```
`include "uvm_macros.svh"

module svid_packet_mon_wrapper(
                        svid_packet_mon_intf mon_intf,
                        input clk
                        );


   import uvm_svid_monitor_package::*;

   wire [7:0]                    tx_frame_0;
   wire [7:0]                    tx_frame_1;
   wire [7:0]                    tx_frame_2;
   wire                      trigger;

   svid_packet_intf i_svid_packet_intf(clk);

   assign tx_frame_0 = mon_intf.tb.tx_frame_0;
   assign tx_frame_1 = mon_intf.tb.tx_frame_1;
   assign tx_frame_2 = mon_intf.tb.tx_frame_2;
   assign trigger    = mon_intf.tb.trigger;


   assign i_svid_packet_intf.start_frame     = tx_frame_2[7:5];
   assign i_svid_packet_intf.addr            = tx_frame_2[4:1];
   assign i_svid_packet_intf.command         =
{tx_frame_2[0],tx_frame_1[7:4]};
   assign i_svid_packet_intf.payload         =
{tx_frame_1[3:0],tx_frame_0[7:4]};
   assign i_svid_packet_intf.parity          = tx_frame_0[3];
   assign i_svid_packet_intf.end_frame       = tx_frame_0[2:0];
   assign i_svid_packet_intf.trigger         = trigger;


   wire       computed_parity =    tx_frame_2[4] ^ tx_frame_2[3] ^
tx_frame_2[2] ^ tx_frame_2[1] ^ tx_frame_2[0] ^
                                   tx_frame_1[7] ^ tx_frame_1[6] ^
tx_frame_1[5] ^ tx_frame_1[4] ^
                                   tx_frame_1[3] ^ tx_frame_1[2] ^
tx_frame_1[1] ^ tx_frame_1[0] ^
                                   tx_frame_0[7] ^ tx_frame_0[6] ^
tx_frame_0[5] ^ tx_frame_0[4];



   //run test method for the monitor
class my_test extends uvm_test;
   `uvm_component_utils( my_test )

   //instantiate the packet monitor
   svid_packet_checking_env    svid_pc_env;
   virtual    svid_packet_intf svid_pc_intf;

   function new( string name, uvm_component parent );
```

```
      super.new( name, parent );
   endfunction: new

   function void build_phase(uvm_phase phase);
      super.build_phase( phase );
      i_svid_packet_checking_env =
svid_packet_checking_env:type_id::create( .name( "svid_environment"  ),
.parent( this ) );
   endfunction

   function void connect_phase(uvm_phase phase);
      super.connect_phase( phase );
      svid_pc_env.ingress_intf = svid_pc_intf;
   endfunction //


   function void run_phase(uvm_phase phase);
      //What needs to go in here ?????
      //We are not driving just monitoring ???
   endfunction

endclass // my_test


   ////////////////////////////////////////////////////////////////////
   //
   ////////////////////////////////////////////////////////////////////
   initial begin
      //??????????????????????????????
      //What goes in here to start up the packet checking environment
????
      //How does the packet checking environment connect to
i_svid_packet_intf ??
   end

endmodule // svid_packet_mon_wrapper
```