Harnessing the power of AMBA CHI: The Protocol Powering Modern CPU Clusters

A SignatureIP technical white paper on coherent interconnect design for multi-core SoCs

SignatureIP Engineering Team
Center of Excellence for Compute and I/O Subsystem IP

***Abstract*, Cache coherency is no longer an implementation detail. A modern compute SoC carries a dozen or more coherent initiators across CPU clusters, GPU compute units, neural accelerators, and high-throughput I/O. The fabric that connects them determines how much of the SoC's nominal performance the application actually sees. AMBA CHI, the AMBA 5 Coherent Hub Interface, is the protocol the industry has converged on for that fabric. CHI is a layered, channel-segregated protocol built around a directory-based home node, designed from the start for SoC-scale coherency at hundreds of agents. Broadcast snooping does not scale beyond roughly eight coherent agents on a shared bus, and even aggressive broadcast variants stop scaling somewhere in the low hundreds of cores. Direct Memory Transfer, Direct Cache Transfer, and Direct Write Transfer remove the home node from the data path on the common case. CHI raises its own system problems in return: I/O coherency without full CHI, AXI migration, DMA address translation, off-die coherent attach, fabric-level functional safety, and topology configuration. SignatureIP's portfolio addresses these problems as one integrated system: C-NOC as the CHI coherent core, NC-NOC as the non-coherent fabric, AXI2CHI and CHI2AXI as the migration and interop bridges, Proxy Cache and ATC as the I/O-coherency and translation enablers, PCIe Controller and CXL Controller (with the pre-integrated CXL Subsystem) as the off-die attach paths, Inoculator as the cloud-based topology generator, and Ethernet IP as a representative non-coherent master.**

***Index Terms*, AMBA CHI, network-on-chip, directory-based snoop filter, DMT, DCT, DWT, RN-F, HN-F, AXI to CHI bridge, IOMMU, topology generation, PCIe 6.x, CXL 3.x, Flex Bus, multi-core SoC, CPU cluster.**

I. EXECUTIVE SUMMARY

Cache coherency is no longer an implementation detail. A modern compute SoC carries a dozen or more coherent initiators across CPU clusters, GPU compute units, neural accelerators, and high-throughput I/O. The fabric that connects them determines how much of the SoC's nominal performance the application actually sees. AMBA CHI, the AMBA 5 Coherent Hub Interface, is the protocol the industry has converged on for that fabric. CHI is not an incremental extension of AXI. It is a layered, channel-segregated protocol built around a directory-based home node, designed from the start for SoC-scale coherency at hundreds of agents.

The promise of CHI is concrete. Broadcast snooping does not scale beyond roughly eight coherent agents on a shared bus, and even aggressive broadcast variants stop scaling somewhere in the low hundreds of cores [1]. Directory-based coherency in CHI removes that ceiling. Direct Memory Transfer (DMT), Direct Cache Transfer (DCT), and Direct Write Transfer (DWT) further remove the home node from the data path on the common case, which is the single largest latency saving available to a coherent fabric without redesigning the protocol [2].

CHI raises its own system problems in return. Most IP on a real SoC is AXI, not CHI. DMA engines need address translation. High-bandwidth accelerators and memory expansion need a coherent attach path that extends off-die over a standardized link. Functional safety has to be planned into the fabric from the start, even when the formal certification arrives later. Topology configuration at this scale is not a manual exercise. SignatureIP's portfolio addresses these problems as one integrated system: C-NOC as the CHI coherent core, NC-NOC as the non-coherent fabric, AXI2CHI and CHI2AXI as the migration and interop bridges, Proxy Cache and ATC as the I/O-coherency and translation enablers, PCIe Controller and CXL Controller (with the pre-integrated CXL Subsystem) as the off-die coherent and non-coherent attach paths, Inoculator as the cloud-based topology generator, and Ethernet IP as a representative non-coherent master integrated through the architecture end to end. Functional safety support across the portfolio is a SignatureIP roadmap item.

II. WHY CPU CLUSTERS NEED HARDWARE COHERENCY

A modern out-of-order CPU core retires instructions in well under a nanosecond. A DRAM access, even with the most aggressive memory controller and the most agreeable access pattern, takes tens of nanoseconds. Every production-grade CPU therefore carries caches, and every multi-core CPU therefore has to reconcile the views that those caches hold of shared data.

