Data Representation: Fixed and Floating Point

Dr. Debapriya Roy

Overview

Why Data Representation?

- Computers store data in binary format (0 and 1).
- Numbers, characters, images, and sounds need encoding.
- Choice of representation affects:
  - Precision
  - Range
  - Storage and speed

Types of Data Representation

- **Integer Representation** (Signed/Unsigned)
- **Fixed-Point Representation**
- **Floating-Point Representation**
- **Character Representation (ASCII, Unicode)**

Binary to Decimal Conversion

Method: Sum of Powers of 2

- **Step 1:** Separate integer and fractional parts
- **Step 2:** Apply powers of 2 from left (positive) and right (negative of radix point)

Example 1: Integer Binary

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = \boxed{13}$$

Example 2: Binary with Fraction

$$1101.01_2 = 13 + 0 \times 2^{-1} + 1 \times 2^{-2} = 13 + 0 + 0.25 = \boxed{13.25}$$

Decimal to Binary Conversion

Step 1: Convert Integer Part (Division by 2)

$$13 \div 2 = 6 \text{ remainder } 1$$

$$6 \div 2 = 3 \text{ remainder } 0$$

$$3 \div 2 = 1 \text{ remainder } 1$$

$$1 \div 2 = 0 \text{ remainder } 1$$

$$\Rightarrow \text{Read from bottom to top: } 13_{10} = \boxed{1101_2}$$

Step 2: Convert Fractional Part (Multiplication by 2)

$$0.25 \times 2 = 0.50 \Rightarrow \text{Digit: } 0$$

$$0.50 \times 2 = 1.00 \Rightarrow \text{Digit: } 1$$

$$\Rightarrow 0.25_{10} = \boxed{0.01_2}$$

$$\boxed{13.25_{10} = 1101.01_2}$$

What is Fixed-Point?

- Represents real numbers with fixed number of bits before and after decimal point.
- Notation: Qm.n (m = integer bits, n = fractional bits)
- Example: Q3.4 format (3 integer bits, 4 fractional bits)

Example: Fixed-Point Encoding

Q3.4 format: 7-bit signed fixed-point number (1 sign + 3 integer + 4 fraction)

- Binary: 0011.1010
- Decimal equivalent:

$$0011.1010 = 3 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 0 \times 2^{-4} = 3.625$$

- Binary: 1100.0110 (Two's complement negative)
- Decimal: -3.375

Advantages and Limitations of Fixed-Point

Pros:

- Simple hardware
- Fast arithmetic operations
- Useful in embedded systems

Cons:

- Limited range and precision
- Overflow is common
- Manual scaling required

What is Floating-Point?

- Represents real numbers using scientific notation:

$$\text{Number} = (-1)^s \times M \times 2^E$$

- **s** = sign bit, **M** = mantissa/significand, **E** = exponent
- Widely used due to large dynamic range

IEEE 754 Standard Formats

Single Precision (32-bit):

- Sign: 1 bit
- Exponent: 8 bits (Bias = 127)
- Mantissa: 23 bits

Double Precision (64-bit):

- Sign: 1 bit
- Exponent: 11 bits (Bias = 1023)
- Mantissa: 52 bits

Decimal to IEEE 754 Floating Point Conversion

Example: Convert -13.25 to IEEE 754 (Single Precision)

1. **Step 1: Convert integer and fractional part to binary**
   - $13_{10} = 1101_2$
   - $0.25_{10} = 0.01_2$
   - $\Rightarrow 13.25_{10} = 1101.01_2$
2. **Step 2: Normalize the binary number**
   - $1101.01_2 = 1.10101 \times 2^3$
3. **Step 3: Encode the components**
   - **Sign bit**: 1 (since number is negative)
   - **Exponent** $= 127 + 3 = 130 \rightarrow 10000010_2$
   - **Mantissa**: 101010 followed by 17 zeros to make 23 bits
4. **Step 4: Final IEEE 754 Format**

   Binary: $\boxed{1\ 10000010\ 10101000000000000000000}$

5. **Hexadecimal:** `C1540000`

Special Values in IEEE 754

- Zero: All exponent and mantissa bits are 0
- Infinity: Exponent all 1s, mantissa all 0s
- NaN: Exponent all 1s, mantissa non-zero
- Denormalized numbers: Exponent all 0s (small values near zero)

Floating-Point Arithmetic Example

Addition of 1.5 and 2.25

- $1.5 = 1.1 = 1.1 \times 2^0$
- $2.25 = 10.01 = 1.001 \times 2^1$
- Align exponents: $1.1 = 0.11 \times 2^1$
- Add: $0.11 + 1.001 = 1.111$
- Result: $1.111 \times 2^1 = 3.75$

Fixed vs Floating Point

Aspect	**Fixed-Point**	**Floating-Point**
Precision	Constant	Dynamic
Range	Small	Large
Hardware	Simple	Complex
Speed	Faster	Slower
Use Case	Embedded, DSP	Scientific, Graphics

Application Domains

Fixed-Point:

- Audio and video codecs
- Microcontrollers
- Real-time systems

Floating-Point:

- Machine Learning
- 3D Graphics and Simulation
- Scientific computation

Conclusion

- Data representation is foundational to computer systems.
- Fixed-point is suitable for speed and simplicity.
- Floating-point is essential for precision and range.
- IEEE 754 provides a standardized and robust format.

Thank You

Questions?