Performance Validation Strategy for Collaborative SoC developments where two worlds meets in GDS only

Chirag Kedia, Lead Engineer, Sr, QUALCOMM India Private Limited
Mukesh Ameria, Engineer, Staff/Manager, QUALCOMM India Private Limited
Rahul Gupta, Director, Engineering, QUALCOMM India Private Limited

accellera SYSTEMS INITIATIVE™
2021 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Problem Statement/Introduction

Performance is one of the most important aspect for qualification of correct architecture and implementation. Growing market and business across Companies are required to collaborate and deliver the HardMacro as collateral to the Partner. Following Challenges are faced in this scenario:

- ***Protected HardMacro***
- ***Bandwidth And Latency***
- ***Memory at other side of HardMacro***
- ***SoC frequency Varies***

This paper proposes an implemented and proven solution to this problem with following:

- BFM Model of HardMacro plugged in SoC TB.
- Own HM Structural latency/bandwidth calculations, DDR model mimicked
- Real traffic from performance team taken in agreed format and tool developed to create test cases and TestBench Infrastructure.

***Abbrevations Used**: HM -> HardMacro, NoC -> Network on Chip, BFM -> Bus Functional Model, GDS -> Graphical Data System*

Proposed Methodology/Advantages

HM BFM model integrated at rest of the SoC environment along with their traffic. Ensure peak traffic use cases, they are able to meet bandwidth requirements. For this solution following work is done:

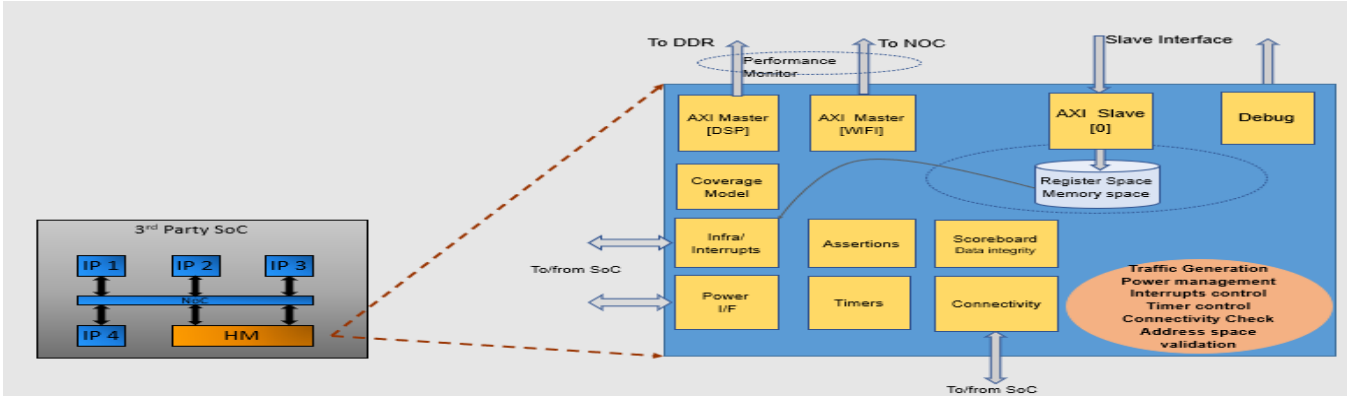
- BFM Environment developed to mimic HM - All AXI Master IF, Side band signals, Power IF and AHB Slave and Interrupts.
- HM BFM is a unified development and supports almost all possible bus interface like: AHB, AXI, CHI etc. hence one solution.
- Traffic generator tool takes ‘xls’ as input provided by performance team to generate bus agnostic UVM test sequence.
- HM BFM replaces HM master and internal HM NoC
- Within HM RTL replace the IP with the Unified transactor to mimic the actual traffic to analyse the bandwidth and check round trip latency within HM Boundary.

