2019 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

SoC Verification Enablement Using HM Model

Vineet Tanwar, Qualcomm India Private Limited, Noida, India (*vtanwar@qti.qualcomm.com*)
Chirag Kedia, Qualcomm India Private Limited, Noida, India (*ckedia@qti.qualcomm.com*)
Rahul Gupta, Qualcomm India Private Limited, Noida, India (*rahgup@qti.qualcomm.com*)

***Abstract*—Thorough pre-silicon verification of a complete system-on-chip (SoC), in the absence of RTL design of one or more sub-blocks/hard macros (HM), is a major challenge. One of the probable reasons of unavailability of HM RTL design could be unaligned HM delivery schedule and SoC schedule. Another reason could be the confidentiality of the HM RTL design that cannot be shared with the SoC verification team. This paper showcases an approach to overcome above challenge by developing and plugging an equivalent UVM based model of HM in SoC. The approach was successfully deployed in one of the projects of the organization which smoothened out the process of SoC verification in the absence of HM RTL design. Critical Specification bugs, Integration bugs and protocol issues were identified using the HM UVM Model. Also, there were interesting gaps related to performance and gaps in configurations of NOC at the SOC level were identified.**

Keywords—SoC verification; HM model; SoC testplan

I. INTRODUCTION

Today, the chipmakers in the semiconductor world release a new chip within a very small-time span of a few months. A very strict timeline is followed for completing the project milestones and for staying ahead of the competition it is made sure that the chip is out in the market as per the scheduled plan without compromising the quality of the product. Before the chip sees the silicon, it is very important for the System-on-Chip (SoC) to undergo a thorough pre-silicon verification so that any bugs in design can be caught and rectified earlier in the initial stages of chip development.

But there can be certain situations where the Register-Transfer Level (RTL) design of a Hard Macro (HM) is under development and its delivery schedule is not aligned with the SoC schedule. The HM could be as complex as it could include Wireless-Fidelity (WIFI) IP, Digital Signal Processors (DSP) processor, Network-on-Chip (NOC), Power controller, Debug Logic and SRAM/ROM. In the absence of the HM, the verification of the complete SoC cannot be halted as it may delay the final tape-out of the chip which eventually increases the Time-To-Market (TTM) of the product.

This paper proposes a solution to this problem where an equivalent UVM based Model of the HM can be plugged in the SoC and verification process may continue before the actual RTL design of HM is available enabling left shift of the verification cycle and shorter TTM without compromising quality. Using this approach, the standalone HM development & verification process and the SoC verification process can happen parallelly and can help cut down schedule time. This model supports all the integration aspects of the actual HM such as Power modes, Debug modes, Master and Slave AXI/AHB transactors, Interrupts and Performance. Apart from the functional features, it also supports the Design For Testability (DFT) features like boundary scan, test-bus, etc. Furthermore, the model is configurable to mimic the traffic profile of standard IP's like WIFI and DSP for use-cases and contains in-built performance monitor which can be configured for the required min/max latencies and average/peak bandwidth. The model also supports the self-checking mechanism for connectivity verification of the bi-directional signals (pads in HM) and the unidirectional signals (pads in SoC IO ring).

Model provides unique approach to validate Boot modes and power modes. There are various boot modes such as cold boot, Boot with DFT mode enable, Abnormal reset/crash reset which are important to be verified at the SoC level to ensure the correctness of specifications at SoC and HM level. Similarly there are various power modes with different wake up sources which need to be verified at the SoC level to ensure power manager at SoC and HM level are aligned including the isolations of various signals in different power domain between SoC and HM. Model also took into account some of the Power Manager IP (PMIC) features.

Along with the HM model, the model development team also provides a top-level test plan to the SoC team with the specific scenarios to ensure complete feature and use-case verification of the HM. The test plan includes scenarios, test procedures, re-usable UVM sequences, functional coverage plan, assertions and checks to be implemented at the SoC level by the SoC team. This aspect is very important for ensuring that SoC team runs all the use case scenarios. Assertions at the soc level ensure that assumptions made by SoC team and HM team are aligned. Functional coverage ensures that there are no gaps in the feature coverage.

The same approach can also be used in another real-life situation where the HM RTL design is confidential and cannot be shared with the SoC team. Only the Graphical Data System (GDS) file is to be provided to the foundry for integrated circuit (IC) fabrication. Again, in this case, an equivalent behavioral model of the HM bundled with the test plan can be delivered to the SoC team to enable the SoC verification process.

II. HM Model Architecture

This section describes the usage of HM model that will be integrated with the SoC. The model is used to verify the correctness of the integration of HM in SoC and to verify the SoC level use case scenarios.

