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

class monitor;

  virtual xge_mac_interface     mon_vi;
  packet                        xge_mac_pkt;
  mailbox                       mon2sb;

  // ======== Contructor ========
  function new( input virtual xge_mac_interface vif,
                input mailbox mon2sb                );
    $display("MONITOR :: inside new() function");
    this.mon_vi = vif;
    this.mon2sb = mon2sb;
    xge_mac_pkt = new();
  endfunction : new


  // ======== Class methods ========
  task collect_packet();
    packet      mon_pkt;
    bit         pkt_in_progress;
    bit         err_in_packet;
    bit [7:0]   rx_data_q[$];
    int         idx;
    bit         packet_captured;
    mon_pkt         = new();
    pkt_in_progress = 0;
    packet_captured = 0;
    err_in_packet   = 0;

    forever begin
      @(mon_vi.cb)
      begin
        if ( mon_vi.cb.pkt_rx_avail ) begin
          mon_vi.cb.pkt_rx_ren <= 1'b1;
        end
        if ( mon_vi.cb.pkt_rx_val ) begin
          if ( mon_vi.cb.pkt_rx_sop && !mon_vi.cb.pkt_rx_eop && pkt_in_progress==0 ) begin
            // -------------------------------- SOP cycle ----------------
            pkt_in_progress = 1;
            mon_vi.cb.pkt_rx_ren <= 1'b1;
            mon_pkt.sop_mark            = mon_vi.cb.pkt_rx_sop;
            mon_pkt.mac_dst_addr        = mon_vi.cb.pkt_rx_data[63:16];
            mon_pkt.mac_src_addr[47:32] = mon_vi.cb.pkt_rx_data[15:0];
            mon_pkt.mac_src_addr[31:0]  = 32'h0;
            mon_pkt.ether_type          = 16'h0;
            mon_pkt.payload = new[0];
            while ( rx_data_q.size()>0 ) begin
              rx_data_q.pop_front();
            end
          end   // ---------------------------- SOP cycle ----------------
          if ( !mon_vi.cb.pkt_rx_sop && !mon_vi.cb.pkt_rx_eop && pkt_in_progress==1 ) begin
            // -------------------------------- MOP cycle ----------------
            pkt_in_progress = 1;
            mon_vi.cb.pkt_rx_ren <= 1'b1;
            if ( rx_data_q.size()==0 ) begin
              mon_pkt.mac_src_addr[31:0]  = mon_vi.cb.pkt_rx_data[63:32];
              mon_pkt.ether_type          = mon_vi.cb.pkt_rx_data[31:16];
              rx_data_q.push_back(mon_vi.cb.pkt_rx_data[15:8]);
              rx_data_q.push_back(mon_vi.cb.pkt_rx_data[7:0]);
            end
            else begin
              for ( int i=0; i<8; i++ ) begin
                rx_data_q.push_back( (mon_vi.cb.pkt_rx_data >> (64-8*(i+1))) & 8'hFF );
              end
            end
          end   // ---------------------------- MOP cycle ----------------
          if ( mon_vi.cb.pkt_rx_eop && pkt_in_progress==1 ) begin
            // -------------------------------- EOP cycle ----------------
            mon_pkt.eop_mark= mon_vi.cb.pkt_rx_eop;
            pkt_in_progress = 0;
            err_in_packet   = mon_vi.cb.pkt_rx_err;
            mon_vi.cb.pkt_rx_ren <= 1'b0;
            if ( mon_vi.cb.pkt_rx_mod==0 ) begin
              for ( int i=0; i<8; i++ ) begin
                rx_data_q.push_back( (mon_vi.cb.pkt_rx_data >> (64-8*(i+1))) & 8'hFF );
              end
            end
            else begin
              for ( int i=0; i<mon_vi.cb.pkt_rx_mod; i++ ) begin
                rx_data_q.push_back( (mon_vi.cb.pkt_rx_data >> (64-8*(i+1))) & 8'hFF );
              end
            end
            //$display("MONITOR DEBUG :: mon_pkt.mac_dst_addr = %0x", mon_pkt.mac_dst_addr);
            //$display("MONITOR DEBUG :: mon_pkt.mac_src_addr = %0x", mon_pkt.mac_src_addr);
            //$display("MONITOR DEBUG :: mon_pkt.ether_type   = %0x", mon_pkt.ether_type);
            //$display("MONITOR DEBUG :: rx_data_q size =%0d", rx_data_q.size());
            mon_pkt.payload = new[rx_data_q.size()];
            idx = 0;
            while ( rx_data_q.size()>0 ) begin
              mon_pkt.payload[idx]  = rx_data_q.pop_front();
              idx++;
            end
            packet_captured  = 1;
            // -------------------------------- EOP cycle ----------------
          end
          if ( mon_vi.cb.pkt_rx_sop && mon_vi.cb.pkt_rx_eop && pkt_in_progress==0) begin
            // -------------------------------- SOP/EOP cycle ------------
            err_in_packet   = mon_vi.cb.pkt_rx_err;
            mon_vi.cb.pkt_rx_ren <= 1'b1;
            mon_pkt.sop_mark            = mon_vi.cb.pkt_rx_sop;
            mon_pkt.eop_mark            = mon_vi.cb.pkt_rx_eop;
            mon_pkt.mac_dst_addr        = mon_vi.cb.pkt_rx_data[63:16];
            mon_pkt.mac_src_addr[47:32] = mon_vi.cb.pkt_rx_data[15:0];
            mon_pkt.mac_src_addr[31:0]  = 32'h0;
            mon_pkt.ether_type          = 16'h0;
            mon_pkt.payload = new[0];
            while ( rx_data_q.size()>0 ) begin
              rx_data_q.pop_front();
            end
            packet_captured = 1;
            // -------------------------------- SOP/EOP cycle ------------
          end
          if ( packet_captured ) begin
            mon_pkt.print("FROM MONITOR");
            // Put the collected packet into the mailbox if the packet has no errors
            if ( !err_in_packet && mon_pkt.sop_mark && mon_pkt.eop_mark ) begin
              mon2sb.put(mon_pkt);
            end
            else begin
              $display("MONITOR :: t=%2t, ERROR PACKET, WILL NOT SEND IT TO SCOREBOARD", $time);
            end
            packet_captured = 0;
          end
        end
      end
    end
  endtask : collect_packet

endclass
```