

องค์ประกอบของเครื่องคอมพิวเตอร์  
และภาษาแอสเซมบลี:  
ARM และ RaspberryPi3

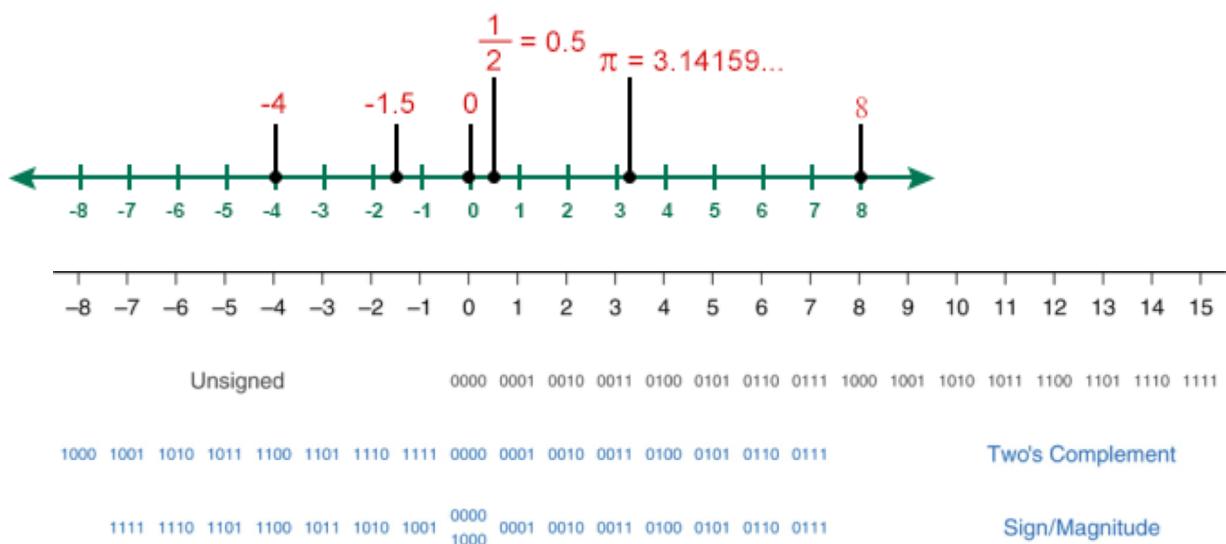
บทที่ 2 ข้อมูลและคณิตศาสตร์ในคอมพิวเตอร์

ผศ.ดร.สุรินทร์ กิตติธรภุล  
ภาควิชาศึกษาดูงานคอมพิวเตอร์ คณะวิศวกรรมศาสตร์  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

## สารบัญ

- บทที่ 2 ข้อมูลและคณิตศาสตร์ในคอมพิวเตอร์
  - 2.1 ตัวแปรชนิดต่างในภาษา C/C++
  - 2.2 เลขจำนวนเต็มฐานสอง
  - 2.3 คณิตศาสตร์เลขจำนวนเต็มฐานสอง
  - 2.4 เลขทศนิยมฐานสองชนิดจุดคงที่ (Binary Fixed Point)
  - 2.5 เลขทศนิยมฐานสองชนิดจุดลอยตัว (Binary Floating Point)
  - 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754
  - 2.7 ตัวอักษร

## 2.1 ตัวแปรชนิดต่างในภาษา C/C++



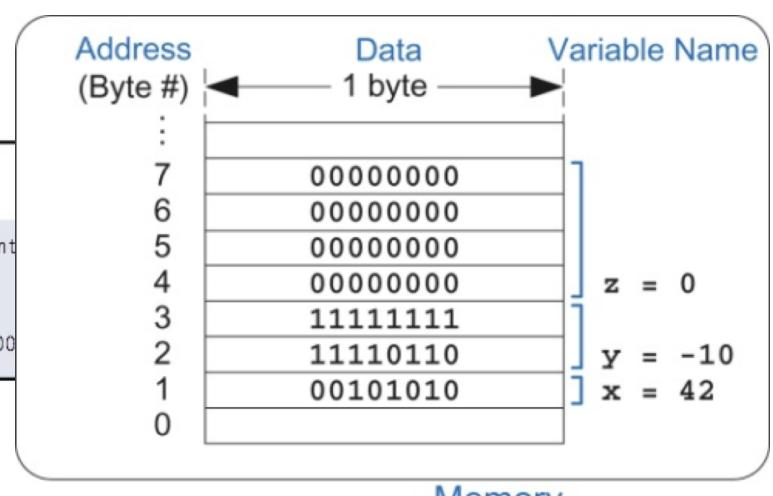
Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

3

## 2.1 ตัวแปรชนิดต่างในภาษา C/C++

### C Code Example eC.3 EXAMPLE DATA TYPES

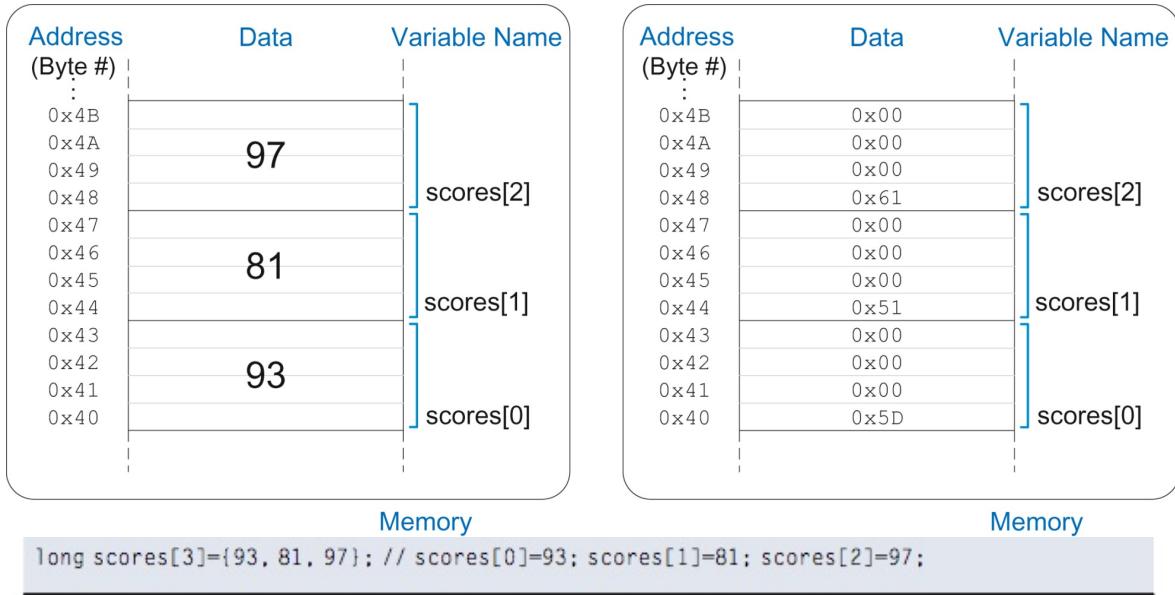
```
// Examples of several data types and their binary representation
unsigned char x = 42;           // x = 00101010
short y = -10;                 // y = 11111111 11110110
unsigned long z = 0;            // z = 00000000 00000000 00000000
```



Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

4

## 2.1 ตัวแปรชนิดต่างในภาษา C/C++



Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

5

## 2.1 ตัวแปรชนิดต่างในภาษา C/C++

### C Code Example eC.18

```
// Example pointer manip
int salary1, salary2;
int *ptr;
salary1 = 67500;
ptr = &salary1;
salary2 = *ptr + 1000;
```

Address (Byte #)	Data	Variable Name
0x7B	0x70	ptr
0x7A	68500	salary2
0x79	67500	salary1
0x78		
0x77		
0x76		
0x75		
0x74		
0x73		
0x72		
0x71		
0x70		

(a)

Memory

(b)

Memory

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

6

unsigned → ឧប្បជ្ជន៍តាមចុងក្រោយនៃ address

2's complement → រាយការប្រគល់សារ (unsigned)

សារ signed

រាយការប្រគល់តាមចុងក្រោយ

overflow

៥៨ = ១២៩ មុនពេល = ៥៨

-8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Unsigned

0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

1000 1001 1010 1011 1100 1101 1110 1111 0000 0001 0010 0011 0100 0101 0110 0111

Two's Complement

1111 1110 1101 1100 1011 1010 1001 0000 0001 0010 0011 0100 0101 0110 0111

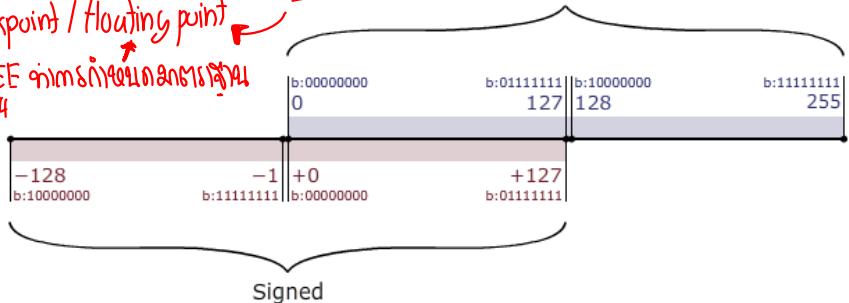
0000 1000 0001 0010 0011 0100 0101 0110 0111

Fixpoint / floating point  
គណនី IEEE តាមរាយការប្រគល់តាមចុងក្រោយ

ទី៤

Sign/Magnitude

Unsigned



Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

7

## 2.2.1 เลขจำนวนเต็มฐานสอง ชนิดไม่มีเครื่องหมาย

```
unsigned char x = 42; // x = 0b00101010
unsigned short y = 10; // y = 0b00000000000000001010
unsigned int z = 0; // z = 0x00000000
```

និយាយទី 2.2.1. ការណែនាំ លេខំនុវត្តន៍ នៅក្នុងប្រព័ន្ធឌីជីថាម (Unsigned Integer)  $X_{2,u}$  ដែល  
លេខ្ទានសងខ្សោយនូវរូបរាង

$$X_{2,u} = x_{n-1}x_{n-2}x_{n-3}\dots x_1x_0 \quad (2.1)$$

ដើម្បី  $x_i$  គឺជាកំណត់ “1” ឬ “0” នៃតាមរុងទី  $i$  និងតាមរុងទី  $0$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

8

## 2.2.1 เลขจำนวนเต็มฐานสอง ชนิดไม่มีเครื่องหมาย

### 2.2.1.1 การแปลงเลขฐานสองเป็นฐานสิบ

จากนิยามที่ 2.2.1 ค่าจำนวนเต็มฐานสิบ  $X_{10,u}$  ของเลข  $X_{2,u}$  สามารถคำนวณได้จาก

$$X_{10,u} = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \dots + x_12^1 + x_02^0 \quad (2.2)$$

ดังนั้น ค่าฐานสิบ  $X_{10,u}$  อยู่ในช่วง  $0 \leq X_{10,u} < 2^n - 1$

ตัวอย่างที่ 2.2.1. เมื่อ  $n = 4$  บิก เลขจำนวนเต็มฐานสองชนิดไม่มีเครื่องหมาย  $X_{2,u} = 1011_2$

