Expedite multi-die coherency verification through adaptive VIP subsystem

Jainender Kumar (Samsung-India, jain.kumar@samsung.com), Sunil Shrirangrao Kashide (Samsung-India, sunil.sk@samsung.com), Garima Srivastava (Samsung-India, s.garima@samsung.com), Dimitry Pavlovsky (Cadence-USA, dimitry@cadence.com), Anunay Bajaj (Cadence-India, anunay@cadence.com)

***Abstract*- Coherency verification plays a pivotal role in ensuring the integrity and performance of the multiple die designs using UCIe (Universal Chiplet Interconnect Express) protocol. As multi-die/multi-chiplet system grows in complexity, keeping a consistent view of shared data across different chiplets or dies becomes increasingly challenging. Coherency ensures that no die contains stale or incorrect information. It also ensures safeguarding system stability and prevents critical issues like data corruption or race conditions. In a homogeneous/heterogeneous multi-chiplet system in package (SIP), each of the processor cores may read and write to a unified memory space. To offer a coherent view of memory in such a system, certain rules about memory reads and writes and how they act upon memory are needed. The multi-chip SIP is nowhere an exception to this mandate. To maintain the memory consistency rules across the dies, the coherency extensions provided by the ARM-based AMBA CHI (coherent hub interface) protocol help stream system traffic over the UCIe interconnect.**

To accelerate interconnect verification, this paper proposes a special UCIe Verification IP (VIP) that is "sandwiched" with streaming protocol VIP and coherent VIPs forming "UCIe VIP sub-system" to achieve verification of system-level coherency even before the complete multi-chip RTL/design environment is ready. The VIP Is highly configurable and it can communicate either directly through Credited eXtensible Stream (CXS) (bypassing UCIe) or through UCIe. This unique utility provides a great head start to verification by allowing test case execution at different stages of RTL.

I. Introduction

The primary goal is to verify the entire data path logic that connects various components which allows shareable and non-shareable memory accesses across the chiplets. This should also ensure the correctness and consistency of data across various levels of caches in a multi-chip verification environment. This requirement demands achieving multi-fold functional verification confidence, especially over the data path which includes the interconnect fabric and the 'UCIe tunnel.' To address this requirement, the realization of secondary chiplet/die is modeled using a UVM (Universal Verification Methodology) based testbench framework.

Including multiple chiplets connected via UCIe in a testbench presents a higher runtime challenge, especially in a simulation-based verification approach. Thus, it becomes very crucial for a verification engineer to evolve the verification strategies and to devise a solution that not only allows verification in the early phase but also offers efficient resource utilization, deterministic runtime, reusability, scalability, and support for coherency.

Owing to the above challenges and requirements, a VIP subsystem is needed to help reduce turnaround time (TAT) in data path verification of multi-chip designs using a single chiplet with a VIP subsystem. Considerable time and effort are involved in coordinating simulation efforts across the chiplets, leading to prolonged simulation periods. Additionally, this solution will help achieve faster link-up and facilitate ease of debugging. The VIP subsystem in this scenario would mimic the secondary chiplet/die environment, generating or responding to coherent and non-coherent requests. This would provide significant advantages in managing and creating complex test scenarios spanning across the chiplets. The verification metrics thus reported will give greater confidence to the test writer for early UCIe-based verification.

