## **EXPERIMENT 6 CODE**

```
task main();
//DESIGN
module full adder(input a,b,cin,
                                                  trans = new();
                                                  trans.randomize();
output s,cy);
                                                  trans.display("Generator");
  assign {cy,s} = a+b+cin;
                                                  gen2driv.put(trans);
endmodule
                                                endtask
//TRANSACTION
                                              endclass
class transaction;
  rand bit a;
                                              //DRIVER
  rand bit b;
                                              class driver;
                                                virtual intf vif;
  rand bit cin;
  bit s;
                                                mailbox gen2driv;
                                                transaction trans;
  bit cy;
  function void display(string name);
      $display("@ %s component:
                                                    function new(virtual intf vif,
     inputs are %0d,%0d,%0d, and
                                                    mailbox gen2driv);
                                                  this.vif = vif;
     outputs: sum=%0d,
     cy=%0d",name,a,b,cin,s,cy);
                                                  this.gen2driv = gen2driv;
                                                endfunction
  endfunction
endclass
                                                task main();
                                                  gen2driv.get(trans);
//GENERATOR
class generator;
  transaction trans;
                                                  vif.a <= trans.a;</pre>
  mailbox gen2driv;
                                                  vif.b <= trans.b;</pre>
  function new(mailbox gen2driv);
                                                  vif.cin <= trans.cin;</pre>
    this.gen2driv = gen2driv;
                                                  trans.s = vif.s;
  endfunction
                                                  trans.cy = vif.cy;
```

```
trans.display("Driver");
                                               mon2scb.get(trans);
  endtask
                                                 if({trans.cy,trans.s} ==
                                             trans.a+trans.b+trans.cin)
endclass
                                                   $display("Success");
                                                 else
//MONITOR
                                                   $error("Wrong Result");
class monitor;
                                                 trans.display("Scoreboard");
  virtual intf vif;
                                               endtask
  mailbox mon2scb;
                                             endclass
  transaction trans;
  function new(virtual intf vif,
                                             //INTERFACE
mailbox mon2scb);
                                             interface intf();
    this.vif = vif;
                                               logic a,b,cin,s,cy;
    this.mon2scb = mon2scb;
                                             endinterface
  endfunction
                                             //ENVIRONMENT
  task main();
                                             `include "transaction.sv"
    #3;
                                             `include "generator.sv"
    trans = new();
                                             `include "driver.sv"
    trans.a = vif.a;
                                             `include "monitor.sv"
    trans.b = vif.b;
                                             `include "scoreboard.sv"
    trans.cin = vif.cin;
    trans.s = vif.s;
                                             class environment;
    trans.cy = vif.cy;
                                               generator gen;
    mon2scb.put(trans);
                                               driver drv;
    trans.display("Monitor");
                                               monitor mon;
  endtask
                                               scoreboard scb;
endclass
                                               mailbox m1,m2;
//SCOREBOARD
                                               virtual intf vif;
class scoreboard;
                                               function new(virtual intf vif);
  mailbox mon2scb;
                                                 this.vif = vif;
  transaction trans;
                                                 m1 = new();
                                                 m2 = new();
  function new(mailbox mon2scb);
                                                 gen = new(m1);
    this.mon2scb = mon2scb;
                                                 drv = new(vif, m1);
  endfunction
                                                 mon = new(vif, m2);
                                                 scb = new(m2);
  task main();
```

```
endfunction
                                                endtask
                                              endclass
  task test();
                                             //TEST
    fork
                                              `include "environment.sv"
      gen.main();
      mon.main();
      drv.main();
                                              program test(intf i_intf);
      scb.main();
                                                environment env;
    join
  endtask
                                                initial begin
                                                  env = new(i_intf);
  task run;
                                                  env.run();
    test();
                                                end
    $finish;
                                              endprogram
//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.cin,i intf.s,i intf.cy);
endmodule
```

## **EXPERIMENT 7 CODE**

```
//GENERATOR
//DESIGN
module full adder(input [3:0] a,b,
                                             class generator;
output [3:0] sum, output cy);
                                               transaction trans;
  assign {cy,sum} = a+b;
                                               mailbox gen2driv;
endmodule
                                               function new(mailbox gen2driv);
                                                 this.gen2driv = gen2driv;
//TRANSACTION
                                               endfunction
class transaction;
  rand bit [3:0] a;
                                               task main();
  rand bit [3:0] b;
  bit [3:0] sum;
                                                 trans = new();
  bit cy;
                                                 trans.randomize();
                                                 trans.display("Generator");
