แสดงเลขจำนวนเต็มในคอมพิวเตอร์นั้น การเปลี่ยนค่าจำนวนเต็มเป็นตรงข้าม ทำได้โดยการเปลี่ยนสลับค่าทุกบิตเป็นค่าตรงข้าม แล้วบวกด้วย 1 เข้ากับค่าที่สลับค่าบิตนั้น (1’s Complement) ก็จะทำให้ได้ค่าจำนวนเต็มเป็นค่าตรงข้ามกับค่าเดิม (2’s Complement) ตัวอย่างเช่น

+21 = 00010101
1’s Complement = 11101010
+
1
2’s Complement = 11101011 = –21

ลองพิจารณาค่าติดลบ
–21 = 11101011
1’s Complement = 00010100
-
1
2’s Complement = 00010101 = +21

2.5 บทสรุป

ข้อมูลนั้นมีทั้ง คาแรกเตอร์ (Character) รูปภาพ (Picture) เสียง (Sound) หรือข้อมูลในแบบต่าง ๆ จะต้องสามารถนำเข้าสู่คอมพิวเตอร์ และแปลงให้อยู่ในรูปแบบที่เหมาะสมด้วยอุปกรณ์อินพุตที่ให้คอมพิวเตอร์สามารถดำเนินการจัดเก็บ หรือนำไปใช้ในระบบคอมพิวเตอร์ได้

รูปแบบมาตรฐานข้อมูล ประกอบด้วย คาแรกเตอร์, รูปภาพ (บิตแมพ), รูปภาพ (อ็อบเจ็กต์), กราฟิกและฟอนท์, เสียง, เพจ และภาพเคลื่อนไหว

ระบบตัวเลขที่เก็บลงในคอมพิวเตอร์เกิดจากการนับจำนวนสิ่งของต่าง ๆ และมีการพัฒนาโดยใช้เครื่องหมายหรือสัญลักษณ์แทนจำนวนที่นับได้ โดยทั่วไปใช้ระบบตัวเลขฐานสิบ โดยมีตัวเลข 10 ตัว (0-9) หลักการพื้นฐานนี้เองสามารถประยุกต์ใช้กับเลขฐานสอง (Binary) ฐานแปด (Octal) และฐานสิบหก (Hexadecimal) ได้อย่างง่ายดาย

ระบบเลขฐานสิบ (Decimal Number) ประกอบด้วยสัญลักษณ์ 10 สัญลักษณ์ คือ เลข 0 ถึง 9

ระบบเลขฐานสอง (Binary Number) เป็นระบบที่คอมพิวเตอร์รู้จักดี เนื่องจากในสภาวะทั่วไปทางไฟฟ้าหรือทางตรรกศาสตร์จะมีสภาวะมาตรฐานสองสภาวะคือ High (สูง) กับ Low (ต่ำ) หรือ On (เปิด) กับ Off (ปิด) หรือ True (จริง) กับ False (เท็จ) ทำให้สัญลักษณ์ที่ใช้กับเลขฐานสองมีสองตัวคือ 0 และ 1

ระบบเลขฐานแปด (Octal Number) ประกอบด้วยตัวเลข 0 , 1 , 2 , 3 , 4 , 5 , 6 และ 7 เลขฐานแปดมีโครงสร้างเช่นเดียวกับเลขฐานสิบและเลขฐานสอง คือ ตัวเลขแต่ละตัวจะมีค่าตัวคูณในลักษณะเลขยกกำลังของเลข 8 เช่น 8^0 , 8^1 , 8^2 , ... ส่วนค่าของเลขฐานแปดที่เป็นทศนิยมจะมีค่าตัวคูณเป็นเลขยกกำลังของค่าติดลบ เริ่มจาก 8^{-1} , 8^{-2} , ... ค่าที่ออกมาเมื่อนำมารวมกันจะเป็นค่าของเลขฐานสิบ

เลขฐานสิบหก (Hexadecimal Number) นำมาใช้งานเพื่อลดข้อยุ่งยากสำหรับเลขฐานสอง ทำให้สะดวกในการใช้งาน เลขฐานสิบหกประกอบด้วย 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , A , B , C , D , E , F โครงสร้างเลขฐานสิบหก คือ ตัวเลขแต่ละตัวจะมีค่าตัวคูณในลักษณะเลขยกกำลังของเลข 16 เช่น 16^0 , 16^1 , 16^2 , ... ส่วนค่าของเลขฐานสิบหกที่เป็นทศนิยมจะมีค่าตัวคูณเป็นเลขยกกำลังของค่าติดลบ เริ่มจาก 16^{-1} ,16^{-2} , ... ค่าที่ออกมาเมื่อนำมารวมกันจะเป็นค่าของเลขฐานสิบ

การแปลงระหว่างเลขฐานสอง, เลขฐานแปด และเลขฐานสิบหก เป็นเลขฐานสิบ ทำได้โดยการใช้ค่าฐานสองยกกำลังตำแหน่งน้ำหนักแต่ละหลัก แล้วนำมารวมกัน