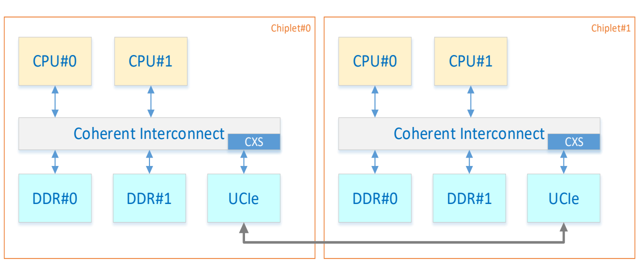

Fig. 1 A typical multi-chiplet skeleton system

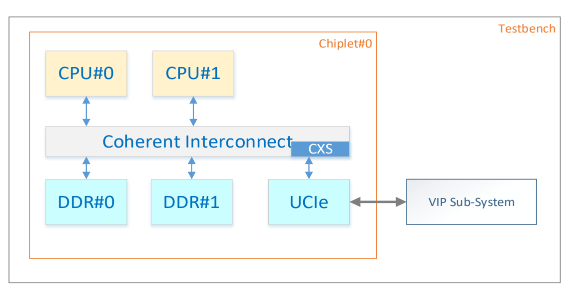

Fig. 2 Multi-chiplet verification using VIP subsystem

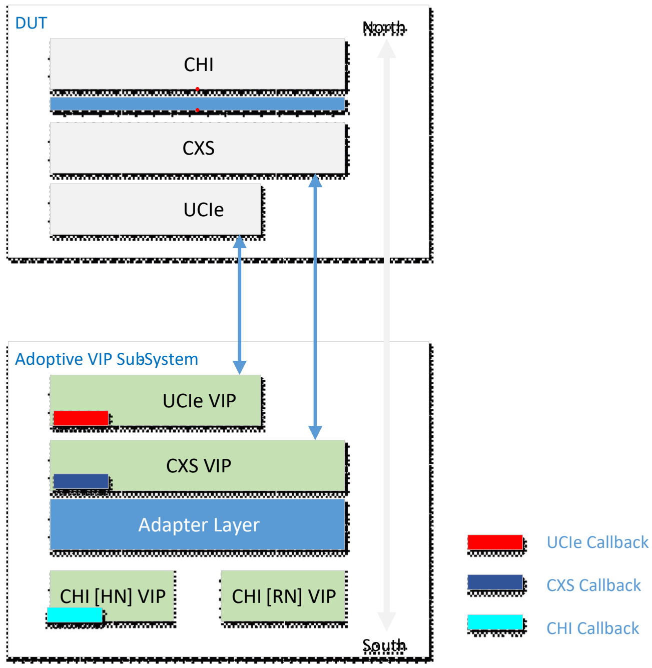

Fig. 3 Topological view of the UCIe VIP subsystem architecture.

II. VIP SUBSYSTEM COMPONENTS

A typical multi-chip skeleton system, as shown in Fig.1, includes multiple CPU clusters connected on both sides of the coherent interconnect and the UCIe controller bridge, facilitating the communication channel across the chips.

Fig. 2 depicts the topological connection of the proposed VIP subsystem alongside other DUT components, mimicking a secondary chiplet (Chiplet#2) setup in a primary chiplet (Chiplet#1) verification environment. The UCIe VIP comprises three layers. [a] Protocol layer, [b] Die-to-die Adapter layer, and [c] Physical layer. As shown in Fig. 3, the north end of the VIP subsystem has the physical layer of UCIe-VIP connected to the RTL UCIe-IP, while the south end of the UCIe VIP is connected to the adapter layer through Credited eXtensible Stream (CXS) port communication.

The UCIe protocol is a transport layer protocol, and any request transferred requires a responder to provide an appropriate response. In this scenario, fabricating a response stimulus from Chiplet#1 is necessary. Requests from any coherent master on Chiplet#0 reach the coherent interconnect and subsequently the UCIe IP via the CXS interface. These requests are routed to the UCIe VIP subsystem, which uses an adapter layer to convert the data into a custom-compatible packet format. The CHI Home Node (HN) VIP receives the transformed packet, mimicking the actual multi-chip behavior in terms of address access. The HN VIP in the VIP subsystem provides appropriate responses back to the coherent interconnect, ensuring compliance with the ARM-based AMBA CHI coherency protocol.

The level of access-based verification indicates that the requests transmitted through the data path from the coherent interconnect via CXS + UCIe controller are successfully verified upon receipt of responses, even before Chiplet#1 is physically available. VIP subsystem components and its functionality can be broadly described as follows:

(i) UCIe VIP performs rapid link initialization and training, bypassing the boot-up procedure. In a multi-chiplet SIP (System-In-Package) DUT case, a full boot sequence is required for cross-chiplet communication. However, the inclusion of the VIP subsystem eliminates the need for the conventional IP programming sequence, significantly reducing simulation time.

(ii) The Adapter logic is an intermediate component, enabling communication between a proprietary interconnect packetized protocol embedded within UCIe and a standard CHI framework.

This layer is responsible for the seamless mapping of information from one protocol to another according to predefined rules, ensuring compatibility and a modularized approach, especially if the logic becomes part of a VIP vendor offering.

