Digital IC Verification

Assignment Three

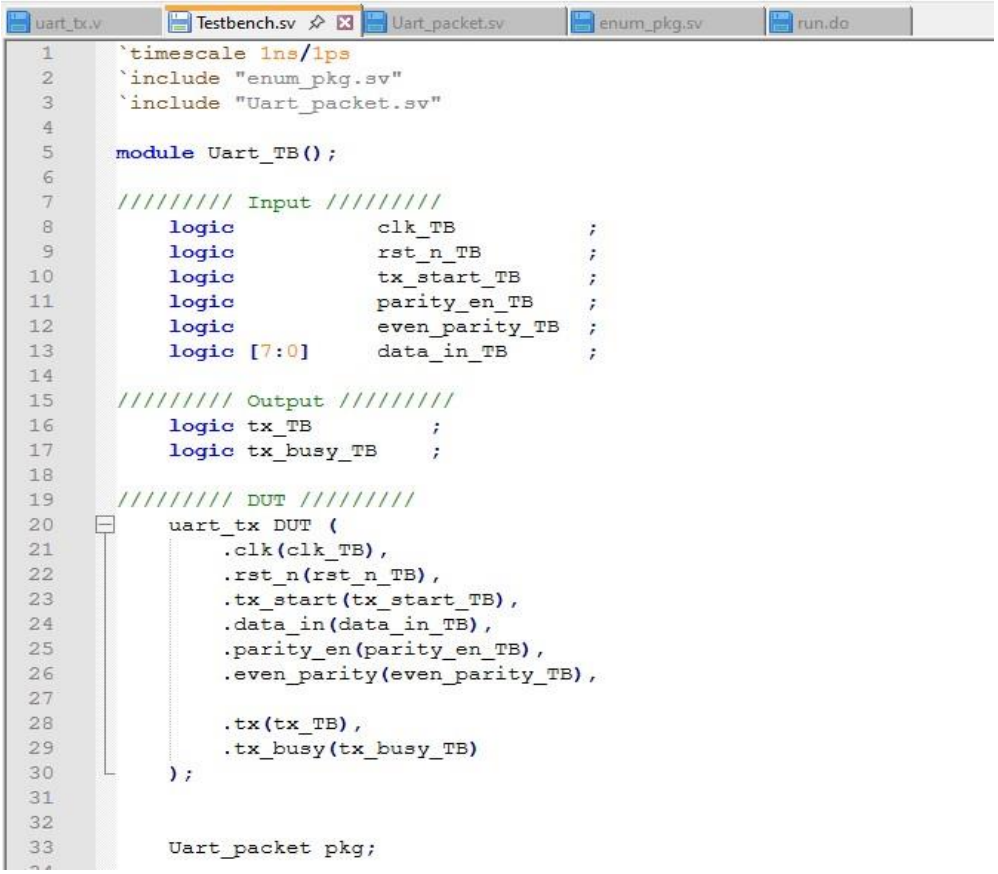
Name : mohamed samy fathy

Under supervision of:

