

These slides are being provided with permission from the copyright for in-class (CS2208B) use only. The slides must not be reproduced or provided to anyone outside of the class.

All download copies of the slides and/or lecture recordings are for personal use only. Students must destroy these copies within 30 days after receipt of final course evaluations.

Tutorial 04: Rounding and Normalization

Computer Science Department

CS2208: Introduction to Computer Organization and Architecture

Winter 2020-2021

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.

0.110101011001000 \Rightarrow 0.11010101

$$\begin{array}{r} + 0.00000001 \\ \hline = 0.11010110 \end{array}$$

1001000
>
1000000

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

0.110101011000000 \Rightarrow 0.11010101

$$\begin{array}{r} + 0.00000001 \\ \hline = 0.11010110 \end{array}$$

1000000
=
1000000

Mid-way \rightarrow round to even significand

0.110101001001000 \Rightarrow 0.11010100

$$\begin{array}{r} + 0.00000001 \\ \hline = 0.11010101 \end{array}$$

1001000
>
1000000

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

0.110101001000000 \Rightarrow 0.11010100

$$\begin{array}{r} + 0.00000000 \\ \hline = 0.11010100 \end{array}$$

1000000
=
1000000

0.110101010xxxxxx \Rightarrow 0.11010101

$$\begin{array}{r} + 0.00000000 \\ \hline = 0.11010101 \end{array}$$

0xxxxxx
<
1000000

0.110101000xxxxxx \Rightarrow 0.11010100

$$\begin{array}{r} + 0.00000000 \\ \hline = 0.11010100 \end{array}$$

0xxxxxx
<
1000000

Normalization

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

$AB.BA_{16}$

→ 10101011.10111010_2

After normalization,

→ $1.010101110111010_2 \times 2^{+7}$

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

In base b , a normalized number will have the form

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

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

Normalization and Rounding

■ Example 3: Consider the unsigned normalized binary value $1.010101110111010_2 \times 2^{+7}$

- 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 $1.010101110111010_2 \times 2^{+7}$
 $= 10101011.10111010_2 = \text{AB.BA}_{16}$

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 6 bits (1 + 5) in total,
- $\rightarrow 1.010101110111010_2 \times 2^{+7}$
- $\rightarrow 1.01010_2 \times 2^{+7}$ (using **truncation / rounding down**)
 $\rightarrow 10101000_2 \rightarrow A8_{16}$
- **Truncation** error = $AB.BA_{16} - A8_{16} = 3.BA_{16}$
- $\rightarrow 1.01011_2 \times 2^{+7}$ (using **rounding up**)
 $\rightarrow 10101100_2 \rightarrow AC_{16}$
- **Rounding up** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$
- As $1110111010_2 > 10000000000_2$
- $\rightarrow 1.01011_2 \times 2^{+7}$ (using **rounding to the nearest**)
 $\rightarrow 10101100_2 \rightarrow AC_{16}$
- **Rounding to the nearest** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$

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 9 bits (1 + 8) in total,
- $\rightarrow 1.010101110111010_2 \times 2^{+7}$
- $\rightarrow 1.01010111_2 \times 2^{+7}$ (using **truncation / rounding down**)
 $\rightarrow 10101011.1_2 \rightarrow AB.8_{16}$
- **Truncation** error = $AB.BA_{16} - AB.8_{16} = 0.3A_{16}$
- $\rightarrow 1.01011000_2 \times 2^{+7}$ (using **rounding up**)
 $\rightarrow 10101100.0_2 \rightarrow AC_{16}$
- **Rounding up** error = $AB.BA_{16} - AC_{16} = -0.46_{16}$
- As $0111010_2 < 1000000_2$
- $\rightarrow 1.01010111_2 \times 2^{+7}$ (using **rounding to the nearest**)
 $\rightarrow 10101011.1_2 \rightarrow AB.8_{16}$
- **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,
- $\rightarrow 1.010101110111010_2 \times 2^{+7}$
- $\rightarrow 1.0101011101110_2 \times 2^{+7}$ (using **truncation / rounding down**)
 $\rightarrow 10101011.101110_2 \rightarrow \text{AB.B8}_{16}$
- **Truncation** error = $\text{AB.BA}_{16} - \text{AB.B8}_{16} = 0.02_{16}$
- $\rightarrow 1.0101011101111_2 \times 2^{+7}$ (using **rounding up**)
 $\rightarrow 10101011.101111_2 \rightarrow \text{AB.BC}_{16}$
- **Rounding up** error = $\text{AB.BA}_{16} - \text{AB.BC}_{16} = -0.02_{16}$
- As $10_2 == 10_2$
- $\rightarrow 1.01010111101110_2 \times 2^{+7}$ (using **rounding to the nearest**)
 $\rightarrow 10101011.110110_2 \rightarrow \text{AB.B8}_{16}$
- **Rounding to the nearest** error = $\text{AB.BA}_{16} - \text{AB.B8}_{16} = 0.02_{16}$
- 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