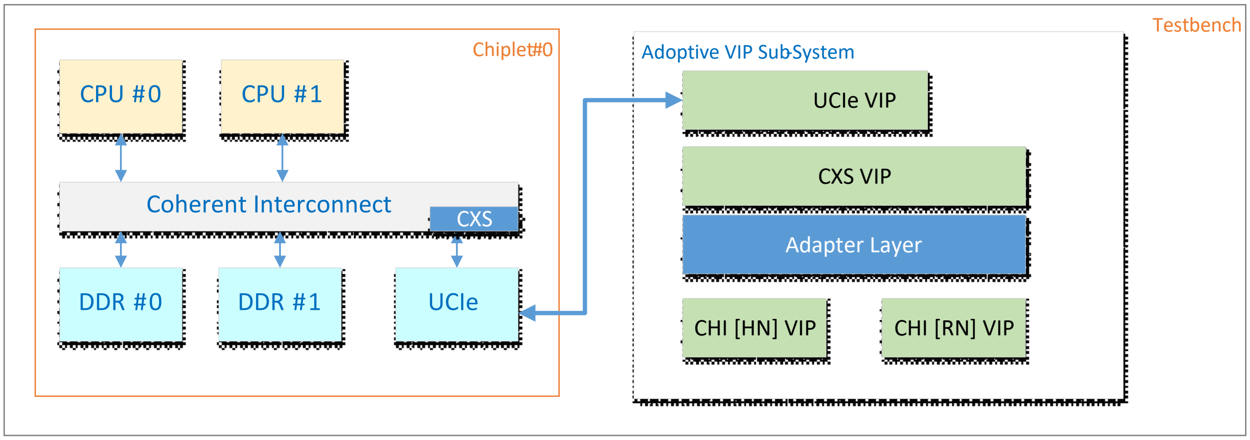

Fig. 4 Framework for multi-chip verification using VIP subsystem

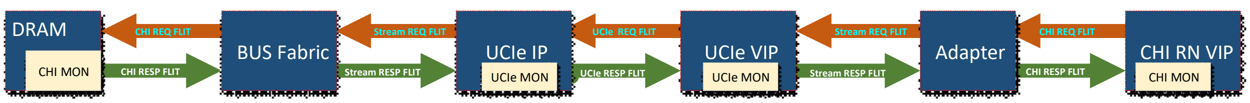

Fig. 5 Data flow from UCIe subsystem's CHI RN node to DRAM Primary chiplet

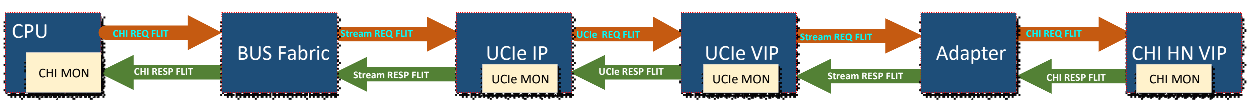

Fig. 6 Data flow from CPU to UCIe subsystem (secondary chiplet)

(iii) The interconnection of AMBA CHI (RN/HN) VIP includes a Request Node (RN) and a Home Node (HN) for handling requests and responses based on configuration in a UVM sequence generation framework in a standard methodology-driven test bench.

Fig. 4 further illustrates the encapsulation of all the components connected to form a UCIe VIP sub-system in a typical test bench environment. This integrated solution incorporates a chiplet-to-chiplet verification infrastructure within a single-chip environment.

III. Modes of Operation – VIP Subsystem

CHI VIP and adapter layer work in sync to process the requests and responses in either direction as configured:

Requestor Node (RN): As shown in Fig. 5, the RN generates CHI requests, which are formatted by the adapter layer to be compatible with the proprietary interconnect packetized protocol and then mapped onto the UCIe payload as either Raw or Streaming protocol. The CHI RN component within the VIP subsystem mimics a requester, initiating traffic (Coherent/non-coherent) from the secondary chip to the primary chip.

CHI Responses: The adapter layer receives responses generated by RTL. It processes the packetized responses and maps them to valid CHI responses. The adapter layer then sends the response to the RN node to complete the request.

Home node as responder (HN): As shown in Fig. 6, the CHI Home Node VIP receives incoming CHI requests processed by the adapter layer and provides appropriate responses back to the adapter layer for further processing. The HN node in the VIP subsystem mimics the response provided by the secondary chiplet to the primary chiplet.

IV. Features of VIP Subsystem

The test bench architecture involving the VIP subsystem ensures the correctness and integrity of communication and coherency establishment across multiple chips. The architecture includes several key components that work together to facilitate data transfer and response collection throughout the system.

In a typical multi-chip verification environment, ensuring proper connectivity and remote access across the chips is essential. When a CHI master or any IP on Chiplet#0 wants to access memory on Chiplet#1, the request must traverse a complex data path involving multiple architectural components. The transfer begins with a CHI master on Chiplet#0 initiating a memory access request to Chiplet#1. The request sent is then processed by chiplet link interconnect (CLI) inside the Chiplet#0 interconnect before being routed to the CXS interface. CLI is responsible for streamlining the request, response, data, and snoop into a single data packet for high-speed transmission and managing tokens for packetized transfer. Next, the streamlined packet is forwarded to CXS IP, which generates a CXS-specific payload, transferring data efficiently and reliably to UCIe IP on Chiplet#0. Finally, UCIe IP encapsulates the data into the necessary protocol format, facilitating seamless communication between chiplets. Once the data is transmitted to Chiplet#1, the UCIe controller forwards it to the respective coherent interconnect through the CXS interface. The CXS interface transfers the data to a coherent interconnect on Chiplet#1. Finally, the coherent interconnect on Chiplet#1 decodes the incoming data and routes it to the appropriate destination (preferably to the caches within and the residing memories). The excitation of the entire path ensures that the memory access request from Chiplet#0 reaches its intended target on Chiplet#1 and vice-versa. During this process, the responsibility of maintaining coherency and data integrity lies with the vendor-specific VIPs and the self-checking components, such as monitors and scoreboards. The developed VIP subsystem not only mimics the functionality of Chiplet#1 but also provides various added features as follows:

*A. **Adaptive configurable mode of VIP subsystem***

The VIP subsystem offers two configurable modes. These modes allow the VIP subsystem to connect to RTL through either at CXS layer or UCIe layer, as shown in Fig. 7 through dotted lines. This configuration enables data integrity and protocol-related checks to be started early even if UCIe IP is under development giving a head start to verification.

*B. **Bypass configuration of VIP subsystem***

UCIe VIP in the VIP subsystem provides necessary configuration knobs to bypass link training and to facilitate faster link-up and reduced simulation time while monitoring and printing the link status information for higher visibility. This parallel operation of VIP will reduce the turnaround time of data path integrity and coherency checks.

*C. **APIs for providing vendor-specific flow control packets***

The adapter layer provides additional APIs that can be utilized to support additional management packet communication as shown in Fig. 7. This becomes useful when certain management packets need to be transferred before starting the actual communication. For an example, in a system supporting packet based protocol, default token value must be configured either to start or to respond to any kind of request. The management packet contains default values of these tokens.

