Computer Architecture and Organization

CS 115

Lecture 4

Instructor: **Gerald John M. Sotto**

Last Updated: September 30, 2025

Table of Contents

Unit II: Machine Level Representation of Data

Fixed and Floating-point Systems

Fixed and Floating-Point Systems

These two systems are how computers represent **real numbers**—numbers that can have a fractional component. The choice between them is a classic **trade-off between speed/simplicity** and **precision/range**.

Fixed-Point Systems

The fixed-point system is a method for **representing real numbers** (numbers that include fractional parts, like 3.14 or −0.5) in binary format, where the position of the radix point **(binary point)** is **fixed** or **implied.**

In this system, a number is represented by a sequence of bits, where a certain number of bits are allocated to the **integer part** and the remaining bits are allocated to the **fractional part.**

$$(N)_{10} \rightarrow (\underbrace{b_{i-1}\ldots b_1 b_0}_{\textbf{Integer Part}}.\underbrace{b_{-1}\, b_{-2}\ldots b_{-f}}_{\textbf{Fractional Part}})_2 \leftarrow \textbf{base of the number}$$

Fixed-Point Systems

$$(N)_{10} \rightarrow (\underbrace{b_{i-1}\ldots b_1 b_0}_{\textbf{Integer Part}}.\underbrace{b_{-1}\, b_{-2}\ldots b_{-f}}_{\textbf{Fractional Part}})_2$$

base of the number

Representation Example:

$$V = \sum_{j=-f}^{i-1} b_j * 2^j$$

Above is a **universal mathematical formula** used to **calculate the decimal value of any fixed-point binary number.** It's not just a setup for a specific problem; it's the formal definition of how a fixed-point binary value is calculated.

Fixed-Point Systems

$(N)_{10} \rightarrow (\underbrace{b_{i-1}\ldots b_1 b_0}_{\text{Integer Part}}.\underbrace{b_{-1}\ b_{-2}\ldots b_{-f}}_{\text{Fractional Part}})_2$ ← **base of the number**

$$V = \sum_{j=-f}^{i-1} b_j * 2^j$$

For example, the binary number $\mathbf{(101.11)_2}$ with $\mathbf{i=3}$ integer bits and $\mathbf{f=2}$ fractional bits is:

Bit Position (j)	Bit Value (b_j)	Positional Weight (2^j)	Calculation ($b_j \cdot 2^j$)	Decimal Value
2 (i−1)	1	2^2	$1 \cdot 4$	4
1	0	2^1	$0 \cdot 2$	0
0	1	2^0	$1 \cdot 1$	1
-1	1	2^{-1}	$1 \cdot 0.5$	0.5
-2 (−f)	1	2^{-2}	$1 \cdot 0.25$	0.25
Sum			**V**	**5.7510**

$$V=(1 \cdot 2^2)+(0 \cdot 2^1)+(1 \cdot 2^0)+(1 \cdot 2^{-1})+(1 \cdot 2^{-2})$$

$$V=4+0+1+0.5+0.25 = \mathbf{(5.75)_{10}}$$

Fixed-Point Systems

$$(N)_{10} \rightarrow (\underbrace{b_{i-1}\ldots b_1 b_0}_{\text{Integer Part}}.\underbrace{b_{-1}\, b_{-2}\ldots b_{-f}}_{\text{Fractional Part}})_2 \leftarrow \text{base of the number}$$

$$V = \sum_{j=-f}^{i-1} b_j * 2^j$$

Another example, the binary number **$(11.01)_2$** with **$i=2$** integer bits and **$f=2$** fractional bits is:

$$\begin{aligned} V = \sum_{j=-2}^{1} b_j * 2^j &= (b_1 \cdot 2^1) + (b_0 \cdot 2^0) + (b_{-1} \cdot 2^{-1}) + (b_{-2} \cdot 2^{-2}) \\ &= (1 \cdot 2^1) + (1 \cdot 2^0) + (0 \cdot 2^{-1}) + (1 \cdot 2^{-2}) \\ &= (1 \cdot 2) + (1 \cdot 1) + (0 \cdot 0.5) + (1 \cdot 0.25) \\ &= 2 + 1 + 0 + 0.25 \\ &= \mathbf{3.25_{10}} \end{aligned}$$

Therefore, the fixed-point binary number **$(11.01)_2$** is equal to **3.25** in the decimal system.

