```
//////////////////////////////////////////////////////////////////////
//                                                                  //
//  File name : bringup_loopback/testcase.sv                        //
//  Author    : G. Andres Mancera                                   //
//  License   : GNU Lesser General Public License                   //
//  Course    : Advanced Verification with SystemVerilog OOP        //
//              Testbench - UCSC Silicon Valley Extension           //
//                                                                  //
//////////////////////////////////////////////////////////////////////

`include "../../testbench/packet.sv"
`include "../../testbench/driver.sv"
`include "../../testbench/monitor.sv"
`include "../../testbench/coverage.sv"
`include "../../testbench/scoreboard.sv"
`include "../../testbench/env.sv"

program testcase (  interface tcif_driver,
                    interface tcif_monitor  );

  // Since this is a bringup test, all the fields of the packet will be
  // heavily constrained.  The payload size is also contrained to match
  // that of the rudimentary testcase that comes along with the design.
  class bringup_packet extends packet;
    constraint C_payload_size
      {
        payload.size()  inside {[45:54]};
      }
    constraint C_bringup_packet
      {
        mac_dst_addr    == 48'hAABB_CCDD_EEFF;
        mac_src_addr    == 48'h1122_3344_5566;
        ether_type      dist { 16'h0800:=40, 16'h0806:=20, 16'h88DD:=40 };  // IPv4, ARP, IPv6
        foreach( payload[j] )
        {
          payload[j]    == j+1;
        }
        ipg             == 10;
      }
  endclass : bringup_packet

  env                 env0;
  int unsigned        num_packets;
  bringup_packet      testcase_packet;

  initial begin
    env0            = new(tcif_driver, tcif_monitor);
    testcase_packet = new();

    // Connect packet handle from driver to testcase_packet
    env0.drv.xge_mac_pkt = testcase_packet;
    num_packets = $urandom_range(40,60);
    tcif_driver.init_tb_signals();
    tcif_driver.make_loopback_connection();
    tcif_driver.wishbone_write_task(8'h00, 32'h1);
    tcif_driver.wait_ns(2000);
    env0.run(num_packets);
    tcif_driver.wait_ns(100000);
    tcif_driver.wishbone_read_task(8'h00);
    tcif_driver.wishbone_read_task(8'h08);
    tcif_driver.wishbone_read_task(8'h0C);
    tcif_driver.wishbone_read_task(8'h10);
    $finish;
  end

  final begin
    int unsigned    num_pkts;
    int unsigned    num_errors;
    num_pkts    =   packet::get_pktid();
    num_errors  =   env0.scbd.num_of_mismatches;
    $display("\nTESTCASE: ----------------- End Of Simulation -----------------");
    $display("TESTCASE: Number of packets sent       :  %0d", num_pkts);
    $display("TESTCASE: Number of mismatched packets :  %0d", num_errors);
    if ( num_errors==0 )
      $display("TESTCASE: ---------------------- PASSED -----------------------\n");
    else
      $display("TESTCASE: ---------------------- FAILED -----------------------\n");
  end

endprogram
```