ค่าฐานสิบของ  $X_{2,u}$  คือ

$$X_{10,u} = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \quad (2.3)$$

$$= 8 + 0 + 2 + 1 \quad (2.4)$$

$$= 11_{10} \quad (2.5)$$

## 2.2.1 เลขจำนวนเต็มฐานสอง ชนิดไม่มีเครื่องหมาย

บิกที่	เลขฐานสิบ	ผลหาร	เศษ
-	123		
0	123/2	61	1
1	61/2	30	1
2	30/2	15	0
3	15/2	7	1
4	7/2	3	1
5	3/2	1	1
6	1/2	0	1
7	0/2	0	0

บิกที่ $i$	$2^i$	ผลลัพธ์- $2^i$	ผลลัพธ์	$x_i$
-		123		
7	$2^7 = 128$	123-128	123	0
6	$2^6 = 64$	123-64	59	1
5	$2^5 = 32$	59-32	27	1
4	$2^4 = 16$	27-16	11	1
3	$2^3 = 8$	11-8	3	1
2	$2^2 = 4$	3-4	3	0
1	$2^1 = 2$	3-2	1	1
0	$2^0 = 1$	1-1	0	1

### 2.2.2 เลขจำนวนเต็มฐานสอง ชนิดมีเครื่องหมาย

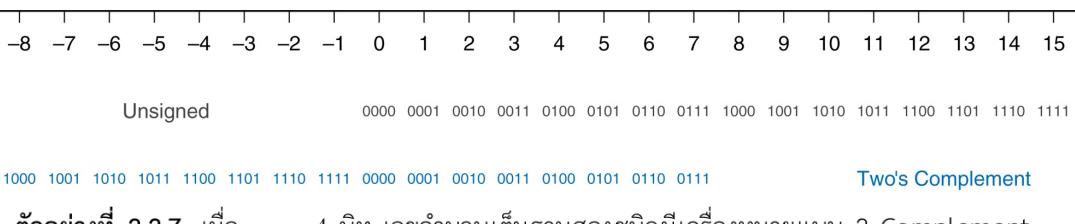
```
char x = -2; // x = 0b11111110  
short y = -2; // y = 0b11111111111111111110  
int z = -2; // z = 0xFFFFFFF
```

นิยามที่ 2.2.2. กำหนดให้ เลขจำนวนเต็มฐานสองชนิดมีเครื่องหมาย (Signed Integer) แบบ 2-Complement  $X_{2,s}$  เอียงอยู่ในรูป

$$X_{2,s} = x_{n-1}x_{n-2}x_{n-3}\dots x_1x_0 \quad (2.15)$$

เมื่อ  $s$  ทำหน้าที่เป็นบิตเครื่องหมาย (Sign bit) และ  $x_i$  คือค่า “1” หรือ “0” ในตำแหน่งที่  $i$  และตำแหน่งขวาเมื่อสุดคือตำแหน่งที่  $i = 0$

### 2.2.2 เลขจำนวนเต็มฐานสอง ชนิดมีเครื่องหมาย



ตัวอย่างที่ 2.2.7. เมื่อ  $n = 4$  บิต เลขจำนวนเต็มฐานสองชนิดมีเครื่องหมายแบบ 2 Complement  $X_{2,s} = 1011_2$  มีค่าฐานสิบเท่ากับเท่าไหร่  
ค่าฐานสิบของ  $X_{2,s}$  คือ

$$X_{10,8} \equiv -1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \quad (2.17)$$

$$\equiv -8 + 0 + 2 + 1 \quad (2.18)$$

$$\equiv -5_{10} \quad (2.19)$$

เลขจำนวนเต็มฐานสองชนิดมีเครื่องหมาย 2-Complement ขนาด  $n = 4$  บิตจะมีค่าฐานสิบอยู่ในช่วง -8 ถึง +7

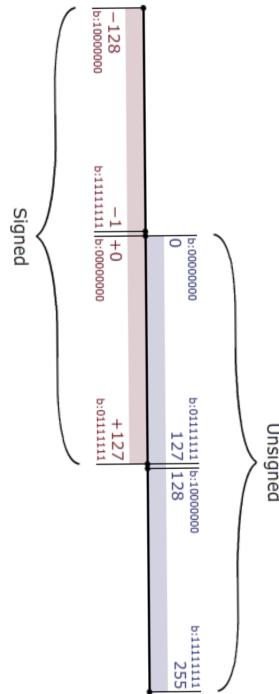
ឧបករណីលេខសែន 61 + ដែល មានលេខចុងខ្លួន

Sign Extension	លេខវត្ថុនៃសង $n=4$	0110		លេខវត្ថុនៃសង $n=5$	$X_{10,s}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	$X_{10,u}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	$X_{10,s}$ គោរពសិប មិត្តគីឡូរងហ្មាយ
		$X_{10,s}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	$X_{10,u}$ គោរពសិប មិត្តគីឡូរងហ្មាយ				
	1000	-8	8	1 0000	-16	16	
	1001	-7	9	1 0111	-9	23	
	1010	-6	10	1 1000	-8	24	
	1011	-5	11	1 1001	-7	25	
	1100	-4	12	1 1010	-6	26	
	1101	-3	13	1 1011	-5	27	
	1110	-2	14	1 1100	-4	28	
	1111	-1	15	1 1101	-3	29	
	0000	0	0	1 1110	-2	30	
	0001	1	1	1 1111	-1	31	
	0010	2	2	0 0000	0	0	
	0011	3	3	0 0001	1	1	
	0100	4	4	0 0010	2	2	
	0101	5	5	0 0011	3	3	
	0110	6	6	0 0100	4	4	
	0111	7	7	0 0101	5	5	
				0 0110	6	6	
				0 0111	7	7	
				...	...	...	
				0 1111	15	15	

ការអនុពត្យនាំងលើលក្ខណន៍ bit

Sign Extension	លេខវត្ថុនៃសង $n=5$			លេខវត្ថុនៃសង $n=8$	$X_{10,s}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	$X_{10,u}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	
		$X_{10,s}$ គោរពសិប មិត្តគីឡូរងហ្មាយ	$X_{10,u}$ គោរពសិប មិត្តគីឡូរងហ្មាយ				
	1 0000	-16	16	1000 0000	-128	128	
	...	...	...	...	...	...	
	1 0111	-9	23	1111 0111	-9		
	1 1000	-8	24	1111 1000	-8	248	
	1 1001	-7	25	1111 1001	-7	249	
	1 1010	-6	26	1111 1010	-6	250	
	1 1011	-5	27	1111 1011	-5	251	
	1 1100	-4	28	1111 1100	-4	252	
	1 1101	-3	29	1111 1101	-3	253	
	1 1110	-2	30	1111 1110	-2	254	
	1 1111	-1	31	1111 1111	-1	255	
	0 0000	0	0	0000 0000	0	0	
	0 0001	1	1	0000 0001	1	1	
	0 0010	2	2	0000 0010	2	2	
	0 0011	3	3	0000 0011	3	3	
	0 0100	4	4	0000 0100	4	4	
	0 0101	5	5	0000 0101	5	5	
	0 0110	6	6	0000 0110	6	6	
	0 0111	7	7	0000 0111	7	7	
	...	...	...	0000 1000	8	8	
	0 1111	15	15	...	...	...	
				0111 1111	127	127	

เลขฐานสอง $n=8$	$X_{10,s}$ ค่าฐานสิบ มีเครื่องหมาย	$X_{10,u}$ ค่าฐานสิบ ไม่มีเครื่องหมาย
1000 0000	-128	128
...	...	...
1111 0111	-9	
1111 1000	-8	248
1111 1001	-7	249
1111 1010	-6	250
1111 1011	-5	251
1111 1100	-4	252
1111 1101	-3	253
1111 1110	-2	254
1111 1111	-1	255
0000 0000	0	0
0000 0001	1	1
0000 0010	2	2
0000 0011	3	3
0000 0100	4	4
0000 0101	5	5
0000 0110	6	6
0000 0111	7	7
0000 1000	8	8
...	...	...
0111 1111	127	127



Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

15

## 2.2.3 เลขจำนวนเต็มฐานสอง ชนิดมีเครื่องหมาย Sign-Magnitude

นิยามที่ 2.2.3. กำหนดให้ เลขจำนวนเต็มฐานสองชนิดมีเครื่องหมาย (Signed Integer) แบบ Sign-Magnitude  $X_{2,sm}$  เขียนอยู่ในรูป

$$X_{2,sm} = s x_{n-2} x_{n-3} \dots x_1 x_0 \quad (2.34)$$

เมื่อ  $s$  คือบิตเครื่องหมาย (Sign bit) และ  $x_i$  คือค่า “1” หรือ “0” ในตำแหน่งที่  $i$  และตำแหน่งของ  $x_0$  มีอสูดคือตำแหน่งที่  $i = 0$

การแปลงเลขจำนวนเต็มฐานสองแบบ Sign-Magnitude ให้เป็นค่าฐานสิบสามารถทำได้โดย

$$X_{10,sm} = (-1)^s \times (x_{n-2} \times 2^{n-2} + \dots + x_1 \times 2^1 + x_0 \times 2^0) \quad (2.35)$$

ดังนั้น ค่าฐานสิบ  $X_{10,s}$  อยู่ในช่วง  $-2^{n-1} + 1$  ถึง  $+2^{n-1} - 1$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

16

## 2.2.3 เลขจำนวนเต็มฐานสอง ชนิดมีเครื่องหมาย Sign-Magnitude

Table 2.8: ค่าฐานสิบแบบมีเครื่องหมาย Sign-Magnitude, แบบ 2-Complement, และแบบไม่มีเครื่องหมาย (Unsigned) ของเลขฐานสองความยาว  $n=4$  บิตทั้งหมด  $2^4=16$  แบบ

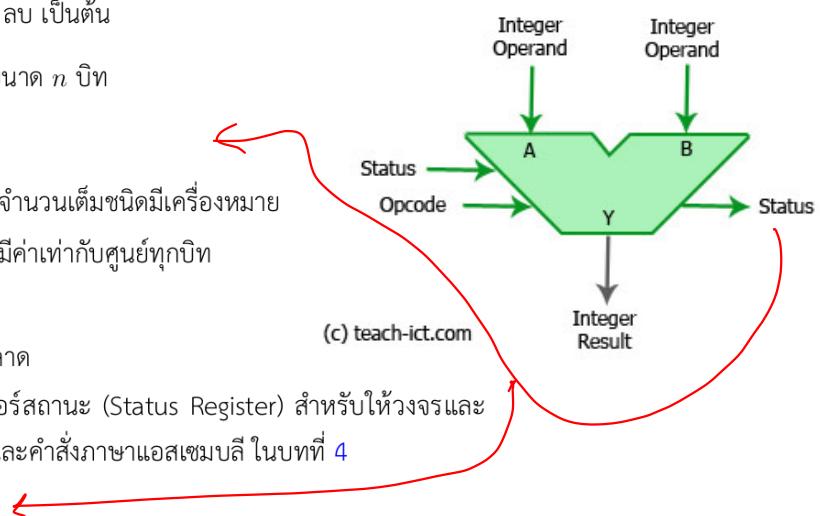
เลขฐานสอง $n=4$ บิต	$X_{10,sm}$ ค่าฐานสิบ Sign-Mag	$X_{10,s}$ ค่าฐานสิบ 2-Comp.	$X_{10,u}$ ค่าฐานสิบ Unsigned
1111	-7	-1	15
1110	-6	-2	14
1101	-5	-3	13
1100	-4	-4	12
1011	-3	-5	11
1010	-2	-6	10
1001	-1	-7	9
1000	-0	-8	8
0000	+0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7

bit pattern 10100000

ค่าฐานสิบ 10

## 2.3 คณิตศาสตร์เลขจำนวนเต็ม

- สัญญาณ Opcode เพื่อสั่งการทำงาน เช่น บวก ลบ เป็นต้น
  - ผลลัพธ์ Y เป็นจำนวนเต็มชนิดไม่มีเครื่องหมายขนาด n บิต
  - สัญญาณ Status ประกอบด้วย
    - หากเครื่องหมาย N (Negative) สำหรับเลขจำนวนเต็มชนิดมีเครื่องหมาย
    - หาก Z (Zero)=1 เพื่อบ่งบอกว่าผลลัพธ์ Y มีค่าเท่ากับศูนย์ทุกบิต
    - หาก Carry  $c_n$  สำหรับตัวดิจิตที่ n
    - หาก Overflow (V) เพื่อบ่งบอกความผิดพลาด
- หากสัญญาณเหล่านี้จะบันทึกลงในรีจิสเตอร์สถานะ (Status Register) สำหรับให้ตรวจสอบและโปรแกรมเมอร์ตรวจสอบด้วยwangจรดิจิทัลและคำสั่งภาษาแอสเซมบลี ในบทที่ 4

