# **Blocking & Non-blocking in Verilog**

## **🧠 1. Concurrent Statements**

* **What:** Verilog assign statements (outside always blocks).
* **Behavior:** All run in **parallel**, independent of order.

✅ Example:

assign sum = a + b;

assign carry = a & b;

## **🔁 2. Procedural Statements**

* **What:** Statements inside always block.
* **Why:** Used for **decision-making**, **case logic**, or **clock-based logic**.

### **✳️ if-else Statement**

Used for conditional logic:

always @(\*) begin

if (a == 1) out = 1;

else out = 0;

end

### **🔘 case Statement**

Used to select output based on input pattern:

always @(\*) begin

case(sel)

2'b00: out = w[0];

2'b01: out = w[1];

default: out = 0;

endcase

end

### **🎭 casex Statement**

* Treats x and z as "don't care".
* Good for **priority encoders**.

casex (input)

4'b1xxx: out = 2'b11; // Highest priority

...

endcase

## **🔗 3. Wires vs Regs**

| **Feature** | **wire** | **reg** |
| --- | --- | --- |
| Purpose | Connect components | Store values |
| Used in | assign (concurrent) | always blocks (procedural) |
| Default Value | z | x |

## **💡 When to Use**

* Use wire if using assign
* Use reg if assigning inside always

## **🔄 4. Blocking vs Non-blocking**

### **🔹 Blocking (=)**

* Executes **immediately**.
* **Order matters**.
* Good for **combinational** logic.

x = a;

y = x + 1; // Uses new x

### **🔸 Non-blocking (<=)**

* Executes **in parallel** (end of time step).
* Order **does not matter**.
* Required for **sequential logic**.

x <= a;

y <= x + 1; // Uses old x

## **🧪 5. Examples**

### **✅ 2:1 MUX (using if-else)**

always @(\*) begin

if (s == 0)

f = w[0];

else

f = w[1];

end

### **✅ 4:1 MUX (using case)**

always @(\*) begin

case (s)

2'b00: f = w[0];

2'b01: f = w[1];

2'b10: f = w[2];

2'b11: f = w[3];

endcase

end

### **✅ Priority Encoder (using casex)**

casex (w)

4'b1xxx: y = 2'b11;

4'b01xx: y = 2'b10;

4'b001x: y = 2'b01;

4'b0001: y = 2'b00;

endcase

## **🧭 6. Combinational vs Sequential**

| **Type** | **Depends on** | **Use Cases** |
| --- | --- | --- |
| Combinational | Only current inputs | MUX, encoder, decoder |
| Sequential | Inputs + past state | Counters, Flip-Flops, FSMs |