Fixed-Point Systems

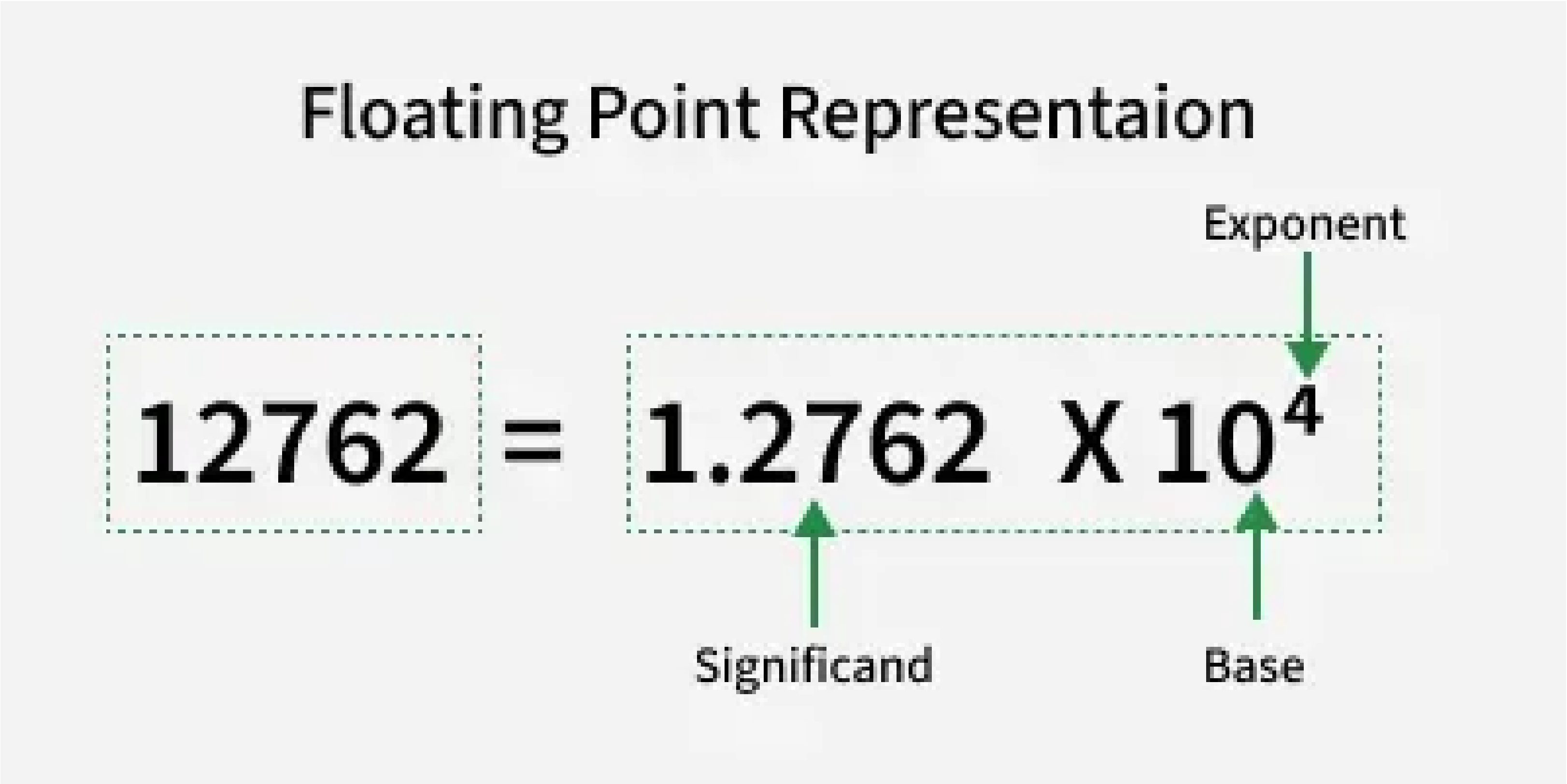
Relevance: It provides predictable precision and avoids the performance overhead of complex floating-point units.

Example: Using an 8-bit system with the binary point fixed after the 4th bit (4 bits for integer, 4 bits for fraction).

