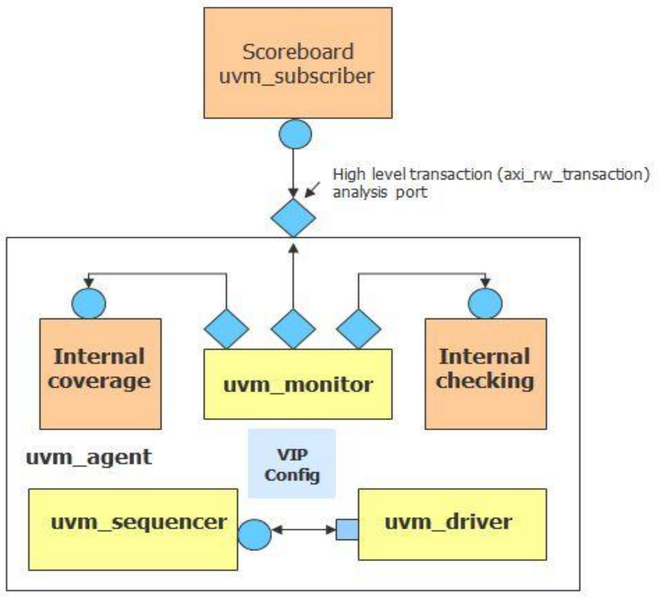
# Monitors, Monitors Everywhere …



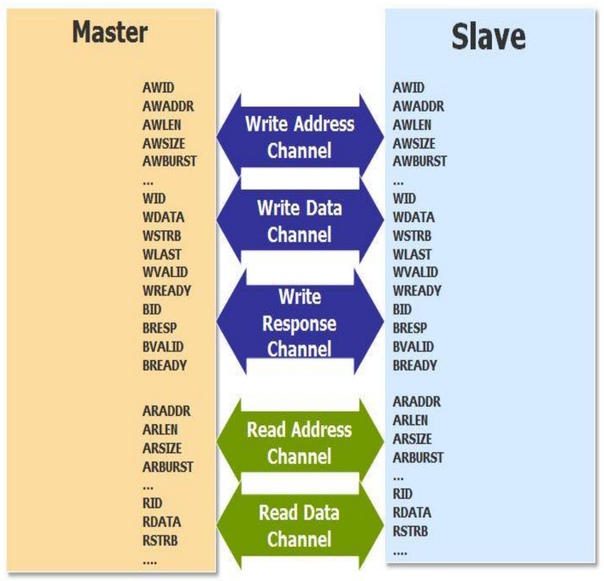

## Transaction Level Monitor

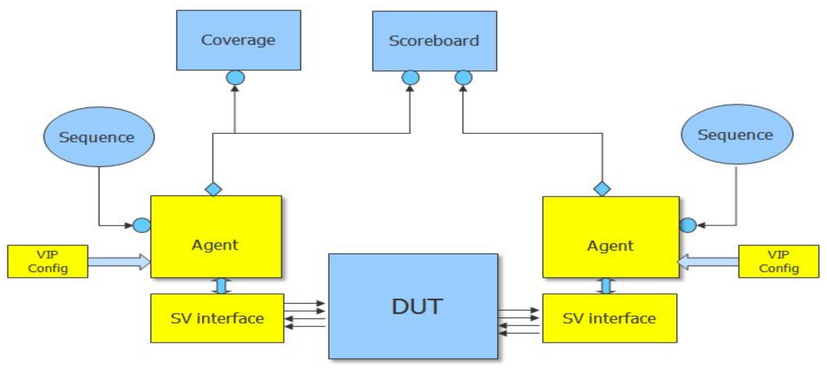
```
class axi_tl_monitor extends uvm_monitor;
  // High Level AXI Monitor
  // Records Transaction Writes and Reads Only
  // Transaction Level Analysis Port
  uvm_analysis_port #(axi_tlp) axi_ap;
  ...
  task run_phase(uvm_phase phase)
    axi_tlp tlp;
    forever @(clk)
    // Detect the Write or Read Transaction Level
    // Packet from the bus pins
    // Send the transaction to the analysis port
    // of the Agent
     axi_ap.write(tlp);
  endtask
endclass
```

## Transaction Level Listener

```
// Transaction Level Listener
// Implements the write method called from monitor
class axi_item_listener extends
    uvm_subscriber #(uvm_sequence_item_base );
  // Method captures packet from Monitor analysis port and
  // performs user defined task - print it out
  function write(uvm_sequence_item_base t );
    `uvm_info("AXI", t.convert2string())
  endfunction
endclass
```