That reconciliation is the coherency problem. If a single agent holds a writable copy of a memory location, no other agent can be allowed to observe stale data past the point where the architecture says it should be gone. The mechanism that delivers the guarantee is a coherency protocol layered onto the interconnect. Each cache line carries a state. State transitions are driven by transactions on the fabric, with the protocol enforcing a single consistent order of writes to each location.

The cost of getting coherency wrong is visible in three places. The first is inter-core latency. On a dual-socket Intel Sandy Bridge system, inter-core communication latency ranges from approximately 36 ns when both cores share an L3 segment to approximately 137 ns when they sit in different sockets, a roughly 3.8x penalty driven entirely by topology [3]. The same effect appears inside a single die between clusters connected through different segments of the interconnect.

The second is memory bandwidth. Measured bandwidth utilization on a representative server-class system saturates near 53 percent of theoretical peak under heavy coherency traffic, against roughly 82 percent without it [3]. That gap is paid in snoop traffic, response traffic, and the serialization points imposed by the protocol.

The third is false sharing. When two agents write to different bytes within the same cache line, MESI forces the line to ping-pong between caches even though there is no data dependency. The application appears correct because the variables are logically independent. The silicon does useless work, and the workload trace gives no obvious cause [4].

These costs are not optional. They are the price of letting multiple cores hold writable copies of shared memory. The question for SoC architects is whether the fabric makes the price visible and bounded, or whether it lets the price grow without limit as agent count rises. CHI is the protocol designed to bound it.

III. INSIDE AMBA CHI

A. The Layered Architecture and the Channel Set

CHI separates protocol, transport, and link layers. The protocol layer defines transactions and coherency state machines. The transport layer defines how transactions are packetized and routed through a fabric. The link layer defines the flow control between adjacent nodes. The separation lets the fabric implementation optimize each layer independently and lets a topology change at the transport layer without disturbing the coherency model.

Transactions ride on four channels: REQ for requests, RSP for responses, DAT for data, and SNP for snoops. Each channel carries its own flow control. The home node can schedule snoops against snoop bandwidth while data flows on its own network, instead of serializing both classes of traffic on a single shared bus. CHI Issue E (E.a, then E.b) added replicated channels and a 12-bit TxnID that allows up to 1024 outstanding transactions per node, which is the working budget assumed by current production CPU clusters [7]. Fig. 1 summarizes the channel and node structure.

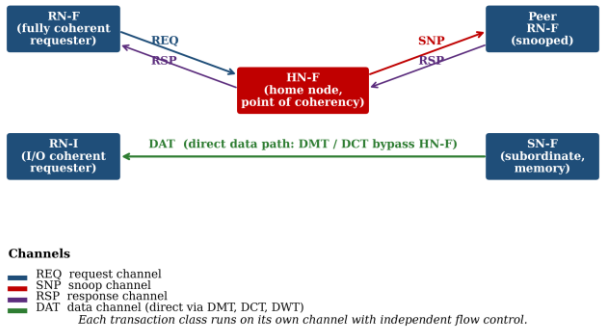

Fig. 1. CHI channel and node model. Four channels (REQ, RSP, DAT, SNP) connect the requester, home node, snooped peer, and subordinate node types.

B. Node Types and the Home Node as the Point of Coherency

CHI defines node types by their role in the protocol. RN-F is a fully coherent request node that holds cacheable copies of memory, is snooped, and responds to snoops. RN-I is an I/O coherent request node that can snoop into other caches but is not itself snooped, on the assumption that I/O agents do not retain modified data over long horizons. HN-F is the fully coherent home node, the point of coherency for an address range, which holds the directory state and arbitrates transactions to that range. HN-I is the equivalent for non-coherent memory. SN-F is a fully coherent subordinate node, typically memory. MN is a miscellaneous node.

The structural point is that coherency is enforced at the HN-F, not at the requester or the subordinate. A request enters the fabric, reaches its home node, and the home node decides what snoops to issue, what data to forward, and what state transitions to apply. The directory at the home node knows which agents currently hold the line and which do not.