  function void display(string name);
                                                 gen2driv.put(trans);
    $display("@ %s component: inputs
                                               endtask
are %0d,%0d, and outputs:
{cy,sum}=%0d",name,a,b,{cy,sum});
                                             endclass
  endfunction
endclass
```

```
//DRIVER
                                              task main();
class driver;
                                                  #3;
  virtual intf vif;
                                                  trans = new();
  mailbox gen2driv;
                                                  trans.a = vif.a;
                                                  trans.b = vif.b;
  transaction trans;
                                                  trans.sum = vif.sum;
                                                  trans.cy = vif.cy;
  function new(virtual intf vif,
mailbox gen2driv);
                                                  mon2scb.put(trans);
    this.vif = vif;
                                                  trans.display("Monitor");
    this.gen2driv = gen2driv;
                                                endtask
                                              endclass
  endfunction
  task main();
                                              //SCOREBOARD
    gen2driv.get(trans);
                                              class scoreboard;
                                                mailbox mon2scb;
    vif.a <= trans.a;</pre>
                                                transaction trans;
                                                function new(mailbox mon2scb);
    vif.b <= trans.b;</pre>
    trans.sum = vif.sum;
                                                  this.mon2scb = mon2scb;
    trans.cy = vif.cy;
                                                endfunction
    trans.display("Driver");
  endtask
                                                task main();
                                                  mon2scb.get(trans);
endclass
                                                  if({trans.cy,trans.sum} ==
                                              trans.a+trans.b)
                                                    $display("Success");
//MONITOR
class monitor;
                                                    $error("Wrong Result");
  virtual intf vif;
                                                  trans.display("Scoreboard");
  mailbox mon2scb;
                                                endtask
  transaction trans;
                                              endclass
  function new(virtual intf vif,
                                              //INTERFACE
mailbox mon2scb);
                                              interface intf();
    this.vif = vif;
                                                logic [3:0] a,b,sum;
    this.mon2scb = mon2scb;
                                                logic cy;
  endfunction
                                              endinterface
```

```
//ENVIRONMENT
                                              fork
                                                gen.main();
`include "transaction.sv"
                                                mon.main();
`include "generator.sv"
                                                drv.main();
`include "driver.sv"
                                                scb.main();
`include "monitor.sv"
                                              join
`include "scoreboard.sv"
                                             endtask
class environment;
                                            task run;
 generator gen;
                                              test();
 driver drv;
                                              #5;
 monitor mon;
                                              test();
  scoreboard scb;
                                              $finish;
 mailbox m1,m2;
                                            endtask
                                           endclass
 virtual intf vif;
 function new(virtual intf vif);
                                           //TEST
   this.vif = vif;
                                           `include "environment.sv"
   m1 = new();
                                           program test(intf i_intf);
   m2 = new();
   gen = new(m1);
                                             environment env;
   drv = new(vif,m1);
   mon = new(vif, m2);
                                             initial begin
   scb = new(m2);
                                              env = new(i_intf);
  endfunction
                                              env.run();
                                            end
 task test();
                                           endprogram
//TB TOP
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
```

## **EXPERIMENT 8 CODE**

```
//DESIGN
                                               task main();
module full adder(input [3:0]
                                                  trans = new();
a,b,input clk,reset, output reg
                                                  repeat(count)begin
[6:0]sum);
                                                    trans.randomize();
                                                    trans.display("Generator");
  always@(posedge clk)begin
                                                    gen2driv.put(trans);
    if(reset) sum = 0;
                                                    @(drv_done);
    else sum = a+b;
  end
                                                  end
endmodule
                                                endtask
                                              endclass
//TRANSACTION
class transaction;
                                              //DRIVER
                                              class driver;
  rand bit [3:0] a;
  rand bit [3:0] b;
                                                virtual intf vif;
  bit [6:0] sum;
                                                mailbox gen2driv;
                                                transaction trans;
  function void display(string name);
     $display("@ %s component: inputs
                                                function new(virtual intf vif,
     are %0d,%0d, and outputs:
                                              mailbox gen2driv);
     sum=%0d",name,a,b,sum);
                                                  this.vif = vif;
  endfunction
                                                  this.gen2driv = gen2driv;
endclass
                                                endfunction
//GENERATOR
                                                task main();
class generator;
                                                  forever begin
                                                  gen2driv.get(trans);
  transaction trans;