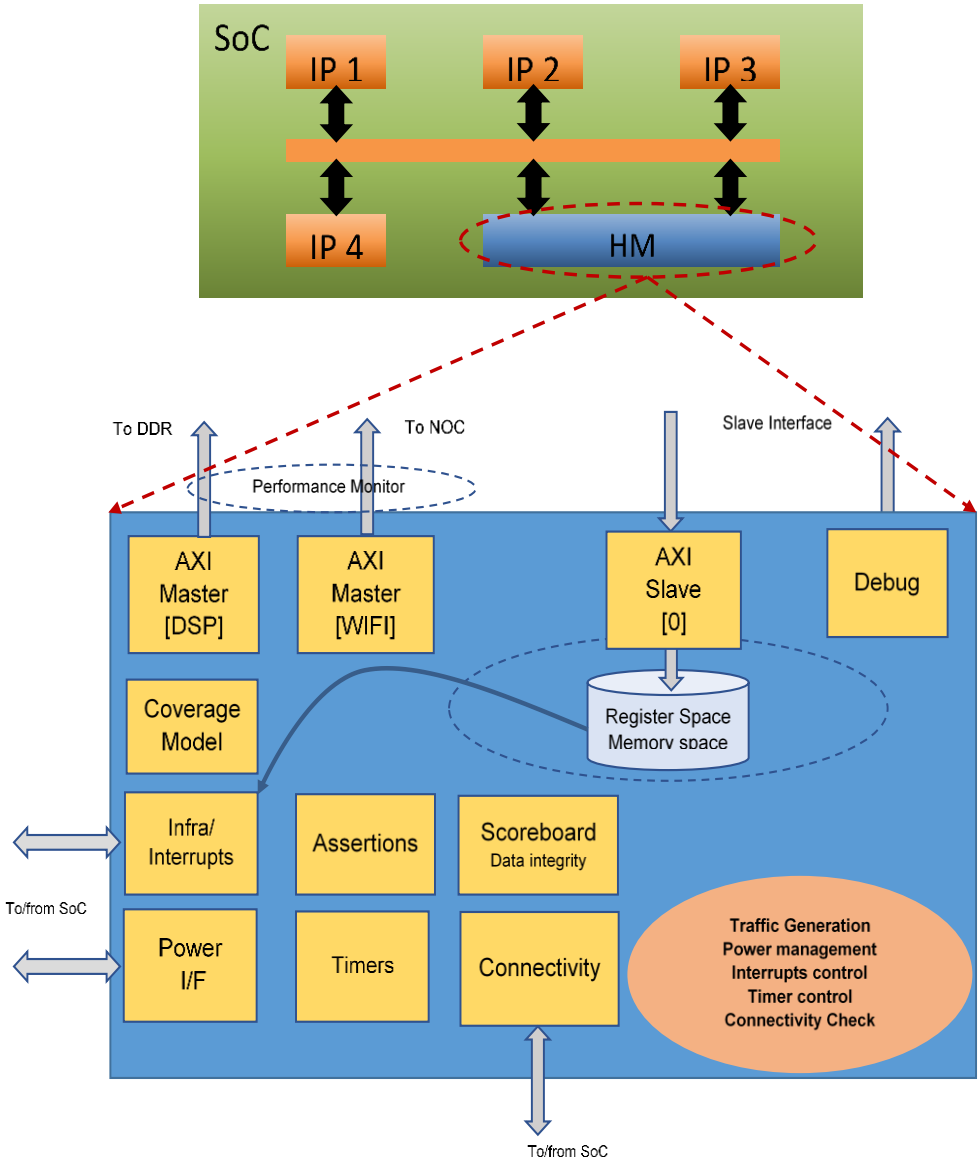

Figure 1. HM Model Architecture

Figure 1 shows the architecture of the UVM based HM model. It comprises of all the valid interfaces and required components as per the HM architecture. Component list as follows:

- AMBA-AXI Master/Slave Agent
- Debug Interface
- Power Management
- Interrupt Generation/Handle Model
- Memory Mapped Register Space
- Bus Monitor
- Performance Monitor
- Functional Coverage
- Assertions

Above features will be discussed in detail in next section of this paper.

A. Features NOT supported by HM Model

Before going to the next section where the HM model features are discussed in detail, below is the list of features that should not be expected from this HM model.

- Model is not bit accurate. It doesn't have the full functionality of actual HM.
- Model does not map the HM design registers. It implements few registers for slave interface, required for HM model integration verification.
- Model doesn't implement the functionality of WIFI. It just mimics the WIFI traffic at the AXI Master interface.
- Model doesn't implement the functionality of DSP. It just mimics the DSP traffic at the AXI Master interface.
- Model doesn't support power aware simulation using UPF at the SOC level.