C. Directory-Based Coherency and the Scaling Ceiling

Broadcast snooping answers the coherency question by sending every request to every cache. It is simple and works well at small agent counts. It does not scale. Snoop traffic grows as the square of the agent count, the response bandwidth required at each cache grows linearly, and the energy cost grows with both. Broadcast variants with probe filtering can stretch the practical limit to roughly 128 cores on some workloads, but they hit a ceiling and do not generalize [1].

Directory-based filtering at the home node answers the same question by keeping a sharer list per line. The home node knows where the line lives and only sends snoops to the agents that hold a copy. Directory storage scales at roughly the square root of agent count under common configurations, which keeps the cost tractable at hundreds of cores. Every

production-grade SoC-scale coherent fabric in current use is directory-based [5]. Fig. 2 shows the resulting flow with a DCT data delivery.

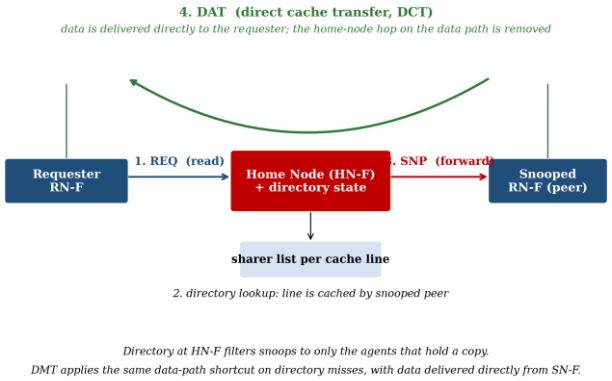

Fig. 2. Directory-based coherency flow with DCT. Control passes through the home node; data is delivered directly from the snooped peer to the requester.

D. DMT, DCT, and DWT, and What They Buy

A naive coherent read flows from the requester to the home node, then to the data source, then back through the home node, then to the requester. The control path needs the home node so the directory stays consistent. The data path does not.

Direct Memory Transfer applies on a directory miss. The subordinate node, typically memory, sends data directly to the requester instead of going through the home node first. Direct Cache Transfer applies on a directory hit. The home node tells the snooped cache to send the line directly to the requester. Direct Write Transfer applies the same principle on the write-back path. In each case the control path keeps coherency state correct while the data path is shortened to the minimum number of hops the topology allows [2]. The three direct-transfer features together are the single largest latency saving available to a CHI fabric without redesigning the protocol, and they are the features a coherent NoC implementation has to expose to be production-grade.

E. Issue History and What Changed When

Issue B (2017) was the first public CHI release. Issue D introduced the AMBA interface parity extension for functional safety, intended for automotive and other resilience-critical applications, and is the basis on which downstream parity-protected coherent fabrics build their detection coverage [6]. Issue E (E.a in 2020) added WriteZero, SnpQuery, MakeReadUnique, two-part Stash, DWT, replicated channels, the 12-bit TxnID, and Memory Tagging Extension support. Issue E.b (2021) was clarification, correction, and inclusive-terminology cleanup, not new features.

For SoCs being architected today, Rev E.b is the production specification, and it is the version this paper treats as in active rollout. Subsequent CHI issues exist and extend the protocol further, but the production-grade target for current designs is Rev E.b.

IV. THE SYSTEM PROBLEMS CHI RAISES

CHI solves the coherency scaling problem. It does not solve the SoC integration problem. Once a designer commits to a CHI coherent fabric, six practical issues land on the desk.

The first is I/O coherency without full CHI. Most non-CPU IP on a real SoC is not CHI native. DMA engines, network controllers, storage controllers, video and image engines, and most third-party accelerators expose AXI4 or AXI4-Stream. Forcing every such block to implement a full RN-F interface to participate in coherency is rarely justified, both because of design effort and because the access patterns do not warrant full coherency.

The second is the AXI to CHI migration path. A green-field SoC can design around CHI from the start. A practical product line cannot. There is always an existing portfolio of AXI subsystems, memory controllers, debug logic, and legacy IP that has to coexist with the new coherent fabric. A clean migration needs bidirectional bridges that handle AXI ordering semantics, transaction ID management, and the channel-to-packet semantic gap, not a forklift redesign.

The third is DMA address translation. DMA-capable masters on the fabric need to operate in virtualized environments, with isolation between guests, and with the same kind of translation services that the CPU MMU provides on the instruction side. Routing every DMA access through a centralized IOMMU adds latency that the rest of the coherent path was designed to remove.

