**Experiment 9 – Full Adder**

**Objective:**

Write the code to design a 4 bit adder and write its test bench is System Verilog

**Tool Used:**

Xilinx ISE.

**Theory:**

An SV Test bench is designed using system Verilog components.

**Code:**

**DUT:**

module add(input [3:0] a,b, input, output [3:0] sum, output cout);

  assign {cout,sum} = a+b;

endmodule

**TB:**

//TB\_TOP

`include "interface.sv"

`include "test"

module tbench\_top;

  intf i\_intf();

  test t1(i\_intf);

  full\_adder f1(i\_intf.a,i\_intf.b,i\_intf.sum,i\_intf.cy);

endmodule

//INTERFACE

interface intf();

  logic [3:0] a,b,sum;

  logic cy;

endinterface

//TRANSACTION

class transaction;

  rand bit [3:0] a;

  rand bit [3:0] b;

  bit [3:0] sum;

  bit cy;

  function void display(string name);

    $display("@ %s component: inputs are %0d,%0d, and outputs: {cy,sum}=%0d",name,a,b,{cy,sum});

  endfunction

endclass

//GENERATOR

class generator;

  transaction trans;

  mailbox gen2driv;

  function new(mailbox gen2driv);

    this.gen2driv = gen2driv;

  endfunction

  task main();

    trans = new();

    trans.randomize();

    trans.display("Generator");

    gen2driv.put(trans);

  endtask

endclass

//DRIVER

class driver;

  virtual intf vif;

  mailbox gen2driv;

  transaction trans;

  function new(virtual intf vif, mailbox gen2driv);

    this.vif = vif;

    this.gen2driv = gen2driv;

  endfunction

  task main();

    gen2driv.get(trans);

    vif.a <= trans.a;

    vif.b <= trans.b;

    trans.sum = vif.sum;

    trans.cy = vif.cy;

    trans.display("Driver");

  endtask

endclass

//TEST

`include "environment.sv"

program test(intf i\_intf);

  environment env;

  initial begin

    env = new(i\_intf);

    env.run();

  end

endprogram

//INTERFACE

interface intf();

  logic [3:0] a,b,sum;

  logic cy;

endinterface

//MONITOR

class monitor;

  virtual intf vif;

  mailbox mon2scb;

  transaction trans;

  function new(virtual intf vif, mailbox mon2scb);

    this.vif = vif;

    this.mon2scb = mon2scb;

  endfunction

  task main();

    #3;

    trans = new();

    trans.a = vif.a;

    trans.b = vif.b;

    trans.sum = vif.sum;

    trans.cy = vif.cy;

    mon2scb.put(trans);

    trans.display("Monitor");

  endtask

endclass

//SCOREBOARD

class scoreboard;

  mailbox mon2scb;

  transaction trans;

  function new(mailbox mon2scb);

    this.mon2scb = mon2scb;

  endfunction

  task main();

    mon2scb.get(trans);

    if({trans.cy,trans.sum} == trans.a+trans.b)

      $display("Success");

    else

      $error("Wrong Result");

    trans.display("Scoreboard");

  endtask

endclass

//ENVIROINMENT

`include "transaction.sv"

`include "generator.sv"

`include "driver.sv"

`include "monitor.sv"

`include "scoreboard.sv"

class environment;

  generator gen;

  driver drv;

  monitor mon;

  scoreboard scb;

  mailbox m1,m2;

  virtual intf vif;

  function new(virtual intf vif);

    this.vif = vif;

    m1 = new();

    m2 = new();

    gen = new(m1);

    drv = new(vif,m1);

    mon = new(vif,m2);

    scb = new(m2);

  endfunction

  task test();

    fork

      gen.main();

      mon.main();

      drv.main();

      scb.main();

    join

  endtask

  task run;

    test();

    #5;

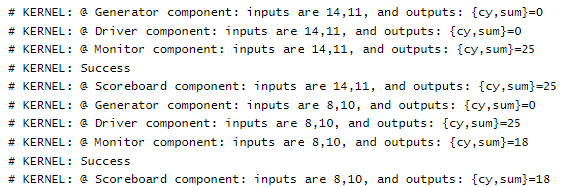
    test();

    $finish;

  endtask

endclass

**Output:**

****

**Result:**

The given problem statement is executed and verified to be correct.