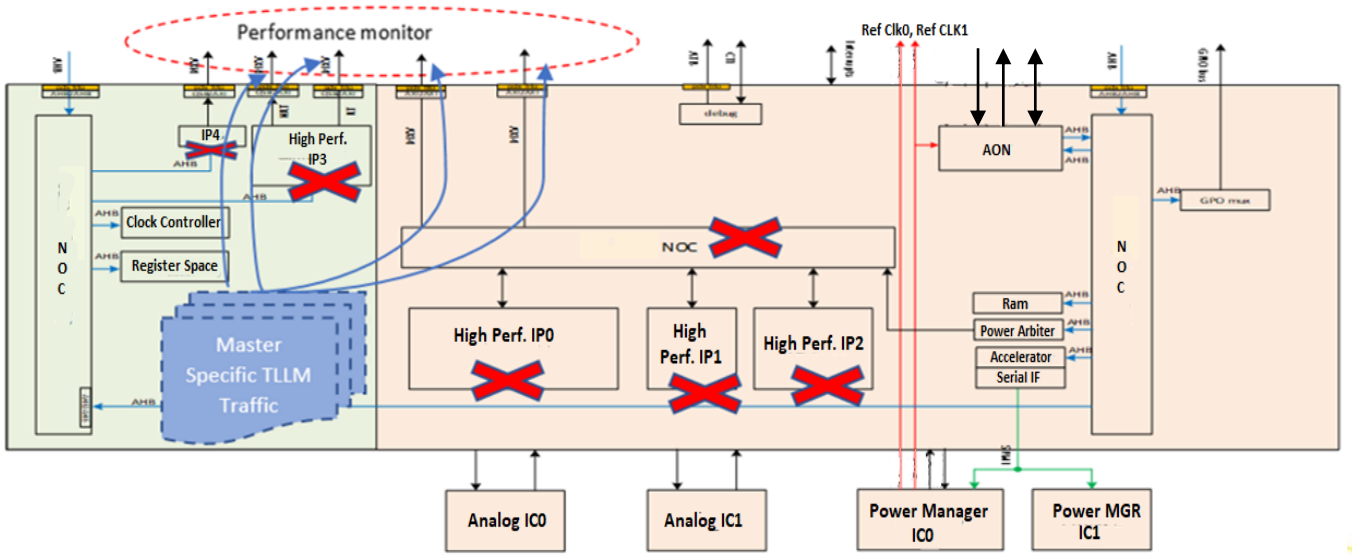
Implementation Details/Diagram

1. Functional and Interface compliant BFM (Replacing HM)

2. Performance Traffic mimicking from HM to SoC

3. Replace specific master within HM with Transactor





Implementation Details/Flow Chart

1. Performance traffic pattern per master from performance Team (in XLS Format) as shown in Figure 3

2. Traffic generator Tool to generate Traffic pattern in Fig. 4(Bus Specific transaction sequence in SV) based on target SoC interface bus type and performance traffic:
   Perl Command: ***./BFM_SEQ_GEN.pl traffic_pattern.xls –o traffic_seq.sv***

3. Along with above traffic from HM side, SoC side traffic is also get integrated and run in SoC testbench. This setup helps to push use-case traffic in SoC without actual RTL from HM side.

	priorit			200 MHz				
aooo=1	areqpri=5	7	16	ana_sysnoc_qxm	awrite=0	tid=0	mid=0x1	adddress=0x18300000
aooo=1	areqpri=5	7	18	ana_sysnoc_qxm	awrite=0	tid=2	mid=0x1	adddress=0x18300080
aooo=1	areqpri=5	7	20	ana_sysnoc_qxm	awrite=0	tid=3	mid=0x1	adddress=0x18300100
aooo=1	areqpri=5	7	22	ana_sysnoc_qxm	awrite=0	tid=4	mid=0x1	adddress=0x18300180
aooo=1	areqpri=5	7	24	ana_sysnoc_qxm	awrite=0	tid=5	mid=0x1	adddress=0x18300200
aooo=1	areqpri=5	7	26	ana_sysnoc_qxm	awrite=0	tid=6	mid=0x1	adddress=0x18300280
aooo=1	areqpri=5	7	28	ana_sysnoc_qxm	awrite=0	tid=7	mid=0x1	adddress=0x18300300
aooo=1	areqpri=5	7	30	ana_sysnoc_qxm	awrite=0	tid=8	mid=0x1	adddress=0x18300380
aooo=1	areqpri=5	7	32	ana_sysnoc_qxm	awrite=0	tid=9	mid=0x1	adddress=0x18300400
aooo=1	areqpri=5	7	34	ana_sysnoc_qxm	awrite=0	tid=10	mid=0x1	adddress=0x18300480
aooo=1	areqpri=5	7	36	ana_sysnoc_qxm	awrite=0	tid=11	mid=0x1	adddress=0x18300500
aooo=1	areqpri=5	7	38	ana_sysnoc_qxm	awrite=0	tid=12	mid=0x1	adddress=0x18300580
aooo=1	areqpri=5	7	40	ana_sysnoc_qxm	awrite=0	tid=13	mid=0x1	adddress=0x18300600

Figure 3

```
task drive_xact(longint addr ='QC_SOC_SLAVE1_START_ADDR, svt_axi_transaction::xact_type_enum xact_type,int b_length,svt_axi_transaction::burst_size_enum b_size,bit ['SVT_AXI_MAX_ID_WIDTH - 1:0] id_val =
'0,int qos = 0,int wait_clks);
begin
  qc_bfm_axi_mstr1_aggre_traffic_perf_sequence m1_aggre_noc_traffic_seq;
  repeat(wait_clks) @(posedge `BFM_ENV_HEIR.bfm_vif.axi_if.master_if[1].aclk);
  m1_aggre_noc_traffic_seq = qc_bfm_axi_mstr1_aggre_traffic_perf_sequence::type_id::create("m1_aggre_noc_traffic_seq",this);
  m1_aggre_noc_traffic_seq.set_xact_param(addr,xact_type,b_length,b_size,id_val,qos,wait_clks);
  m1_aggre_noc_traffic_seq.randomize();
  m1_aggre_noc_traffic_seq.start(`BFM_ENV_HEIR.axi_system_env.master[1].sequencer);
end
endtask
    drive_xact(32'hd0300000,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,);
    drive_xact(32'hd1d5c000,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,2,17);
    drive_xact(32'hd0300080,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,1);
    drive_xact(32'hd0300100,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300180,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300200,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300280,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300300,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300380,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300400,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300480,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300500,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
    drive_xact(32'hd0300580,svt_axi_transaction::READ,8,svt_axi_transaction::BURST_SIZE_128BIT,16'h1,5,2);
```

Figure 4

Results

With the mentioned approach we were able to find System Level issues such as:

1. Mismatch in the System bus parameters which were causing Performance degradation
2. Memory Bandwidth mismatch at different recommended frequencies.
3. Dynamic QoS not supported which in turn was not giving expected bandwidth and latency.

Planned to adopt this approach for derivatives of this project and similar kind of collaborative projects.

Conclusion

- **ROI of above-mentioned solution:**

✓ **Quality:** Helped uncover bugs in HM+SOC flow execution which could come as surprise in silicon/Emulation/Validation.

✓ **Productivity:** Entire verification process is left shifted with HM model delivery. Assertions & coverage model developed for HM model can be reused in RTL TB which helped in verification closure signoff. It also helped to develop a measure to assess HM RTL, model and spec are all aligned.

✓ **Scalability:** The model can be ported with required feature updates for derivative Projects as approach of development is unified.