w09-Lec

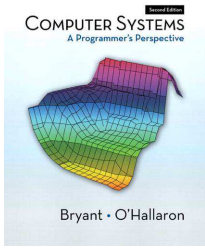
Data Representation

Part II – Floating Point

Assembled for 204111

by Kittipitch Kuptavanich

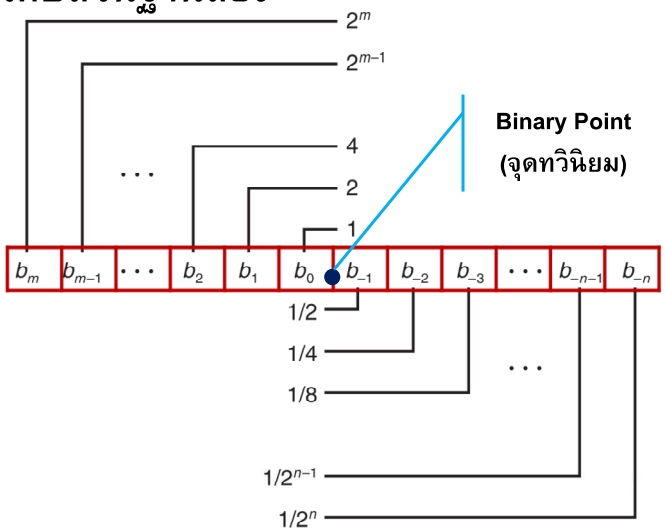
Floating Point



- **Fractional Binary Number**
- IEEE Floating Point
- Normalized Value
- Denormalized Value
- Special Value
- Rounding
- Floating Point Operation
- Floating Point in C

Fractional Binary Numbers

- เลขเศษส่วนฐานสอง



Fractional Binary Numbers [2]

- พิจารณาการแทนข้อมูลฐาน 10
  - เราไม่สามารถใช้เลขทศนิยมที่มีการจำกัดตำแหน่ง เพื่อแทนค่าที่แท้จริงของ 1/3 หรือ 5/7
  - เนื่องจากเป็นทศนิยมไม่รู้จบ: $0.\dot{3}$ และ $0.\dot{7}14285\dot{}$
  - 0.33333 ใกล้เคียงค่าจริงมากกว่า 0.33
- ในระบบเลขฐาน 2 นั้นมีข้อจำกัดเช่นเดียวกัน
  - แทน<u>ค่าจริง</u>ของตัวเลขได้ เฉพาะตัวเลขที่สามารถเขียนในรูป $x \times 2^y$ เท่านั้น
  - นอกจากนั้นจะเป็น<u>ค่าประมาณ</u> - ความใกล้เคียงกับค่าจริง<u>*ขึ้นกับ*</u>จำนวนตำแหน่งที่ใช้แสดงค่า

Fractional Binary Number [3]

Representation	Value	Decimal
0.0_2	$\frac{0}{2}$	0.0_{10}
0.01_2	$\frac{1}{4}$	0.25_{10}
0.010_2	$\frac{2}{8}$	0.25_{10}
0.0011_2	$\frac{3}{16}$	0.1875_{10}
0.00110_2	$\frac{6}{32}$	0.1875_{10}
0.001101_2	$\frac{13}{64}$	0.203125_{10}
0.0011010_2	$\frac{26}{128}$	0.203125_{10}
0.00110011_2	$\frac{51}{256}$	0.19921875_{10}

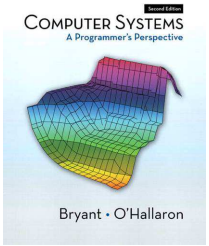
Practice Problem 1

เติมตารางต่อไปนี้ให้สมบูรณ์

Fractional value	Binary representation	Decimal representation
$\frac{1}{8}$	0.001	0.125
$\frac{3}{4}$		
$\frac{25}{16}$		
	10.1011	
	1.001	
		5.875
		3.1875