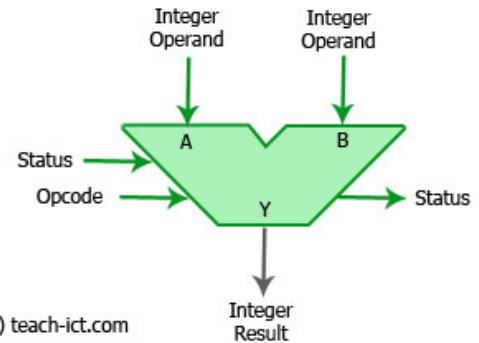


overflow ไม่ได้มีตัวอย่าง software จึงไม่สามารถแสดงตัวอย่างได้

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

จำนวนบิต 9 บิต กับ 2 บิต ที่เกิดจาก overflow (2.47)

	$c_n$	$c_{n-1}$	$c_{n-2}$	..	$c_2$	$c_1$	$c_0$	+
$X_{2,u} +$		$x_{n-1}$	$x_{n-2}$	..	$x_2$	$x_1$	$x_0$	
$Y_{2,u}$		$y_{n-1}$	$y_{n-2}$	..	$y_2$	$y_1$	$y_0$	
$Z_{2,u}$		$z_{n-1}$	$z_{n-2}$	..	$z_2$	$z_1$	$z_0$	



( pos )

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

19

A	B	CarryIn	Out	CarryOut
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$\bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + ABC$   
 $\bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC)$   
 $\bar{A}(B \oplus C) + A(\bar{B} \oplus \bar{C})$   
 $A \oplus (B \oplus C)$

$\bar{A}\bar{B}C + A(\bar{B} + B\bar{C}) + ABC$   
 $(\bar{A} + A)BC + A(B \oplus C)$   
 $BC + A(B \oplus C)$  จันๆ ?

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

$$c_{i+1}z_i = x_i + y_i + c_i \quad (2.40)$$

เมื่อ  $i=0, 1, 2, \dots, n-1$  โดย  $c_0 = 0$  และสัญลักษณ์ + คือการบวกเลข ไม่มีการ OR กันเชิงตรรกศาสตร์  
ในวิชาออกแบบจรดิจิทัล เราเรียกว่าการบวกเลขชนิดนี้ว่า วงจร Full Adder โดยวงจรจะนำบิต  
ข้อมูลจำนวน 3 บิตมากระทำการทางตรรกศาสตร์ได้ผลลัพธ์  $z_i$  โดย

$$z_i = x_i \oplus y_i \oplus c_i$$

เมื่อ  $\oplus$  คือ กระบวนการ Exclusive-OR และบิตตัวท้าย  $c_{i+1}$

$$c_{i+1} = (x_i \& y_i) | (x_i \& c_i) | (y_i \& c_i) \quad (2.42)$$

เมื่อ  $\&$  คือ กระบวนการ AND และ  $|$  คือ กระบวนการ OR วงจรบวกเลขชนิดไม่มีเครื่องหมายขนาด  $n$  บิต  
นี้สามารถตรวจสอบการเกิดโอเวอร์โฟล์วได้โดย

$$V = c_n \quad (2.43)$$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

20

จันๆ  $\oplus$  กับเงา ก็จะเท่ากัน  
ก็ต้องเป็น 0 ถึง 0 ไป

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

ตัวอย่างที่ 2.3.1. จงคำนวณหาค่าของ  $5 + 9$  ด้วยเลขจำนวนเต็มชนิดมีเครื่องหมาย แบบ Unsigned ขนาด 4 บิต  $5 + 9 = 14$  ดังนั้น ในเครื่องคอมพิวเตอร์ขนาด 4 บิต สามารถคำนวณได้ดังนี้

	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$	Overflow
	0	0	0	1	0	$V=0$
$X=5 +$		0	1	0	1	+
$Y=9$		1	0	0	1	
$Z=14$		1	1	1	0	

บวกแล้ว ว่างๆ

การบวกเลขขนาด 4 บิตแบบไม่มีเครื่องหมาย:  $5 + 9 = 14$  พร้อมตัวทด และผลลัพธ์ถูกต้องเนื่องจากไม่เกิดโอเวอร์เฟลว์ ( $V=c_n=0$ )

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

ตัวอย่างที่ 2.3.2. จงคำนวณหาค่าของ  $7 + 9$  ด้วยเลขจำนวนเต็มชนิดมีเครื่องหมาย แบบ Unsigned ขนาด 4 บิต  $7 + 9 = 0$  ดังนั้น ในเครื่องคอมพิวเตอร์ขนาด 4 บิต ซึ่งไม่สามารถแสดงผลค่า  $16_{10}$  ได้ ดังนี้

detect overflow

	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$	Overflow
	1	1	1	1	0	$V=c_n=1$
$X=7 +$		0	1	1	1	+
$Y=9$		1	0	0	1	
$Z=16$		0	0	0	0	

สาเหตุของการเกิด Overflow เนื่องจากผลลัพธ์มีค่าอยู่นอกย่านที่เป็นไปได้ โดยสามารถตรวจสอบอย่างง่ายดายโดย  $c_4 = 1$  ( $V$ : Overflow) เมื่อเกิดโอเวอร์เฟลว์ ผลลัพธ์ที่ได้จึงมีค่าไม่ถูกต้อง (Invalid)

## 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

ตัวอย่างที่ 2.3.2. จงคำนวณหาค่าของ  $7 + 9$  ด้วยเลขจำนวนเต็มชนิดมีเครื่องหมาย แบบ Unsigned ขนาด 4 บิต  $7 + 9 = 16 = 0$  ดังนั้น ในเครื่องคอมพิวเตอร์ขนาด 4 บิต ซึ่งไม่สามารถแสดงผลค่า  $16_{10}$  ได้ ดังนี้

	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$	Overflow
	1	1	1	1	0	$V=c_n=1$
$X=7+$	0	1	1	1		+
$Y=9$	1	0	0	1		
$Z=16$	0	0	0	0		

สาเหตุของการเกิด Overflow เนื่องจากผลลัพธ์มีค่าอยู่นอกอย่างที่เป็นไปได้ โดยสามารถตรวจสอบอย่างง่ายดายโดย  $c_4 = 1$  ( $V$ : Overflow) เมื่อเกิดโอเวอร์เฟลว์ ผลลัพธ์ที่ได้จะมีค่าไม่ถูกต้อง (Invalid)

## 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

วงจรคูณเลขชนิดที่ 1

การคูณเลขมีความซับซ้อนสูงสำหรับการใช้งานจริงต้องใช้วงจรดิจิทัลและเวลาในคำนวณ การคูณเลขที่ง่ายที่สุดคือการบวกเลขแล้ววนบวกซ้ำ ตามวงจรที่จะอธิบายในรูปที่ 2.5 และ 2.6 ต่อไปนี้

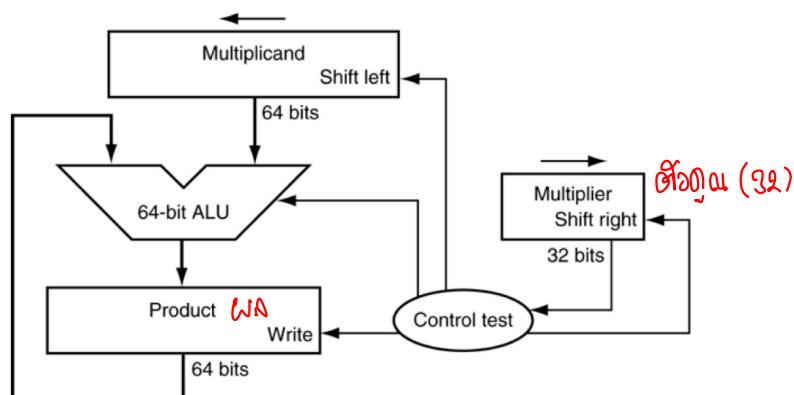


Figure 2.5: วงจรคูณเลขขนาด  $n = 32$  บิต ชนิดที่ 1 โดยใช้ ALU ขนาด  $2n = 64$  บิต และรีจิสเตอร์ตัวตั้งขนาด  $2n = 64$  บิต (ที่มา: Patterson and Hennessy (2000))

## 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

ตัวอย่างที่ 2.3.3. จงคูณ  $1010_2$  ( $10_{10}$ ) ด้วย  $1101_2$  ( $13_{10}$ ) ตามวิธีที่ 2.5

ผลคูณเลขขนาด 4 บิต ตัวตั้ง =  $1010_2$  ( $10_{10}$ ) ตัวคูณ =  $1101_2$  ( $13_{10}$ ) เท่ากับ  $130_{10}$  ด้วยวิธีในรูปที่ 2.5 หมายเหตุ Shift R คือ การเลื่อนบิตไปทางขวา

2.5

Table 2.9: ผลคูณเลขขนาด 4 บิต ตัวตั้ง =  $1010_2$  ( $10_{10}$ ) ตัวคูณ =  $1101_2$  ( $13_{10}$ ) เท่ากับ  $130_{10}$  ด้วยวิธีในรูปที่ 2.5 หมายเหตุ Shift R คือ การเลื่อนบิตไปทางขวา