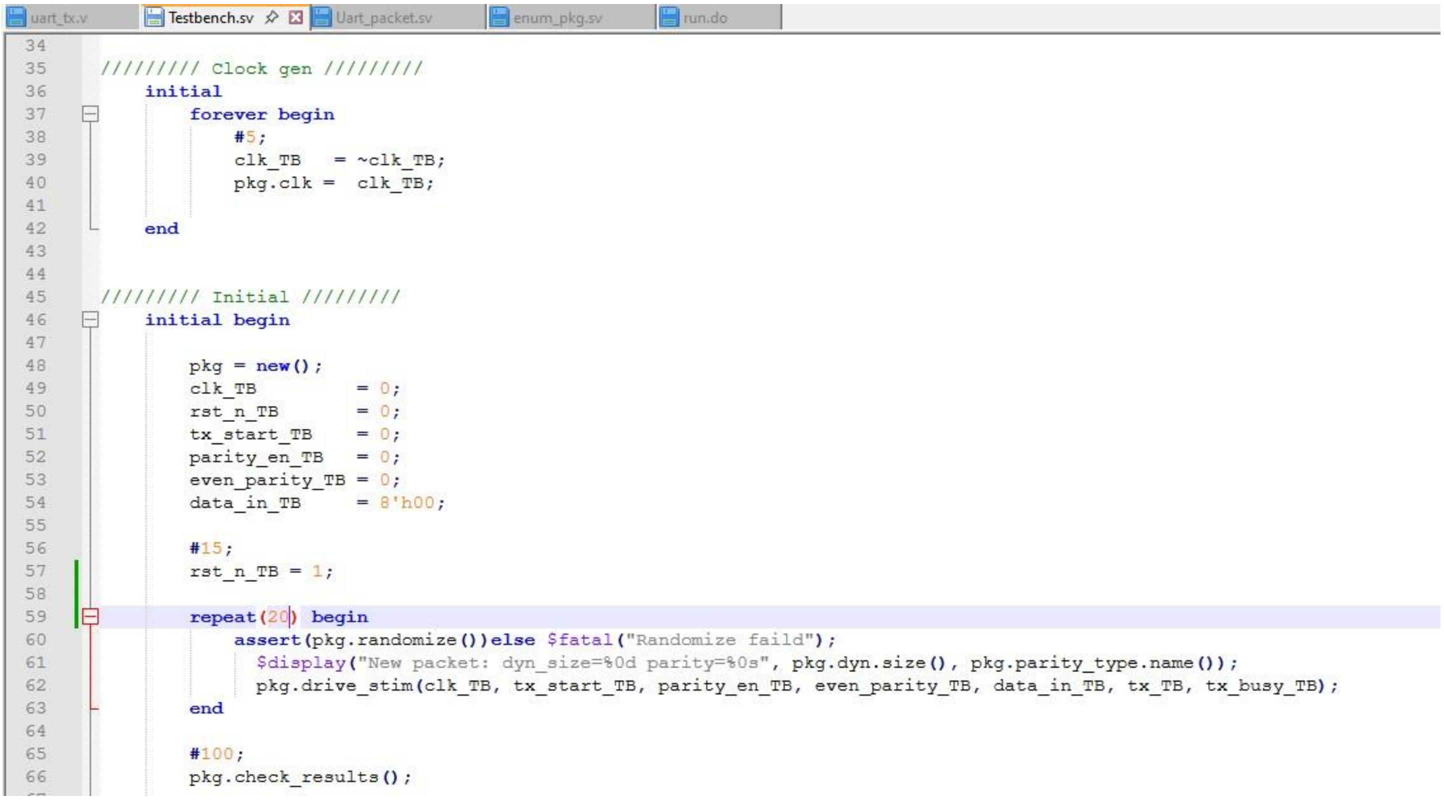
Eng . Hassan Khaled

1 - Snippets for the implemented code.

// Testbench

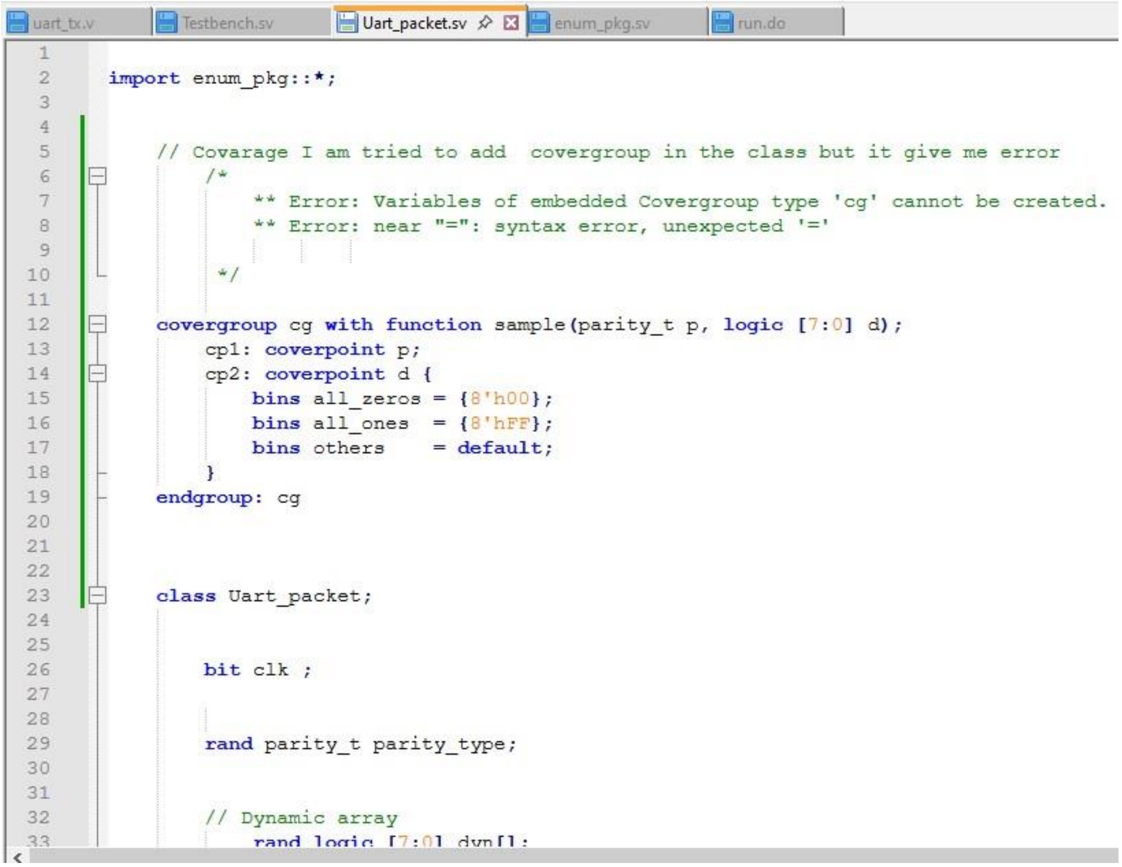


```
`timescale 1ns/1ps
`include "enum_pkg.sv"
`include "Uart_packet.sv"

module Uart_TB();

///////// Input /////////
    logic           clk_TB           ;
    logic           rst_n_TB         ;
    logic           tx_start_TB      ;
    logic           parity_en_TB     ;
    logic           even_parity_TB   ;
    logic [7:0]     data_in_TB       ;

///////// Output /////////
    logic tx_TB         ;
    logic tx_busy_TB    ;

///////// DUT /////////
    uart_tx DUT (
        .clk(clk_TB),
        .rst_n(rst_n_TB),
        .tx_start(tx_start_TB),
        .data_in(data_in_TB),
        .parity_en(parity_en_TB),
        .even_parity(even_parity_TB),

        .tx(tx_TB),
        .tx_busy(tx_busy_TB)
    );


    Uart_packet pkg;
```