Floating Point

- Fractional Binary Number
- **IEEE Floating Point**
- Normalized Value
- Denormalized Value
- Special Value
- Rounding
- Floating Point Operation
- Floating Point in C



IEEE Floating Point

- **ในอดีตผู้ผลิตคอมพิวเตอร์แต่ละราย ต่างก็มีวิธีการแสดงค่าเลข Floating Point ที่แตกต่างกันออกไป** (ไม่มีมาตรฐานกลาง)
  - **ให้ความสนใจกับความเร็วมากกว่าค่าที่ถูกต้อง [-_-]**
- **IEEE standard 754**
  - **ตั้งขึ้นในปี 1985**
  - **มาตรฐานกลางสำหรับ Floating Point**
  - **รองรับโดย CPU ส่วนมาก**

Floating Point Representation

- การแทนค่า **Floating Point** อยู่ในรูปแบบ

$$(-1)^s \times M \times 2^E$$

ลักษณะเดียวกันกับการเขียน $204111 = 2.04111 \times 10^5$ ในฐาน **10**

- ***Sign Bit*** s เป็น bit ที่บอกว่าเป็นจำนวนบวกหรือลบ
- ***Significand*** M เป็นจำนวนในช่วง 1.0 ถึง 2.0 **(M: mantissa)**
- ***Exponent*** E กำหนดขนาดของจำนวนที่ต้องการแทนค่าในรูปกำลังที่ E ของ 2 (E สามารถมีค่าเป็น **+ 0** หรือ **-** ได้)

Bit-Level Representation

- **MSB** ใช้แทน ***Sign Bit*** s
- **Bit** ในช่วง `exp` ใช้แสดงค่า E ด้วยวิธีแปลงค่า
- **Bit** ในช่วง `frac` ใช้แสดงค่า M ด้วยวิธีแปลงค่า

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Precision options

- **Single precision: 32 bits**

- **Double precision: 64 bits**

from Carnegie Mellon University's 15213 course slide by Greg Kesden

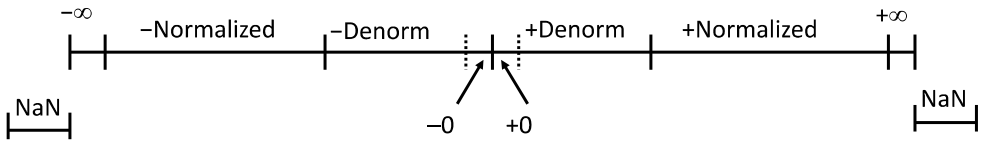
3 Cases of Value Represented

เราสามารถแบ่งจำนวนที่สามารถแสดงค่าด้วย มาตรฐาน **floating point** ได้ออกเป็น **3** กรณี โดยพิจารณาจากค่า `exp` เป็นหลัก

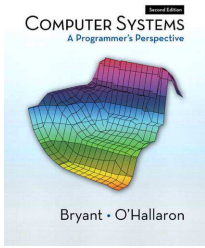
- **Case 1: Normalized Value**
  - กรณี `exp` ไม่ได้มีค่าเป็น **0** ทุก **bit** หรือ **1** ทุก **bit** ($exp \neq 000\ldots0$ **and** $exp \neq 111\ldots1$)
  - เป็นกรณีที่พบได้บ่อยที่สุด
- **Case 2: Denormalized Value**
  - กรณี `exp` มีค่าเป็น **0** ทุก **bit** ($exp = 000\ldots0$)
  - ใช้ แทนค่า **0** และตัวเลขที่ใกล้เคียง **0** มาก ๆ
- **Case 3: Special**
  - กรณี `exp` มีค่าเป็น **1** ทุก **bit** ($exp = 111\ldots1$)
  - **NaN (Not a number), infinities** (∞ หรือ $-\infty$)

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Visualization: Floating Point Encodings



Floating Point



- Fractional Binary Number
- IEEE Floating Point
- **Normalized Value**
- Denormalized Value
- Special Value
- Rounding
- Floating Point Operation
- Floating Point in C