The fourth is functional safety. Once a coherent fabric carries safety-critical traffic, every buffer, every control path, and every state machine in the fabric becomes part of the safety case. Protection mechanisms have to be designed into the fabric from the start, not added at the end.

A fifth issue is topology complexity. NC-NOC and C-NOC are configurable along many axes, including agent count, traffic profile, protocol mix, address map, QoS class, and power domains. Configuring them by hand for a real SoC is error prone, and the errors are expensive to find late in the schedule. A production flow needs a topology generator that produces a validated configuration from a system specification, not a configuration spreadsheet.

A sixth issue is off-die attach. CPU clusters and accelerators that sit on the CHI coherent fabric increasingly need to reach memory and accelerators on a different die, package, or socket. On the non-coherent side, PCIe 6.x is the production transport for high-bandwidth I/O and storage. On the coherent side, CXL 3.x layered on the Flex Bus carries CXL.io (PCIe-equivalent), CXL.cache (coherent device caches), and

CXL.mem (memory expansion), with device Types 1, 2, and 3. Once an SoC commits to CHI on-die, the off-die path needs a bridge that connects the CHI fabric to PCIe and to CXL without dropping back to a software-managed coherency model.

These six issues frame the rest of the paper. The next section walks through how the SignatureIP portfolio addresses each of them.

V. THE SIGNATUREIP ANSWER

The SignatureIP portfolio is one integrated system organized around a CHI coherent core. C-NOC is the coherent fabric. NC-NOC is the non-coherent fabric. Proxy Cache and ATC sit at the boundary between the two. AXI2CHI and CHI2AXI are the protocol bridges. PCIe Controller and CXL Controller (with the pre-integrated CXL Subsystem) extend the architecture off-die. Inoculator generates the topology. Ethernet IP is the representative I/O master that exercises the integration end to end. Fig. 3 shows the architecture in one SoC context.

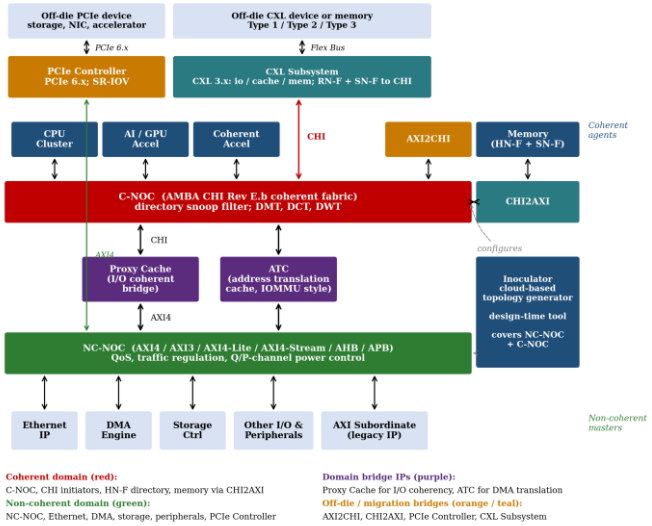

Fig. 3. SignatureIP portfolio in one SoC context. C-NOC carries CHI coherent traffic; NC-NOC carries AXI4 non-coherent traffic; Proxy Cache and ATC bridge the two domains; AXI2CHI and CHI2AXI are the bidirectional protocol bridges; PCIe Controller and CXL Subsystem extend the architecture off-die; Inoculator generates the topology at design time.

A. C-NOC: The Coherent Core

C-NOC is SignatureIP's coherent network-on-chip and implements AMBA CHI Rev E.b. It carries the coherent traffic between RN-F initiators (CPU clusters, GPU compute, AI accelerators) and HN-F home nodes, with directory-based snoop filtering at the home node and DMT, DCT, and DWT enabled on the data path. The home node tracks directory state and issues snoops. Data is delivered along the shortest topological path the fabric allows, between the source and the requester, bypassing the intermediate home-node hop that dominates latency on naive directory implementations.

The design target for C-NOC is high-bandwidth, low-latency multi-core CPU clusters and AI/ML accelerator domains. The CHI channel separation, the directory at the home node, and the direct-transfer features together let the fabric carry sustained coherent traffic without saturating any single link.