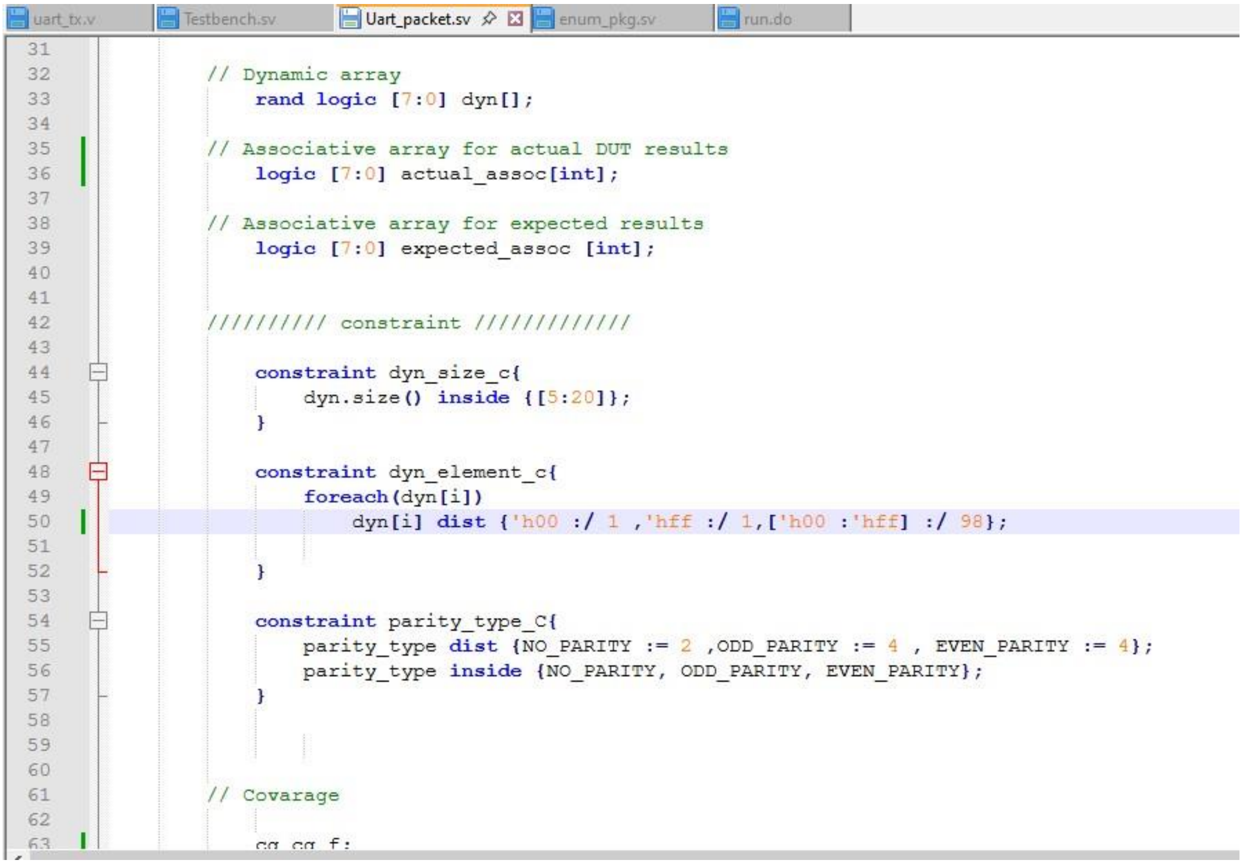


```

///////// Clock gen /////////
    initial
        forever begin
            #5;
            clk_TB   = ~clk_TB;
            pkg.clk =  clk_TB;

    end


///////// Initial /////////
    initial begin

        pkg = new();
        clk_TB         = 0;
        rst_n_TB       = 0;
        tx_start_TB    = 0;
        parity_en_TB   = 0;
        even_parity_TB = 0;
        data_in_TB     = 8'h00;

        #15;
        rst_n_TB = 1;

        repeat(20) begin
            assert(pkg.randomize())else $fatal("Randomize faild");
              $display("New packet: dyn_size=%0d parity=%0s", pkg.dyn.size(), pkg.parity_type.name());
              pkg.drive_stim(clk_TB, tx_start_TB, parity_en_TB, even_parity_TB, data_in_TB, tx_TB, tx_busy_TB);
        end

        #100;
        pkg.check_results();
```