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Case 1: Normalized Value

- กรณี: exp ≠ 000…0 **and** exp ≠ 111…1
- ค่า E คำนวณได้โดยการ แปลงจาก exp:

$$E = \text{exp} - \textit{Bias}$$

  - exp: **unsigned value**
  - ***Bias*** = $2^{k-1} - 1$, **where** k จำนวน **bit ของ** exp
    - **Single precision:** 127 (exp: 1…254, E: -126…127)
    - **Double precision:** 1023 (exp: 1…2046, E: -1022…1023)

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Case 1: Normalized Value [2]

- ค่า M คำนวนได้จาก สูตร:

$$M = 1.xxx\ldots x_2$$

  - $xxx\ldots x$: **bit จากส่วน** frac
  - ค่า **minimum เมื่อ** frac = 000…0 (M = 1.0)
  - ค่า **maximum เมื่อ** frac = 111…1 (M = almost 2.0)
  - เนื่องจากตัวเลขตัวแรก (หน้าสุด) มีค่า 1 เสมอ
    - ทำให้ไม่จำเป็นต้องใส่ bit นี้ลงไป ในการแทนค่าระดับ bit (เพิ่มเข้าไปอัตโนมัติตอนคำนวณ)

$(-1)^s \times M \times 2^E$

Example 1: Decoding

- **8 bit format**

  - **4 exp bit** (k)
  - **3 frac bit** (n)
- **คำนวณหาค่า** 0 0110 111
  - **bias** = $2^{4-1} - 1 = 7$

Bias = $2^{k-1} - 1$

E = exp – **Bias**

M = $1.xxx\ldots x_2$

e: exponent bits
f: fraction bits

Bit representation	Exponent e	E	2^E	Fraction f	M	Value $2^E \times M$	V	Decimal
0 0110 111	6	-1	1/2	7/8	15/8	15/16	15/16	0.9375

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Example 2: Encoding

- **Value:** `float F = 204111.0;`

204111_{10} = 11000111010100111_2
= $1.10001110101001111_2 \times 2^{17}$

- **Significand**

M = 1.10001110101001111
frac = 10001110101001111000000

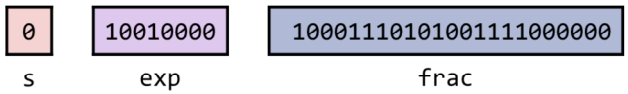
- **Exponent**

E = 17

Bias = 127

exp = 144 = 10010000_2

- **Result**



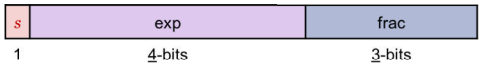
Practice Problem 2

Bias = $2^{k-1} - 1$

E = exp – **Bias**

M = $1.xxx\ldots x_2$

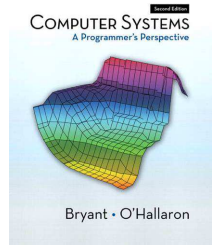
เติมตารางให้สมบูรณ์



Bit Representation	Exponent e	E	2^E	Fraction f	M	Value $2^E \times M$	V	Decimal
0 0001 000								
0 0110					14/8			
0		000		1				
0		010	7					
0		110				1792/8		
0 1110								240.0

e: exponent bits
f: fraction bits

Floating Point



- Fractional Binary Number
- IEEE Floating Point
- Normalized Value
- **Denormalized Value**
- Special Value
- Rounding
- Floating Point Operation
- Floating Point in C