When enabled, these management specific APIs allows the adapter layer of the VIP subsystem to identify management packets using the packet header and respond accordingly (or as configured). As per the configuration, the adapter layer allows request packets to be responded to either with accept/deny link status information or to exchange credits in a predefined packet format. This feature provides additional packet support that is not part of any other VIP and is primarily used to address vendor-specific management packets.

D. Error injection at various levels

The VIP Subsystem provides various callbacks for each of the layers, be it CHI, CXS, or UCIe. These callbacks can be used to inject errors at various levels, which would have been difficult otherwise in the real RTL design. These callbacks enable comprehensive negative test scenarios to be exercised, such as data checksum errors, poison bit propagation, etc. at the CHI level; parity check and miss-compare error checks at the CXS interface; and various timeout and data checksum errors at the UCIe level.

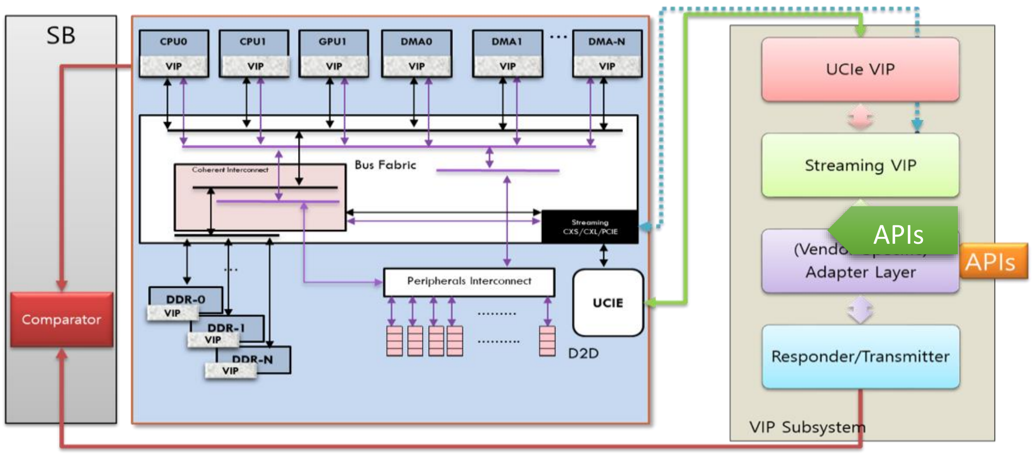

Fig. 7 VIP subsystem adoptive connection configuration

E.   Plug n Play system scoreboard

As depicted in Fig. 7, the proposed subsystem allows a UVM-based system-level scoreboard to be instantiated across the VIP monitor interfaces for data integrity and protocol checks. The use of standard AMBA CHI protocol for development makes it possible to easily port the CHI sequences and test scenarios, ensuring compatibility with existing scoreboards.

With this, the correctness and completeness of data while it transfers through the coherent interconnect across the chiplets are guaranteed without any additional design or checker overhead. It will act as an overall system monitor, monitoring the entire AMBA-based transactions. At active or passive interfaces, this scoreboard provides necessary information about protocol traffic at the relevant ports. The same is expected and applicable throughout the multi-chip communication process, replicated at the VIP subsystem level.

F.   CHI sequence backward compatibility:

The end node of the VIP subsystem comprises a CHI Request/Home node, making it compatible with any pre-existing CHI-based sequence.

V. Advantages of the Proposed Methodology

The proposed VIP subsystem and its features offer various advantages that can be categorized as follows:

A.   Scalable

The solution demonstrates excellent scalability, making it well-suited for handling the complexity associated with verifying coherency across multiple chiplets. By leveraging a flexible architecture, the solution ensures that even as the number of chiplets increases, the verification process remains streamlined and effective. Fig. 8, shows how the proposed solution can be used in a typical star topology DUT connection.

B.   Reusability

Reusing the CHI sequences, test scenarios and system scoreboard is especially advantageous in a scalable solution. As the number of chiplet/die or system complexity increases, the reuse of these components becomes increasingly valuable.

C.   Portability

The proposed solution is robust across different verification levels, whether it is IP level, block level, or SoC level. The test stimulus as well can be ported to a multi-chiplet DUT top environment. This consistency simplifies debugging and ensures coherent results at every level of verification.

D. Error injection

With multiple layers of VIP providing various callbacks, it becomes easy to inject errors at any level, enabling the realization of various types of negative scenarios, such as back pressure, payload corruption, retry/re-train, timeout, and checksum errors. This ensures thorough coverage of potential failure points, making the system more robust and reliable in real-world applications.

E. Accelerate verification

The VIP subsystem allows incremental verification, where the data path and connection integrity can be verified with CXS mode as per the different stages of design development. This allows faster identification of bugs. This approach accelerates the verification process by enabling parallel progress on different aspects of DUT verification, reducing bottlenecks, and preventing delays typically caused by waiting for the complete system design.

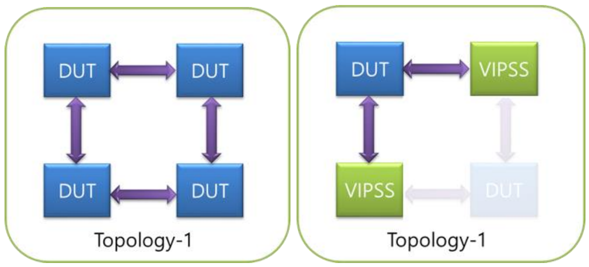

Fig. 8 Multiple chiplet test bench setup configuration in a typical star topology vs using VIP subsystem (VIPSS)

F. Efficient resource utilization

The implementation of the VIP subsystem is resource-efficient as compared to connecting a complete system-on-chip (SoC) for multi-die verification. The solution minimizes the need for the excessive computational power or hardware resources typically demanded by multi-die SoC verification. This efficiency becomes even more critical as the number of dies in the setup increases exponentially.