การแปลงระหว่างเลขฐานสิบเป็นเลขฐานสอง, เลขฐานแปด และเลขฐานสิบหก ทำได้โดยการหารเลขฐานสิบด้วย 2 ของเลขฐานสิบ และนำผลลัพธ์ที่เป็นเศษมาจัดเรียงตำแหน่งเป็นเลขฐานสอง

การแปลงเลขฐานสองเป็นเลขฐานแปดหรือฐานสิบหก เลขฐานแปดจะประกอบด้วยเลขฐานสอง 3 ตัว ($8 = 2^3 = 3$ บิต) และเลขฐานสิบหกประกอบด้วยเลขฐานสอง 4 ตัว ($16 = 2^4 = 4$ บิต) โดยนับจากตัวที่มีค่านัยสำคัญต่ำสุด (หรือจากจุดทศนิยม) เลื่อนไปทางซ้ายหรือขวา

การแปลงเลขฐานแปดหรือเลขฐานสิบหกเป็นเลขฐานสอง ให้กระจายเลขฐานแปดหรือเลขฐานสิบหกเป็นเลขฐานสอง โดยจะต้องใส่ให้ครบจำนวนบิตของเลขฐานที่เราต้องการ

การแปลงเลขฐานแปดเป็นเลขฐานสิบหก จะต้องแปลงให้เป็นเลขฐานสองก่อน หลังจากนั้นจัดกลุ่มใหม่เป็นกลุ่มละ 4 ตัว (บิต) เพื่อแปลงเป็นเลขฐานสิบหก

ในระบบเลขฐานสอง สามารถแสดงค่าเลขจำนวนเต็มโดยใช้ตัวเลขเพียง 2 ตัว คือ 0 และ 1 โดยอาจจะมีเครื่องหมาย + และ – รวมทั้งมีจุด การเก็บข้อมูลและประมวลผลในคอมพิวเตอร์จะเก็บข้อมูลเฉพาะค่า (Value) และเครื่องหมายในรูปแบบจำนวนเต็มที่ไม่มีเครื่องหมาย (Unsigned Integer), จำนวนเต็มที่มีเครื่องหมาย (Sign Integer) และ 2’Complement

คำถามทบทวนประจำบท

~~~~~~

คำสั่ง

<u>ตอนที่ 1</u> จงแปลงเลขฐานต่อไปนี้ โดยแสดงวิธีทำอย่างละเอียด

1. 53_{10} → $\_\_\_\__{2}$
2. 259.6875_{10} → $\_\_\_\__{2}$
3. 135_{10} → $\_\_\_\__{8}$
4. 584.995_{10} → $\_\_\_\__{8}$
5. 0.69_{10} → $\_\_\_\__{16}$
6. 2860.153_{10} → $\_\_\_\__{16}$
7. 10010.1011_{2} → $\_\_\_\__{10}$
8. 110101_{2} → $\_\_\_\__{10}$
9. 72_{8} → $\_\_\_\__{10}$
10. 5732.246_{8} → $\_\_\_\__{10}$
11. 93_{16} → $\_\_\_\__{10}$
12. $5EB.2_{16}$ → $\_\_\_\__{10}$
13. 11010_{2} → $\_\_\_\__{8}$
14. 10000111.1_{2} → $\_\_\_\__{8}$
15. 23_{8} → $\_\_\_\__{2}$
16. 6501.23_{8} → $\_\_\_\__{2}$
17. 10010100011_{2} → $\_\_\_\__{16}$
18. $110111011011000.0111010_{2}$ → $\_\_\_\__{16}$
19. $A90_{16}$ → $\_\_\_\__{2}$
20. $53BD.20A_{16}$ → $\_\_\_\__{2}$
21. 34_{8} → $\_\_\_\__{16}$
22. 417_{8} → $\_\_\_\__{16}$
23. 5246.23_{8} → $\_\_\_\__{16}$
~~~~~~

24. 1101.101_{16} → _____$_8$

25. $ABCD.EF_{16}$ → _____$_8$

<u>ตอนที่ 2</u> จงหาผลลัพธ์จากโจทย์ต่อไปนี้ โดยแสดงวิธีทำอย่างละเอียด

1. 11011_2 + 10011_2
2. 110111.11_2 + 101111.11_2
3. 354_8 + 437_8
4. 23246.201_8 + 765432.107_8
5. 374_{16} + 697_{16}
6. $738B.2A_{16}$ + $91CD.E4_{16}$
7. 10110_2 - 10011_2
8. 111000_2 - 100111_2
9. 6151_8 - 2454_8
10. 20702.361_{16} - 20574.142_{16}
11. 594_{16} - $38B_{16}$
12. 3184_{16} - 2090.34_{16}
13. 1's Complement ของ 1101101_2
14. 1's Complement ของ 110010101_2
15. 2's Complement ของ 10111_2
16. 2's Complement ของ 10111000_2