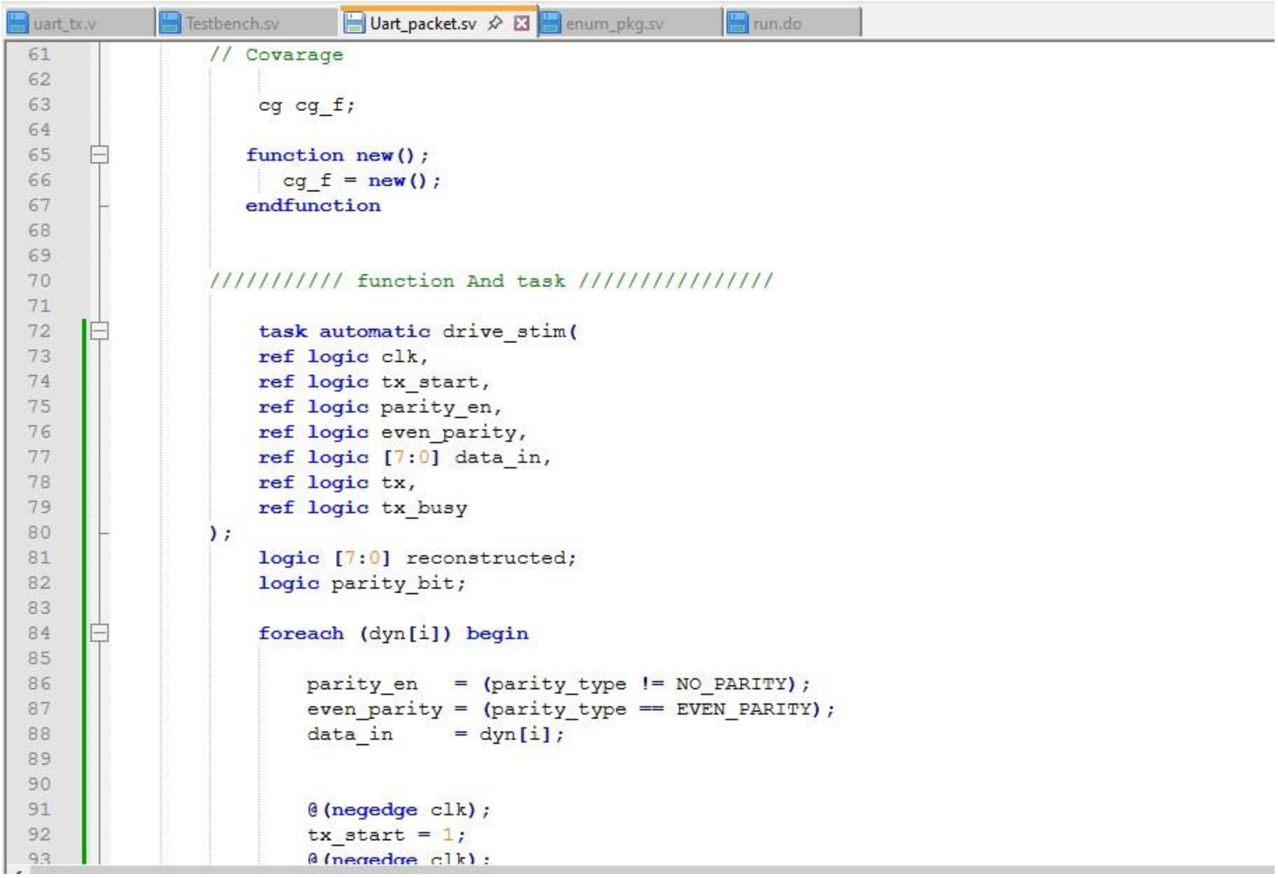
//Uart_packet



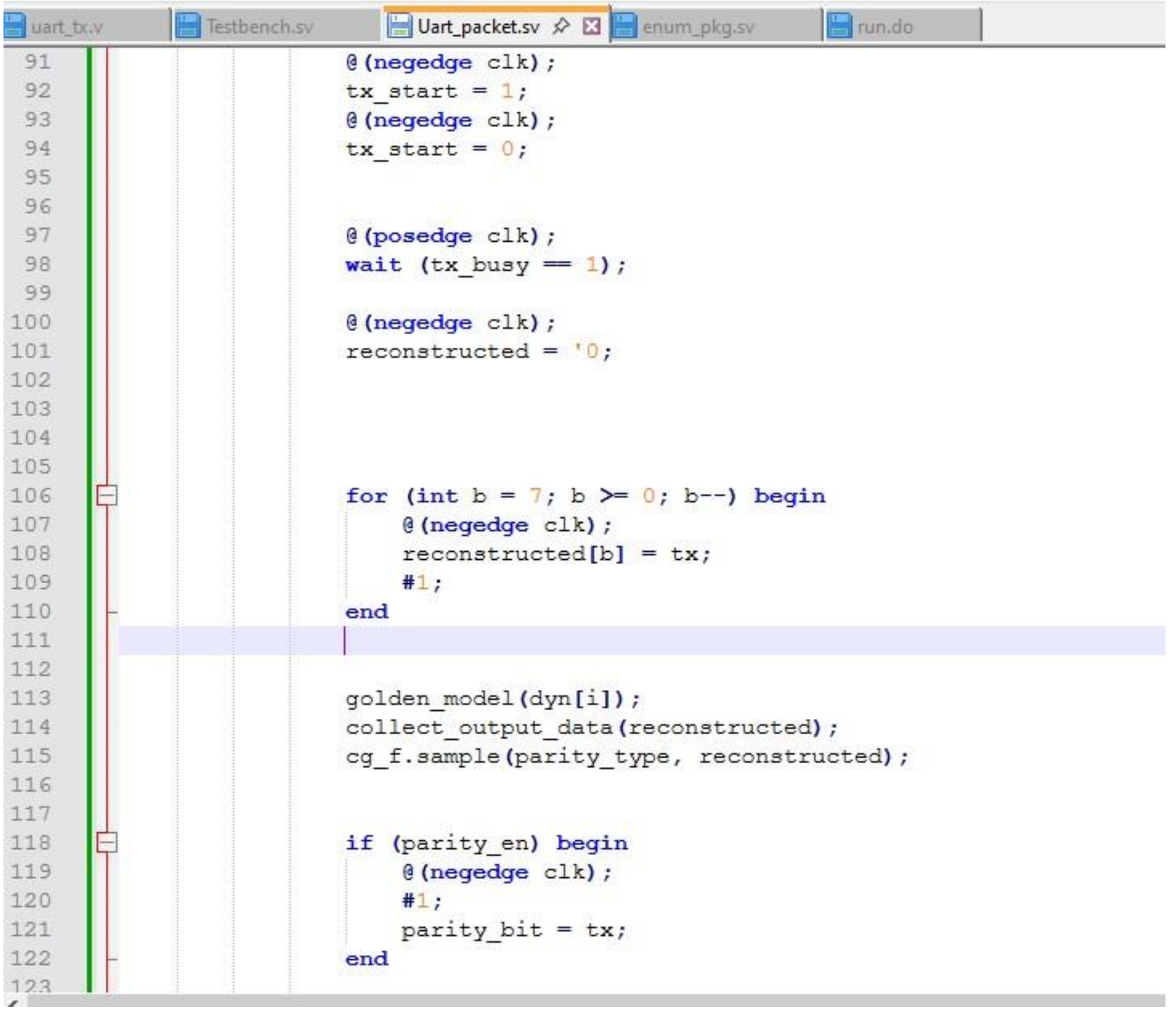
```

import enum_pkg::*;


    // Covarage I am tried to add  covergroup in the class but it give me error
        /*
            ** Error: Variables of embedded Covergroup type 'cg' cannot be created.
            ** Error: near "=": syntax error, unexpected '='

         */

    covergroup cg with function sample(parity_t p, logic [7:0] d);
        cp1: coverpoint p;
        cp2: coverpoint d {
            bins all_zeros = {8'h00};
            bins all_ones  = {8'hFF};
            bins others    = default;
        }
    endgroup: cg



    class Uart_packet;


        bit clk ;


        rand parity_t parity_type;


        // Dynamic array
            rand logic [7:0] dyn[];
```