G. Supports all the coherency features.

The solution's support for all coherency traffic types provides comprehensive coverage, ensuring that every possible coherency scenario is thoroughly verified. This broad traffic support strengthens the confidence in the verification of multi-die systems. By handling diverse coherency interactions, the solution ensures that the system remains stable and consistent across different configurations.

H. Easy Debug

In a multi-chiplet SoC, where every individual chiplet is connected through a packet-based streaming protocol like UCIe and CXS, it becomes challenging to decode the packet at various stages. The VIP subsystem provides simplified information traveling across the different layers, offering different types of payloads that would otherwise be difficult to understand through waveforms. This also makes it easier to track the status of different state machines involved at each layer, thereby aiding in faster debugging. For example, below are the attached waveform and corresponding log messages printed by the UCIe monitor.

Despite of above mentioned advantages, below are the few limitations of the VIPSS:

A. Modeling of the real system traffic delay cannot be realized exactly.

B. Development of VIP adapter layer logic adds the delay at the start of execution whereas back-to-back RTL does not have (precisely in homogenous design).

VI. SIMULATION WITH VIP SUBSYSTEM

This section depicts the various implementation-specific log prints and messages in a multi-die setup.

A. VIP Subsystem: UCIe prints and Waveform

The intended stimulus/scenarios execution status can be understood and analyzed using the VIPSS layered logging capability for the ease of debug.

- *Physical layer:*

```
UCIE_PROC  state change : UNKNOWN to Undefined
UCIE_PROC  LTSM state change : UNKNOWN to UNDEFINED
[TX] LTSM state change : UNDEFINED to RESET
[TX] LTSM state change : RESET to SBINIT
[TX] LTSM state change : SBINIT to MBINIT_PARAM
[TX] LTSM state change : MBINIT_PARAM to MBINIT_CAL
[TX] LTSM state change : MBINIT_CAL to MBINIT_REPAIRCLK
[TX] LTSM state change : MBINIT_REPAIRCLK to MBINIT_REPAIRVAL
[TX] LTSM state change : MBINIT_REPAIRVAL to MBINIT_REVERSALMB
[TX] LTSM state change : MBINIT_REVERSALMB to MBINIT_REPAIRMB
[TX] LTSM state change : MBINIT_REPAIRMB to MBTRAIN_VALVREF
[TX] LTSM state change : MBTRAIN_VALVREF to MBTRAIN_DATAVREF
[TX] LTSM state change : MBTRAIN_DATAVREF to MBTRAIN_SPEEDIDLE
[TX] LTSM state change : MBTRAIN_SPEEDIDLE to MBTRAIN_TXSELFCAL
[TX] LTSM state change : MBTRAIN_TXSELFCAL to MBTRAIN_RXCLKCAL
[TX] LTSM state change : MBTRAIN_RXCLKCAL to MBTRAIN_VALTRAINCENTER
[TX] LTSM state change : MBTRAIN_VALTRAINCENTER to MBTRAIN_VALTRAINVREF
[TX] LTSM state change : MBTRAIN_VALTRAINVREF to MBTRAIN_DATATRAINCENTER1
[TX] LTSM state change : MBTRAIN_DATATRAINCENTER1 to MBTRAIN_DATATRAINVREF
[TX] LTSM state change : MBTRAIN_DATATRAINVREF to MBTRAIN_RXDESKEW
[TX] LTSM state change : MBTRAIN_RXDESKEW to MBTRAIN_DATATRAINCENTER2
[TX] LTSM state change : MBTRAIN_DATATRAINCENTER2 to MBTRAIN_LINKSPEED
[TX] LTSM state change : MBTRAIN_LINKSPEED to LINKINIT
RDISSM state change : Undefined to Reset
RDISSM state change : Reset to Active
[TX] LTSM state change : LINKINIT to ACTIVE
```

- *RDI layer:*

```
ucieAgent>@0 ns : ADP RDISSM state change : Undefined to Reset
ucieAgent>@0 ns : RDISSM state change : Undefined to Reset
ucieAgent>@628920 fs : ADP RDISSM Loading Protocol layer register default values from SOMA
ucieAgent>@628920 fs : ADP RDISSM Info! REG_WRITE : FDI_SB_RX_LOCAL_CREDITS - AdpCredits=32
ucieAgent>@628920 fs : ADP RDISSM Info! REG_WRITE : FDI_SB_RX_LOCAL_CREDITS - ProtoCredits=32
ucieAgent>@628920 fs : ADP RDISSM Info! REG_WRITE : FDI_SB_TX_LOCAL_CREDITS - AdpCredits=32
ucieAgent>@628920 fs : ADP RDISSM Info! REG_WRITE : FDI_SB_TX_LOCAL_CREDITS - ProtoCredits=32
ucieAgent>@8500 ps : ADP RDISSM state change : Reset to Active
ucieAgent>@8500 ps : RDISSM state change : Reset to Active
UVM_INFO (waitForAdpParamExchangeHandshake : WAIT_FOR_ADP_PARAM_EXCHANGE_HANDSHAKE) begun
ucieAgent>@72 ns : ADP RDISSM Protocol set to StreamingProtocol
ucieAgent>@72 ns : ADP RDISSM Flit Format set to 68BFlit, i.e. format 2
UVM_INFO (waitForAdpParamExchangeHandshake : WAIT_FOR_ADP_PARAM_EXCHANGE_HANDSHAKE) ended
```