รอบ Iteration	ตัวตั้ง Multicand	ตัวคูณ Multiplier	ผลคูณ <sub>2</sub> Product <sub>2</sub>	ผลคูณ <sub>10</sub> Product <sub>10</sub>	กระทำ Action
-	0000 1010 <sub>2</sub>	1101 <sub>2</sub>	0000 0000 <sub>2</sub>	0 <sub>10</sub>	Initialized
0	0000 1010 <sub>2</sub>	1101 <sub>2</sub>	0000 1010 <sub>2</sub>	10 <sub>10</sub>	Add
	0001 0100 <sub>2</sub>	0110 <sub>2</sub>	0000 1010 <sub>2</sub>	10 <sub>10</sub>	Shift R
1	0001 0100 <sub>2</sub>	0110 <sub>2</sub>	0000 1010 <sub>2</sub>	10 <sub>10</sub>	None
	0010 1000 <sub>2</sub>	0011 <sub>2</sub>	0000 1010 <sub>2</sub>	10 <sub>10</sub>	Shift R
2	0010 1000 <sub>2</sub>	0011 <sub>2</sub>	0011 0010 <sub>2</sub>	50 <sub>10</sub>	Add
	0101 0000 <sub>2</sub>	0001 <sub>2</sub>	0011 0010 <sub>2</sub>	50 <sub>10</sub>	Shift R
3	0101 0000 <sub>2</sub>	0001 <sub>2</sub>	1000 0010 <sub>2</sub>	130 <sub>10</sub>	Add
	1010 0000 <sub>2</sub>	0000 <sub>2</sub>	1000 0010 <sub>2</sub>	130 <sub>10</sub>	Shift R

แบบนี้จะได้ผลลัพธ์ตัวตั้ง  
ก็ต้องดูบิตที่ 32 นี่  
ตัวตั้งจะ clock ตัวบิตที่ 32 ลง  
ด้วยห้ามกด

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

25

## 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

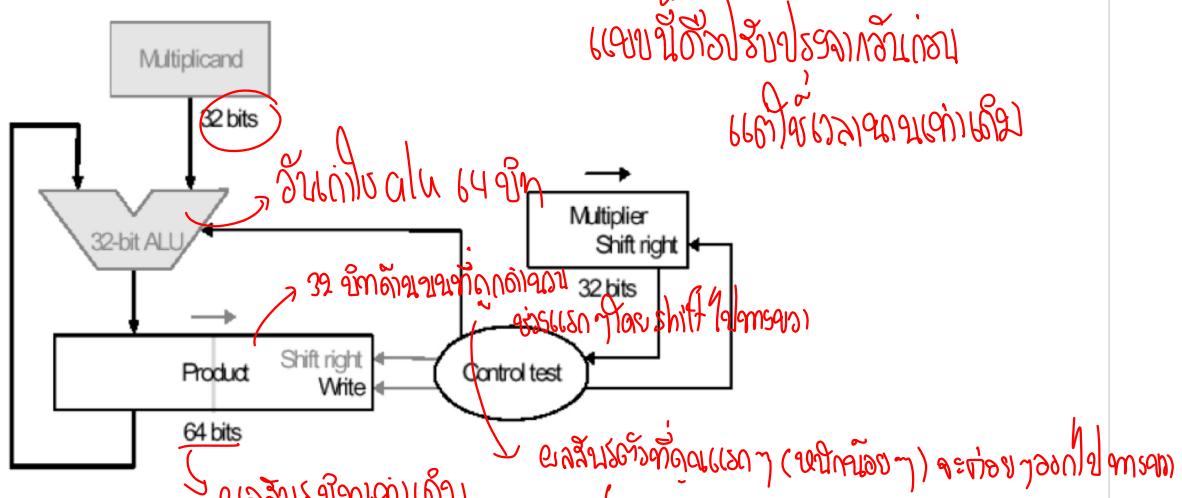


Figure 2.6: วิธีคูณเลขขนาด  $n = 32$  บิต ชนิดที่ 2 โดยใช้ ALU ขนาด  $n = 32$  บิต และรีจิสเตอร์ตัวตั้งขนาด  $n = 32$  บิต (ที่มา: Patterson and Hennessy (2000))

ห้ามกดตัวตั้ง แต่เวลาต้องป้อนตัวตั้ง

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

26

ပုဂ္ဂန်စောင်မြန်မာနိုင်ငံ + အာနိုင်ဒါနပူရ

## ରୋଗର୍ତ୍ତିଙ୍କ ଯେହିଏବୁଦ୍ଧିମେତ୍ରାଳୟ ଲେଖନ

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

ตัวอย่างที่ 2.3.4. จงคูณ  $1010_2$  ( $10_{10}$ ) ด้วย  $1101_2$  ( $13_{10}$ ) ตามวิธีที่ 2.6

ผลคูณเลขฐาน 4 บิท ตัวตั้ง=1010<sub>2</sub> (10<sub>10</sub>) ตัวคูณ=1101<sub>2</sub> (13<sub>10</sub>) เมื่อกับ 130<sub>10</sub> ด้วยวงจรในรูปที่ 2.6

Table 2.10: ผลคูณเลขฐาน 4 บิต ตัวตั้ง =  $1010_2$  ( $10_{10}$ ) ตัวคูณ =  $1101_2$  ( $13_{10}$ ) เท่ากับ  $130_{10}$  ด้วยวงจรในรูปที่ 2.6 หมายเหตุ Shift R คือ การเลื่อนบิตไปทางขวา  $\rightarrow$  3.2 ห้องเรียน เลือกภาษา shift

→ 32 ດືກໂຫຼດ ແລະ ຂອງ shift right

รอบ Iteration	ตัวตั้ง Multicand	ตัวคูณ Multiplier	ผลคูณ Product <sub>2</sub>	ผลคูณ Product <sub>10</sub>	กระทำ Action
-	<u>1010<sub>2</sub></u>	1101 <sub>2</sub>	0000 0000 <sub>2</sub>	0 <sub>10</sub>	Initialized
0	1010 <sub>2</sub>	1101 <sub>2</sub>	1010 0000 <sub>2</sub>	160 <sub>10</sub>	Add
	1010 <sub>2</sub>	0110 <sub>2</sub>	0101 0000 <sub>2</sub>	80 <sub>10</sub>	Shift R
1	1010 <sub>2</sub>	0110 <sub>2</sub>	0101 0000 <sub>2</sub>	80 <sub>10</sub>	None
	1010 <sub>2</sub>	0011 <sub>2</sub>	0010 1000 <sub>2</sub>	40 <sub>10</sub>	Shift R
2	1010 <sub>2</sub>	0011 <sub>2</sub>	1100 1000 <sub>2</sub>	200 <sub>10</sub>	Add
	1010 <sub>2</sub>	0001 <sub>2</sub>	0110 0100 <sub>2</sub>	100 <sub>10</sub>	Shift R
3	1010 <sub>2</sub>	0001 <sub>2</sub>	1 0000 0100 <sub>2</sub>	260 <sub>10</sub>	Add
	1010 <sub>2</sub>	0000 <sub>2</sub>	1000 0010 <sub>2</sub>	130 <sub>10</sub>	Shift R

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

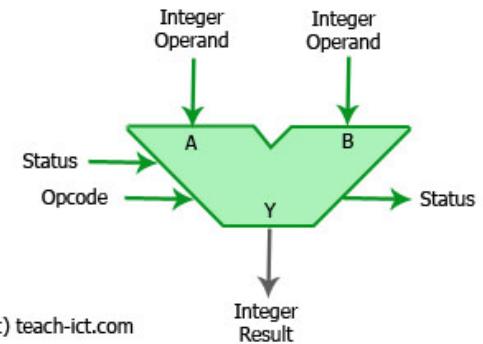
27

ଓম্বা মিকার্য়েল চৰা হৈনো তথ্য প্ৰযোগ কৰা দেৱ ৫৫৫ +  
৭৬ টক্কা কালু clock ১০ হিঁড়ুল কৰা

### 2.3.1 คณิตศาสตร์เลขจำนวนเต็ม ชนิดไม่มีเครื่องหมาย

### 2.3.2 คณิตศาสตร์เลขจำนวนเต็ม ชนิดมีเครื่องหมาย 2-Complement

	$c_n$	$c_{n-1}$	$c_{n-2}$	..	$c_2$	$c_1$	$c_0$	+
$X_{2,s} +$		$x_{n-1}$	$x_{n-2}$	..	$x_2$	$x_1$	$x_0$	
$Y_{2,s}$		$y_{n-1}$	$y_{n-2}$	..	$y_2$	$y_1$	$y_0$	
$Z_{2,s}$		$z_{n-1}$	$z_{n-2}$	..	$z_2$	$z_1$	$z_0$	



(c) teach-ict.com

### 2.3.2 คณิตศาสตร์เลขจำนวนเต็ม ชนิดมีเครื่องหมาย 2-Complement

$$c_{i+1} z_i = x_i + y_i + c_i \quad (2.44)$$

เมื่อ  $i=0, 1, 2, \dots, n-1$  โดย  $c_0 = 0$

ผลลัพธ์ของการบวกเลขจำนวน 3 บิต สามารถคำนวณได้จากการ Full Adder ดังนี้

$$z_i = x_i \oplus y_i \oplus c_i \quad (2.45)$$

เมื่อ  $\oplus$  คือ กระบวนการ Exclusive-OR

$$c_{i+1} = (x_i \& y_i) | (x_i \& c_i) | (y_i \& c_i) \quad (2.46)$$

เมื่อ  $\&$  คือ กระบวนการ AND และ  $|$  คือ กระบวนการ OR

การเกิดโอเวอร์ฟล็อวของการบวกเลขชนิดมีเครื่องหมาย 2-Complement ได้โดย

$$V = c_n \oplus c_{n-1} \quad (2.47)$$

$$\begin{array}{r}
 \text{C}_4 \text{ C}_3 \text{ C}_2 \text{ C}_1 \\
 \hline
 0 \ 1 \ 1 \ 1 \ 0 \\
 0 \ 1 \ 1 \ 0 \\
 \hline
 0 \ 0 \ 1 \ 0 \oplus \\
 \hline
 1 \ 0 \ 0 \ 0 = -7
 \end{array}$$

## 2.3.2 คณิตศาสตร์เลขจำนวนเต็ม ชนิดมีเครื่องหมาย 2-Complement

ตัวอย่างที่ 2.3.5. จงคำนวณหาค่าของ  $7 + 3$  ด้วยเลขจำนวนเต็มชนิดมีเครื่องหมายแบบ 2-Complement ขนาด 4 บิต ดังนั้น ในเครื่องคอมพิวเตอร์ขนาด 4 บิต สามารถคำนวณได้ดังนี้

	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$	Overflow
	0	1	1	1	1	$V=1$
$X = 7$		0	1	1	1	+
$+Y = +3$		0	0	1	1	
$Z = -6$		1	0	1	0	

ผลลัพธ์ของบวก  $7 + 3 = 10$  แต่ใน 4 บิตจะเก็บได้แค่  $0-7$  ดังนั้นเกิด Overflow

การคำนวณ  $0 + 3 = 3$  แต่ใน 4 บิตจะเก็บได้แค่  $0-7$  ดังนั้นเกิด Overflow

ผลลัพธ์ของบวก  $-6 + 3 = -3$  แต่ใน 4 บิตจะเก็บได้แค่  $0-7$  ดังนั้นเกิด Overflow

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

31

## 2.3.2 คณิตศาสตร์เลขจำนวนเต็ม ชนิดมีเครื่องหมาย 2-Complement

ตัวอย่างที่ 2.3.7. จงคำนวณหาค่าของ  $-3 - 6 = (-3) + (-6)$  ดังนั้น ในเครื่องคอมพิวเตอร์ขนาด 4 บิต สามารถคำนวณได้แต่ผลลัพธ์กลับไม่ถูกต้องเนื่องจากเกิดโอเวอร์โฟล์ว ดังนี้