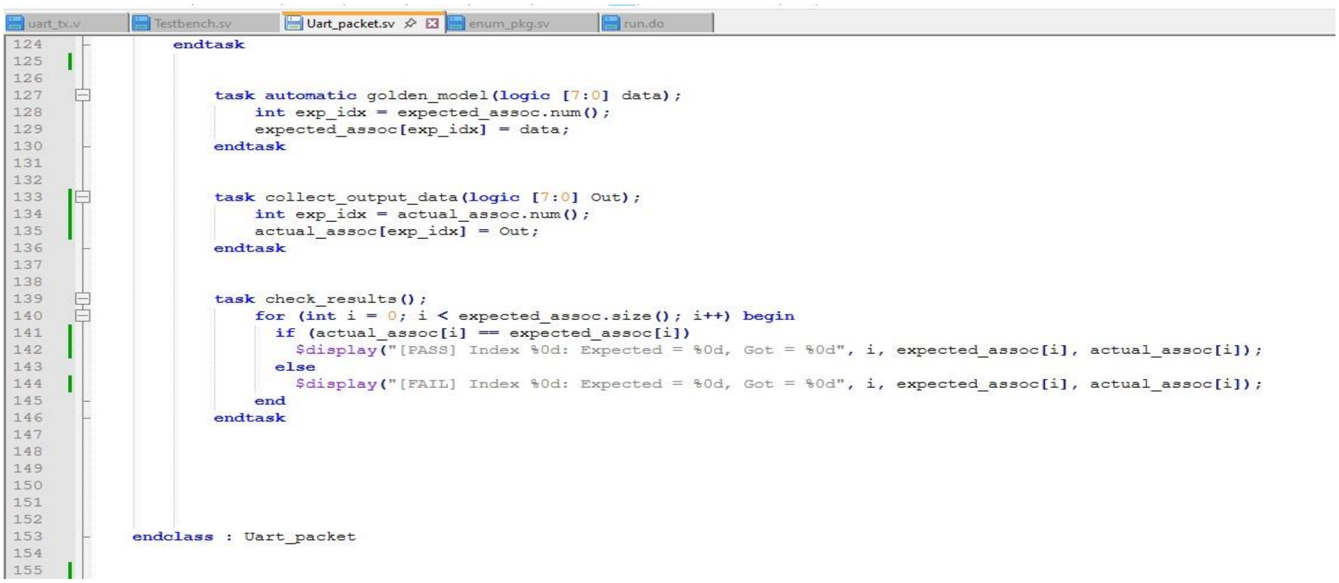
```

        // Dynamic array
            rand logic [7:0] dyn[];

        // Associative array for actual DUT results
            logic [7:0] actual_assoc[int];

        // Associative array for expected results
            logic [7:0] expected_assoc [int];


        ////////// constraint ////////////

            constraint dyn_size_c{
                dyn.size() inside {[5:20]};
            }

            constraint dyn_element_c{
                foreach(dyn[i])
                    dyn[i] dist {'h00 :/ 1 ,'hff :/ 1,['h00 :'hff] :/ 98};

            }

            constraint parity_type_C{
                parity_type dist {NO_PARITY := 2 ,ODD_PARITY := 4 , EVEN_PARITY := 4};
                parity_type inside {NO_PARITY, ODD_PARITY, EVEN_PARITY};
            }



        // Covarage

            cg cg_f;
```



```
// Covarage

    cg cg_f;

   function new();
      cg_f = new();
   endfunction


/////////// function And task ///////////////

    task automatic drive_stim(
    ref logic clk,
    ref logic tx_start,
    ref logic parity_en,
    ref logic even_parity,
    ref logic [7:0] data_in,
    ref logic tx,
    ref logic tx_busy
);
    logic [7:0] reconstructed;
    logic parity_bit;

    foreach (dyn[i]) begin

        parity_en   = (parity_type != NO_PARITY);
        even_parity = (parity_type == EVEN_PARITY);
        data_in     = dyn[i];


        @(negedge clk);
        tx_start = 1;
        @(negedge clk);
```