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Case 2: Denormalized Value

- กรณี: exp = 000…0
- ค่า E คำนวณได้โดยการ แปลงจาก exp:

$$E = 1 - Bias$$

- ค่า M คำนวนได้จาก สูตร:

$$M = 0.xxx\ldots x_2$$

  - $xxx\ldots x$: bit จากส่วน frac

from Carnegie Mellon University's 15213 course slide by Greg Kesden

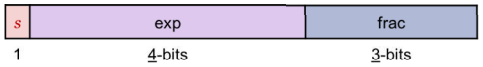
Case 2: Denormalized Value

- แบ่งเป็น 2 กรณี
  - exp = 000…0 และ frac = 000…0
    - แทนค่า 0
    - สังเกตการมีค่า +0 และ -0
  - exp = 000…0 และ frac ≠ 000…0
    - จำนวนที่มีค่าใกล้เคียง 0
    - ช่วงห่างระหว่างจำนวนเท่ากัน

Practice Problem 3

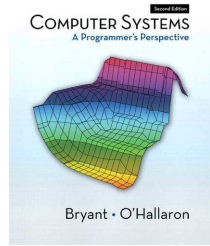
- $Bias = 2^{k-1} - 1$
- $E = 1 - Bias$
- $M = 0.xxx\ldots x_2$

เติมตารางให้สมบูรณ์



Bit Representation			Exponent			Fraction		Value		
			e	E	2^E	f	M	$2^E \times M$	V	Decimal
0	0000	000								
0	0000	001								
0	0000	010								
0	0000	011								
0	0000	110								
0	0000	111								

Floating Point



- Fractional Binary Number
- IEEE Floating Point
- Normalized Value
- Denormalized Value
- **Special Value**
- Rounding
- Floating Point Operation
- Floating Point in C

from Carnegie Mellon University's 15213 course slide by Greg Kesden

case 3: Special Value

- กรณี: exp = 111…1
- แบ่งเป็น 2 กรณี
  - exp = 111…1 และ frac = 000…0
    - แทนค่า ∞ (infinity)
    - สำหรับ operation ที่ทำให้เกิดการ overflow
    - ทั้งกรณี +∞ และ -∞
    - เช่น 1.0/0.0 = -1.0/-0.0 = +∞, 1.0/-0.0 = -∞
  - exp = 111…1 และ frac ≠ 000…0
    - Not-a-Number (NaN)
    - กรณีที่ไม่สามารถแทนผล Operation ด้วยตัวเลข Floating Point ได้
    - เช่น $\sqrt{(-1)}$, ∞ - ∞, ∞ × 0

Visualization of 6 Bit Representation

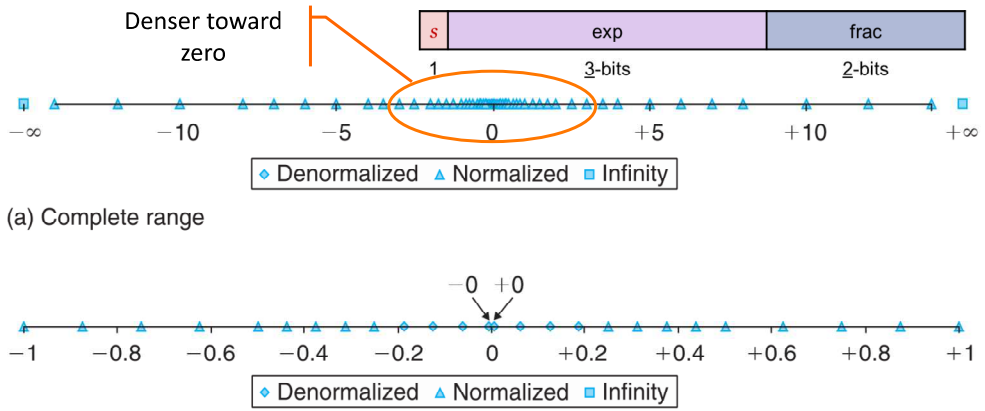


Figure 2.33 **Representable values for 6-bit floating-point format.** There are $k = 3$ exponent bits and $n = 2$ fraction bits. The bias is 3.