III. FEATURES SUPPORTED BY HM MODEL

This section provides detailed information about each component encapsulated inside the HM model.

A. AXI Master/Slave Agent

- AXI Master mimics the traffic generation for DSP and WIFI IPs.
- AXI Slave is used for configuring register space implemented for interrupts generation, status etc. These registers are implemented in HM model to ease out the model integration verification in SoC.
- All the AXI interfaces have system monitor, assertions and scoreboard mechanism to verify the protocol compliance and the data integrity.
- Model can generate the performance summary and checks the transaction Max/Average latency and bandwidth.
- The AXI UVC checks for any violation done on the bus for each transaction done. If the transaction violates any of the parameters UVM Error gets flashed.
- AXI Master UVC placed for DSP and WIFI has a built-in monitor which looks for the traffic on the bus and send it to analysis port of the scoreboard on the other hand to check the integrity of the data. SoC user has the responsibility to feed their memory data through backdoor access.

B. ATB Master Agent

- ATB Master agent mimics the traffic generation for Trace from HM model.

- It has system monitor, assertions and scoreboard mechanism to verify the protocol compliance and the data integrity.

C. Debug Agent

Debug agent support the feature required at HM interface level for effective debug communication from SoC to HM and vice versa.

- 56-bit Gray code time count

  56-bit input to the HM model coming from the SoC. SoC will have the mechanism to generate the 56-bit gray code time-count. This bus is directly connected to the model input. Connectivity check will ensure the proper connectivity of this gray code counter.

- Boundary Scan

  The main Test Access Port (TAP) is part of the SoC and boundary scan control signals are provided at the HM boundary for the Boundary scan. The pads inside HM are also modelled in the HM model and having support for boundary scan. Boundary scan can be checked by using the SoC main TAP.

- Narrow Timestamp (NTS)

  The input 7-bit encoded data with sync bits gets decoded into 64-bit value by the NTS Decoder driver implemented in the HM model environment.

D. Power Management Agent

Power supply rail, chip reset and oscillator clock to SoC is driven by HM. So, HM model provides appropriate response to requests by SoC for power supply control, reset requests and oscillator clock on/off requests. It also handles handshaking between HM and SOC during the sleep-wakeup sequences, HM DDR-BW (Bandwidth) voting, Crash reset sequences and SSR (Sub-System Reset). HM model handles the below power related functions:

- Power Supply Control

  Appropriate response by HM model for transaction initiated by SoC to control power supply as per its operational state.

- XO clock vote control

  SOC XO clock shutdown when SOC sends requests to turn on/off the clock via input port.

- Cold Boot Reset control

  Based on the mode in which the chip is initialized, HM model performs the appropriate handshaking of reset signals with SoC.

- Sub System Reset (SSR) control

  Model supports the SSR feature where SoC can assert-deassert the reset signal any time for HM boot in case of any abnormal behavior of HM.

- Full Chip Reset by SoC

  Model caters the SoC request to issue a chip reset in response to the full chip reset request by HM.

- DDR BW voting by HM

  Model supports the 4-way handshake performed between HM and SoC for DDR BW voting by which HM can control the DDR clock frequency.

- HM wakeup via SoC

  When HM is in sleep, it supports the wakeup scheme where the interrupt source is SoC via interrupt ports.

- SoC sleep vote

Model supports the feature where SoC asserts the sleep vote by asserting a sleep vote I/O port, and if HM is in Sleep, it acknowledges the sleep vote request via ack I/O signal.

E. Infra Agent

Infra agent handles the assertion and de-assertion of interrupt going from HM to SoC. Also, it monitors the interrupts coming from SoC to HM and executes the ISR (Interrupt Service Routine) to clear the incoming interrupts. HM model also implements two sets of status registers which stores the current status of both incoming and outgoing interrupt pins. SoC test developer can read these registers any time in the test to check the status of both type of interrupts.

Apart from interrupt handling, infra agent supports the feature for IO connectivity verification of non-functional ports.

- Assertion and de-assertion of outgoing interrupts from HM

  Model supports the feature to generate the outgoing interrupts using SV task. SoC test writer can use this task to set the desired outgoing interrupt.

  For de-assertion/clearing of outgoing interrupts, there are two provisions provided to the SoC user. SoC test developer can either use the SV task to clear the interrupts or can write to the interrupt clear register defined in the model register space.

