Computational Accuracy in DSP Implementations

Dr. Deepa Kundur

University of Toronto

Number Formats

- In a DSP, signals are represented as discrete sets of numbers from the input stage along through intermediate processing stages to the output.
- Even DSP structures such as filters require numbers to specify coefficients for operation.
- Two typical formats for these numbers:
  - fixed-point format
  - floating-point format

Fixed-Point Format

- simplest scheme
- number is represented as an integer or fraction using a fixed number of bits
- An n-bit fixed-point signed integer $-2^{n-1} \leq x \leq 2^{n-1} - 1$ is represented as:

$$x = -s \cdot 2^{n-1} + b_{n-2} \cdot 2^{n-2} + b_{n-3} \cdot 2^{n-3} + \cdots + b_1 \cdot 2^1 + b_0 \cdot 2^0$$

where s represents the sign of the number ($s = 0$ for positive and $s = 1$ for negative)

Fixed-Point Integer Format

- What is the most negative value that can be represented?

$$\begin{aligned} x &= -s \cdot 2^{n-1} + b_{n-2} \cdot 2^{n-2} + b_{n-3} \cdot 2^{n-3} + \cdots + b_1 \cdot 2^1 + b_0 \cdot 2^0 \\ x &= -1 \cdot 2^{n-1} + 0 \cdot 2^{n-2} + 0 \cdot 2^{n-3} + \cdots + 0 \cdot 2^1 + 0 \cdot 2^0 \\ &= \boxed{-2^{n-1}} \quad \text{represented with } [1\ 0\ 0\ \cdots\ 0\ 0] \end{aligned}$$

- What is the most positive value that can be represented?

$$\begin{aligned} x &= -s \cdot 2^{n-1} + b_{n-2} \cdot 2^{n-2} + b_{n-3} \cdot 2^{n-3} + \cdots + b_1 \cdot 2^1 + b_0 \cdot 2^0 \\ x &= -0 \cdot 2^{n-1} + 1 \cdot 2^{n-2} + 1 \cdot 2^{n-3} + \cdots + 1 \cdot 2^1 + 1 \cdot 2^0 \\ &= \boxed{2^{n-1} - 1} \quad \text{represented with } [0\ 1\ 1\ \cdots\ 1\ 1] \end{aligned}$$

- Why are only integers represented in this range?

$$x = -s \cdot 2^{n-1} + b_{n-2} \cdot 2^{n-2} + b_{n-3} \cdot 2^{n-3} + \cdots + b_1 \cdot 2^1 + b_0 \cdot \underbrace{2^0}_{=1}$$

Fixed-Point Integer Format

Example: $n = 3$

$$\begin{aligned} -2^{n-1} &\le x \le 2^{n-1} - 1 \\ -2^{3-1} &\le x \le 2^{3-1} - 1 \\ -2^{2} &\le x \le 2^{2} - 1 \\ -4 &\le x \le 3 \implies x \in \{-4, -3, -2, -1, 0, 1, 2, 3\} \end{aligned}$$

How do you represent the number $x = -2$?

Fixed-Point Integer Format

How do you represent the number $x = -2$?

There is always a unique way of assigning values to $s, b_{n-2}, b_{n-3}, \ldots, b_1, b_0$.

$$\begin{aligned} x &= -s \cdot 2^2 + b_1 \cdot 2^1 + b_0 \\ x &= -1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \end{aligned}$$

$$[\ 1\ \ 1\ \ 0\ ]$$

Fixed-Point Fractional Format

- Similarly a n-bit fixed-point signed fraction $-1 \le x \le (1 - 2^{-(n-1)})$ is represented as:

$$\begin{aligned} x = & -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + b_{-2} \cdot 2^{-2} + \cdots \\ & \cdots + b_{-(n-2)} \cdot 2^{-(n-2)} + b_{-(n-1)} \cdot 2^{-(n-1)} \end{aligned}$$

where s represents the sign of the number ($s = 0$ for positive and $s = 1$ for negative)

Fixed-Point Fractional Format

- What is the most negative value that can be represented?