- *FDI layer:*

```
ucieAgent>@00628920 fs : PROTO 0 FDI LSM 1 state change : Undefined to Reset
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Reset Protocol registers
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Loading Protocol layer register default values from SOMA
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Info! REG_WRITE : FDI_SB_RX_LOCAL_CREDITS - AdpCredits=32
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Info! REG_WRITE : FDI_SB_RX_LOCAL_CREDITS - ProtoCredits=32
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Info! REG_WRITE : FDI_SB_TX_LOCAL_CREDITS - AdpCredits=32
ucieAgent>@328 ns : PROTO 0 FDI LSM 0 Info! REG_WRITE : FDI_SB_TX_LOCAL_CREDITS - ProtoCredits=32
UVM_INFO (waitForAdpLsmState : WAIT_FOR_ADP_LSM_STATE) stack 0 DENALI_UCIE_SSM_STATE_Active Started
ucieAgent>@660500 ps : PROTO 0 FDI LSM 0 state change : Reset to Active
ucieAgent>@660500 ps : ADP STACK LSM 0 state change : Reset to Active
```

B. *VIP Subsystem: CXS-Adapter layer*

In the VIP subsystem, the UCIe VIP communicates internally with CXS, through internal callbacks and processing queues without the need of interface signals, to generate a CXS complaint streaming packet.

- **Link management packet by CXS**

```
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 :
##### VIP is sending Initial link management packet #####
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[0] = 0x1
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[1] = 0x0
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[2] = 0x0
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[3] = 0x6a
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[4] = 0x92
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[5] = 0x6d
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[6] = 0xb3
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[7] = 0x2b
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[8] = 0x1
cdn_ps_ucie_if_env[1].ucieAgent.activeUp1 : linkMngmt0[9] = 0x0
```

- **Packet reception by CXS VIP**

```
--------------------------------------------------------------------------------
denaliCxsTransaction printInfo (Time: 7853512895000) :
--------------------------------------------------------------------------------
Type : DENALI_CXS_TR_FlitRx
Callback : DENALI_CXS_CB_FlitEndedRx
ErrorId : DENALI_CXS_NO_ERROR
FlitDataRx (8'h) :  01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
CurStartPtrsRx (5'h) :  00 1f
CurEndPtrsRx (6'h) :  07 3f
CurEndPktSizeRx (32'h) :  00000020 ffffffff
CurEndErrsRx : 4'h0
StartedPerFlitRx : 4'h1
EndedPerFlitRx : 4'h1
FlitControlRx (8'h) :  11 07
FlitEndedErr : DENALI_CXS_FLIT_ENDED_NoErr
FlitProtocolType (8'h) :  00 00
FlitLast (8'h) :  01 00
```