	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$	Overflow
	1	0	0	0	0	$V=1$
$X = -3 +$		1	1	0	1	+
$-Y = -6$		1	0	1	0	
$Z = +7$	1	0	1	1	1	

ผลลัพธ์ของบวก  $-3 + -6 = -9$  แต่ใน 4 บิตจะเก็บได้แค่  $0-7$  ดังนั้นเกิด Overflow

เนื่องจากฮาร์ดแวร์คำนวณ  $c_4 \text{ xor } c_3 = 1 \text{ xor } 0 = 1$  ทำให้ตรวจสอบได้ว่าเกิด Overflow ทำให้ผลลัพธ์มีค่าไม่ถูกต้อง

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

32

## 2.4 เลขทศนิยมฐานสองชนิดจุดคงที่ (Fixed Point)

นิยามที่ 2.4.1. กำหนดให้  $F_2$  เป็นเลขทศนิยมฐานสองชนิด Signed Magnitude เขียนอยู่ในรูป

$$F_2 = [s][x_{n-1}x_{n-2}x_{n-3}\dots x_1x_0].[y_{-1}y_{-2}y_{-3}\dots y_{-m}] \quad (2.48)$$

นิยมใช้ชนิดขนาด-เครื่องหมาย ประกอบด้วย 3 ส่วน คือ บิตเครื่องหมาย (Sign bit:  $s$  หรือ ±) โดย  $s=0$  แทนเครื่องหมายบวก และ  $s=1$  แทนเครื่องหมายลบ ส่วนจำนวนเต็ม (Integer:  $X_{2,u}$ ) มีความยาว  $n$  บิต และส่วนทศนิยม (Fraction:  $F_2$ ) ยาว  $m$  บิต รวมความยาวทั้งหมด  $m+n+1$  บิต

โดย  $F_{10}$  คือ ค่าฐานสิบของเลขทศนิยมฐานสองชนิดจุดคงที่  $F_2$  สามารถคำนวณได้จาก

$$F_{10} = \pm[x_{n-1}2^{n-1} + \dots + x_12^1 + x_02^0 + y_{-1}2^{-1} + y_{-2}2^{-2} + y_{-3}2^{-3} + \dots + y_{-m}2^{-m}] \quad (2.49)$$

หรือ

$$F_{10} = \pm[x_{n-1}2^{n-1} + \dots + x_12^1 + x_02^0 + \frac{y_{-1}}{2} + \frac{y_{-2}}{4} + \frac{y_{-3}}{8} + \dots + \frac{y_{-m}}{2^m}] \quad (2.50)$$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

33

ตัวต่อๆ กัน fraction  $\rightarrow$  1010 y คือ ปีนัง 4 เซนติเมตร

## 2.4 เลขทศนิยมฐานสองชนิดจุดคงที่ (Fixed Point)

ตัวอย่างที่ 2.4.1. จงแปลงเลขฐานสิบต่อไปนี้เป็นเลขทศนิยมฐานสองชนิดจุดคงที่  $m=n=2$  บิต

$$\frac{1}{2} + \frac{1}{4}$$

$$+0.75_{10} = 000.11_2 \quad (2.51)$$

$$+3.00_{10} = 011.00_2 \quad (2.52)$$

$$-3.75_{10} = 111.11_2 \quad (2.53)$$

ผู้อ่านสามารถสามารถเขียนเลขทศนิยมฐานสองชนิดจุดคงที่ได้ตามรูปแบบนี้

$$F_2 = [s][X_{2,u}].[Y_2] \quad (2.54)$$

ค่าทศนิยม ( $Y_2$ ) มีความยาว  $m$  บิต เขียนเป็นสัญลักษณ์ได้ดังนี้

$$Y_2 = y_{-1}y_{-2}y_{-3}\dots y_{-m} \quad (2.55)$$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

34

## 2.5 เลขทศนิยมฐานสองชนิดจุดลอยตัว (Floating Point)

นิยามที่ 2.5.1. เลขทศนิยมชนิดจุดลอยตัวฐานสองที่อยู่ในรูปนอมัลไลซ์ ประกอบด้วย 3 ส่วน คือ บิทเครื่องหมาย (Sign bit:  $s$ ) ค่านัยสำคัญ (Significand:  $S_2$ ) และค่ายกกำลัง (Exponent:  $E_2$ ) มีลักษณะดังนี้

$$\pm [1.y_{-1}y_{-2}y_{-3}\dots y_{-m}]_2 \times 2^{E_2} \quad (2.59)$$

เลขยกกำลังเป็นเลขจำนวนเต็มฐานสองชนิด sign-magnitude ความยาว  $n$  บิท ดังนี้

$$E_2 = \pm[e_{n-1}e_{n-2}\dots e_0]_2 \quad (2.60)$$

เมื่อ  $e_i$  แต่ละบิทมีค่า “1” หรือ “0” ในตำแหน่งที่  $i$   $s$  คือ Sign bit และ  $n$  คือ จำนวนบิทซึ่งกำหนดไว้ก่อนจะออกแบบโครงสร้าง

## 2.5 เลขทศนิยมฐานสองชนิดจุดลอยตัว (Floating Point)

จากนิยามที่ 2.59 ค่านัยสำคัญ (Significand)  $S_2$  ความยาว  $m+1$  บิท สามารถเขียนใหม่ได้ดังนี้

$$S_2 = [1.y_{-1}y_{-2}y_{-3}\dots y_{-m}]_2 \quad (2.61)$$

ซึ่งมีความสำคัญต่อรูปแบบการเขียน เนื่องจากจะต้องทำการนอมัลไลซ์ผลลัพธ์ที่ได้จากการคำนวณเสมอ ค่านัยสำคัญที่น้อมัลไลซ์ขึ้นต้นสามารถคำนวณหาค่าฐานสิบ ได้ดังนี้

$$S_{10} = \pm\left[1 + \frac{y_{-1}}{2} + \frac{y_{-2}}{4} + \frac{y_{-3}}{8} + \dots + \frac{y_{-m}}{2^m}\right] \times (2^{\pm E_2}) \quad (2.62)$$

กฎ IEEE

$\downarrow$  (แทนค่าสักตัว  $(y_{-1} + \dots + y_{-m}) + bias \rightarrow \oplus$  เมื่อ)

## 2.5 เลขศนิยมฐานสองชนิดจุดลอยตัว (Floating Point)

ตัวอย่างที่ 2.5.2. จงแปลงเลขศนิยมฐานสองชนิดจุดลอยตัวที่น้อมลัลซ์แล้ว  $k=4$  บิตให้เป็นเลขศนิยมฐานสิบ

วิธีทำ

$$(-1) \times 1.1010_2 \times 2^{-3} \rightarrow \text{แปลงเป็นฐาน } 10 \text{ ต่อ ?}$$

- ② 1. ปรับจุดศนิยม เพื่อให้เป็นเลขฐานสองชนิด Sign-Magnitude และเลขยกกำลังเท่ากับ 0  
~~○ ห้ามซ้อน~~ ① ป้องเป็นตัว 0

2. แปลงค่าเลขฐานสองชนิดที่เลื่อนตำแหน่งแล้วให้เป็นฐานสิบ

$$(1 \times 2^3) + (1 \times 2^{-4}) + (1 \times 2^{-6})$$

$$= 0.125 + 0.0625 + 0.015625$$

$$= 0.203125_{10}$$

(บล็อก) สรุปเรื่องจุดลอยตัว

## 2.5 เลขศนิยมฐานสองชนิดจุดลอยตัว (Floating Point)

ตัวอย่างที่ 2.5.4. จงบวกเลขศนิยมฐานสองชนิดจุดลอยตัวที่น้อมลัลซ์แล้ว ดังต่อไปนี้

วิธีทำ

สมมติว่ามีเลขศนิยมฐานสองขนาด 4 บิต (4-bit)

บล็อก ①

$$1.000_2 \times 2^{-1} + -1.110_2 \times 2^{-2} \text{ หรือ } (0.5_{10} + -0.4375_{10})$$

1. เลื่อนตำแหน่งจุดศนิยมและปรับเลขยกกำลังที่มีค่าน้อยกว่า ให้ตรงกับเลขยกกำลังของเลขที่มีค่ามากกว่า

บล็อก ②

$$1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-1}$$

ตัวอย่างเช่น กดตัวหัวหิน  $\rightarrow 0.111_2 - 1 \rightarrow -2 \rightarrow$  (แต่ต้องกดหนึ่งครั้ง)  
กดหนึ่งครั้ง

$\begin{array}{r} 1 \\ 0 \\ 0 \\ 0 \\ - \end{array}$

2. บวกค่านัยสำคัญที่เลื่อนตำแหน่งแล้ว

$$(1.000_2 - 0.111_2) \times 2^{-1} = 0.001_2 \times 2^{-1} \quad \text{ตัวอย่างเช่น } 0.111_2 \text{ ถูกบวกเข้าไป } 1.000_2 \text{ แล้ว } 0.111_2 \text{ จะหายไป} \rightarrow \text{ตัวอย่างเช่น } 0.001_2 \times 2^{-1}$$

3. น้อมลัลซ์ค่าผลลัพธ์และตรวจสอบให้การบิดโดยเวอร์ไฟล์ร์และอันเดอร์ไฟล์ร์

$1.000_2 \times 2^{-4}$ , ไม่เกิดโดยเวอร์ไฟล์ร์และอันเดอร์ไฟล์ร์  $\rightarrow$  ตัวอย่างเช่น  $0.001_2 \times 2^{-1}$  จึงบวกเข้าไปในตัว 1 ได้

4. ปัดค่าให้เหลือ 4 บิตและอาจต้องน้อมลัลซ์เมื่อจำเป็น

$$1.000_2 \times 2^{-4} (\text{ไม่เปลี่ยนแปลง}) = 0.0625_{10}$$

ตัวอย่างเช่น  $0.001_2 \times 2^{-1}$

ພາບທີ່ແດນຕອນກາເຮົາ image, GPU, audio ອາການຊ່ວຍງັນກັບ  
CPU, arm ແລະ ເງິນ co-processor

↳ DSP + CPU = SoC

↳ SoC (System on Chip)  
ກ່ຽວຂ້ອງໂຄສະນາໄປຈະກຳປະກຳຈະຫຼວງຂະໜາດ CUM  
ແລ້ວ ອິເລັດ ລຽບນີ້ chip ເຕັມ  
↳ ຖະໜານສົດໃຈໆ, ດັ່ງໆຈຶ່ງ, ຊື້ນັ້ນ

## 2.6 ເລຂທສນີຍມჰູນສອງໜີດຈຸດລອຍຕ້ວມາຕຣჲນ IEEE 754

- ໝົດ Single precision (32-bit) ຕຽບກັບຕ້ວແປຮນິດ float ໃນພາກສາ C/C++ ແລະ Java
- ໝົດ Double precision (64-bit) ຕຽບກັບຕ້ວແປຮນິດ double ໃນພາກສາ C/C++ ແລະ Java ເວົ້ວໜ້າ  
ລ່າສຸດຂອງ IEEE 754 ຄື່ອ ປີ ປ.ສ. 2019 ຮາຍລະເວີດເພີ່ມເຕີມ

ຕ້ວອຍ່າງການປະກາສແລະຕັ້ງຄ່າຕ້ວແປຣທີ່ໃໝ່ມາຕຣჲນນີ້

*int a = -5; a ~ 0xFFFFFFFF*  
float a = -5; /\* a = 0xCOA00000 \*/  
double b = -0.75; /\* b = 0xBFE8000000000000 \*/

ກໍາລຟ ກໍາໄລມາແນ່ວ່າ int → -5 ຂີ່:

↳ ໂນກວ່າມີມາດນີ້ລະວະລາຍງານໄວ້ວສ → ຕອນສິ່ງດັ່ງຕາໆເປົ້ອດັດ ສີ່ນິ້ງ assembly ດ້ວຍຕົວລົງໄຟການຕັ້ງຕາມ

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

add add

39

for every cpu.