$$\begin{aligned} x &= -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + \cdots + b_{-(n-2)} \cdot 2^{-(n-2)} + b_{-(n-1)} \cdot 2^{-(n-1)} \\ &= -1 \cdot 2^0 + 0 \cdot 2^{-1} + \cdots + 0 \cdot 2^{-(n-2)} + 0 \cdot 2^{-(n-1)} \\ &= \boxed{-1} \quad \text{represented with } [1\ 0\ \cdots\ 0\ 0] \end{aligned}$$

- What is the most positive value that can be represented?

$$\begin{aligned} x &= -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + \cdots + b_{-(n-2)} \cdot 2^{-(n-2)} + b_{-(n-1)} \cdot 2^{-(n-1)} \\ &= -0 \cdot 2^0 + 1 \cdot 2^{-1} + \cdots + 1 \cdot 2^{-(n-2)} + 1 \cdot 2^{-(n-1)} \\ &= \boxed{1 - 2^{-(n-1)}} \quad \text{represented with } [0\ 1\ \cdots\ 1\ 1] \end{aligned}$$

- What granularity of numbers can be represented in this range?

$$x = -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + \cdots + b_{-(n-2)} \cdot 2^{-(n-2)} + b_{-(n-1)} \cdot \underbrace{2^{-(n-1)}}_{\text{resolution}}$$

Fixed-Point Fractional Format

Example: $n = 3$

$$
\begin{aligned}
-1 &\leq x \leq (1 - 2^{-(n-1)}) \\
-1 &\leq x \leq (1 - 2^{-(3-1)}) \\
-1 &\leq x \leq (1 - 2^{-2}) \implies -1 \leq x \leq \frac{3}{4}
\end{aligned}
$$

$$x \in \{-1, -\frac{3}{4}, -\frac{1}{2}, -\frac{1}{4}, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$$

$$x = -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + b_{-2} \cdot 2^{-2}$$

$\frac{1}{4} = \frac{1}{2^{n-1}}$ is the smallest precision of measurement for $n = 3$.

Fixed-Point Fractional Format

How do you represent the number $x = \frac{1}{4}$?

$$
\begin{aligned}
x &= -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + b_{-2} \cdot 2^{-2} \\
\frac{1}{4} &= 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} \\
& \quad [\ 0 \ \ 0 \ \ 1\ ]
\end{aligned}
$$

Fixed-Point Format

$n-1$	$n-2$		2	1	0
s	b_{n-2}	···	b_2	b_1	b_0

(a) Fixed-point format to represent signed integers

implied binary point

$n-1$	$n-2$		2	1	0
s	b_{-1}	···	$b_{-(n-3)}$	$b_{-(n-2)}$	$b_{-(n-1)}$

implied binary point

(b) Fixed-point format to represent signed fractions

Fixed-Point Format

Q: Why would one use fractional representation instead of integer representation of numbers?

- multiplication of integers in DSPs may result in overflow error that will manifest as wrap-around bit error;
  - for $n = 3$, $-4 \leq x < 3$; consider $2 \times (-3) = -6$ outside range!
- a fractional representation can be used instead along with proper scaling
  - proper fraction $\times$ proper fraction $=$ proper fraction
  - for $n = 3$, $x \in \{-1, -\frac{3}{4}, -\frac{1}{2}, -\frac{1}{4}, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$; consider $-\frac{3}{4} \times \frac{1}{2} = -\frac{3}{8}$ outside possible precision!
  - the least significant bits (LSBs) will be discarded (i.e., will be approx with $-\frac{1}{2}$)
  - trade-off overflow error for rounding error

Fixed-Point Format and Size

Q: How would one increase the range of numbers that can be represented in integer fixed-point format?

- increase its size (i.e., the number of bits n); doubling the size substantially increases the range of numbers represented
  - for $n = 3$, $-4 \leq x \leq 3$
  - for $n = 6$, $-2^{6-1} \leq x \leq 2^{6-1} - 1 \implies -32 \leq x \leq 31$
- doubling the size has implications:
  - need double the storage for the same data
  - may need to double the number of accesses using the original size of data bus

Fixed-Point Format and Size

Q: Is there a number format with a different compromise between overflow, precision and storage needs?

A: floating-point!

