Arithmetic for Computers

Welcome to the fascinating world of computer arithmetic! This presentation will guide you through the fundamentals of how computers perform calculations, covering everything from basic binary operations to advanced floating-point arithmetic.

Introduction to Computer Arithmetic

1

1. Understanding the Basics

We'll delve into the foundational concepts of computer arithmetic, exploring how computers represent numbers and perform calculations.

2

2. Digital Representation

Discover how computers represent numbers using a binary system, a fundamental concept in digital computing.

3

3. The Power of Binary

Uncover the advantages of binary representation, which simplifies calculations and makes it easy for computers to process data.

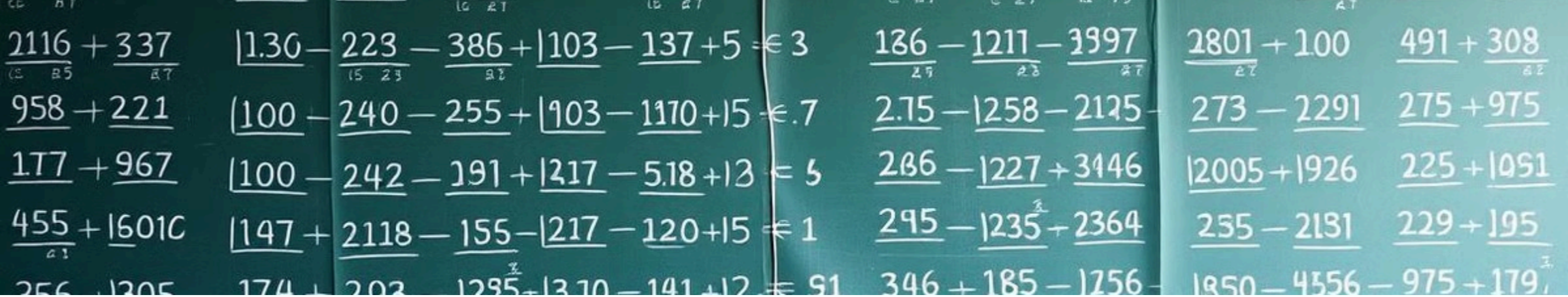
Number Representation in Computers

Binary System

Computers use a binary system, which represents numbers using only two digits: 0 and 1. Each digit is called a bit, and combinations of bits represent different values.

Decimal to Binary Conversion

We'll learn how to convert numbers from decimal (base 10) to binary (base 2), enabling understanding of how computers store and process data.



Binary Addition and Subtraction

1

2

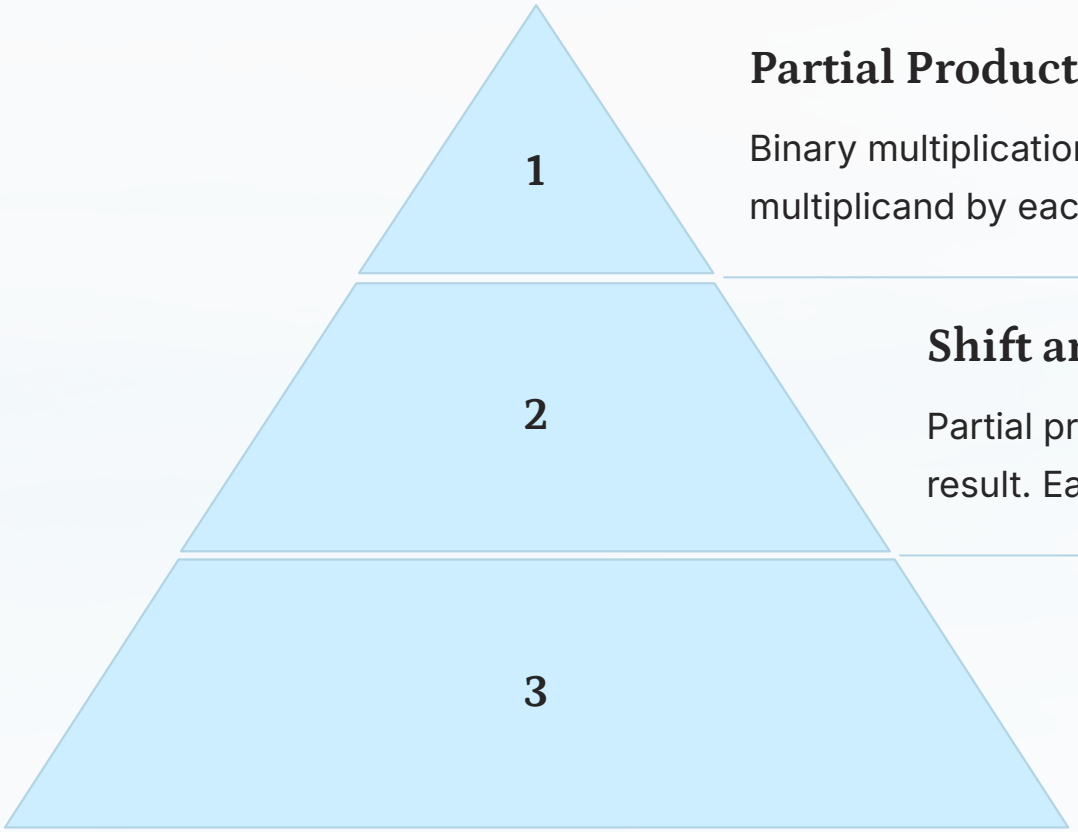
Addition

Binary addition follows similar principles as decimal addition, with carries for sums exceeding 1. 1 + 1 = 10 (binary).

Subtraction

Binary subtraction involves borrowing, similar to decimal subtraction. We'll explore techniques for two's complement subtraction.

Binary Multiplication



Partial Products

Binary multiplication involves calculating partial products by multiplying the multiplicand by each bit of the multiplier.

Shift and Add

Partial products are shifted and added together to obtain the final result. Each shift corresponds to a power of 2.

Final Product

The final product is obtained by summing all the partial products, taking into account the shifts.

Binary Division

1

Repeated Subtraction

Binary division uses repeated subtraction of the divisor from the dividend until the remainder is smaller than the divisor.

2

Shifting and Quotient

Each subtraction corresponds to a bit in the quotient. The dividend is shifted left before each subtraction.

3

Remainder

The final remainder is the result of the last subtraction, indicating any amount that was not fully divisible.

Floating-Point Representation

1

Mantissa

Represents the significant digits of a number, capturing its precision.

2

Exponent

Indicates the power of the base (typically 2) to which the mantissa is raised.

3

Sign Bit

Determines whether the number is positive or negative.

Floating-Point Arithmetic Operations

Addition

Aligning exponents, adding mantissas, and potentially normalizing the result to maintain precision.

Subtraction

Similar to addition, but involves subtracting mantissas after exponent alignment.

$$2 - 363 = 1 - 4{,}00$$

Multiplication

Multiplying mantissas, adding exponents, and potentially normalizing the result.

Division

Dividing mantissas, subtracting exponents, and potentially normalizing the result.

Rounding and Precision Errors

Rounding

Rounding is necessary when a result exceeds the available precision. Different rounding methods (round-to-nearest, round-up, round-down) can be used.

Precision Errors

Precision errors occur when a number cannot be represented exactly in floating-point format, resulting in inaccuracies.

Overflow and Underflow

Overflow occurs when a result exceeds the maximum representable value. Underflow occurs when a result is too small to represent.

Hardware Implementation of Arithmetic