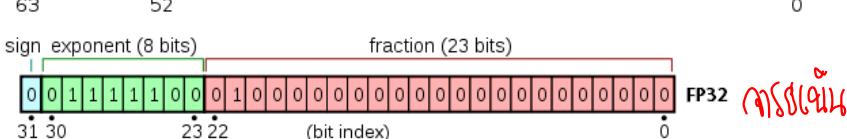
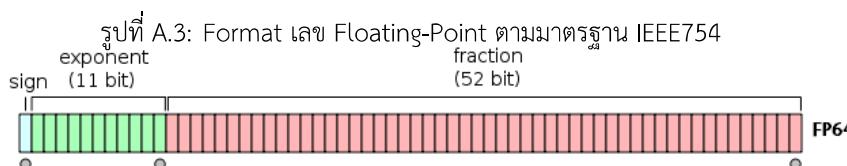
## 2.6 ເລຂທສນີຍມჲນສອງໜີດຈຸດລອຍຕ້ວມາຕຣჲນ IEEE 754

- ໝົດ Single precision (32-bit) ຕຽບກັບຕ້ວແປຮນິດ float ໃນພາກສາ C/C++ ແລະ Java
- ໝົດ Double precision (64-bit) ຕຽບກັບຕ້ວແປຮນິດ double ໃນພາກສາ C/C++ ແລະ Java

single: 8 ບີທ  
double: 11 ບີທ

single: 23 ບີທ  
double: 52 ບີທ

S	Exponent	Fraction
---	----------	----------



ອັນດັບຕົວຢ່າງ

ກ່ຽວຂ້ອງແນວໃຈ single

40

ຕົ້ນໄຫຍ້ມາ ທີ່ລະອົບອົບ

ລວມທີ່ເກີຍ  $-0.75 \rightarrow (-1)^1 \times (0.11_2)$

ຕາງການ normalized  $(-1)^1 \times (1.1_2) \times (2)^{-1}$

Fraction = 100 0000 0000 0000 0000 ...  
Exponent = -1 + Bias

## 2.6 ເລຂທສນິຍມຈູານສອງໜິດຈຸດລອຍຕ້ວມາຕຣຈູານ IEEE 754

ເລຂຈູານສີບ -0.75  
 $-0.75 = (-1)^1 \times 0.11_2 = (-1)^1 \times 1.1_2 \times 2^{-1}$  ນີ້ແຫຼ່ງຈະຫົວ\*  
 Sign = 1 ້າ້າ  
 Fraction = 100 0000 0000 0000 0000 #ອັນນັສຄູຕາຕື່ອງ  
 Exponent = -1 + Bias ດີກໍາເນົາ ນີ້ແຫຼ່ງຂອງນອກ  
 Single (8 ບີທ):  $-1 + 127 = 126 = 0111 1110_2$  \* if we understand, teacher happy  
 Double (11 ບີທ):  $-1 + 1023 = 1022 = 011 1111 1110_2$   
 Single: [1][011 1111][0100 0000 0000 0000 0000] = BF40 0000<sub>16</sub>  
 Double: [1][011 1111 1110][1000 0000 0000 .. 0000] = BFE8 0000 0000 0000<sub>16</sub>

ຂາດໂທແບ່ນຊີ້ານ 2  $\rightarrow$  10

## 2.6 ເລຂທສນິຍມຈູານສອງໜິດຈຸດລອຍຕ້ວມາຕຣຈູານ IEEE 754

ຕ້ວຍຢ່າງທີ່ 2.6.1. ຈຶ່ງທາຄາ່ເລຂທສນິຍມຈູານສີບຂອງເລຂ FP-32 ທີ່ເຕີມໃນຮູບທີ່ 2.7 ອື່ນ ເລຂທສນິຍມມາຕຣຈູານ IEEE 754 ຊືດ Single Precision ດັ່ງຕໍ່ໄປ

$F_{10, IEEE} = 1.01 \times 2^{(124-127)}$  ກຳນົດເກີຍເກີຍ normalized ແລະ 1/2 ຕາມແປຣກສັນ ທີ່ມີນັກສູງອັນດັບຂະໜາດ

$\begin{array}{c} \text{(+)} \\ \text{exp} \\ \downarrow \\ \text{exponent} \end{array} \quad \begin{array}{c} \text{Fractn} \\ \downarrow \\ F_{10, IEEE} = 1.01 \times 2^{(124-127)} \\ = 1.01 \times 2^{-3} \\ = 0.00101_2 \\ = 2^{-3} + 2^{-5} \\ = \frac{1}{2^3} + \frac{1}{2^5} \\ = 0.125_{10} + 0.03125_{10} = 0.15625_{10} \end{array} \quad \begin{array}{l} \text{ມີມູນຄຸນ} \\ \text{ລົງທະບຽນ} \\ \text{ກຳນົດເກີຍ} \\ \text{ກຳນົດເກີຍ} \\ \text{ກຳນົດເກີຍ} \\ \text{ກຳນົດເກີຍ} \end{array}$

(2.68) (2.69) (2.70) (2.71) (2.72) (2.73)

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

ตัวอย่างที่ 2.6.2. จงแปลงเลข  $-0.75_{10}$  เป็นเลขทศนิยมฐานสองชนิดจุดลอยตัวตามมาตรฐาน IEEE 754  
ทั้งสองชนิด

วิธีทำ

$$\begin{aligned}-0.75_{10} &= (-1)^1 \times (0.\underline{5}_{10} + 0.\underline{25}_{10}) \\&= (-1)^1 \times (\frac{1}{2} + \frac{1}{4}) \\&= (-1)^1 \times (0.1_2 + 0.01_2) \\&= (-1)^1 \times 0.11_2 \times 2^0\end{aligned}$$

ทำการน้อมล้ำเล็กตามสมการที่ 2.59

$$-0.75_{10} = (-1)^1 \times 1.1_2 \times 2^{-1} \rightarrow \text{ใช้คลาคีต์ normalize นำ expo + bias}$$

ดังนั้น บิตเครื่องหมาย  $s = 1$

ค่าทศนิยม  $Y_2 = 100\ 0000\ 0000\ 0000\ 0000\ 0000_2$

ค่ายกกำลัง  $E_2 = -1 + E_{bias}$

โดยชนิด Single ค่ายกกำลัง (8 บิต):  $E_2 = -1 + 127 = 126$  หรือ  $E_2 = 0111\ 1110_2$

โดยชนิด Double ค่ายกกำลัง (11 บิต):  $E_2 = -1 + 1023 = 1022$  หรือ  $E_2 = 011\ 1111\ 1110_2$

ดังนั้น  $-0.75_{10}$  เชียนในรูปของเลขทศนิยมชนิดจุดลอยตัวตามมาตรฐาน IEEE 754 ชนิด

Single:  $[1][011\ 1111\ 0][100\ 0000\ 0000\ 0000\ 0000]_2 = BF40\ 0000_{16}$

Double:  $[1][011\ 1111\ 1110][1000\ 0000\ 0000\ ...0000]_2 = BFE8\ 0000\ 0000\ 0000_{16}$

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

ตัวอย่างที่ 2.6.3. เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754 ชนิด Single-Precision ต่อไปนี้ มีค่าเท่าไรในฐานสิบ

$$C\ 0\ A\ 0\ 0\ 0\ 0\ 0_{16} = [1100\ 0000\ 1010\ 0000\ 0000\ 0000\ 0000\ 0000]_2$$

วิธีทำ

แปลงจากเลขฐานสิบให้เป็นฐานสองและองค์ประกอบต่างๆ ได้ดังนี้

$$s = [1][E_2 = 100\ 0000\ 1][Y_2 = 010\ 0000\ 0000\ 0000\ 0000]_2$$

จะพบว่าบิตเครื่องหมาย  $s = 1$

$$\text{ค่ายกกำลังจริง } E_{true} = \underline{1000\ 0001}_2 - E_{bias} = 129 - 127 = 2$$

$$\text{ค่าทศนิยม } Y_2 = 010\ 0000\ 0000\ 0000\ 0000\ 0000_2$$

$$F_{10, IEEE} = (-1)^1 \times (1 + .01_2) \times 2^2 \quad \text{ผลลัพธ์ต่อไป} \quad (2.74)$$

$$= (-1)^1 \times (1.01_2) \times 2^2 \quad (2.75)$$

$$= (-1)^1 \times (101.0_2) \quad (2.76)$$

$$= (-1) \times 5.0 \quad (2.77)$$

$$= -5.0_{10} \quad (2.78)$$

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

ตัวอย่างที่ 2.6.5. การบวกเลขทศนิยมฐานสองชนิดจุดลอยตัว (Single Precision) มาตรฐาน IEEE754

$-5_{10} + -0.75_{10}$  หรือเท่ากับ  $\text{COA}0\ 0000_{16} + \text{BF}40\ 0000_{16} = ?$

$$\begin{array}{r} 1100\ 0000\ 1010\ 0000\ 0000\ 0000\ 0000_2 \\ + \\ 1011\ 1111\ 0100\ 0000\ 0000\ 0000\ 0000_2 \end{array}$$

วิธีทำ

① ทำให้เป็น normalized ก่อน

ผู้อ่านจะต้องแปลงเลขฐานสองให้เป็นรูปแบบเลขฐานสองที่น่อมัลลิลีซ์ ดังต่อไปนี้

$$(-1)^1 \times (1.010\ 0000\ 0000\ 0000\ 0000)_2 \times 2^2 +$$

$$(-1)^1 \times (1.100\ 0000\ 0000\ 0000\ 0000)_2 \times 2^{-1}$$

1. เลื่อนตำแหน่งจุดทศนิยมและปรับเลขยกกำลังที่มีค่าน้อยกว่า ให้ตรงกับเลขยกกำลังของเลขที่มีค่ามากกว่า

$$(-1)^1 \times (1.010\ 0000\ 0000\ 0000\ 0000)_2 \times 2^2 +$$

$$(-1)^1 \times (0.001\ 1000\ 0000\ 0000\ 0000)_2 \times 2^2$$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

45

เขียนลงในกระดาษได้ ก็จะได้

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

2. บอกค่านัยสำคัญที่เลื่อนตำแหน่งแล้ว

$$= (-1)^1 \times (1.011\ 1000\ 0000\ 0000\ 0000)_2 \times 2^2$$

3. น้อมัลลิลีซ์ค่าผลลัพธ์ และตรวจเช็คการเกิดโอเวอร์ไฟล์ว์และอันเดอร์ไฟล์ว์

$$(-1)^1 \times (1.011\ 1000\ 0000\ 0000\ 0000)_2 \times 2^2 \Rightarrow \text{ไม่เกิดโอเวอร์ไฟล์ว์และอันเดอร์ไฟล์ว์}$$