from Carnegie Mellon University's 15213 course slide by Greg Kesden

Dynamic Range (Positive Only)

	s	exp	frac	E	Value	
Denormalized numbers	0	0000	000	-6	0	
	0	0000	001	-6	1/8*1/64 = 1/512	closest to zero
	0	0000	010	-6	2/8*1/64 = 2/512	
	…					
	0	0000	110	-6	6/8*1/64 = 6/512	
	0	0000	111	-6	7/8*1/64 = 7/512	largest denorm
Notice smooth transition	0	0001	000	-6	8/8*1/64 = 8/512	smallest norm
	0	0001	001	-6	9/8*1/64 = 9/512	
	…					
	0	0110	110	-1	14/8*1/2 = 14/16	
	0	0110	111	-1	15/8*1/2 = 15/16	closest to 1 below
Normalized numbers	0	0111	000	0	8/8*1 = 1	
	0	0111	001	0	9/8*1 = 9/8	closest to 1 above
	0	0111	010	0	10/8*1 = 10/8	
	…					
	0	1110	110	7	14/8*128 = 224	
	0	1110	111	7	15/8*128 = 240	largest norm
	0	1111	000	n/a	inf	

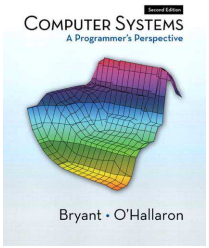
from Carnegie Mellon University's 15213 course slide by Greg Kesden

Special Properties of the IEEE Encoding

- Floating Point Zero (+0) is Same as Integer Zero
  - ทุก bit = 0
- สามารถใช้การเปรียบเทียบจำนวน ในลักษณะใกล้เคียงกับ Unsigned Integer
  - ต้องพิจารณา sign bit ก่อน
  - ต้องพิจารณา -0 ว่ามีค่า = 0
  - ปัญหากรณี NaNs
    - Will be greater than any other values
    - What should comparison yield?
  - กรณีอื่น ๆ เป็นไปตามปกติ
    - Denormalized vs. normalized
    - Normalized vs. infinity

Floating Point

- Fractional Binary Number
- IEEE Floating Point
- Normalized Value
- Denormalized Value
- Special Value
- **Rounding**
- Floating Point Operation
- Floating Point in C



Rounding

- เนื่องจากการแสดงค่าแบบ **Floating Point** สามารถแทนค่าได้ในช่วงความละเอียดที่จำกัด ทำให้จำเป็นต้องมีการประมาณค่าโดยการปัดเศษทิ้ง (Rounding) เช่น ปัด 1.4 → 1 หรือ ปัด 1.6 → 2
- ต้องพิจารณาตัดสินใจ กรณีค่าที่ต้องการปัดเศษอยู่กึ่งกลางระหว่างผลลัพธ์ที่เป็นไปได้ 2 จำนวน
  - เช่น 1.5 (1 vs 2)
  - ปัดไปทางไหน? เพราะอะไร?
  - 1.5 + 2.5 + 3.5 + 4.5 vs 2 + 3 + 4 + 5

Rounding [2]

- ใน **IEEE floating-point format** มีวิธีการปัดเศษ ที่ต่างกันถึง 4 วิธี

IEEE Default Mode

Mode	$1.40	$1.60	$1.50	$2.50	$-1.50
Round-to-even	$1	$2	$2	$2	$-2
Round-toward-zero	$1	$1	$1	$2	$-1
Round-down	$1	$1	$1	$2	$-2
Round-up	$2	$2	$2	$3	$-1

Round to Even

- การปัดเศษเลขคู่ เป็น **Default Mode**
  - แก้ปัญหากรณีค่าความคลาดเคลื่อนจากการปัดเศษของเลขหลาย ๆ จำนวน (ลงเสมอ หรือ ขึ้นเสมอ) สะสมรวมกัน
  - ใช้ในกรณีตัวเลขที่ต้องการปัดอยู่กึ่งกลางเท่านั้น
    - เช่น กรณีปัดให้เป็นทศนิยม 2 ตำแหน่ง