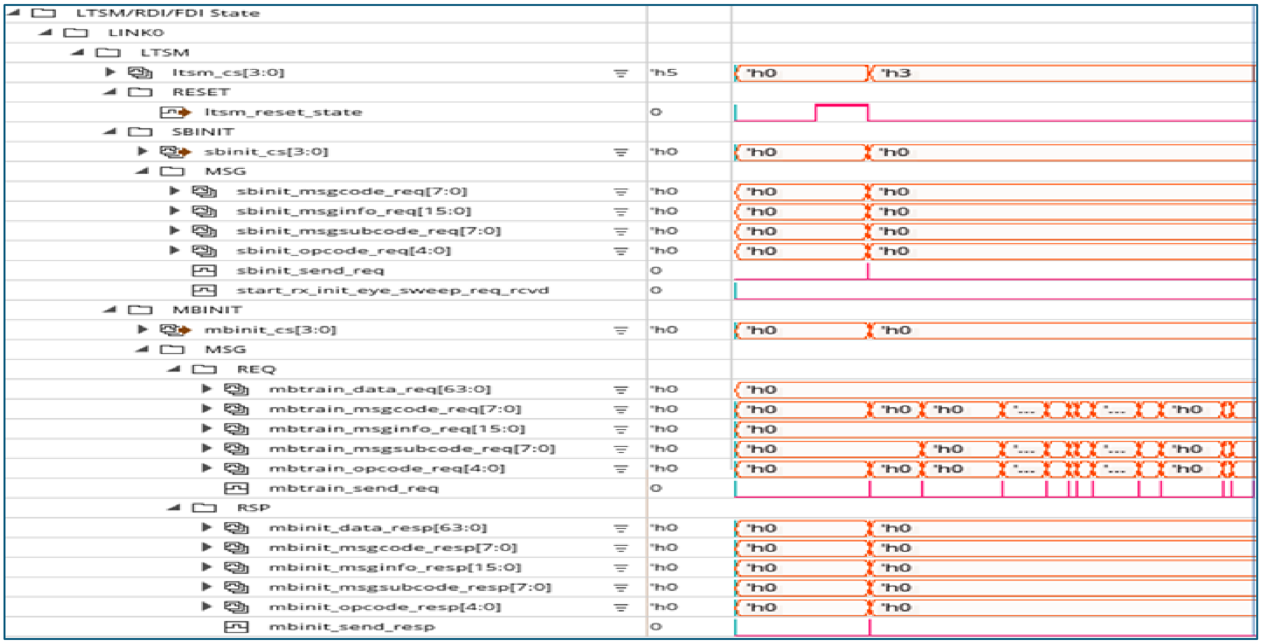

Fig. 9 UCIe sideband and main band waveform

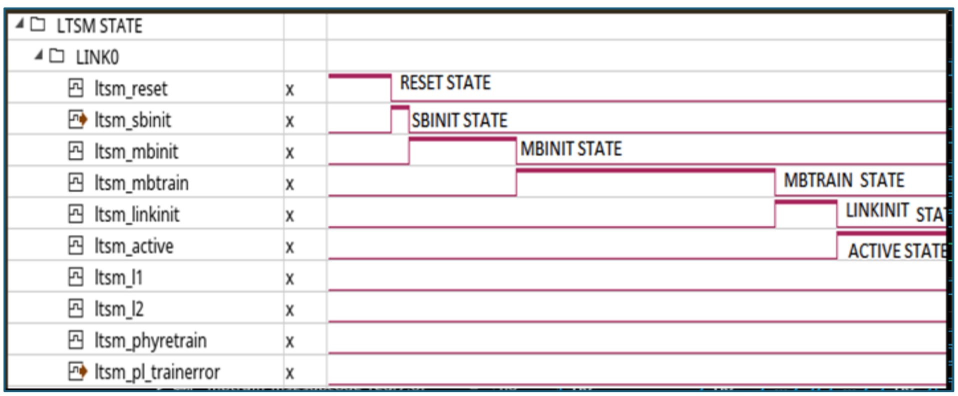

Fig. 10 UCIe PHY waveform

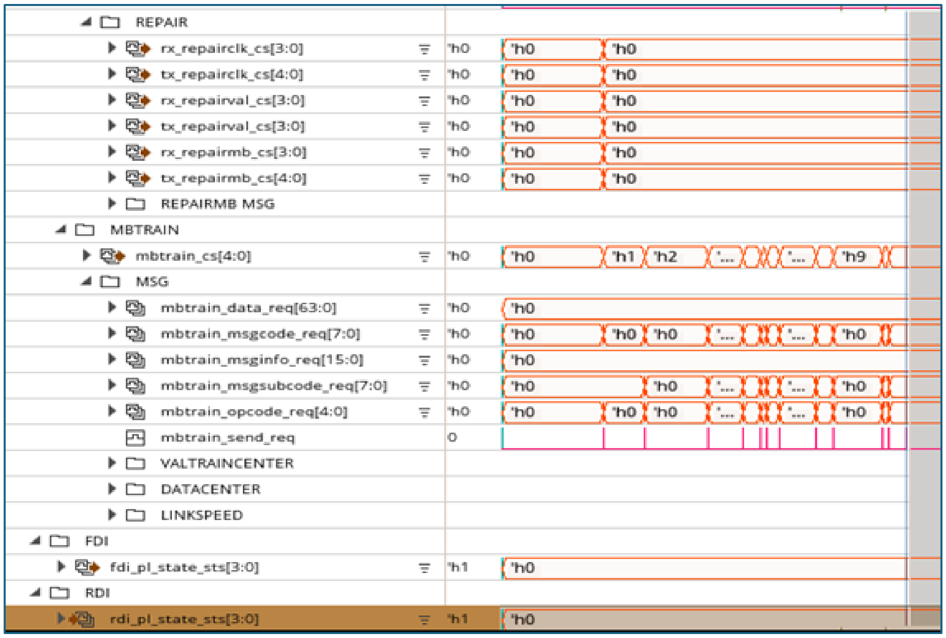

Fig. 11 UCIe FDI waveform

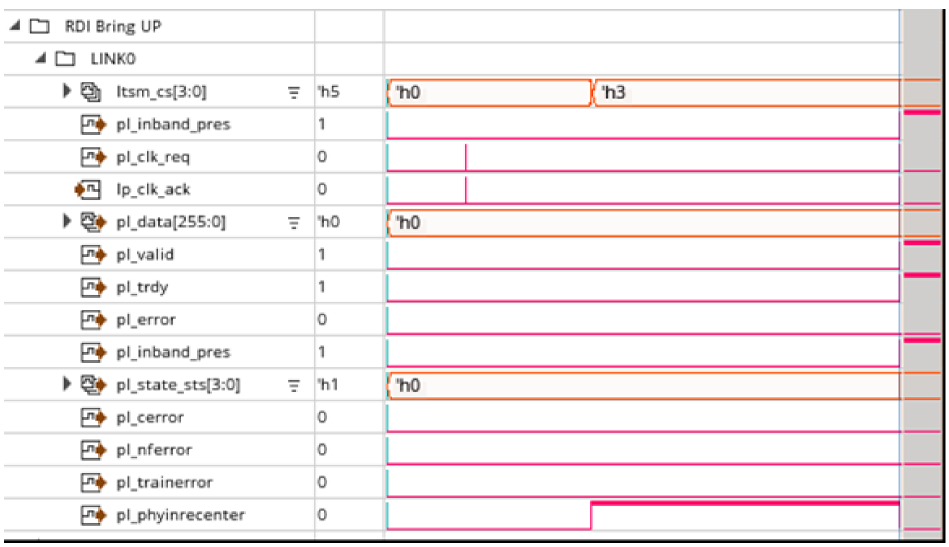

Fig. 12 UCIe RDI waveform

C. *VIP Subsystem: Adapter layer plus CHI VIP*

The CXS-complaint streaming packet is then received by the adapter layer, processed, and finally delivered to the CHI VIP or vice versa. This communication can be enabled and logged as the requests are being processed. Below are the prints related to a request being sent by CHI-RN in the VIP subsystem.

uvm_test_top.tb.ucieSve0.active_CHI_Rn : Inside chiCxsSendRequest
uvm_test_top.tb.ucieSve0.active_CHI_Rn : Transaction id from CHI trans = 0x32d
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Start Request flit info: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Cmd = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== QoS = 1: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== TgtID = 16: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== SrcID = 24: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== TxnID = 813: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== ReturnNID = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Endian = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== ReturnTxnID = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== ReqOpCode = 4: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Size = 7: ====

uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Address = 0x446dedb8: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== addrExt = 0x0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== NS = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== LikelyShared = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== AllowRetry = 1: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Order = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== PCrdType = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== MemAttr = 5: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== SnpAttr = 1: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== LPID = 10: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== Excl = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== ExpCompAck = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== TagOp = 1: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== TraceTag = 0: ====
uvm_test_top.tb.ucieSve0.active_CHI_Rn : ==== End Of Request flit info ====

VII. RESULTS

The multi-die has been verified with the proposed VIP subsystem, and it has been observed that the runtime was recorded to be 4.75X faster compared to multi-die simulations.

Furthermore, as the VIP subsystem can be configured to operate directly through CXS, the verification was started even before the UCIe IP matures. The list of design bugs found is listed below.

TABLE I

COMPARISON OF SIMULATION BOOT, LINKUP AND RUN TIME