Chiplet and multi-die coherency extensions for C-NOC are on the SignatureIP roadmap as a future release. The architectural foundation in the current monolithic C-NOC, including directory placement and home-node distribution, is being designed against that path. Customers committing to C-NOC today are positioned to extend into chiplet topologies without reworking the coherency protocol.

B. NC-NOC: The Non-Coherent Fabric

NC-NOC is the high-performance non-coherent fabric for I/O, peripheral, and legacy IP integration. It carries AXI4, AXI3, AXI4-Lite, AXI4-Stream, AHB, and APB traffic, with configurable QoS classes, address interleaving, traffic regulation, performance counters, and Q-Channel and P-Channel power-domain control built in. The architectural value of NC-NOC is that it lets the coherent fabric stay free for traffic that actually requires coherency.

C. Proxy Cache: I/O Coherency Without Full CHI

Most SoCs carry DMA engines and accelerators that speak AXI, not CHI. Forcing every such agent to implement a full RN-F interface to join the coherent domain is rarely justified. Proxy Cache sits at the NC-NOC boundary and presents those agents to the coherent fabric as I/O-coherent participants. It absorbs their repeated reads of shared lines and answers from local state, which cuts snoop traffic into C-NOC and lowers the effective latency the agent sees. The designer gets I/O coherency without re-architecting the agent.

On the C-NOC side, Proxy Cache is a CHI agent with directory state tracked by the home node. On the NC-NOC side, it presents AXI4. The coherent fabric is not forced to maintain ordering around every I/O burst, and the non-coherent path is not forced to handle snoops.

D. ATC: Address Translation for DMA Masters

DMA-capable masters on NC-NOC need address translation in any system that runs under virtualization, isolates guests, or supports a process-level memory model on the I/O side. ATC is the IOMMU-style address translation cache that holds recently used translations close to the requester, so the common case does not pay the latency of a full page-walk on every transaction. The presence of ATC on the fabric is what makes DMA latency tolerable in a virtualized system without giving up the isolation guarantees that justify virtualization in the first place.

E. AXI2CHI: The Migration On-Ramp

AXI2CHI translates AXI4 transactions into CHI Rev E.b transactions, so AXI-native initiators can reach a CHI fabric without redesign. AXI uses a channel-based protocol with same-ID ordering and per-destination tracking. CHI uses a layered packet architecture with explicit transaction IDs and directory-driven coherency. AXI2CHI handles the mapping in both directions: ordering, ID management, and the conversion between AXI burst semantics and CHI transaction types. A customer migrating from an AXI-centric SoC to a CHI-centric one does not have to redesign their AXI subsystems.

F. CHI2AXI: Keeping AXI Subordinates in the System

Most existing memory controllers, debug logic, trace logic, and legacy subordinate IP is AXI. CHI2AXI translates CHI transactions to AXI4, so a CHI-native fabric can reach those subordinates without forcing them to migrate. Together with AXI2CHI, this gives the SoC team a complete interop story: AXI initiators can drive a CHI fabric, and CHI initiators can drive AXI subordinates, without dropping back to a bus.

G. Ethernet IP: A Non-Coherent Master in Practice

SignatureIP Ethernet IP is the representative non-coherent master that exercises the NC-NOC plus Proxy Cache plus C-NOC pattern end to end. The Ethernet block runs at packet-rate throughput. It needs to read coherent memory for transmit data and write coherent memory for received data. Software cache maintenance at that rate is not viable: every TX burst would need a clean, every RX burst would need an invalidate, and any missed maintenance call is silent data corruption.

Proxy Cache gives the Ethernet block a coherent memory view without software flushes. NC-NOC carries the Ethernet traffic on its own network so it does not contend with coherent traffic on C-NOC.

H. Inoculator: Topology Generation as an Engineered Step

NC-NOC and C-NOC are configurable along enough axes that hand-configuration at production scale is not viable. Inoculator is SignatureIP's cloud-based topology generation tool. Today it covers both NC-NOC and C-NOC: it takes a system specification (agent count, traffic profiles, protocol mix, address map, QoS classes) and produces a validated NC-NOC and C-NOC topology configuration. Coverage for the rest of the SignatureIP portfolio (Proxy Cache, ATC, AXI2CHI, CHI2AXI, Ethernet IP, PCIe Controller, CXL Controller, CXL Subsystem) is on the Inoculator roadmap and will be added incrementally.