4. ปัดค่าให้เหลือ 23 บิตและอาจต้องน้อมัลลิลีซ์เมื่อจำเป็น

$$(-1)^1 \times (1.011\ 1000\ 0000\ 0000\ 0000)_2 \times 2^2$$

$$s = 1$$

$$\text{ค่ายกกำลัง } E_2 = 2 + 127 = 129 = 1000\ 0001_2$$

$$\text{ค่าทศนิยม } Y_2 = 011\ 1000\ 0000\ 0000\ 0000_2$$

$$F_{2,\text{IEEE}} = [1][100\ 0000\ 1][011\ 1000\ 0000\ 0000\ 0000]_2 \Rightarrow C0B8\ 0000_{16}$$

แปลงเป็นบิต + bias (คงไว้)

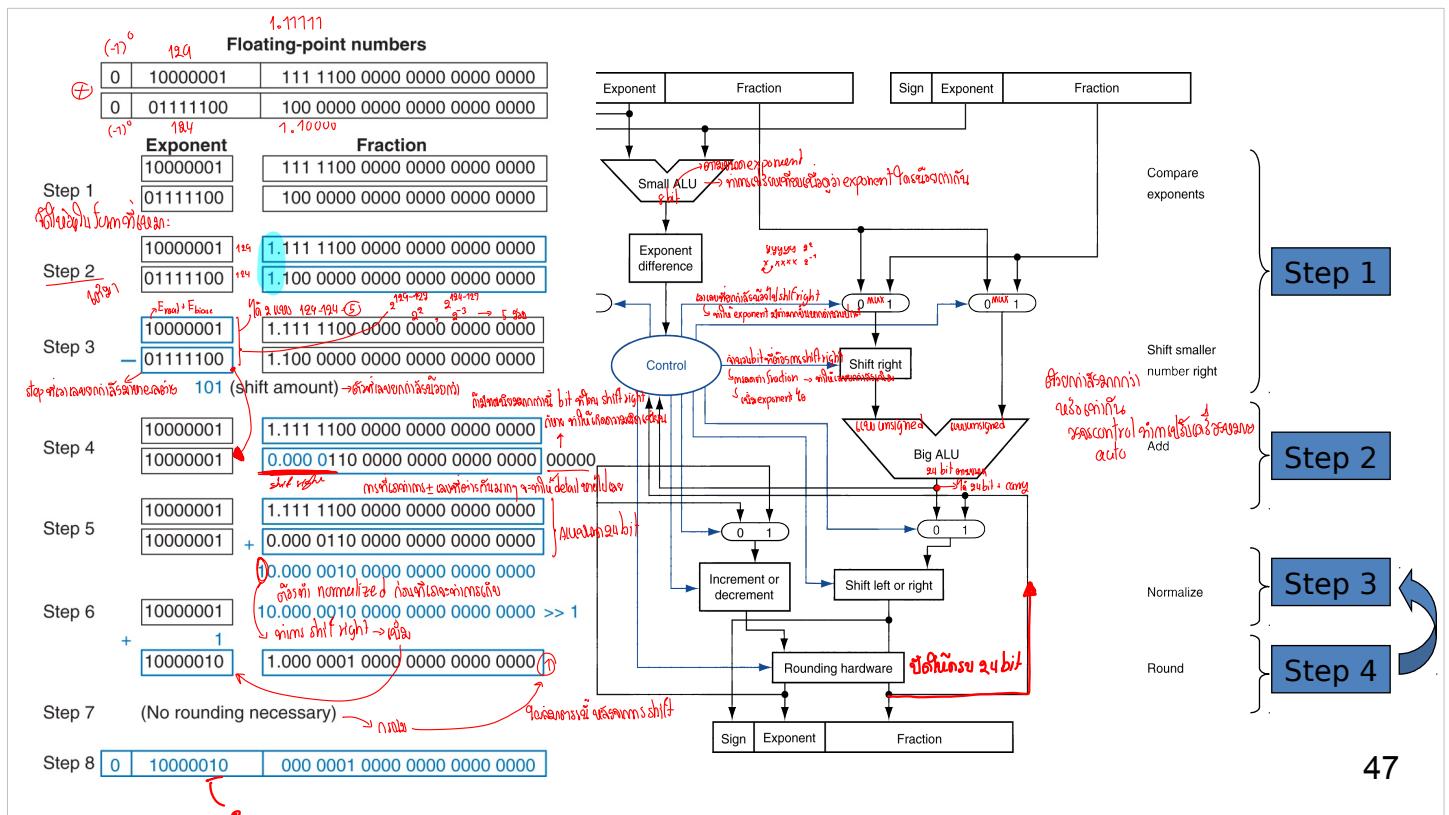
Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

46

ตัว fraction จะมีส่วนหนึ่งที่ต้องเดินไปต่อไป รู้ว่าในรูป normalized จะเหลือ 1

จะ剩ลง ที่จะไม่ต้องนำไปเสียเวลา, ซึ่งดีในการคำนวณ

## \* ចារចាយបិន្នោម + លេ single



47

## 2.6 លេខសិនិមត្តនាសងខ្លួនឯកជុំលើតំវោមាព្យាន IEEE 754

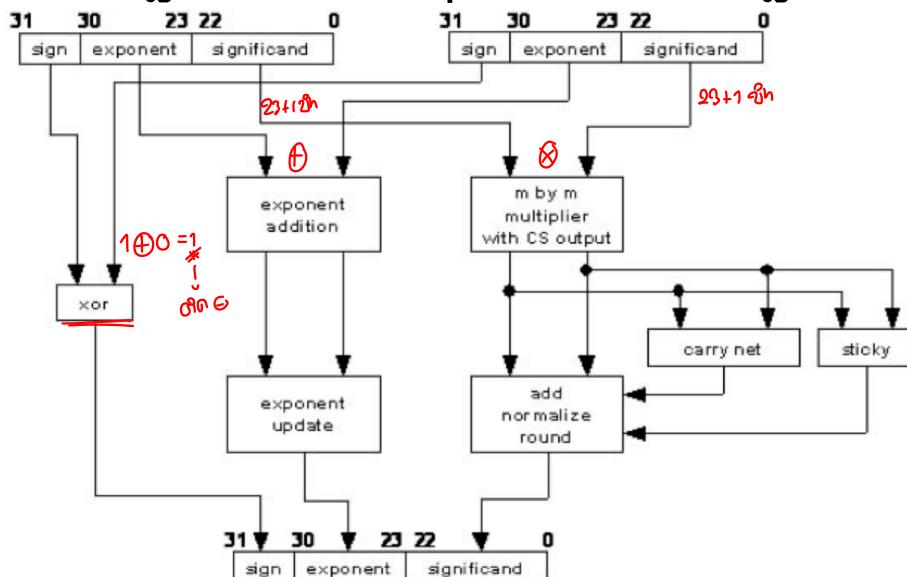


Figure 2.10: វងគ្រុណលេខសិនិមត្តនាសងខ្លួនឯកជុំលើតំវោមាព្យាន IEEE 754 ໂដយរៀបចាប់អិនុញ្ញា  
ចាប់ពីលក្ខណៈទី 2 ដោយបិន្នោម និងខ្លួនឯកជុំលើពុំផែលក្ខណៈទី 1 ។ (ម៉ា: Patterson and Hennessy (2000))

48

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

ตัวอย่างที่ 2.6.7. จงคูณเลขทศนิยมฐานสองชนิดจุดลอยตัวแบบ Single Precision ดังต่อไปนี้

$$EX: -5_{10} \times -0.75_{10} = COA0\ 0000_{16} \times BF40\ 0000_{16} = ?$$

1. แปลงเลขฐานสิบหกให้เป็นเลขฐานสอง

$$\begin{array}{r} 1100\ 0000\ 1010\ 0000\ 0000\ 0000\ 0000_2 \times \\ |2^4| \\ 1011\ 1111\ 0100\ 0000\ 0000\ 0000\ 0000_2 \end{array}$$

2. แปลงเลขฐานสองตามกฎของ IEEE 754 Single Precision รูปแบบเลขฐานสองที่น้อมลัลไล์ซ์ ดังต่อไปนี้

$$\begin{aligned} &= (-1)^1 \times (1.010\ 0000\ 0000\ 0000\ 0000)_2 \times 2^{127} \\ &= (-1)^1 \times (1.100\ 0000\ 0000\ 0000\ 0000)_2 \times 2^{-1} \end{aligned}$$

3. บวกเลขยกกำลังเข้าด้วยกัน:  $[2^2 \times 2^{-1}] = 2^1$

$$\begin{aligned} &= (-1)^1 \times (-1)^1 \times [(1.010\ 0000\ 0000\ 0000\ 0000)_2 \times \\ &\quad (1.100\ 0000\ 0000\ 0000\ 0000)_2] \times [2^2 \times 2^{-1}] \end{aligned}$$

ผลลัพธ์ของเลขยกกำลัง

การบวกสัมประสิทธิ์

$\begin{array}{r} 1010 \\ 1100 \\ \hline 0000 \end{array}$  Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

49

$\begin{array}{r} 1010 \\ 1100 \\ \hline 0000 \end{array}$  กดปุ่ม 2x

→ ดูรายละเอียดการปักติ → ตัวอย่าง → คำอธิบายละเอียด → มีนาทีๆ เวลาใช้เวลาโดยประมาณ

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

4. คูณค่านัยสำคัญเข้าด้วยกัน

$$\begin{aligned} &1.010\ 0000\ 0000\ 0000\ 0000_2 \times \\ &\quad _{-1.100\ 0000\ 0000\ 0000\ 0000_2} \\ &= 1.111\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_2 \end{aligned}$$

5. น้อมลัลไล์ซ์ค่าผลลัพธ์ และตรวจสอบการเกิดโอลิฟลิวและอันเดอร์โอลิว

$$1.111\ 0000\ 0000\ 0000\ 0000_2 \times 2^1 \text{ (ค่าเป็นปกติ)}$$

6. ปัดค่าให้เหลือ 23 บิตและอาจต้องน้อมลัลไล์ซ์อีกรอบหากจำเป็น

$$1.111\ 0000\ 0000\ 0000\ 0000_2 \times 2^1 \text{ (ไม่มีการเปลี่ยนแปลงเกิดขึ้น)}$$

7. ปรับเครื่องหมายของผลลัพธ์ให้ถูกต้องจากตัวตั้งและตัวคูณ

$$(-1)^0 \times (1.111\ 0000\ 0000\ 0000\ 0000_2 \times 2^1)$$

ดังนั้น  $-5_{10} \times -0.75_{10} = 3.75_{10}$  และในเครื่องคอมพิวเตอร์จะเห็นเป็นค่า

บิตเครื่องหมาย  $s = 0$

$$COA0\ 0000_{16} \times BF40\ 0000_{16} = 4070\ 0000_{16}$$

$$\text{ค่ายกกำลัง } E_2 = 1+127 = 128 = 1000\ 0000_2$$

$$\text{ค่าทศนิยม } Y_2 = 111\ 0000\ 0000\ 0000\ 0000_2$$

$$F_{2,IEEE} = 0100\ 0000\ 0111\ 0000\ 0000\ 0000\ 0000_2 \Rightarrow 4070\ 0000_{16}$$