1.2349999 1.23 (Less than half way)
1.2350001 1.24 (Greater than half way)
1.2350000 1.24 (Half way—round up)
1.2450000 1.24 (Half way—round down)

กรณีอื่น ๆ ปัดตามหลักคณิตศาสตร์ปกติ

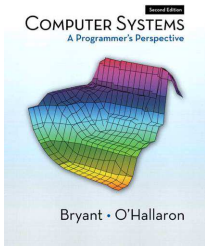
from Carnegie Mellon University's 15213 course slide by Greg Kesden

Rounding Binary Numbers

- **Binary Fractional Numbers**
  - เป็นเลขคู่ก็ต่อเมื่อบิทขวาสุดหลังจากการปัดเศษ มีค่าเป็น 0
  - เศษที่ต้องการปัดจะมีค่ากึ่งกลางก็ต่อเมื่ออยู่ในรูป = $100\ldots0_2$
- **Examples**
  - ปัดเหลือแค่ 2 ตำแหน่งหลัง binary point

Value	Binary	Rounded	Action	Rounded Value
2 3/32	10.00011_2	10.00_2	(<1/2—down)	2
2 3/16	10.00110_2	10.01_2	(>1/2—up)	2 1/4
2 7/8	10.11100_2	11.00_2	(1/2—up)	3
2 5/8	10.10100_2	10.10_2	(1/2—down)	2 1/2

Floating Point



- Fractional Binary Number
- IEEE Floating Point
- Normalized Value
- Denormalized Value
- Special Value
- Rounding
- **Floating Point Operation**
- Floating Point in C

Floating Point Operations

- **Basic Idea**
  - ให้ x และ y เป็นจำนวน floating point
  - ผลลัพธ์ของ operation $\odot$ ใด ๆ ระหว่าง x และ y จะได้จาก
    - การหาค่าจริงของ $x \odot y$
    - แล้วนำผลลัพธ์ที่ได้มาปัดเศษ (Round) ให้เข้ากับจำนวนตำแหน่งที่แสดงค่าได้ (frac)

Floating Point Addition

การบวก

- กำหนดให้
  - $x +^f y$ แทน Round$(x + y)$
  - มีสมบัติการสลับที่ของการบวก (Commutative)
    - $x +^f y = y +^f x$
  - ไม่มีสมบัติการเปลี่ยนกลุ่ม (Associative)
    - `(3.14 + 1e10)- 1e10` ≠ `3.14 + (1e10 - 1e10)`
    - เนื่องจากการปัดเศษ

Floating Point Addition [2]

// ดังนั้นในการคำนวณ จาก code:

```
x = a + b + c;
y = b + c + d;
```

// มีค่าไม่เท่ากับ

```
t = b + c;
x = a + t;
y = t + d;
```

Floating Point Operation จะมีการ Round ทุกครั้ง

//why?

Floating Point Multiplication

การคูณ

- **Monotonicity Properties**

$$a \geq b \text{ and } c \geq 0 \Rightarrow a \times^f c \geq b \times^f c$$
$$a \leq b \text{ and } c \leq 0 \Rightarrow a \times^f c \leq b \times^f c$$

- **Also guaranteed that**

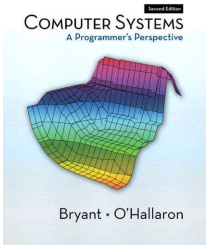
$$a \times^f a \geq 0$$

As long as $a \neq NaN$

เครื่องหมายไม่เปลี่ยน แม้จะเกิดการ overflow

Floating Point

- Fractional Binary Number
- IEEE Floating Point
- Normalized Value
- Denormalized Value
- Special Value
- Rounding
- Floating Point Operation
- **Floating Point in C**



Floating Point in C

C Guarantees Two Levels

- `float` single precision
- `double` double precision

Conversions/Casting

- Casting between `int`, `float`, and `double` ทำให้มีการเปลี่ยนแปลงรูปแบบของ bit
  - `int` → `float`
    - ไม่มีการ overflow แต่มีการปัดเศษ
  - `int/float` → `double`
    - คงค่าเดิมไว้ เนื่องจาก double สามารถแสดงค่าได้ในช่วงที่กว้างกว่า และมีความเที่ยงตรง (precision) มากกว่า

