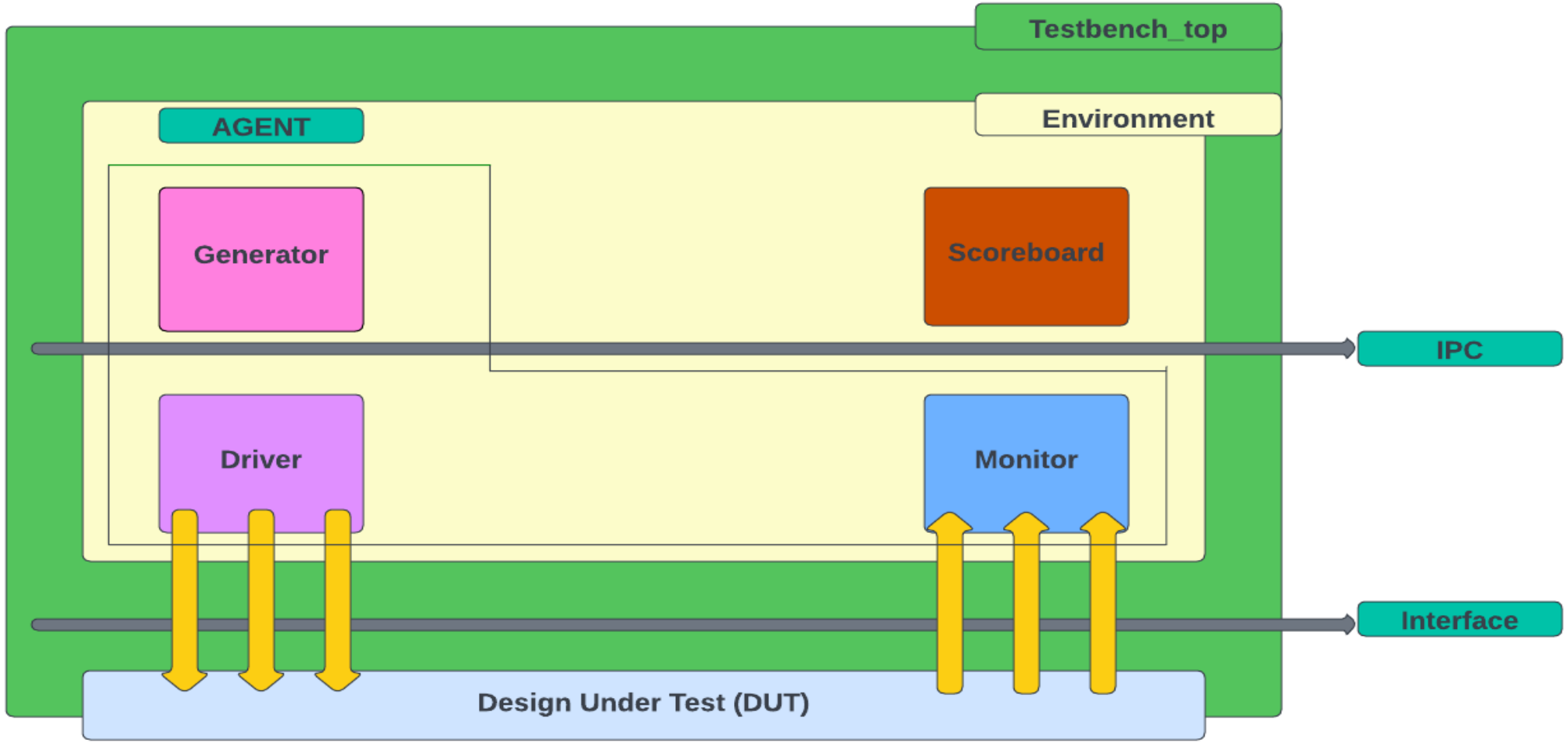
System Verilog Lab

Verification of Full Adder using System Verilog Testbench Components

Name: Mallipudi Janaki Vara Prakash
Roll no: 19131A04D8
Branch/Section: ECE-03

Testbench Components



Design Under Test (DUT)

```
module my_adder (adder_if _if);
  always @(posedge _if.tb_clk); begin
    always_comb begin
    if (_if.rstn) begin
      _if.sum <= 0;
      _if.carry <= 0;
    end else begin
     {_if.carry, _if.sum} <= _if.a + _if.b;
    end
  end
  end
endmodule
```

```
interface adder_if();
  logic     rstn;
  logic [7:0]   a;
  logic [7:0]   b;
  logic [7:0]   sum;
  logic     carry;
   logic tb_clk;
  initial tb_clk <= 0;
  always #10 tb_clk = ~tb_clk;
endinterface

interface clk_if();
  logic tb_clk;
  initial tb_clk <= 0;
  always #10 tb_clk = ~tb_clk;
endinterface
```

Packet class

- Contains Variables for all the input/outputs port present in DUT to share among classes.

```
class Packet;
  rand bit     rstn;
  rand bit[7:0] a;
  rand bit[7:0] b;
  bit [7:0]   sum;
  bit       carry;
  function void print(string tag="");
    $display ("T=%0t %s a=0x%0h b=0x%0h sum=0x%0h carry=0x%0h", $time,
tag, a, b, sum, carry);
  endfunction
  function void copy(Packet tmp);
    this.a = tmp.a;
    this.b = tmp.b;
    this.rstn = tmp.rstn;
    this.sum = tmp.sum;
    this.carry = tmp.carry;
  endfunction
endclass
```

