# Don’t delay catching bugs : Using UVM based architecture to model external board delays

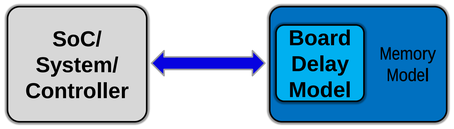
**Amit Paunikar, Saurabh Arya, Vikas Makhija & Shaily Khare**
**Synopsys India Pvt. Ltd.**

## Introduction

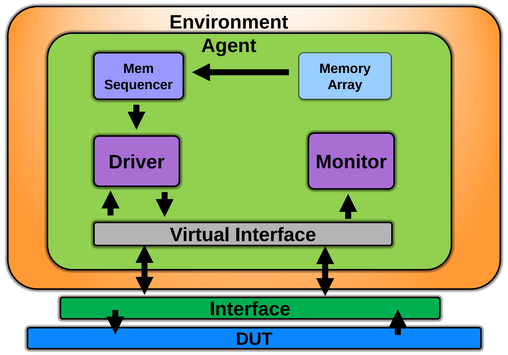
A typical SoC has different electrical delays. Their cumulative electrical delay can result in a liner shift within each system signal, Which can be combined to model as **Board Delays**.
This presentation focusses on

- Modeling external board delays using UVM (Universal Verification Methodology) for single and multi-channel environment.
- Multiple approaches :- fixed board delays, configurable board delays & randomized external board delays modelling.
- Demonstration of external board delay modeling using LPDDR (Low Power DRAM) and HBM (High Bandwidth Memory)

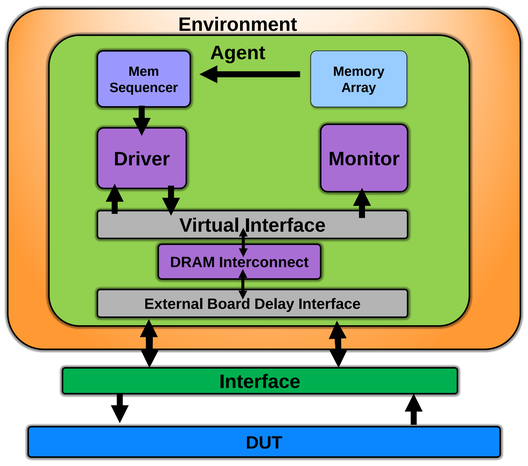
## Board Delay Model Encapsulated within Memory System



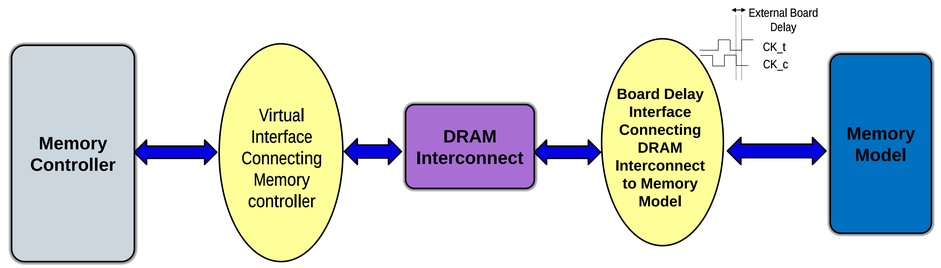

## UVM Architecture with Memory VIP




## Board delay architecture with UVM approach




## External board delay interface & Interconnect class




## Interconnect class code

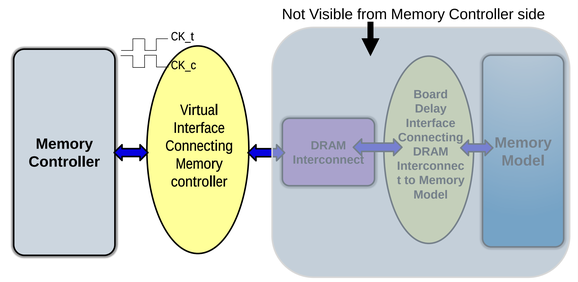
Introduced two additional components in VIP :

- External Board Delay Interface :
  - Pin compatible interface with original memory interface
  - Instantiated inside original memory interface
- Intermediate Interconnect Class (DRAM Interconnect)
  - Connects board delay interface to original interface
  - Provides a mechanism to add desired delays on each pin
  - It’s a UVM component encapsulated inside agent, hence delays can be changed during runtime as well
- These delays are enabled according to the type of signal that could be categorized in three groups- Input, Output and Inout signals.
- The delays are mentioned as <signal_name>_fly_by_delay_ps and are part of timing configuration class.
- Write and read data paths have separate delays.