- Monitoring and De-assertion of incoming interrupts to HM

  HM model continuously monitors the incoming interrupts from SoC and executes the ISR in case of any incoming interrupts. This ISR will finally clear the interrupt by writing to the appropriate registers of SoC.

- Connectivity Check

  Infra agent also supports the feature to check the proper connection of non-functional pins between HM and SoC.

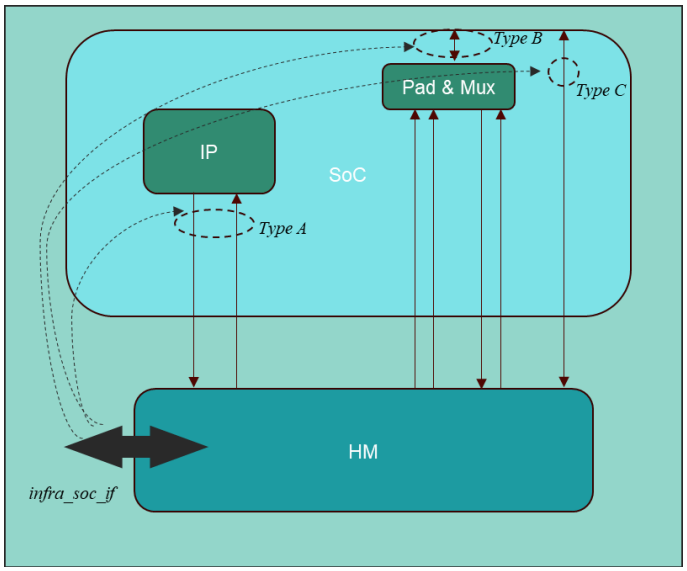

Figure 2. Connectivity check for non-functional pins.

As shown in Figure 2, there can be three types of connection between HM and SoC.

Type A signals – Consumed within SoC logic.

Type B signals – Pads in SoC IO ring.

Type C signals - Pads in HM

HM model team provides directed tests for connectivity check of these signals to the SoC team. SoC team just connects the model interface "infra_soc_if" with the appropriate signal hierarchy in SoC and run the tests. These tests are self-checking and throws an error in case of any open port or cross connection.

F. Assertions and Functional Coverage

Several assertions are plugged in the model which ensures that both the SoC team and the HM team are aligned with the specifications and there is no mismatch.

Also, to ensure that there are no gaps in the feature coverage, modelling team shares various cover-groups with the SoC team.

Both, assertion and coverage help to test and monitor the feature completeness with correctness of the traffic generated and the response by the SoC.

IV. ARTIFACTS FOR SOC VERIFICATION ENABLEMENT

HM modelling team provides the following artifacts to the SoC team for SoC verification enablement.

- Behavioral UVM-Model of the actual HM
- Top level SoC testplan with specific scenarios to ensure complete feature and use-case verification of HM in SoC. Testplan includes:
    - Test scenarios and procedures.
    - Re-usable UVM sequences.
    - Functional Covergroups.
    - Assertion properties and checks.

  An example SoC level testplan is shown in Figure 3.
- A user guide covering:
    - Architectural details of the model.
    - Integration aspects of the model.
    - Coverage plan
    - Assertion plan
    - Steps for creating new tests at SoC level.

S. No.	Feature Coverage	IO's to be Covered	Test Scenario	Test Procedure	Sequence Name	Model Check	SoC Check	Functional Coverage
1	Cold Boot Reset	sig_a, sig_b, sig_c, sig_d, sig_e, sig_f, sig_sts, sig_g, sig_h	Cold boot triggers with sig_a de-assertion followed by sig_b de-assertion, DSP self boot triggers and start booting form boot-ROM, full chip should be out of reset and clock controller starts running default clocks	1) Release sig_a 2) Wait for sig_b 3) Release sig_c 4) Release sig_d 5) Apply cold_boot sig_e=0x00 6) Release sig_f 7) Boot DSP and update sig_sts during boot 8) Ensure sig_g should high on top. 9) Set sig_h at end of boot 10) Access HM registers from each block 11) Ensure all updates for sig_sts reflecting on top	trig_cold_boot_reset_seq	Assertion: P1_sva_cold_boot	Data Sanity Check	cold_boot_cg
2	Feature 2	sig_k, sig_l	Feature 2 scenario	Feature 2 test procedure	feature_2_seq	feature_2_check	Assertion: P1_sva_feature_2	feature_2_cg
...	...	...	...	...	...	...	...	...

Figure 3. SoC level testplan.

REFERENCES

[1] Universal Verification Methodolgy (UVM) 1.2 User's Guide

[2] Chris Spear, System Verilog for Verification, 3rd ed., Springer