Floating-Point Format

- suitable for computations where a large number of bits (in fixed point format) would be required to store intermediate and final results
  - example: algorithm involves summation of a large number of products (a.k.a multiply and accumulate)
- A floating point number x is represented as:

$$x = M_x 2^{E_x}$$

  where M_x is called the mantissa and E_x is called the exponent.

Floating-Point Format

- The product of two floating point numbers $x = M_x 2^{E_x}$ and $y = M_y 2^{E_y}$ is given by

$$xy = M_x M_y 2^{E_x + E_y}$$

- A floating-point multiplier must contain a multiplier for the mantissa and an adder for the exponent.
- A floating-point adder requires normalization of the numbers to be added so that they have the same exponents.

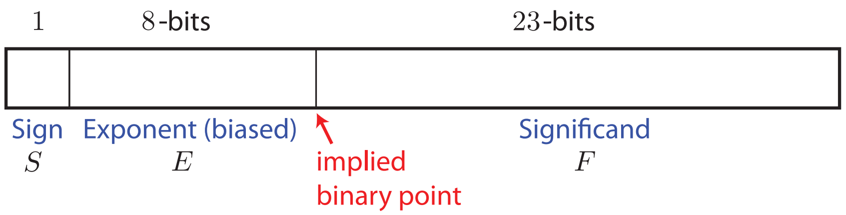
$$\begin{aligned} x + y &= M_x 2^{E_x} + M_y 2^{E_y} = (M_x 2^{E_x - E_s}) 2^{E_s} + (M_y 2^{E_y - E_s}) 2^{E_s} \\ &= (M_x 2^{E_x - E_s} + M_y 2^{E_y - E_s}) 2^{E_s} \end{aligned}$$

Floating-Point Format

- A commonly used single-precision floating-point representation is the IEEE 754-1985 format given as:

$$x = (-1)^{S} \times 2^{(E-bias)} \times (1+F)$$

- S, E and F are all in unsigned fixed-point format.



Floating-Point Format

- A commonly used single-precision floating-point representation is the IEEE 754-1985 format given as:

$$x = (-1)^{S} \times 2^{(E-bias)} \times (1+F)$$

  - F is the magnitude fraction of the mantissa
    - Note: In determining the full mantissa value, a 1 is placed immediately before the implied binary point
  - E is the biased exponent
    - Note: The bias makes sure that the exponent is signed to represent both small and large numbers.
    - The *bias* is set to 127 (largest positive number represented by (8 − 1)-bits).
  - S gives the sign of the fractional part of the number

Floating-Point Format

Example:
Find the decimal equivalent of the floating-point binary number with $bias = 2^3 - 1 = 7$:

$$1011000011100$$

$$\underbrace{1}_{\text{sign}} \quad \underbrace{0110}_{\text{biased exponent}} \quad \underbrace{00011100}_{\text{significand}}$$

$$\begin{aligned} F &= 0 \cdot 2^{-1} + 0 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} + \\ &\quad 1 \cdot 2^{-5} + 1 \cdot 2^{-6} + 0 \cdot 2^{-7} + 0 \cdot 2^{-8} = 0.109375 \\ E &= 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6 \end{aligned}$$

$$\therefore x = -1 \times 1.109375 \times 2^{6-7} = -0.5546875$$

Floating-Point Format

Q: What is the main disadvantage of using floating-point over fixed-point?

- speed reduction:
  - floating point multiplication requires addition of exponents and multiplication of mantissas
  - floating point addition requires exponents to be normalized prior to addition

Dynamic Range

- ratio of the maximum value to the minimum non-zero value that the signal can take in a given number representation scheme:

$$\text{dynamic range} = \frac{\max_{\forall x}\{|x|\}}{\min_{\forall x \neq 0}\{|x|\}}$$

where

$$x = -s \cdot 2^{n-1} + b_{n-2} \cdot 2^{n-2} + b_{n-3} \cdot 2^{n-3} + \cdots + b_1 \cdot 2^1 + b_0 \cdot 2^0,$$

$s, b_k \in \{0, 1\}$, $k = 0, 1, \ldots, n-2$ and $x \neq 0$.

