Tutorial 04: Rounding and Normalization

Computer Science Department
CS2208: Introduction to Computer Organization and Architecture
Winter 2021-2022
Instructor: Mahmoud R. El-Sakka
Office: MC-419
Email: elsakka@csd.uwo.ca
Phone: 519-661-2111 x86996

Rounding

- The rounding mechanisms include
  - *Truncation* (i.e., *dropping unwanted bits*) by *rounding towards zero*; a.k.a., *rounding down*
  - *Rounding towards positive or negative infinity*, the nearest valid floating-point number in the direction positive or negative infinity, respectively, is chosen to decide the rounding ; a.k.a., *rounding up*.
  - *Rounding to nearest*, the closest floating-point representation to the actual number is used.

Rounding

- ***Example 1***: ***Round to the nearest*** the following numbers value to 8 digits after the binary point.

```
0.11010101 1001000 ==> 0.11010101
                     + 0.00000001
                     = 0.11010110
```

1001000 > 1000000

If it is == case, and this bit = 1, you round up.

```
0.11010101 1000000 ==> 0.11010101
                     + 0.00000001
                     = 0.11010110
```

1000000 == 1000000

```
0.11010101 0xxxxxx ==> 0.11010101
                     + 0.00000000
                     = 0.11010101
```

0xxxxxx < 1000000

```
0.11010100 1001000 ==> 0.11010100
                     + 0.00000001
                     = 0.11010101
```

1001000 > 1000000

If it is == case, and this bit = 0, you round down.

```
0.11010100 1000000 ==> 0.11010100
                     + 0.00000000
                     = 0.11010100
```

1000000 == 1000000

Mid-way ➔ round to even significand

```
0.11010100 0xxxxxx ==> 0.11010100
                     + 0.00000000
                     = 0.11010100
```

0xxxxxx < 1000000

Normalization

- ***Example 2***: Convert the unsigned value $AB.BA_{16}$ to binary. ***Normalize** your answer.*

$AB.BA_{16}$

$\rightarrow 10101011.10111010_2$

After normalization,

$\rightarrow 1.010101110111010_2 \times 2^{+7}$

In base b, a normalized number will have the form

$\pm b_0 . b_1 b_2 b_3 \ldots \times b^n$

where $b_0 \neq 0$, and $b_0, b_1, b_2, b_3 \ldots$ are integers between 0 and b -1

0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001
A = 1010
B = 1011
C = 1100
D = 1101
E = 1110
F = 1111

Normalization and Rounding

- ***Example 3***: Consider the unsigned normalized binary value $1.01010111011010_2 \times 2^{+7}$
  - limit it (using **truncation / rounding down**) to 6 bits (1 + 5 bits) in total
  - limit it (using **rounding up**) to 6 bits (1 + 5 bits) in total
  - limit it (using **rounding to the nearest**) to 6 bits (1 + 5 bits) in total
  - limit it (using **truncation / rounding down**) to 9 bits (1 + 8 bits) in total
  - limit it (using **rounding up**) to 9 bits (1 + 8 bits) in total
  - limit it (using **rounding to the nearest**) to 9 bits (1 + 8 bits) in total
  - limit it (using **truncation / rounding down**) to 14 bits (1 + 13 bits) in total
  - limit it (using **rounding up**) to 14 bits (1 + 13 bits) in total
  - limit it (using **rounding to the nearest**) to 14 bits (1 + 13 bits) in total

Calculate the rounding error in each case.

Note that: The binary value $1.01010111011010_2 \times 2^{+7}$

$= 10101011.10111010_2 = \text{AB.BA}_{16}$

0 = 0000	8 = 1000
1 = 0001	9 = 1001
2 = 0010	A = 1010
3 = 0011	B = 1011
4 = 0100	C = 1100
5 = 0101	D = 1101
6 = 0110	E = 1110
7 = 0111	F = 1111

Normalization and Rounding

- *Limiting the answer to 6 bits (1 + 5) in total,*
- ➔ $1.010101110111010_2 \times 2^{+7}$

- ➔ $1.01010_2 \times 2^{+7}$ *(using **truncation / rounding down**)*
  ➔ 10101000_2 ➔ $A8_{16}$
- ***Truncation*** error = $AB.BA_{16} - A8_{16} = 3.BA_{16}$

- ➔ $1.01011_2 \times 2^{+7}$ *(using **rounding up**)*
  ➔ 10101100_2 ➔ AC_{16}
- ***Rounding up*** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$

- As $1110111010_2 > 1000000000_2$
- ➔ $1.01011_2 \times 2^{+7}$ *(using **rounding to the nearest**)*
  ➔ 10101100_2 ➔ AC_{16}
- ***Rounding to the nearest*** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$

0 = 0000	8 = 1000
1 = 0001	9 = 1001
2 = 0010	A = 1010
3 = 0011	B = 1011
4 = 0100	C = 1100
5 = 0101	D = 1101
6 = 0110	E = 1110
7 = 0111	F = 1111

Normalization and Rounding

- *Limiting the answer to 9 bits (1 + 8) in total*,
- ➔ $1.010101110111010_2 \times 2^{+7}$

- ➔ $1.01010111_2 \times 2^{+7}$ (*using* ***truncation / rounding down***)
  ➔ 10101011.1_2 ➔ $AB.8_{16}$
- ***Truncation*** error = $AB.BA_{16} - AB.8_{16} = 0.3A_{16}$

- ➔ $1.01011000_2 \times 2^{+7}$ (*using* ***rounding up***)
  ➔ 10101100.0_2 ➔ AC_{16}
- ***Rounding up*** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$

- As $0111010_2 < 1000000_2$
- ➔ $1.01010111_2 \times 2^{+7}$ (*using* ***rounding to the nearest***)
  ➔ 10101011.1_2 ➔ $AB.8_{16}$
- ***Rounding to the nearest*** error = $AB.BA_{16} - AB.8_{16} = 0.3A_{16}$

0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001
A = 1010
B = 1011
C = 1100
D = 1101
E = 1110
F = 1111

Normalization and Rounding

- *Limiting the answer to 14 bits (1 + 13) in total,*
- ➔ $1.0101011101110\,10_2 \times 2^{+7}$

- ➔ $1.0101011101110_2 \times 2^{+7}$ *(using **truncation / rounding down**)*
  ➔ 10101011.101110_2 ➔ $AB.B8_{16}$
- ***Truncation*** error = $AB.BA_{16} - AB.B8_{16} = 0.02_{16}$

- ➔ $1.0101011101111_2 \times 2^{+7}$ *(using **rounding up**)*
  ➔ 10101011.101111_2 ➔ $AB.BC_{16}$
- ***Rounding up*** error = $AB.BA_{16} - AB.BC_{16} = -0.02_{16}$

- As $10_2 == 10_2$
- ➔ $1.0101011110110_2 \times 2^{+7}$ *(using **rounding to the nearest**)*
  ➔ 10101011.110110_2 ➔ $AB.B8_{16}$
- ***Rounding to the nearest*** error = $AB.BA_{16} - AB.B8_{16} = 0.02_{16}$

- *Which **rounding mechanism** produces less error?*

0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001
A = 1010
B = 1011
C = 1100
D = 1101
E = 1110
F = 1111