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

50

This is summary

- msl នៃ underflow
  - សរុបតម្លៃចាប់ពី 10 ដល់ 100
  - ការបង្ហាញនៃតម្លៃដែលបានបង្ហាញជាអត្ថបទ normalized
  - ការបង្ហាញជាតម្លៃដែលបានបង្ហាញជាអត្ថបទ denormalized
  - ដែលក្នុងសំណង់នេះ “underflow”

## 2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

Table 2.11: ตารางสรุปความแตกต่างระหว่างเลขทศนิยมฐานสองชนิดด้วยตัวตามมาตรฐาน IEEE 754

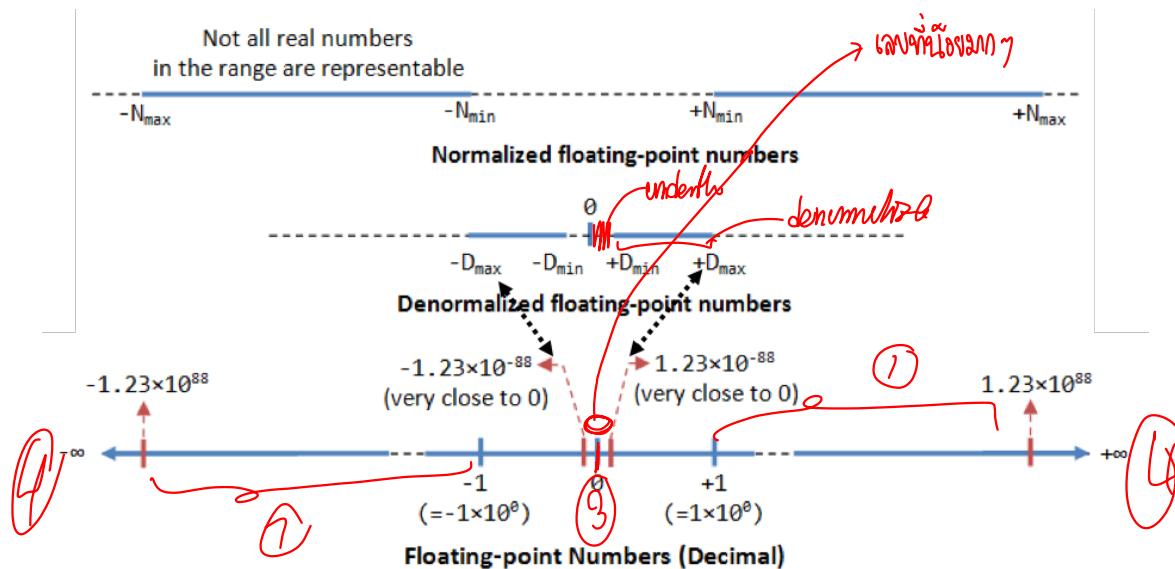
ชนิด Single Precision โดย  $E_2$  คือ ค่ายกกำลัง และ  $Y_2$  คือ เทคนิคที่เป็นส่วนประกอบของค่านัยสำคัญ ( $x$  หมายถึง บิทข้อมูลมีค่าเท่ากับ '0' หรือ '1') style precision

ลำดับที่	เครื่องหมาย $s$ (1 บิต)	ค่ายกกำลัง $E_2$ (8 บิต)	ทศนิยม $Y_2$ (23 บิต)	ความหมาย
1	±	$1\text{-}0000\ 0001_2$ ± $1111\ 1110_2$	xx..2	เลขฐานสิบหัวไป (น้อมลайл์ช)
2	±	$0000\ 0000_2$	xx..2	เลขฐานสิบหิน้อยมาก แต่ไม่เท่ากับศูนย์ (น้อมลайл์ช) → บวกลบยก
3	0	$0000\ 0000_2$	0...0 <sub>2</sub>	$0.0_{10}$ (ศูนย์จุดศูนย์)
4	±	$1111\ 1111_2$	$0\text{-}0_2$	$\pm\infty$ (±อินฟินิตี้)
5	0	$1111\ 1111_2$	xx..2	Nan (Not a Number)

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL  
ເລືອກທີ່ຕິດກ່າວສັນຍາທີ່ຈະນຳໃຈນີ້ໄດ້

51

2.6 เลขทศนิยมฐานสองชนิดจุดลอยตัวมาตรฐาน IEEE 754

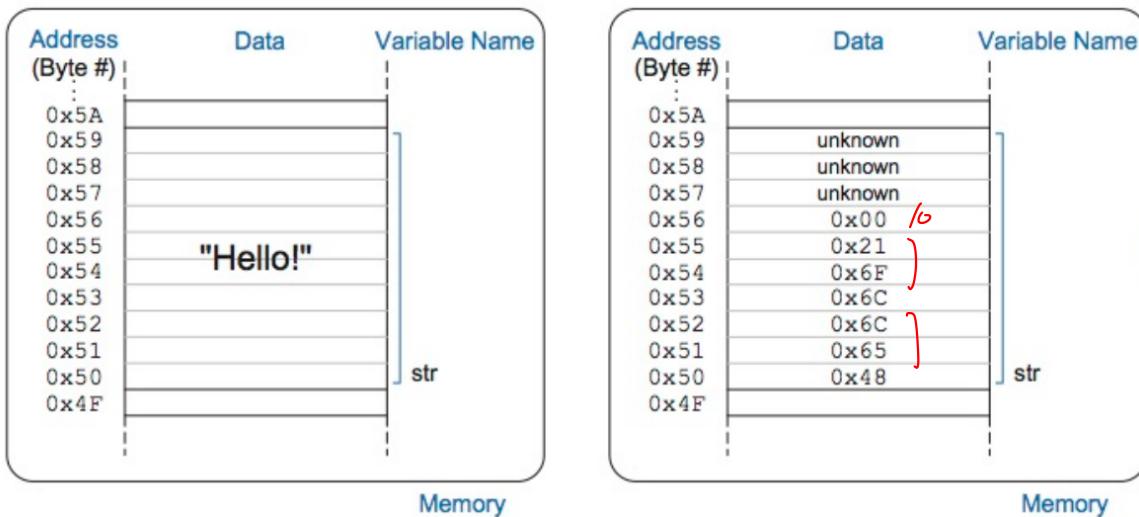


Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

52

## 2.7 ตัวอักษร (Character)

char [10] str="Hello!"



Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

53

กรอกข้อต่อ เลขแบบ follow with column

Codepage 874 - Thai

-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F		
0-	0011	0002	0003	0004	0005	0006	0007	0008	0009	000A	000B	000C	000D	000E	000F		
1-	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	001A	001B	001C	001D	001E	001F	
2-	0020	!	"	#	\$	%	&	'	(	)	*	+	,	-	/	002F	
3-	0030	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	? 003F
4-	0040	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O 004F
5-	0050	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	^	_	- 005F
6-	0060	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o 006F
7-	0070	p	q	r	s	t	u	v	w	x	y	z	{	}	~		007F
8-																	
9-																	
A-	0E48	ก	ຂ	҆	҇	҈	҉	Ҋ	ҋ	Ҍ	ҍ	Ҏ	ҏ	Ґ	ґ	Ғ	ғ 0E5F
B-	0E10	ໜ	ໝ	໘	໙	໚	໛	ໜ	໚	໗	໘	໙	໚	໛	ໜ	໚	໗ 0E1F
C-	0E20	ກ	ມ	ຢ	ຮ	ດ	ຕ	ຖ	ກ	ຈ	ນ	ບ	ປ	ຜ	ພ	ຟ 0E2F	
D-	0E30	ດ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ	ກ 0E3F	
E-	0E40	ເ	ແ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ	ໂ 0E4F	
F-	0E50	ອ	ກ	ໂ	ຕ	ແ	ແ	ວ	ແ	ກ	ສ	ທ	ພ	ອ	ຫ	ໝ 0EAF	

## 2.7 ตัวอักษร (Character)

รหัส Unicode ถูกกำหนดให้เป็นมาตรฐานโดย ISO (International Standard Organization) เพื่อมาตรฐานทั่วโลกสำหรับ ASCII เนื่องจากความต้องการใช้ภาษาที่ไม่ใช่英文เรื่อยๆ รหัส Unicode กำหนดวิธีการเข้ารหัสที่หลากหลายตามวัตถุประสงค์การใช้งาน เช่น รหัส UTF-8 รหัส UTF-16 รหัส UCS-2 เป็นต้น

- รหัส UTF-8 นิยมใช้ในเว็บเพจต่างๆ โดยแต่ละตัวอักษรจะให้ความยาว 1 ไบต์ ดังต่อไปนี้ ไบต์ 1 โอดตัวอักษร 128 ตัวแรกของรหัส ASCII ใช้ความยาว 1 ไบต์ ส่วนตัวอักษรในภาษาอื่นๆ จะให้จำนวนไบต์เพิ่มขึ้น โดยตำแหน่งเริ่มต้นของตารางรหัส Unicode จะเหมือนกับตารางรหัส ASCII ตัวภาษาอังกฤษและภาษาไทย ในรูปที่ 2.12 ภายใต้แต่ละตัวอักษร มีค่ารหัส Unicode กำกับอยู่ด้วยยกหัวอย่างเช่น ตัวอักษรไทยเริ่มต้นที่รหัส ASCII เท่ากับ A1 คือ ก ในรูปของเลขฐานสองขนาด 8 บิต ตรงกับรหัส Unicode 0E01 ความยาว 16 บิต

- รหัส UCS-2 จะใช้พื้นที่ 2 ไบต์ หรือ 16 บิต ต่อ 1 ตัวอักษร ซึ่งทำให้สามารถใช้เฉพาะภาษาไทยได้ 216 หรือ 65,536 แบบมาตราฐานตัวอักษร ซึ่งทำให้วิธีการนี้ได้รับความนิยม

- รหัส UTF-16 คือ การขยายรหัส UCS-2 ให้ทันสมัยมากขึ้น โดยเพิ่มการเข้ารหัสเป็นขนาด 4 ไบต์ คือ

Computer Organization & ARM Assembly Language: RaspberryPi3, Surin K., KMITL

54

## สรุปท้ายบท

Table 2.12: ชนิด ความยาว ข้อมูล และการประยุกต์ใช้งานเลขฐานสองชนิดต่างๆ ในคอมพิวเตอร์

ชนิด	บิต	ข้อมูล	การประยุกต์ใช้งาน
char	8	ตัวอักษร	ข้อความ อีเมล
unsigned char	8	จุดภาพ	รูปภาพขาวดำ และ Gray Scale
unsigned char	8		รูปภาพสี RGB Bitmap JPEG
unsigned int	32/64	แอดเดรส	พอยท์เตอร์ซึ่งตำแหน่งข้อมูล ระบบ 32/64 บิต
unsigned int	32/64	จำนวน	จำนวนอุปกรณ์ IoT จำนวนดาวต่างๆ
int	32/64		
ทศนิยม Fixed	16-32	เสียง	ข้อมูลเสียงดนตรี
ทศนิยม Fixed	16-32	จุดภาพ	ข้อมูลภาพความละเอียดสูง
float	32	จุดภาพ	ข้อมูลภาพความละเอียดสูง
float	32	ระยะทาง	เกม 3 มิติ
double	64	± ระยะทาง	ระยะทางไปยังดาวต่างๆ นอกโลก
double	64	± นำ้มนัก	นำ้มนักดาวต่างๆ นำ้มนักอนุภาคเล็กๆ