In Verilog, numbers are a fundamental aspect of describing digital hardware. Unlike general-purpose programming languages, where numbers primarily represent abstract mathematical values, in Verilog, they often represent actual physical signals or values stored in hardware elements. This means their **bit-width** and **base** are critical.

Here's a breakdown of how numbers are represented in Verilog:

1. Number Literal Syntax

The general syntax for a number literal in Verilog is:

<size>'<radix><value>

Where:

- **<size> (Optional):** An integer representing the **number of bits** in the value. If omitted, the default size is typically 32 bits (though this can vary by simulator/synthesizer, so it's best to explicitly specify).
- **' (Single Quote):** A mandatory separator between the size and the radix.
- **<radix>:** A single character indicating the base of the number. It is case-insensitive.
  - b or B: Binary (base 2)
  - o or O: Octal (base 8)
  - d or D: Decimal (base 10) - This is the default if no radix is specified.
  - h or H: Hexadecimal (base 16)
- **<value>:** The actual numerical value in the specified radix.

Examples:

- **Decimal:**
  - 10: An unsized decimal number (default 32-bit).
  - 8'd123: An 8-bit decimal number with value 123.
  - 16'd255: A 16-bit decimal number with value 255.
  - -5'd3: A 5-bit decimal number, interpreted as signed. Verilog typically handles negative numbers internally as two's complement.
- **Binary:**
  - 4'b1011: A 4-bit binary number (11 decimal).
  - 8'b0101_1100: An 8-bit binary number. Underscores _ are ignored and used for readability.
  - 'b101: An unsized binary number (default 32-bit). This would be 0...0101 (32 bits).
- **Octal:**
  - 6'o71: A 6-bit octal number.
  - 12'o777: A 12-bit octal number.
- **Hexadecimal:**
  - 8'hFF: An 8-bit hexadecimal number (255 decimal).
  - 16'hDEAD_BEEF: A 16-bit hexadecimal number.

- 'hA: An unsized hexadecimal number (default 32-bit). This would be 0...000A (32 bits).