Generator class

- Generate random stimulus and send it to driver using IPC.

```
class generator;
  int   loop = 10;
  event drv_done;
  mailbox drv_mbx;

  task run();
    for (int i = 0; i < loop; i++) begin
      Packet item = new;
      item.randomize();
      $display ("T=%0t [Generator] Loop:%0d/%0d create next item", $time, i+1,
loop);
      drv_mbx.put(item);
      $display ("T=%0t [Generator] Wait for driver to be done", $time);
      @(drv_done);
    end
  endtask
endclass
```

Driver class

- **Receive stimulus from Generator and Trigger respective signals of DUT with help of Interface.**

```
class driver;
  virtual adder_if m_adder_vif;
  virtual clk_if  m_clk_vif;
  event drv_done;
  mailbox drv_mbx;
  task run();
    $display ("T=%0t [Driver] starting ...", $time);
    forever begin
      Packet item;
      $display ("T=%0t [Driver] waiting for item ...", $time);
      drv_mbx.get(item);
      @ (posedge m_clk_vif.tb_clk);
    item.print("Driver");
      m_adder_vif.rstn <= item.rstn;
      m_adder_vif.a <= item.a;
      m_adder_vif.b <= item.b;
      ->drv_done;
    end
  endtask
endclass
```

Monitor class

- Receive response from DUT and send it to Scoreboard using IPC.

```
class monitor;
  virtual adder_if   m_adder_vif;
  virtual clk_if   m_clk_vif;
  mailbox scb_mbx;
  task run();
    $display ("T=%0t [Monitor] starting ...", $time);
    forever begin
    Packet m_pkt = new();
      @(posedge m_clk_vif.tb_clk);
      #1;
        m_pkt.a   = m_adder_vif.a;
        m_pkt.b   = m_adder_vif.b;
        m_pkt.rstn   = m_adder_vif.rstn;
        m_pkt.sum   = m_adder_vif.sum;
        m_pkt.carry = m_adder_vif.carry;
        m_pkt.print("Monitor");
      scb_mbx.put(m_pkt);
    end
  endtask
endclass
```

Scoreboard class

- Compare response of DUT with Expected/Golden data.

```
class scoreboard;
  mailbox scb_mbx;
  task run();
    forever begin
      Packet item, ref_item;
      scb_mbx.get(item);
      item.print("Scoreboard");
      ref_item = new();
      ref_item.copy(item);
      if (ref_item.rstn)
        {ref_item.carry, ref_item.sum} = ref_item.a +
ref_item.b;
      else
      {ref_item.carry, ref_item.sum} = 0;
  if (ref_item.carry != item.carry) begin
        $display("[%0t] Scoreboard Error! Carry mismatch
ref_item=0x%0h item=0x%0h", $time, ref_item.carry,
item.carry);
      end else begin
        $display("[%0t] Scoreboard Pass! Carry
match ref_item=0x%0h item=0x%0h", $time,
ref_item.carry, item.carry);
      end
  if (ref_item.sum != item.sum) begin
        $display("[%0t] Scoreboard Error! Sum
mismatch ref_item=0x%0h item=0x%0h", $time,
ref_item.sum, item.sum);
      end else begin
        $display("[%0t] Scoreboard Pass! Sum match
ref_item=0x%0h item=0x%0h", $time,
ref_item.sum, item.sum);
      end
    end
  endtask
endclass
```

Environment class

- Hold Generator, Driver, Monitor, Scoreboard together.

```
class env;
 generator     g0;
 driver        d0;
 monitor       m0;
 scoreboard    s0;
 mailbox       scb_mbx;
 virtual adder_if   m_adder_vif;
 virtual clk_if   m_clk_vif;
 event drv_done;
 mailbox drv_mbx;
 function new();
   d0 = new;
  m0 = new;
   s0 = new;
  scb_mbx = new();
   g0 = new;
   drv_mbx = new;
 endfunction
```

```
 virtual task run();
   d0.m_adder_vif = m_adder_vif;
   m0.m_adder_vif = m_adder_vif;
   d0.m_clk_vif = m_clk_vif;
   m0.m_clk_vif = m_clk_vif;
   d0.drv_mbx = drv_mbx;
   g0.drv_mbx = drv_mbx;
   m0.scb_mbx = scb_mbx;
   s0.scb_mbx = scb_mbx;
   d0.drv_done = drv_done;
   g0.drv_done = drv_done;
   fork
     s0.run();
   d0.run();
    m0.run();
      g0.run();
   join_any
  endtask
endclass
```

Test class

- **Hold Environment and control simulation process.**

```
class test;
  env e0;
  mailbox drv_mbx;
  function new();
    drv_mbx = new();
    e0 = new();
  endfunction
  virtual task run();
    e0.d0.drv_mbx = drv_mbx;
    e0.run();
  endtask
endclass
```

Testbench_Top

```
module tb;
  clk_if   m_clk_if   ();
  adder_if   m_adder_if  ();
  my_adder   u0(m_adder_if);
  initial begin
    test t0;
    t0 = new;
    t0.e0.m_adder_vif = m_adder_if;
    t0.e0.m_clk_vif = m_clk_if;
    t0.run();
    #50 $finish;
  end
  initial begin
    $dumpvars;
    $dumpfile ("dump.vcd");
  end
endmodule
```