Inoculator is a productivity and correctness tool. It is not part of the ECC, parity, or lockstep path, and it is not a functional safety, fault injection, or error-protection block.

I. PCIe Controller: High-Bandwidth Non-Coherent Off-Die Transport

PCIe is the production transport for high-bandwidth off-die I/O, storage, and accelerator attach, and it is the physical and link substrate that CXL builds on. SignatureIP's PCIe Controller is a PCIe 6.x controller covering the full Transaction Layer, Data Link Layer, and MAC. It runs at 64 GT/s per lane on PAM4 with link widths from x1 to x16 and is backward compatible to Gen1. At Gen6 it uses 256-byte FLIT framing with FEC, with the standard 128b/130b encoding on Gen3 through Gen5. Reliability comes from FEC, CRC, ACK/NAK replay, and Advanced Error Reporting. The application interface is 512-bit AXI4 with an APB configuration port. The controller can be configured as Root Complex or Endpoint, supports up to four PFs and 64 VFs per PF (SR-IOV with ARI), and exposes a PIPE-compatible PHY interface.

In the CHI-centric architecture the PCIe Controller sits on NC-NOC alongside other non-coherent masters, with translation through ATC and coherent memory access through Proxy Cache where required. Two delivery tracks are supported: an ASIC track (RTL with SDC, UPF, lint and CDC, conformance plan) and an FPGA track (AMD VP1902 reference design). FuSa support for the PCIe Controller is on the SignatureIP roadmap.

J. CXL Controller: Extending Coherency Off-Die

CXL is the off-die extension of the coherency story. CXL.io carries PCIe-equivalent transactions, CXL.cache carries coherent traffic from a device cache to a host fabric, and CXL.mem carries memory-expansion traffic. The SignatureIP CXL Controller covers CXL 3.x across all three protocols and supports device Types 1 (compute-only with device cache), 2 (compute with device-attached memory and HDM-DB), and 3 (memory expansion). It runs on a Flex Bus PHY up to 64 GT/s (Gen6) at x2 and x4. Backward compatibility extends to CXL 2.0 and 1.1.

The architecturally important point for a CHI paper is that the CXL Controller is the bridge between a CHI-coherent fabric and the Flex Bus. It appears to C-NOC as both an RN-F and an SN-F: an RN-F when the off-die device is caching host memory through CXL.cache, an SN-F when host transactions are addressing device-attached memory through CXL.mem. A bias-tracker manages the flush path for Type 2 HDM-DB devices. Application-side interfaces are 512-bit AXI4 for CXL.io and dual APB. Built-in address translation includes an ATU with 20 inbound and 20 outbound regions, a local ATC, and PCIe ATS over DTI. Up to four PFs and 64 VFs per PF support PASID and SR-IOV. FuSa support is on the SignatureIP roadmap.

K. CXL Subsystem: Pre-Integrated CXL Attach

The CXL Subsystem (sig_cxl_system_top) is a drop-in subsystem that packages the CXL Controller with the rest of the components a customer would otherwise have to wire together by hand: AXI-SFI bridge, CPI-CHI bridge, ATU, ATC, and the Flex Bus PHY interface. Two independent instances (cxl_0 and cxl_1) can be configured as x4, x2+x2, x2+x1, or x1+x1, with six wrapper variants covering the

common combinations. Host or device operation is a single parameter (cxl_updn_comp); the DCOH engine manages bias state. Interfaces are 512-bit AXI4, CHI, dual APB, MSI/MSI-X, and DTI/ATS.

The CXL Subsystem is designed to integrate with C-NOC for heterogeneous-platform and chip-to-chip scaling. It is the productized form of the CXL attach story: a single block that a designer can drop in and configure rather than building the bridge integration from individual IPs. FuSa support is on the SignatureIP roadmap.

VI. CONCLUSION AND ROADMAP NOTE

AMBA CHI is the coherency protocol for SoCs that have moved beyond what a shared bus can carry. Its layered architecture, channel separation, directory-based home node, and direct-transfer features address the scaling, latency, and bandwidth costs that broadcast-based fabrics cannot. Rev E.b is the production specification today and is the version this paper targets throughout.