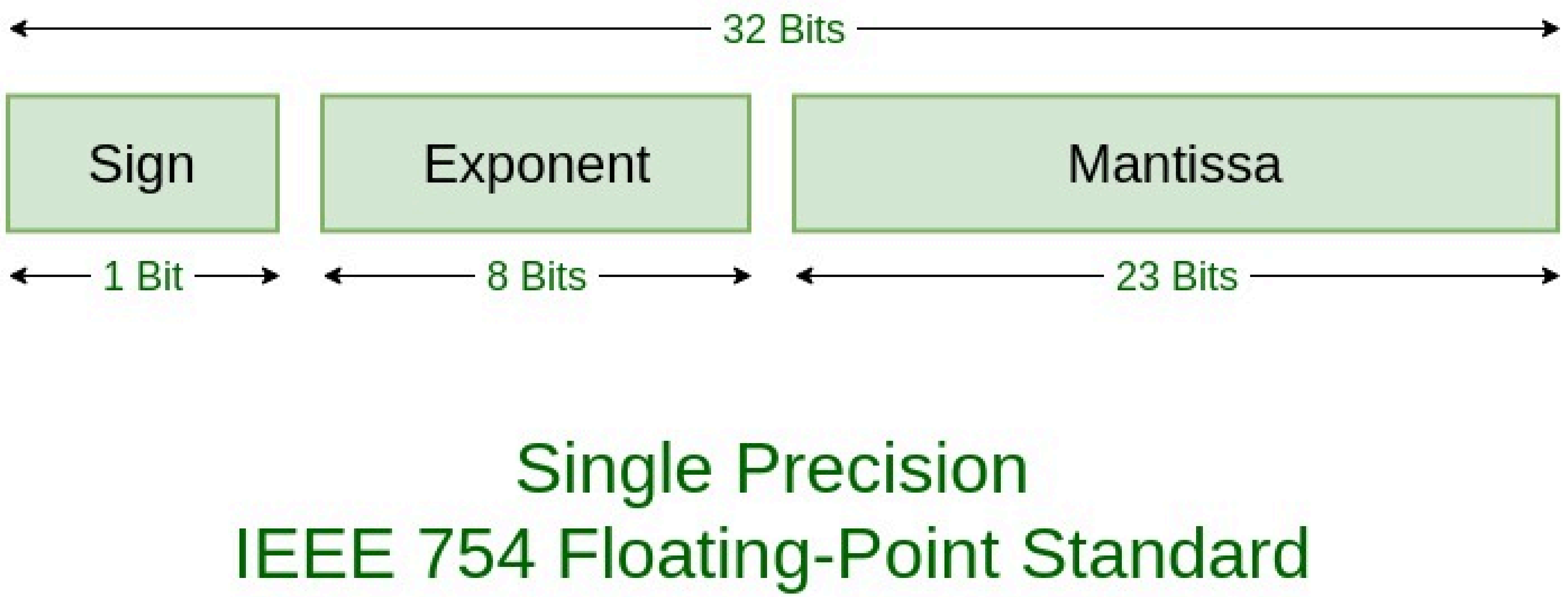
Floating-Point Systems

A system that represents a number as a sign, a **significand** (or **mantissa**), and an exponent. This is the computer's equivalent of scientific notation (e.g., 6.02×10^{23}). The widely accepted standard is the **IEEE 754 standard.**

Allows for a **vast range of values**, from **extremely tiny fractions** to **enormous integers**, at the e**xpense of a constant number of significant digits (precision).** It's essential for scientific computing, 3D graphics, simulations, and virtually all modern high-level programming.



Floating-Point Systems



$$M \times B^E$$

Where:

- **M** is the **Mantissa** (or coefficient), holding the significant digits.
- **B** is the **Base** (**typically 2** for computers, or **10 for standard scientific notation**).
- **E** is the **Exponent**, determining the number's magnitude (**where the decimal point "floats"**).

Floating-Point Systems

$$M \times B^E$$

Where:

- **M** is the **Mantissa** (or coefficient), holding the significant digits.
- **B** is the **Base** (**typically 2** for computers, or **10 for standard scientific notation**).
- **E** is the **Exponent**, determining the number's magnitude (**where the decimal point "floats"**).

570,000 this number is represented as: $\mathbf{5.7 \times 10^5}$

M = 5.7

E = 5

0.00000000032 this number is represented as: $\mathbf{3.2 \times 10^{-10}}$

M = 3.2 (The significant digits)

E = −10 (Pulls the decimal point 10 places left)

Fixed and Floating-point Systems

Data Type	Size (Bytes)	Total Bits (W)	Sign Bit	Integer Bits (i)	Fractional Bits (f)	Qi.f Notation
char	1	8	1	7	0	Q7.0
unsigned char	1	8	0	8	0	Q8.0
short int	2	16	1	15	0	Q15.0
unsigned short int	2	16	0	16	0	Q16.0
int	4	32	1	31	0	Q31.0
unsigned int	4	32	0	32	0	Q32.0
long long int	8	64	1	63	0	Q63.0
unsigned long long int	8	64	0	64	0	Q64.0
---	---	---	---	---	---	---
float	4	32	1	N/A	N/A	N/A (IEEE 754)
double	8	64	1	N/A	N/A	N/A (IEEE 754)

Fixed and Floating-point Systems

Data Type	Total Bits (W)	Standard Name	Mantissa Bits	Approximate Significant Decimal Digits	Maximum Exact Integer Digits (Approx.)
float	32 bits	Single-Precision	23 bits	≈7 digits	7 to 8 digits
double	64 bits	Double-Precision	52 bits	≈15–17 digits	15 to 16 digits

End of Presentation

Questions...?

sana po sir madali lang
exam hihi