HALF ADDER VERIFICATION USING SYSYTEM VERILOG

1.Transaction:

```
class transaction;
  rand bit a;
  rand bit b;
  bit sum;
  bit carry;

  constraint ab_values{ a dist {1:=100};
                      b dist {1:=100};}

  function void display(string name);
    $display("----------------------");
    $display("%s",name);
    $display("----------------------");

    $display("a=%0d,b=%0d",a,b);
    $display("sum=%0d,carry=%0d",sum,carry);
    $display("--------------------------");

  endfunction
endclass
```

2.Generator

```
class generator;
  transaction trans;
 mailbox  gen2driv;

  function new(mailbox  gen2driv);
```

```
    this.gen2driv = gen2driv;
  endfunction

  task main();
    repeat(1)
      begin
      trans = new();
      trans.randomize();
      trans.display("Generator");
      gen2driv.put(trans);
    end
  endtask
endclass
```

3.driver

```
class driver;
  virtual intf vif;
 mailbox gen2driv;

  function new(virtual intf vif, mailbox  gen2driv);
    this.vif = vif;
    this.gen2driv = gen2driv;
  endfunction

  task main();
    repeat(1)
      begin
      transaction trans;
      gen2driv.get(trans);
      vif.a <= trans.a;
      vif.b <= trans.b;
      trans.sum = vif.sum;
        trans.carry=vif.carry;
        trans.display("Driver");

    end
  endtask
endclass
```

4.Interface.

```
interface intf();
    logic a;
    logic b;
    logic sum;
    logic carry;
endinterface
```

5.monitor.

```
class monitor;
  virtual intf vif;
  mailbox  mon2scb;

  function new(virtual intf vif, mailbox  mon2scb);
    this.vif = vif;
    this.mon2scb = mon2scb;
  endfunction

  task main();
    repeat(1)

      #10;
    begin
      transaction trans;
      trans = new();
      trans.a = vif.a;
      trans.b = vif.b;
      trans.sum = vif.sum;
      trans.carry = vif.carry;
      mon2scb.put(trans);
      trans.display("Monitor");
    end
  endtask
endclass
```

6.Scoreboard

```
class scoreboard;
  mailbox  mon2scb;

  function new(mailbox  mon2scb);
    this.mon2scb = mon2scb;
  endfunction

  task main();
    transaction trans;
    repeat(1)
      begin
        mon2scb.get(trans);
        if(((trans.a ^ trans.b) == trans.sum) && ((trans.a & trans.b) == trans.carry))
          $display("Result is Expected");
        else
          $error("Wrong is as Expected");
        trans.display("Scoreboard");
      end
  endtask
```

```
endclass
```

7.Environment.

```
`include "transaction.sv"
`include "generator.sv"
`include "driver.sv"
`include "monitor.sv"
`include "scoreboard.sv"
class environment;

  generator gen;
  driver driv;
  monitor mon;
  scoreboard scb;

  mailbox m1;
  mailbox m2;
  virtual intf vif;

  function new(virtual intf vif);
    this.vif = vif;
    m1 = new();
    m2 = new();
    gen = new(m1);
    driv = new(vif,m1);
    mon = new(vif,m2);
    scb = new(m2);
  endfunction

  task test();
    fork
      gen.main();
      driv.main();
      mon.main();
      scb.main();
    join
  endtask

  task run();
    test();
    $finish;
  endtask
endclass
```

8.Test

```
`include "environment.sv"

program  test(intf i_intf);
  environment env;

  initial begin
    env = new(i_intf);
    env.run();
  end
endprogram
```

9.Top Level Testbench.

```
`include "interface.sv"
`include "test"

module top;
  intf i_intf();
  test t1(i_intf);
  half_adder h1(
    .a(i_intf.a),
    .b(i_intf.b),
    .s(i_intf.sum),
    .c(i_intf.carry)
  );
Endmodule
```