Floating Point in C [2]

- **Conversions/Casting [ต่อ]**
  - `double` → `float`
    - อาจมีการ **overflow** หากเกินช่วงค่าของ **float** หากไม่เกินอาจมีการปัดเศษ เนื่องจาก **precision** ที่น้อยกว่า (จำนวน **bit** `frac` น้อยกว่า)
  - `float/double` → `int`
    - ค่าจะถูก **round toward 0 (truncate)** - ตัดหลังจุดทศนิยมให้เหลือแต่จำนวนเต็ม
    - กรณี **overflow** หรือ **NaN** ไม่ **define** ใน **C**

Summary

- **Floating Point** เป็นค่าประมาณของจำนวนจริง ในรูปของ $x \times 2^y$
- มาตรฐาน **IEEE 754** เป็นมาตรฐานของ **floating point** ที่ใช้กันมากที่สุดโดย **single precision = 32 bit** และ **double precision = 64 bit**
- **IEEE floating point** สามารถแทนค่าพิเศษ ได้แก่ $+\infty$ และ $-\infty$ รวมถึง **NaN**
- การคำนวณต่าง ๆ ที่เกี่ยวข้องกับ **floating point** ควรทำด้วยความระมัดระวังเนื่องจาก **range** และ **precision** ที่จำกัด อีกทั้งยังขาดคุณสมบัติทางคณิตศาสตร์บางประการ

Practice Problem 1: KEY

Fractional value	Binary representation	Decimal representation
$\frac{1}{8}$	0.001	0.125
$\frac{3}{4}$	0.11	0.75
$\frac{25}{16}$	1.1001	1.5625
$\frac{43}{16}$	10.1011	2.6875
$\frac{9}{8}$	1.001	1.125
$\frac{47}{8}$	101.111	5.875
$\frac{51}{16}$	11.0011	3.1875

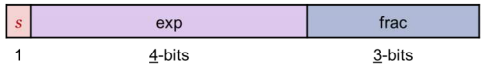
Practice Problem 2: KEY

$Bias = 2^{k-1} - 1$

$E = exp - Bias$

$M = 1.xxx...x_2$

เติมตารางให้สมบูรณ์



Bit Representation			Exponent			Fraction		Value		
			e	E	2^E	f	M	$2^E \times M$	V	Decimal
0	0001	000	1	-6	1/64	0/8	8/8	8/512	1/64	0.015625
0	0110	110	6	-1	1/2	6/8	14/8	14/16	7/8	0.875
0	0111	000	7	0	1	0/8	8/8	8/8	1	1.0
0	0111	010	7	0	1	2/8	10/8	10/8	5/4	1.25
0	1110	110	14	7	128	6/8	14/8	1792/8	224	224.0
0	1110	111	14	7	128	7/8	15/8	1920/8	240	240.0

Practice Problem 3: KEY

- **Bias** = $2^{k-1} - 1$

E = 1 – **Bias**

M = 0.$xxx\ldots x_2$

เติมตารางให้สมบูรณ์

Bit Representation			Exponent			Fraction		Value		
			E	E	2^E	f	M	$2^E \times M$	V	Decimal
0	0000	000	0	-6	1/64	0/8	0/8	0/512	0	0.0
0	0000	001	0	-6	1/64	1/8	1/8	1/512	1/512	0.001953
0	0000	010	0	-6	1/64	2/8	2/8	2/512	1/256	0.003906
0	0000	011	0	-6	1/64	3/8	3/8	3/512	3/512	0.005859
0	0000	110	0	-6	1/64	6/8	6/8	6/512	3/256	0.011719
0	0000	111	0	-6	1/64	7/8	7/8	7/512	7/512	0.013672