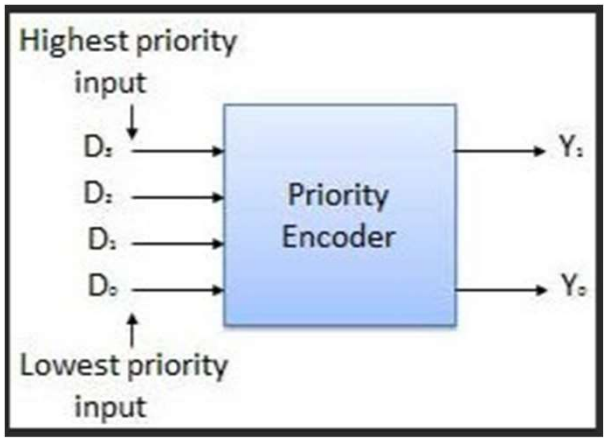
PRIORITY ENCODER LAYERED TESTBENCH

Teammates
Keerthika U PES2UG22EC066
Chandhana S PES2UG22EC040

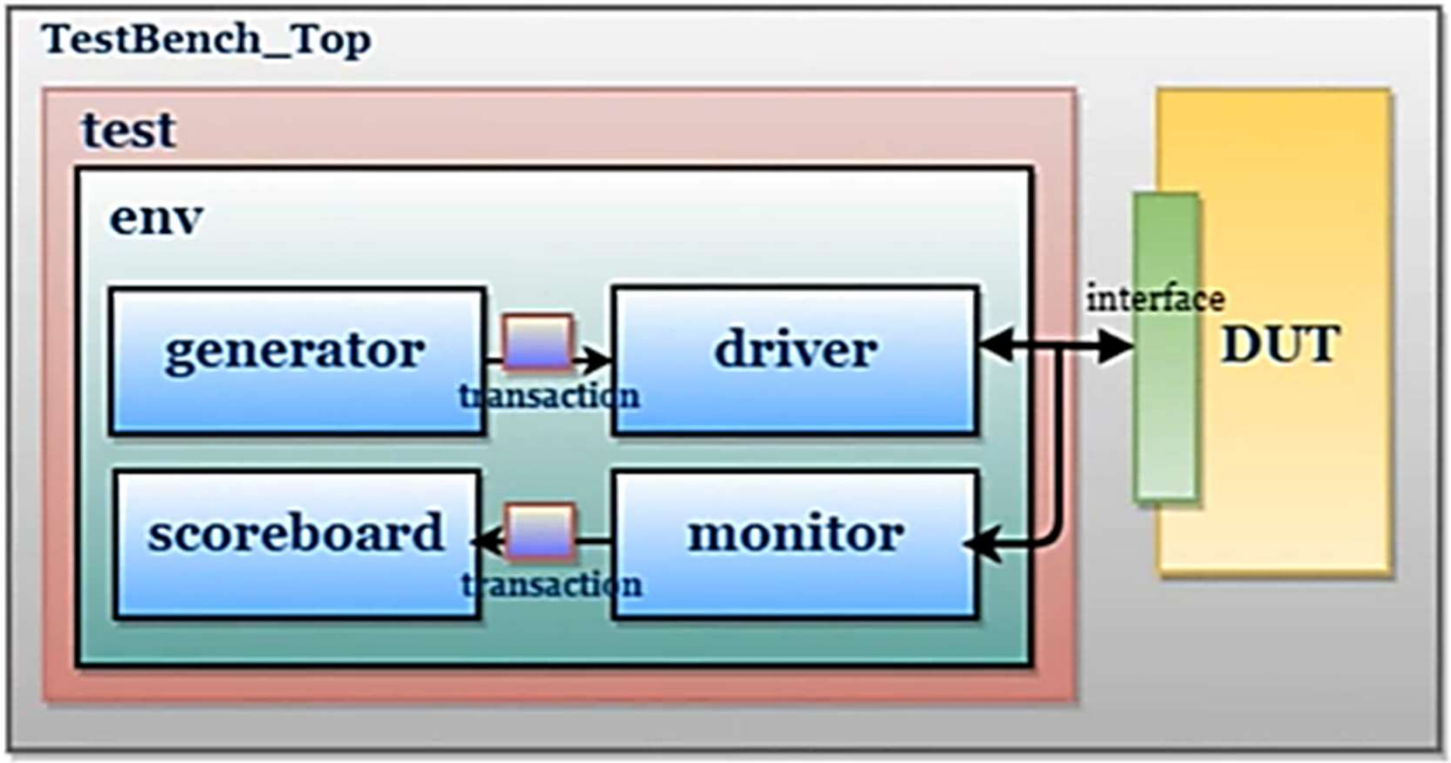
WHAT IS PRIORITY ENCODER

- A priority encoder is a type of digital circuit that takes multiple input lines and encodes the position of the highest-priority active input. The output of a priority encoder is typically a binary representation of the highest-priority active input, along with a "valid" signal to indicate that at least one input is active.