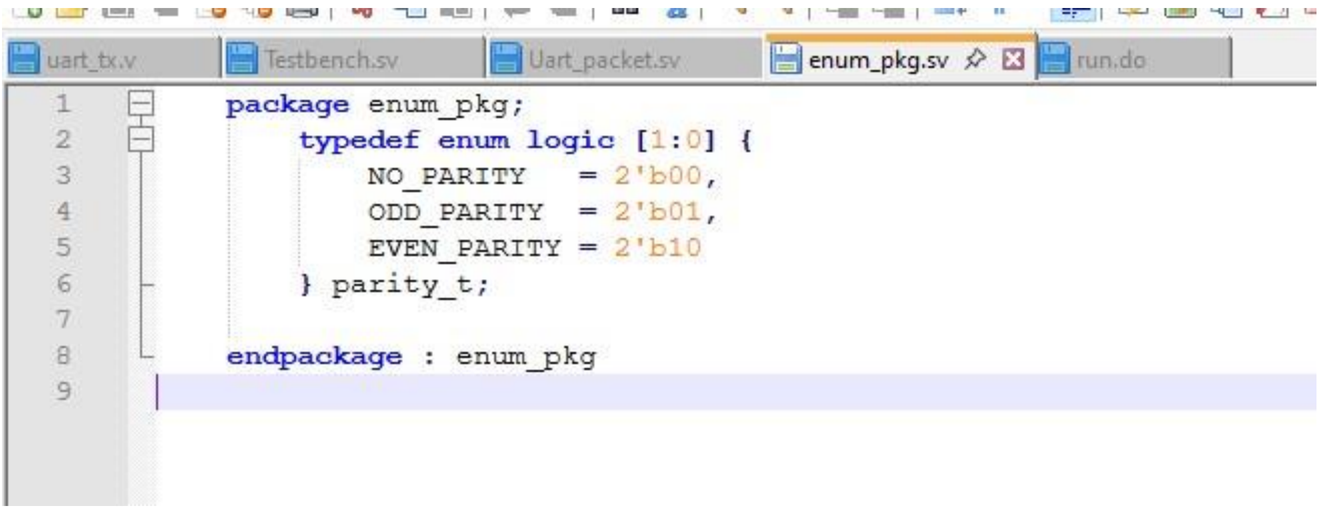


```
        @(negedge clk);
        tx_start = 1;
        @(negedge clk);
        tx_start = 0;


        @(posedge clk);
        wait (tx_busy == 1);

        @(negedge clk);
        reconstructed = '0;




        for (int b = 7; b >= 0; b--) begin
            @(negedge clk);
            reconstructed[b] = tx;
            #1;
        end


        golden_model(dyn[i]);
        collect_output_data(reconstructed);
        cg_f.sample(parity_type, reconstructed);


        if (parity_en) begin
            @(negedge clk);
            #1;
            parity_bit = tx;
        end
```



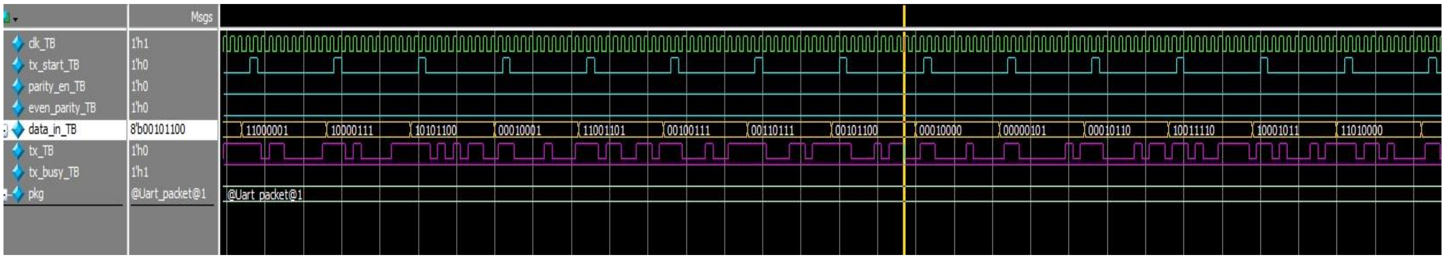
```
    endtask


        task automatic golden_model(logic [7:0] data);
            int exp_idx = expected_assoc.num();
            expected_assoc[exp_idx] = data;
        endtask


        task collect_output_data(logic [7:0] Out);
            int exp_idx = actual_assoc.num();
            actual_assoc[exp_idx] = Out;
        endtask


        task check_results();
            for (int i = 0; i < expected_assoc.size(); i++) begin
              if (actual_assoc[i] == expected_assoc[i])
                $display("[PASS] Index %0d: Expected = %0d, Got = %0d", i, expected_assoc[i], actual_assoc[i]);
              else
                $display("[FAIL] Index %0d: Expected = %0d, Got = %0d", i, expected_assoc[i], actual_assoc[i]);
            end
        endtask





endclass : Uart_packet
```