## AXI Master/Slave

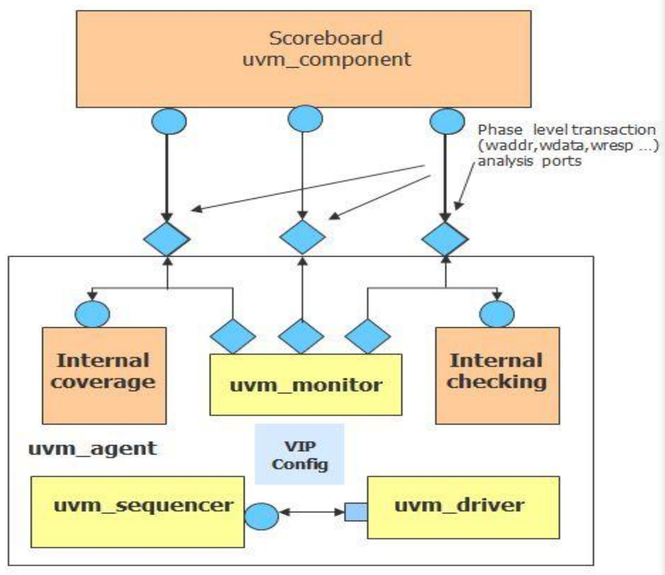



## Transaction Level Scoreboard

```
// Transaction Level AXI Scoreboard
// Scoreboards only records Reads and Writes after completion
// and receipt of response
// Simulataneous reads and writes to same address results
// in unknown response

class axi_scoreboard extends uvm_subscriber#(axi_tlp);
  function write(axi_tlp t);
   if (t.read_or_write == AXI_TRANS_WRITE) begin
   // Record and AXI write transaction and update scoreboard
     case (t.burst)
       AXI_FIXED: process_write_fixed(t);
       AXI_INCR:  process_write_incr(t);
       AXI_WRAP:  process_write_wrap(t);
       ...
    end else
      // Record a read transaction and check if
      // read data == write data
      process_read(t);
  endfunction
  ...
endclass
```




## Phase Level Monitor

```
// Phase level AXI Monitor
// Records each of the AXI channels
class axi_monitor extends uvm_monitor;
  `uvm_component_utils(axi_monitor)
  // Analysis Ports for each of the AXI Channels
  uvm_analysis_port #(axi_write_addr_tlp) waddr_ap;
  uvm_analysis_port #(axi_write_data_tlp) wdata_ap;
  uvm_analysis_port #(axi_write_resp_tlp) wresp_ap;
  uvm_analysis_port #(axi_read_addr_tlp)  raddr_ap;
  uvm_analysis_port #(axi_read_data_tlp)  rdata_ap;
  ...
  task run_phase(uvm_phase phase)
    // Phase level packets for each of the
    //   AXI channels
    axi_write_addr_tlp waddr_tlp;
    axi_write_data_tlp wdata_tlp;
    axi_write_resp_tlp wresp_tlp;
    axi_read_addr_tlp  raddr_tlp;
    axi_read_data_tlp  rdata_tlp;
    ...
      forever @(clk) ...
      // Detect the any of the 5 phase transactions
      // from the bus pins and transmit thru the
      // appropriate analysis port
    ...
    waddr_ap.write(waddr_tlp);
    wdata_ap.write(wdata_tlp);
    wresp_ap.write(wresp_tlp);
    raddr_ap.write(raddr_tlp);
    rdata_ap.write(rdata_tlp);
  endtask
endclass
```

## Phase Level Scoreboard

```
// Phase Level AXI Scoreboard
// Triggers a unique write method each time an AXI
// Phase (one for each channel) is detected
// Can handle simulataneous reads and writes to
// same address

`uvm_analysis_imp_decl(_write_addr_ph)
`uvm_analysis_imp_decl(_write_data_ph)

class axi_phase_scoreboard extends uvm_component;
 //Analysis exports corresponding to each AXI Channel
  uvm_analysis_imp_write_addr_ph #(uvm_sequence_item_base,
    axi_phase_sb) write_addr_export;
  uvm_analysis_imp_write_data_ph #(uvm_sequence_item_base,
    axi_phase_sb) write_data_export;

  // Write methods corresponding to each channel
  // Traps address and data phases and records each beat
  // Can keep track of present and new data in memory to
  // handle simultaneous reads and writes to an address
  function void write_write_addr_ph(
    uvm_sequence_item_base t ); ... endfunction

  function void write_write_data_ph(
    uvm_sequence_item_base t);  ... endfunction
  ...
endclass
```


Rich Edelman
rich_edelman@mentor.com
Raghu Ardeishar
raghu_ardeishar@mentor.com
Mentor Graphics


DVCon 2013 Design & Verification Conference & Exhibition

Sponsored By:

accellera SYSTEMS INITIATIVE