Topology	Boot + Linkup (Hr)	Simulation run time (Hr)	Total Time (Hr)
Multi-Die (dual die)	37	1	38
VIP subsystem	7	1	08

TABLE II

NUMBER OF DESIGN BUGS WITH PROPOSED VIP SUBSYSTEM

Defect detect penetration	#
SFR Path	10
IP Design	5
System Level	7
Error injection scenario	10

VIII. CONCLUSION

In summary, the proposed VIP subsystem demonstrates significant advancements in multi-chip die verification by providing a faster approach for uncovering early integration issues and bugs in a multi-chip environment, with a focus on data integrity and protocol adherence checks at various levels. Furthermore, the VIP subsystem can be configured to operate at the CXS or UCIe level, allowing designs to be verified at different stages of development, thereby expediting coherency verification. Additionally, cross-chiplet test cases, which typically require 2-3 days to complete in a traditional multi-chiplet test bench, can be successfully executed within just 10-12 hours when migrated to the proposed VIP subsystem. The subsystem's scalable, reusable, and portable design ensures seamless operation across various verification levels, from IP to SoC. By leveraging the CHI and UCIe protocols, the subsystem facilitates efficient error injection, accelerates verification, and optimizes resource utilization. Additionally, the various advantages offered by the UCIe subsystem make it an ideal candidate for multi-die verification. Ultimately, the VIP subsystem's comprehensive coverage of coherency features makes it a powerful tool for enhancing the reliability and efficiency of multi-die system verification.

REFERENCES

[1] Kapoor, H.K., Kanakala, P., Verma, M. et al. Design and formal verification of a hierarchical cache coherence protocol for NoC based multiprocess.

[2] Delautre, Antoine, et al. "Verification of System coherency at early Architecture Design Stage." SDR Forum contribution to HAL RFI, Mainz, Germany. 2004.

[3] Hamid, Adnan, David Koogler, and Thomas L. Anderson. "Using portable stimulus to verify cache coherency in many-core soc." Design and Verification Conference. 2016.

[4] Hunter, A., Piziali, A., Ziv, A., Larson, K. and Hemmady, S., 2008, December. Ensuring functional closure of a multi-core soc through verification planning, implementation and execution. In 2008 Ninth International Workshop on Microprocessor Test and Verification (pp. 7-13). IEEE.

[5] Pamula, P., Gorthy, D.P., Ngangbam, P.S. and Alagarsamy, A., 2023. Verification of SoC Using Advanced Verification Methodology. Engineering Proceedings, 34(1), p.12.
[6] Abdennadher, S., 2021, June. Testing Inter-Chiplet communication interconnects in a disaggregated SoC design. In 2021 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS) (pp. 1-7). IEEE.
[7] UCIe Consortium, "UCIe Specification," *UCIe Consortium*, [online]. Available: https://ucie.org/specification.
[8] Arm Limited, "AMBA Coherent Hub Interface (CHI) Specification," *Arm Developer*, [online]. Available: https://developer.arm.com/documentation/ihi0050/latest/.