- dynamic range is proportional to the number of bits n used to represent it

Dynamic Range

Example: $n = 24$ (fixed-point format)

$$\begin{aligned} -2^{n-1} &\leq x \leq 2^{n-1} - 1 \\ -2^{24-1} &\leq x \leq 2^{24-1} - 1 \\ -8,388,608 &\leq x \leq 8,388,607 \end{aligned}$$

$$x \in \{-8,388,608, -8,388,607, \ldots, -1, 0, 1, \ldots 8,388,607\}$$

$x_{max} = 8,388,608$ and $x_{min} = 1$

$$\begin{aligned} \text{dynamic range} &= \frac{8,388,608}{1} = 8,388,608 \\ &= 20 \log_{10}(8,388,608) = \mathbf{138} \text{ dB} \end{aligned}$$

Dynamic Range

Example: $n = 24$ (floating-point format)

$$x = (-1)^S \times 2^{(E-bias)} \times (1 + F)$$

with

- 15-bit significand F (fractional representation);
- 8-bit exponent E (unsigned integer);
- one-bit for S; and
- bias $= 128$.

$$\text{dynamic range} = \frac{\max_{\forall x}\{|x|\}}{\min_{\forall x \neq 0}\{|x|\}}$$

Dynamic Range

- In the computation of dynamic range, F is conventionally set to zero and the sign bit S is irrelevant (set to 0) due to absolute values.

$$\begin{aligned} x &= (-1)^S \times 2^{(E-bias)} \times (1 + F) \\ &= (-1)^0 \times 2^{(E-bias)} \times (1 + 0) \end{aligned}$$

$$x_{max} = 1 \cdot 2^{2^8-1-bias} \cdot 1 \qquad x_{min} = 1 \cdot 2^{0-bias} \cdot 1$$

$$\begin{aligned} \text{dynamic range} &= 20 \log_{10}\left(\frac{2^{2^8-1-bias}}{2^{-bias}}\right) = 20 \log_{10}(2^{2^8-1}) \\ &= 20 \cdot (2^8 - 1) \cdot \log_{10}(2) = 20 \cdot 255 \cdot 0.30102 \\ &= \mathbf{1535} \text{ dB} \end{aligned}$$

Resolution

- general definition: smallest non-zero value that can be represented using a number representation format
- **Q:** What is the resolution if k-bits (signed fractional fixed-point) are used to represent a number between 0 and 1?

$$x = -s \cdot 2^0 + b_{-1} \cdot 2^{-1} + \cdots + b_{-(k-2)} \cdot 2^{-(k-2)} + b_{-(k-1)} \cdot 2^{-(k-1)}$$

$$\text{Resolution} = \frac{1}{2^{k-1}}$$

Precision

- computed as percentage resolution:

$$\text{Precision} = \text{Resolution} \times 100\% = \frac{1}{2^{k-1}} \times 100\%$$

- relates to accuracy of computations
- usually, the greater the precision, the slower the speed or the more complex the support hardware such as bus architectures

Precision

Example: $n = 24$ (signed fractional fixed-point format)

$$\text{Precision} = \frac{1}{2^{n-1}} \times 100 = 2^{-23} \times 100 = 1.2 \times 10^{-5}\,\%$$

Example: $n = 24$ (floating-point format),

$$x = (-1)^S \times 2^{(E-bias)} \times (1 + F)$$

with 15-bit significand, 8-bit exponent (unsigned integer representation), $bias = 128$; convention is to neglect E and S.

$$\text{Precision} = \frac{1}{2^{15}} \times 100 = 3.0 \times 10^{-3}\,\%$$

Fixed-Point vs. Floating Point

Example: $n = 24$

	Fixed-Point	Floating-Point
Dynamic Range	138 dB	1535 dB
Precision	$1.2 \times 10^{-5}\,\%$	$3.0 \times 10^{-3}\,\%$

Fixed-Point vs. Floating Point

- Dynamic Range:
  - determined by exponent of floating-point number
  - since floating-point representations involve exponents, they are superior to fixed-point format schemes in terms of dynamic range
- Resolution/Precision:
  - determined by mantissa of floating-point number
  - mantissa typically uses fewer bits than a fixed point representation, the precision of floating-point is smaller than compared to a fixed point representation

■