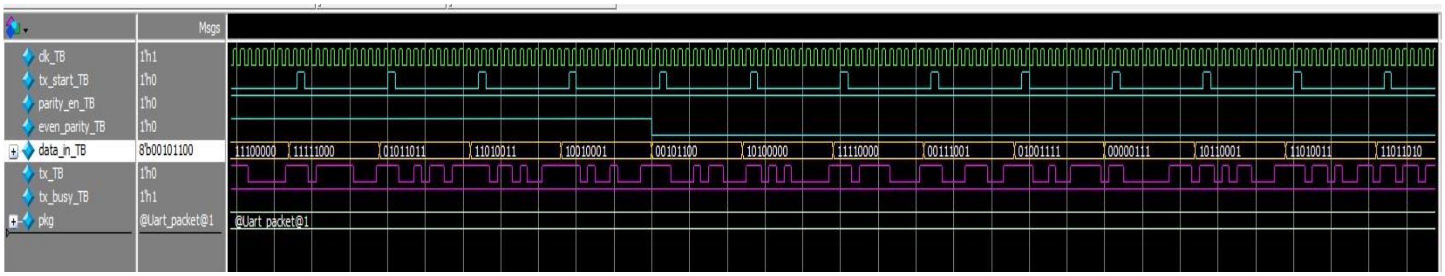
//enum_pkg



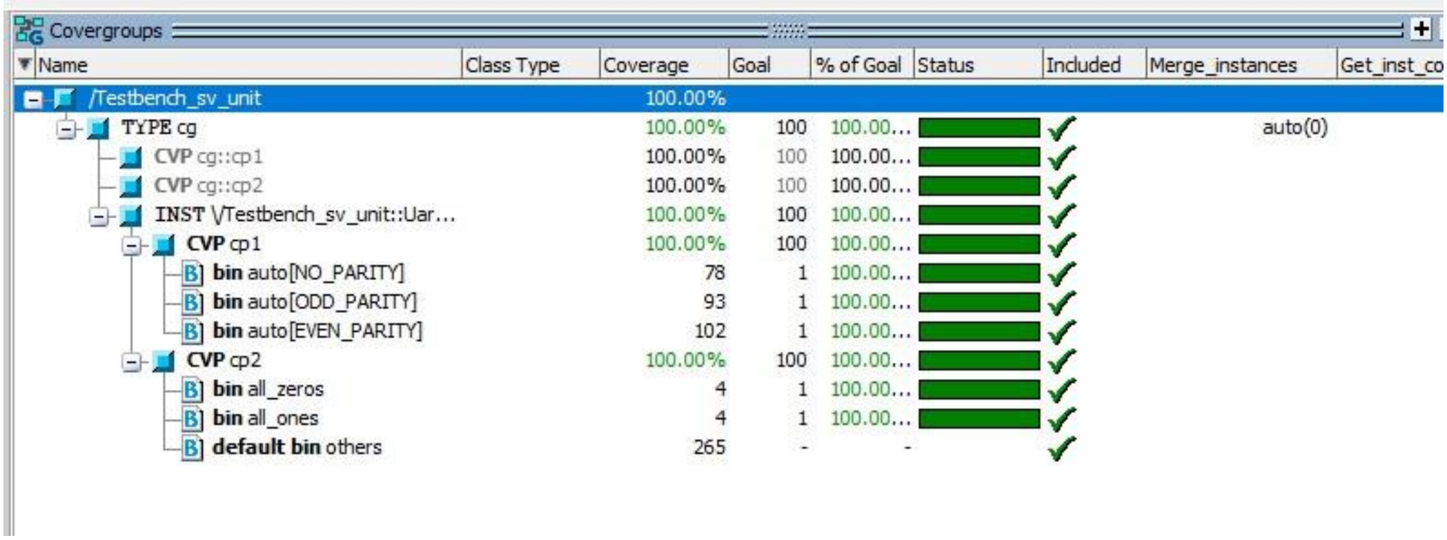
```
package enum_pkg;
    typedef enum logic [1:0] {
        NO_PARITY    = 2'b00,
        ODD_PARITY   = 2'b01,
        EVEN_PARITY  = 2'b10
    } parity_t;

endpackage : enum_pkg
```