CHI shifts work from the protocol to the SoC integration. I/O coherency, AXI migration, DMA translation, off-die accelerator and memory attach, functional safety in the fabric, and topology configuration all become first-order architectural concerns. SignatureIP's portfolio addresses these concerns as one system: C-NOC as the coherent core, NC-NOC as the non-coherent fabric, Proxy Cache and ATC at the domain boundary, AXI2CHI and CHI2AXI as the bidirectional protocol bridges, PCIe Controller as the high-bandwidth non-coherent off-die transport, CXL Controller and the pre-integrated CXL Subsystem as the off-die coherent attach extending the CHI fabric onto the Flex Bus, Ethernet IP as the representative non-coherent master, and Inoculator as the cloud-based topology generator covering NC-NOC and C-NOC today, with the rest of the portfolio on its roadmap.

Two capabilities are explicitly on the SignatureIP roadmap rather than in the current product. Chiplet and multi-die coherency extensions for C-NOC are a future release. Functional-safety support across the portfolio (SECDED ECC, parity, safety status registers, ISO 26262 certification artifacts) is also a roadmap item.

For SoC and CPU-cluster architects evaluating coherent interconnect IP, the criteria are concrete: CHI Rev E.b conformance with DMT, DCT, and DWT enabled, a directory-based home node sized for the agent count, an I/O coherency path that does not force flush traffic onto the coherent fabric, bidirectional AXI bridges that preserve ordering, address translation close to the DMA masters, an off-die coherent attach path that bridges CHI to CXL without dropping back to software-managed coherency, and a topology generator that produces a validated configuration from a system specification. SignatureIP's portfolio is designed to meet those criteria, with chiplet and multi-die coherency and full functional-safety support on the roadmap.

REFERENCES

[1] EECG University of Toronto, snoop-filtering reference. Broadcast-based probe filtering provides reasonable performance up to approximately 128 cores on some benchmarks but is not generally scalable; directory-based filtering is the scalable path. [Online]. Available: https://www.eecg.utoronto.ca/~moshovos/filter/doku.php?id=source-based_snoop_filters

[2] Arm Limited, *Data Memory Transfer, Direct Cache Transfer, PrefetchTgt, and Direct Write Transfer*, Arm Developer Documentation (102407). [Online]. Available: https://developer.arm.com/documentation/102407/latest/

[3] R. K. Somarouthu, "Demystifying Cache Coherency in Modern Multiprocessor Systems," *European Journal of Computer Science and Information Technology*, vol. 13, no. 37, 2025.

[4] T. Liu et al., "Leveraging Cache Coherence to Detect and Repair False Sharing On-the-fly," in *Proc. IEEE/ACM MICRO*, 2024. [Online]. Available: https://www.cse.iitk.ac.in/users/swarnendu/files/papers/micro24.pdf

[5] Arteris, *Snoop Filters in Coherent NoC Systems: Types, Trade-Offs, and Directory-Based Design*. [Online]. Available: https://www.arteris.com/learn/snoop-filter/

[6] Arm Limited, *AMBA moves forward with major revisions to AXI and CHI specifications*, Arm Architectures and Processors blog. On Issue D introducing the AMBA interface parity extension for resilience and functional-safety applications such as automotive. [Online]. Available: https://developer.arm.com/community/arm-community-blogs/b/architectures-and-processors-blog/posts/updates-to-amba-axi-and-chi-specifications

[7] Arm Limited, *AMBA CHI Architecture Specification* (IHI0050), latest revision. [Online]. Available: https://developer.arm.com/documentation/ihi0050/latest/

About SignatureIP

SignatureIP is a provider of semiconductor IP solutions for chiplet and multi-die design across today's advanced SoC architectures. SignatureIP's network-on-chip (NoC) interconnect IP and high-speed interface IP — spanning PCIe Gen 6, CXL 3.0, and coherent and non-coherent NoC — enable system architects to build higher-performance, lower-power designs with faster time to silicon. SignatureIP's IP portfolio allows its customers to focus on product differentiation while SignatureIP handles the complexity of on-chip and die-to-die communication. Learn more at www.signatureip.ai.