Inputs				Output	
D3	D2	D1	Do	B	A
0	0	0	0	x	x
0	0	0	1	0	0
0	0	1	x	0	1
0	1	x	x	1	0
1	x	x	x	1	1



LAYERED TESTBENCH



Testbench

```
`include "intf.sv"
`include "test.sv"
module tbench_top;

  intf i_intf();

  test t1(i_intf);

   priority_encoder p1 (
    .in(i_intf.in),
    .out(i_intf.out),
    .valid(i_intf.valid)
   );

  initial begin
    $dumpfile("dump.vcd"); $dumpvars;
  end


endmodule
```

Generator

```
class generator;

transation trans;

mailbox gen2driv;

    function new(mailbox gen2driv);//coustom constructor
        this.gen2driv=gen2driv;
    endfunction

    task main();
        repeat(4)
            begin
              trans=new();//handle creation
              trans.randomize();
              trans.display("Generator");
              gen2driv.put(trans);
              end
        endtask

endclass
```

Interface

```
interface intf();
logic [3:0] in;  // 4-bit input
logic [1:0] out;  // 2-bit output for encodedvalue
logic valid ;
endinterface
```

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)
        #5;
        begin
        transation trans;
        trans=new();
        trans.in=vif.in;
        trans.out=vif.out;
        trans.valid=vif.valid;
        mon2scb.put(trans);
        trans.display("Monitor");
        end
    endtask
endclass

```

Scoreboard

```
class scoreboard;
    mailbox mon2scb;

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

    task main;
        transation trans;
      repeat(1)
            begin
            mon2scb.get(trans);

                if(((trans.in[3]|trans.in[1])==trans.out[1])&&
                (trans.in[3]|(~trans.in[2]&trans.in[1]))&&

((trans.in[0]|trans.in[1]|trans.in[2]|trans.in[3])==trans.valid))
                $display("Result is as Expeted");
                else
                $error("Wrong Result");

                trans.display("Scoreboard");
            end
    endtask
endclass
```

Test

```
`include "environment.sv"

program test(intf i_intf);
  environment env;

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

endprogram
```

Transaction

```
class transation;
    rand bit [3:0] in;          // 4-bit random input
    bit [1:0] out;              // 2-bit output
    bit valid;                  // Valid signal


    // Display function to show the transaction details
    function void display(string name);
        $display("---------------------");
        $display("Name: %s", name);
        $display("---------------------");
        $display("input= %4b, out= %2b", in, out);
        $display("---------------------");
    endfunction
endclass

```

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
                transation trans;

                gen2driv.get(trans);

                vif.in <= trans.in;

                 trans.out =vif.out;
                 trans.valid=vif.valid;
                 trans.display("driver");
            end
        endtask
endclass

```

Environment

```
`include "transation.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
```

Design

```
module priority_encoder (
  input  logic [3:0]in, // 4-bit input
  output logic[1:0]out,          // 2-bit output
    output logic valid              // Valid output (1 if input is non-zero)
);

    always_comb begin
      out[1] = in[3] | in[2];
      out[0] = in[3] | ((~in[2]) & in[1]);

        // Valid signal (1 if any input is active)
      valid= in[3] | in[2] | in[1] | in[0];
    end
endmodule

```

OUTPUT

```
#
# run -all
# --------------------
# Name: Generator
# --------------------
# input= 1010, out= 00
# --------------------
# --------------------
# Name: Generator
# --------------------
# input= 0110, out= 00
# --------------------
# --------------------
# Name: Generator
# --------------------
# input= 0101, out= 00
# --------------------
# --------------------
# Name: Generator
# --------------------
# input= 0100, out= 00
# --------------------
# --------------------
# Name: driver
# --------------------
# input= 1010, out= 00
# --------------------
# --------------------
```

```
# --------------------
# --------------------
# Name: driver
# --------------------
# input= 1010, out= 00
# --------------------
# --------------------
# Name: Monitor
# --------------------
# input= 1010, out= 11
# --------------------
# Result is as Expeted
# --------------------
# Name: Scoreboard
# --------------------
# input= 1010, out= 11
# --------------------
# ** Note: $finish    : environment.sv(38)
#    Time: 5 ns  Iteration: 0  Instance: /tbench_top/t1
# End time: 03:25:22 on Dec 02,2024, Elapsed time: 0:00:01
# Errors: 0, Warnings: 0
# *** Summary ********************************************
#     qrun: Errors:    0, Warnings:    0
#     vlog: Errors:    0, Warnings:    1
#     vopt: Errors:    0, Warnings:    3
#     vsim: Errors:    0, Warnings:    0
#   Totals: Errors:    0, Warnings:    4
Done
```

“

THANK YOU