**Introduction to Verilog**

What is Verilog?

Verilog is a Hardware Description Language (HDL) used to model and simulate digital logic circuits. It has C-like syntax and is used in designing and testing circuits like adders, multiplexers, ALUs, etc.

🧱 1. Module Declaration

Verilog designs are built from `modules`.

verilog

module my\_module(input a, output b);

assign b = a; // simple buffer

endmodule

## **🔌 2. Ports**

Verilog modules use **ports** to communicate with other components:

* input — receives data
* output — sends data
* inout — bidirectional (rare in beginners' designs)

input a, b;

output sum;

## **🔤 3. Identifiers & Comments**

* Names must start with a letter or underscore
* Verilog is **case-sensitive**
* Comments:  
  + Single-line: //
  + Multi-line: /\* ... \*/

## **🔢 4. Number Representation**

4'b1010 // binary

8'd255 // decimal

16'hABCD // hexadecimal

Format: <size>'<base><value>

## **📶 5. Data Types**

* wire: for continuous connections (default for outputs)
* reg: used for storage (inside always blocks)

wire [3:0] bus;

reg clk;

## **⚙️ 6. Operators**

### **🔹 Bitwise Operators**

& | ^ ~

### **🔹 Logical Operators**

&& || !

### **🔹 Relational Operators**

== != > < >= <=

### **🔹 Shift Operators**

<< >> // Shift left / right

### **🔹 Conditional Operator**

result = (A > B) ? A : B;

## **🧠 7. Behavioral Modeling**

Write logic using equations and operators.

assign f = (x1 & x2) | (~x2 & x3);

## **🧱 8. Structural Modeling**

Use gate-level modeling with primitives.

and(g, x1, x2);

not(y, x2);

and(h, y, x3);

or(f, g, h);

## **🔁 9. Continuous Assignments**

Verilog executes assign statements **in parallel**.

assign sum = a ^ b; // concurrent logic

## **🧩 10. Vectors**

Multibit wires/regs are declared with bit ranges:

input [3:0] a;

wire [7:0] result;

## **➕ 11. Example: 1-bit Full Adder**

module full\_adder(sum, carry, a, b, cin);

input a, b, cin;

output sum, carry;

assign sum = a ^ b ^ cin;

assign carry = (a & b) | (cin & (a ^ b));

endmodule

## **🧮 12. Submodules**

Use one module inside another:

full\_adder FA0(sum[0], c1, A[0], B[0], cin);