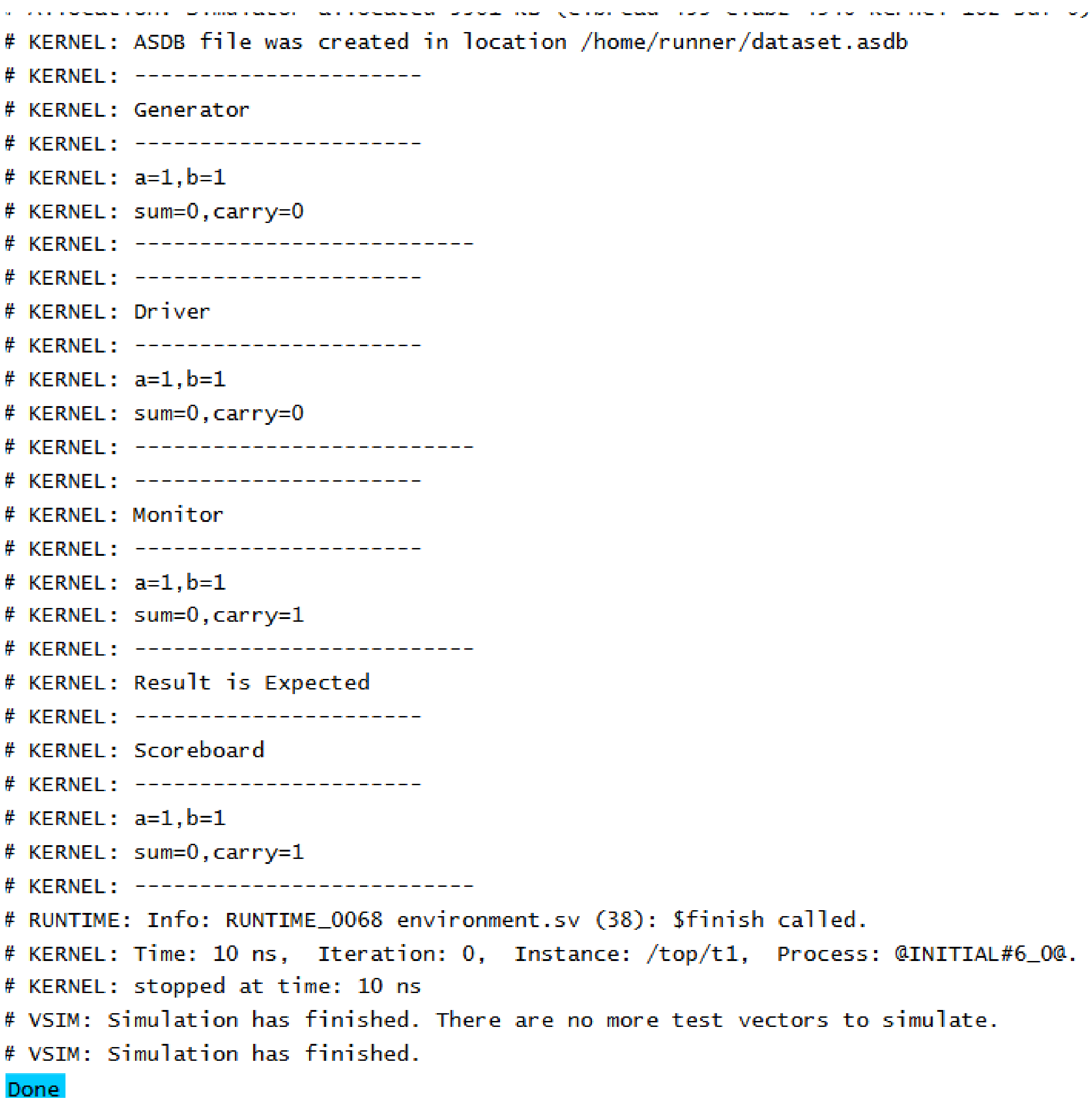
10.Design

```
module half_adder(a,b,c,s);

  input a,b;
  output s,c;
  xor x1(s,a,b);
  and a1(c,a,b);

endmodule
```

OUTPUT:



```
# KERNEL: ASDB file was created in location /home/runner/dataset.asdb
# KERNEL: ----------------------
# KERNEL: Generator
# KERNEL: ----------------------
# KERNEL: a=1,b=1
# KERNEL: sum=0,carry=0
# KERNEL: --------------------------
# KERNEL: ----------------------
# KERNEL: Driver
# KERNEL: ----------------------
# KERNEL: a=1,b=1
# KERNEL: sum=0,carry=0
# KERNEL: --------------------------
# KERNEL: ----------------------
# KERNEL: Monitor
# KERNEL: ----------------------
# KERNEL: a=1,b=1
# KERNEL: sum=0,carry=1
# KERNEL: --------------------------
# KERNEL: Result is Expected
# KERNEL: ----------------------
# KERNEL: Scoreboard
# KERNEL: ----------------------
# KERNEL: a=1,b=1
# KERNEL: sum=0,carry=1
# KERNEL: --------------------------
# RUNTIME: Info: RUNTIME_0068 environment.sv (38): $finish called.
# KERNEL: Time: 10 ns,  Iteration: 0,  Instance: /top/t1,  Process: @INITIAL#6_0@.
# KERNEL: stopped at time: 10 ns
# VSIM: Simulation has finished. There are no more test vectors to simulate.
# VSIM: Simulation has finished.
Done
```