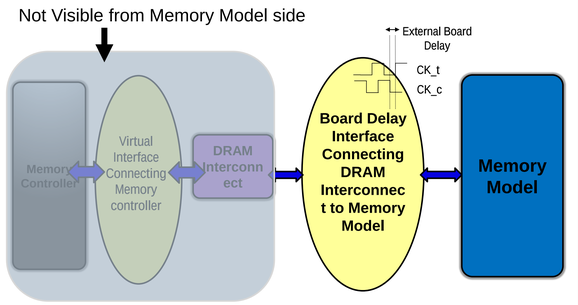
Code Snippet :

```
class hbm_dram_interconnect;
 task hbm_dram_interconnect::run();
  hbm_dram_interconnect_ext_board_delay_connect();
 endtask

task
hbm_dram_interconnect::hbm_dram_interconnect_ext_board_delay_connect();
  begin
   //*** Connecting dq , considering DQ_WIDTH = 128 ***//
   for(int i=0; i<128; i++) begin
     automatic int j = i;
     fork
      forever @(cfg.hbm_if.ext_board_delay_if.dq[j])
       //Model to Controller
       cfg.hbm_if.dq[j] <= #(cfg.timing_cfg.tdq_rd_fly_by_delay_ps[j]*1ps)
(cfg.hbm_if.ext_board_delay_if.dq_en[j]===1) ?
cfg.hbm_if.ext_board_delay_if.dq[j] : 'z;

      //*** Delaying enable signal dq_en_d ***//
      forever @(cfg.hbm_if.ext_board_delay_if.dq_en[j])
       //Model to Controller
       cfg.hbm_if.ext_board_delay_if.dq_en_d[j] <=
#(cfg.timing_cfg.tdq_rd_fly_by_delay_ps[j]*1ps)
(cfg.hbm_if.ext_board_delay_if.dq_en[j]===1) ? '1 : '0;

      forever @(cfg.hbm_if.dq[j])
       //Controller to Model
       cfg.hbm_if.ext_board_delay_if.dq[j] <=
#(cfg.timing_cfg.tdq_wr_fly_by_delay_ps[j]*1ps)
(cfg.hbm_if.ext_board_delay_if.dq_en_d[j]===0) ? cfg.hbm_if.dq[j] : 'z;
     join_none
   end
endtask
endclass
```

## Board Delay Interface from Memory Controller Perspective

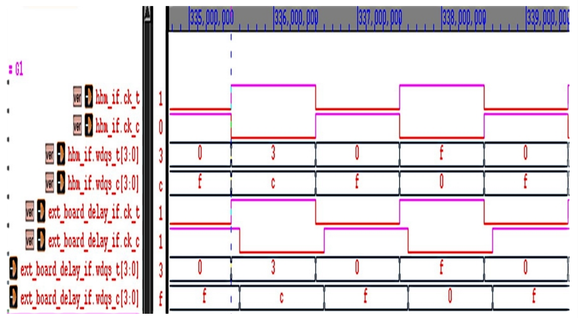



## Board Delay Interface from Memory Model Perspective



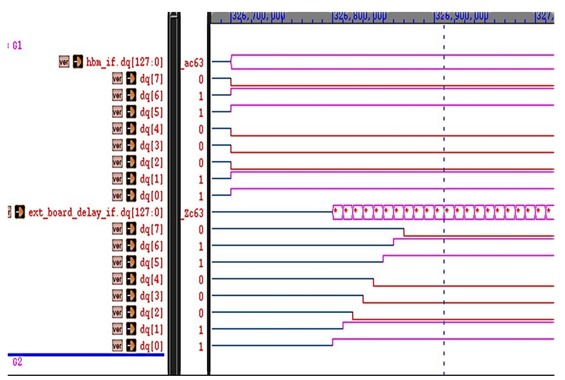

## Experiments & Results

- Differential signals skew: provided a skew of 0.25UI (Unit Intervals) between differential signals, ck_t/ck_c, wdqs_t/wdsq_c and rdqs_t/rdqs_c.




- Per transaction delays: Provided different delays during different transactions by reconfiguring these delays in every transaction.


- DQ skew: We monitored the data sampling on both reading and writing by applying data skew between individual DQ pins .




## Summary

- Demonstrated UVM based model to generate random board delays on each pin of a memory oriented interface.
- These delays are independent and entirely user configurable.
- These methods could be extended to multi-rank and multi-channel environments also.
- We have examined this model on a parallel memory protocol, but this approach can be extended to other parallel as well as serial protocols as future work.
- The board delay model can be modelled for multiple VIPs such as DDR, LPDDR 2/3/4 , HBM , USB , PCIE etc.