  mailbox gen2driv;
                                                  vif.a <= trans.a;</pre>
  int count;
                                                  vif.b <= trans.b;</pre>
                                                  trans.sum = vif.sum;
                                                  trans.display("Driver");
event drv_done;
  function new(mailbox gen2driv);
                                                endtask
    this.gen2driv = gen2driv;
                                              endclass
  endfunction
```

```
//MONITOR
                                             task main();
class monitor:
                                                 forever begin
  virtual intf vif;
                                                 mon2scb.get(trans);
  mailbox mon2scb;
                                                 no trans++;
  transaction trans;
                                                 trans.display("Scoreboard");
 event drv_done;
                                                 if(trans.sum ==
                                             trans.a+trans.b)begin
  function new(virtual intf vif,
                                                   X++;
mailbox mon2scb);
                                                   $display("Success \n\n");
    this.vif = vif;
                                                 end
    this.mon2scb = mon2scb;
                                                 else
                                                   $error("Wrong Result");
  endfunction
                                                 end endtask
  task main();
                                             endclass
    forever begin
    trans = new();
                                             //INTERFACE
    @(posedge vif.clk);
                                             interface intf(input logic clk,reset);
    #1;
                                               logic [3:0] a,b; logic [6:0] sum;
    trans.a = vif.a;
                                             endinterface
    trans.b = vif.b;
                                             //ENVIRONMENT
    trans.sum = vif.sum;
                                             `include "transaction.sv"
                                             `include "generator.sv"
    mon2scb.put(trans);
                                             `include "driver.sv"
    trans.display("Monitor");
    ->drv done;
                                             `include "monitor.sv"
    end
                                             `include "scoreboard.sv"
                                             class environment;
  endtask
endclass
                                               generator gen;
//SCOREBOARD
                                               driver drv;
class scoreboard;
                                               monitor mon;
  mailbox mon2scb;
                                               scoreboard scb;
  transaction trans;
                                               mailbox m1,m2;
  int no trans,x;
                                               event dr;
  function new(mailbox mon2scb);
    this.mon2scb = mon2scb;
  endfunction
```

```
gen.drv_done = dr;
virtual intf vif;
  function new(virtual intf vif);
                                                 mon.drv done = dr;
    this.vif = vif;
                                                 test();
    m1 = new(); m2 = new();
                                                 wait(gen.count == scb.no trans);
    gen = new(m1);
                                                 $display("%0d ON %0d TRANSACTIONS
                                             SUCCESSFULL",scb.x,scb.no_trans);
    drv = new(vif, m1);
    mon = new(vif, m2);
                                                  $finish;
    scb = new(m2);
                                               endtask
  endfunction
                                             endclass
  task test();
                                             //TEST
    fork
                                             `include "environment.sv"
      gen.main(); mon.main();
                                             program test(intf i_intf);
      drv.main(); scb.main();
                                               environment env;
    join any
                                               initial begin
                                                 env = new(i_intf);
  endtask
                                                 env.run();
  task run;
                                               end
    gen.count = 6;
                                             endprogram
                                             full adderf1(i intf.a,i intf.b,i intf.
//TB TOP
`include "interface.sv"
                                             clk,i intf.reset,i intf.sum);
`include "test"
                                               initial begin
                                                 $dumpfile("dump.vcd"); $dumpvars;
module tbench top;
  bit clk,reset;
                                                 reset = 1; #5; reset = 0;
  always #5 clk = !clk;
                                               end
  intf i_intf(clk,reset);
                                             endmodule
  test t1(i intf);
```

## **EXPERIMENT 9 CODE**

```
//constraint 1
class packet;
  rand bit [3:0] addr;
  constraint addr range { addr > 5; }
endclass
module const blocks;
  initial begin
    packet pkt;
    pkt = new();
    repeat(10) begin
      pkt.randomize();
      $display("\taddr = %0d",pkt.addr);
    end
  end
endmodule
//constraint 2
class packet;
  rand bit [3:0] addr;
  constraint addr_range;
endclass
constraint packet :: addr_range { addr > 5; }
module const blocks;
  initial begin
    packet pkt;
    pkt = new();
    repeat(10) begin
      pkt.randomize();
      $display("\taddr = %0d",pkt.addr);
    end
  end
endmodule
```

```
//constraint_3
class packet;
  rand bit [3:0] addr;
  rand bit [3:0] start;
  rand bit [3:0] stop;
  constraint addr_range;
endclass
constraint packet :: addr range { addr inside {[start:stop]}; }
module const blocks;
  initial begin
    packet pkt;
    pkt = new();
    repeat(10) begin
      pkt.randomize();
      $display("start is %0d, stop is %0d",pkt.start, pkt.stop);
      $display("\taddr = %0d",pkt.addr);
    end
  end
endmodule
//constraint 4
class packet;
  rand bit [3:0] addr;
  rand bit [3:0] start;
  rand bit [3:0] stop;
  constraint addr_range;
endclass
constraint packet :: addr range {! (addr inside {[start:stop]});
                                   start < stop;}</pre>
module const blocks;
  initial begin
    packet pkt;
    pkt = new();
    repeat(10) begin
      pkt.randomize();
      $display("start is %0d, stop is %0d",pkt.start, pkt.stop);
      $display("\taddr = %0d",pkt.addr);
    end
  end
endmodule
```