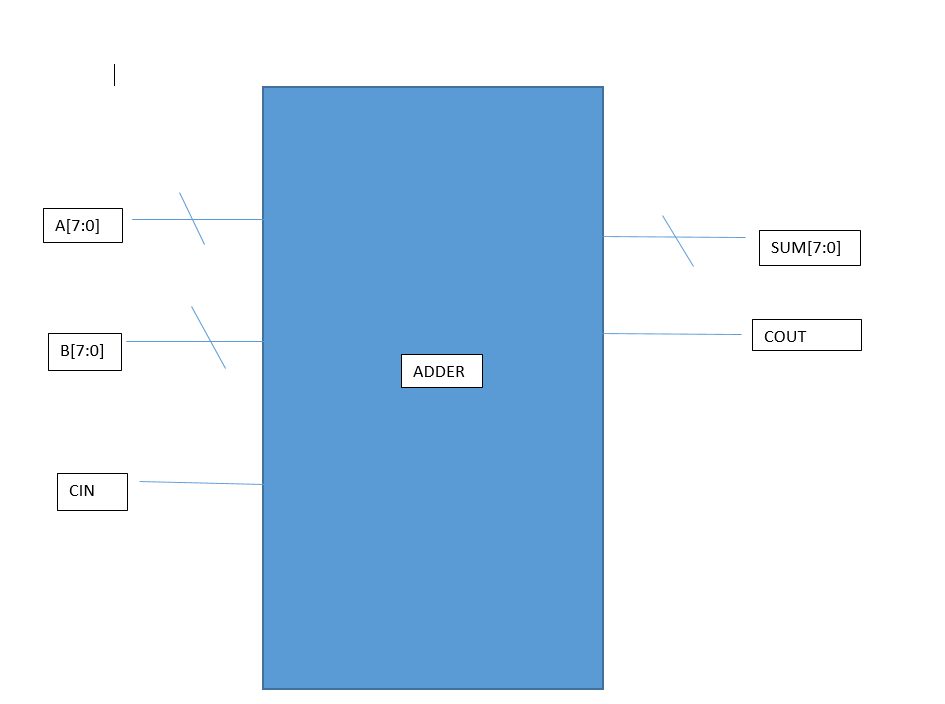
**8-Bit Adder**

**BATCH-05**

**Objective:-**

**Design, implement, and verify a combinational digital circuit that performs the addition of two 8-bit binary numbers and produces an 8-bit sum along with a carry-out signal.**



**A[7:0]**  
→ First 8-bit input operand, part of the addition operation.

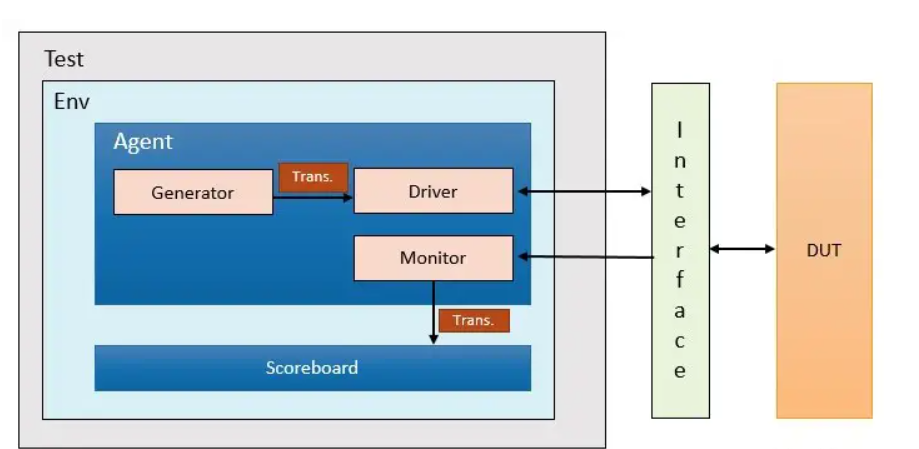
**B[7:0]**  
→ Second 8-bit input operand, added together with A and CIN.

**CIN**  
→ Single-bit input carry used to extend addition beyond 8 bits.

**SUM[7:0]**  
→ 8-bit result of adding A, B, and optional CIN input.

**COUT**  
→ Output carry indicating overflow beyond 8-bit sum capacity result.

**TB Architecture:-**



**1. DUT (Design Under Test)**

* This is the actual 8-bit adder you're testing.
* It takes two 8-bit inputs and produces a 9-bit output (8-bit sum + 1-bit carry, if any).

### ****2. Interface****

* This connects the UVM testbench to the DUT.
* It groups signals like a, b, sum, carry, and possibly clk or reset.

### ****3. Test****

* The top-level module that configures and runs the environment.
* It decides the type of tests (e.g., corner cases, random inputs).

### ****4. Env (Environment)****

* Contains all UVM components needed to drive and monitor the DUT.
* Includes Agent and Scoreboard.

### ****5. Agent****

* Contains three main components:
  + **Generator**: Creates random or constrained 8-bit input vectors a and b.
  + **Driver**: Takes transactions from the generator and applies them to the DUT via the interface.
  + **Monitor**: Observes DUT signals and captures outputs (sum and carry).