2 – Snippets for the waveform shows the values for created variables



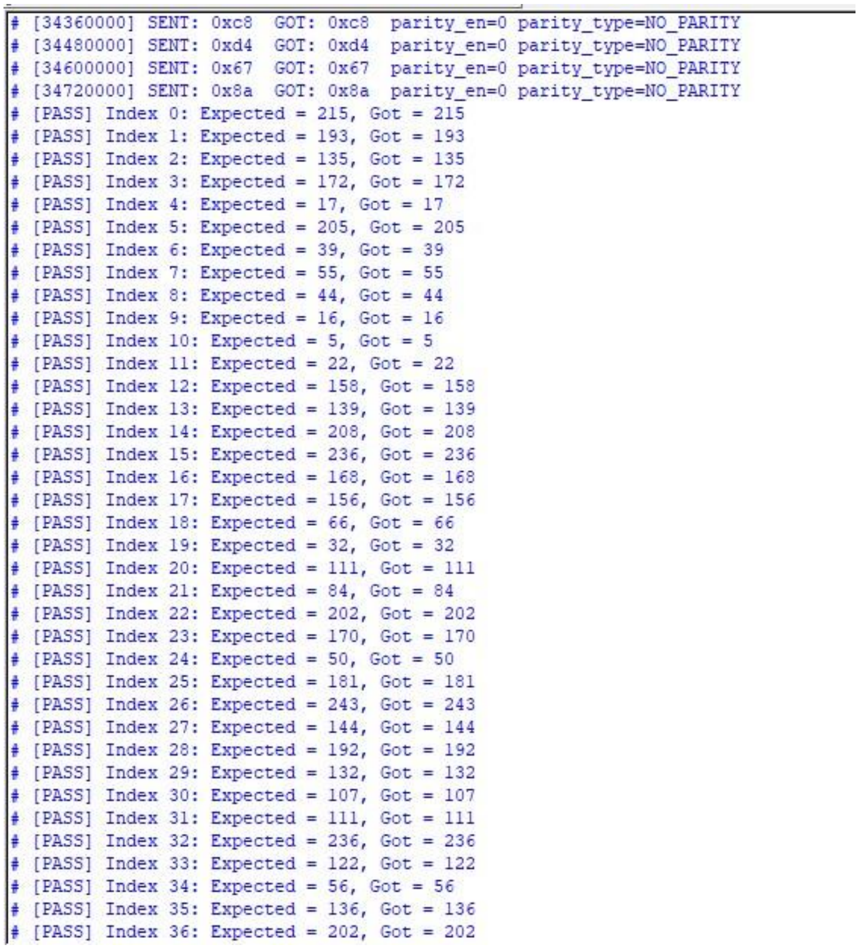


3-Snippets for Functional Coverage achieving



Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_co
/Testbench_sv_unit		100.00%						
TYPE cg		100.00%	100	100.00...	🟩	✓	auto(0)	
CVP cg::cp1		100.00%	100	100.00...	🟩	✓		
CVP cg::cp2		100.00%	100	100.00...	🟩	✓		
INST \/Testbench_sv_unit::Uar...		100.00%	100	100.00...	🟩	✓		
CVP cp1		100.00%	100	100.00...	🟩	✓		
bin auto[NO_PARITY]		78	1	100.00...	🟩	✓		
bin auto[ODD_PARITY]		93	1	100.00...	🟩	✓		
bin auto[EVEN_PARITY]		102	1	100.00...	🟩	✓		
CVP cp2		100.00%	100	100.00...	🟩	✓		
bin all_zeros		4	1	100.00...	🟩	✓		
bin all_ones		4	1	100.00...	🟩	✓		
default bin others		265	-	-		✓		

4- Snippets for the logs show the all printed values.



```
# [34360000] SENT: 0xc8  GOT: 0xc8  parity_en=0 parity_type=NO_PARITY
# [34480000] SENT: 0xd4  GOT: 0xd4  parity_en=0 parity_type=NO_PARITY
# [34600000] SENT: 0x67  GOT: 0x67  parity_en=0 parity_type=NO_PARITY
# [34720000] SENT: 0x8a  GOT: 0x8a  parity_en=0 parity_type=NO_PARITY
# [PASS] Index 0: Expected = 215, Got = 215
# [PASS] Index 1: Expected = 193, Got = 193
# [PASS] Index 2: Expected = 135, Got = 135
# [PASS] Index 3: Expected = 172, Got = 172
# [PASS] Index 4: Expected = 17, Got = 17
# [PASS] Index 5: Expected = 205, Got = 205
# [PASS] Index 6: Expected = 39, Got = 39
# [PASS] Index 7: Expected = 55, Got = 55
# [PASS] Index 8: Expected = 44, Got = 44
# [PASS] Index 9: Expected = 16, Got = 16
# [PASS] Index 10: Expected = 5, Got = 5
# [PASS] Index 11: Expected = 22, Got = 22
# [PASS] Index 12: Expected = 158, Got = 158
# [PASS] Index 13: Expected = 139, Got = 139
# [PASS] Index 14: Expected = 208, Got = 208
# [PASS] Index 15: Expected = 236, Got = 236
# [PASS] Index 16: Expected = 168, Got = 168
# [PASS] Index 17: Expected = 156, Got = 156
# [PASS] Index 18: Expected = 66, Got = 66
# [PASS] Index 19: Expected = 32, Got = 32
# [PASS] Index 20: Expected = 111, Got = 111
# [PASS] Index 21: Expected = 84, Got = 84
# [PASS] Index 22: Expected = 202, Got = 202
# [PASS] Index 23: Expected = 170, Got = 170
# [PASS] Index 24: Expected = 50, Got = 50
# [PASS] Index 25: Expected = 181, Got = 181
# [PASS] Index 26: Expected = 243, Got = 243
# [PASS] Index 27: Expected = 144, Got = 144
# [PASS] Index 28: Expected = 192, Got = 192
# [PASS] Index 29: Expected = 132, Got = 132
# [PASS] Index 30: Expected = 107, Got = 107
# [PASS] Index 31: Expected = 111, Got = 111
# [PASS] Index 32: Expected = 236, Got = 236
# [PASS] Index 33: Expected = 122, Got = 122
# [PASS] Index 34: Expected = 56, Got = 56
# [PASS] Index 35: Expected = 136, Got = 136
# [PASS] Index 36: Expected = 202, Got = 202
```