Improper Request Propagation before Data Reception in Write Transactions in a Bus Architecture

Francesco Restuccia

UC San Diego

High-level summary of the proposed weakness

The product uses a multi-manager shared bus architecture with an on-chip interconnect that speculatively propagates a request for a write transaction, but it does not ensure that the remaining steps required by the manager to complete the request are fulfilled in a timely manner.

Key idea: speculative forwarding of write address before data arrival lead to DoS/delays in the access of shared resources

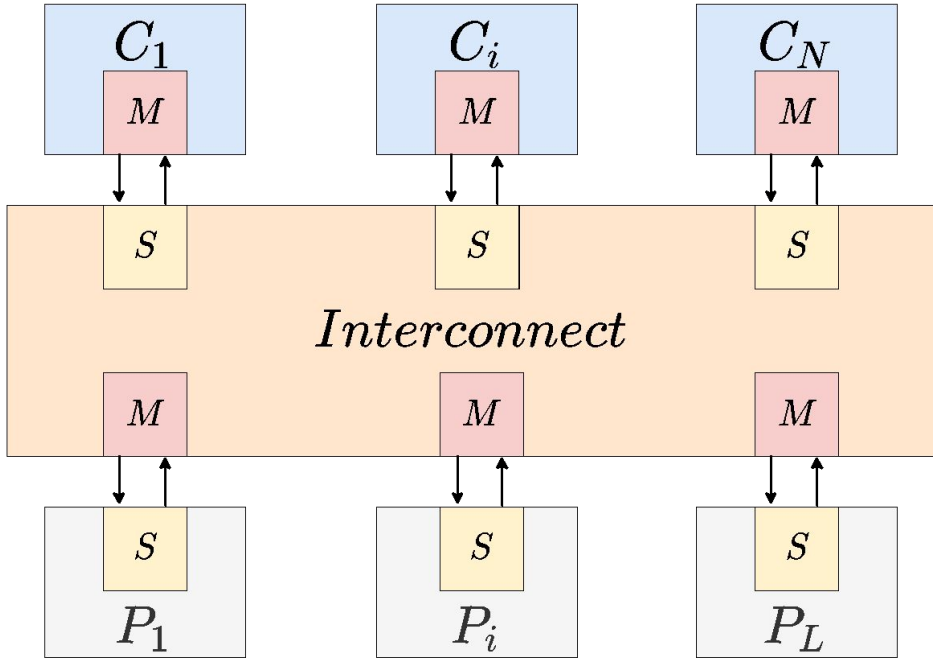
- Shared port to resource becomes reserved; stalled data phase blocks others
- Turns local (IP) misbehavior into system-wide availability issue (DoS)
- Recovery might require system interconnect reset if not handled properly

Reference architecture

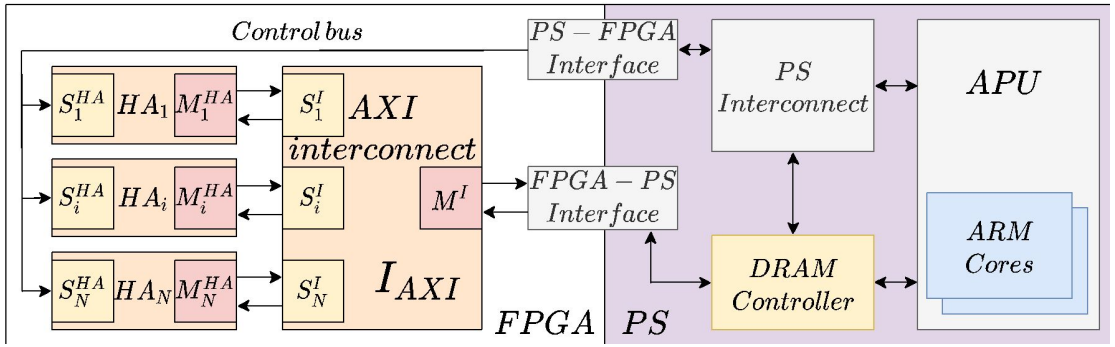
Managers/Controllers (active): processors, DMAs, hardware accelerators, etc.

Interconnect: arbitrates the access of controllers to peripherals

Peripheral (passive): shared, e.g., memories, I/Os, etc.



Anatomy of a typical heterogeneous platform (simplified)

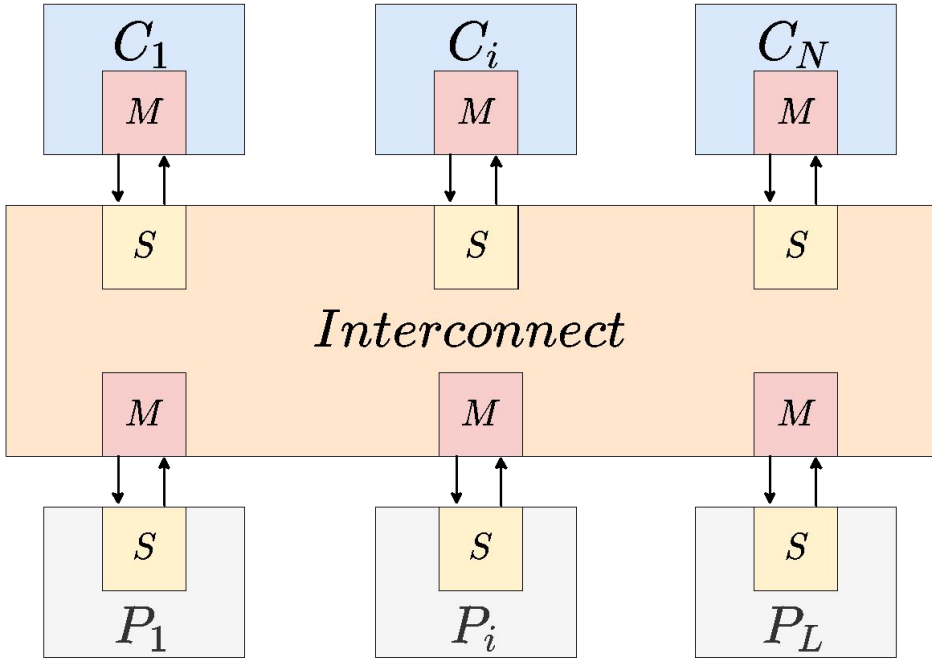


A sample multi-manager architecture deployed on an FPGA SoC platform

Minimal Threat Model and Assumptions

Multi-manager SoC interconnect (e.g., AXI) with shared resources.

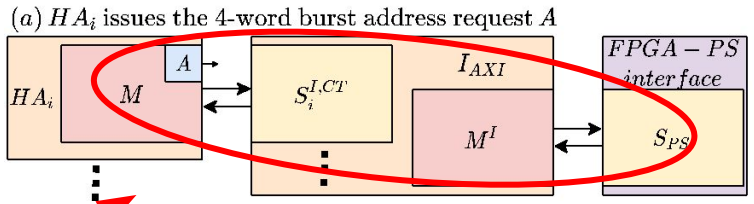
- Interconnect assumes cooperative managers and timely data provision.
- Interconnect is trusted
- Peripherals are trusted
- Managers might not be trusted
  - Malicious/faulty/misbehaving
  - Third-party IPs

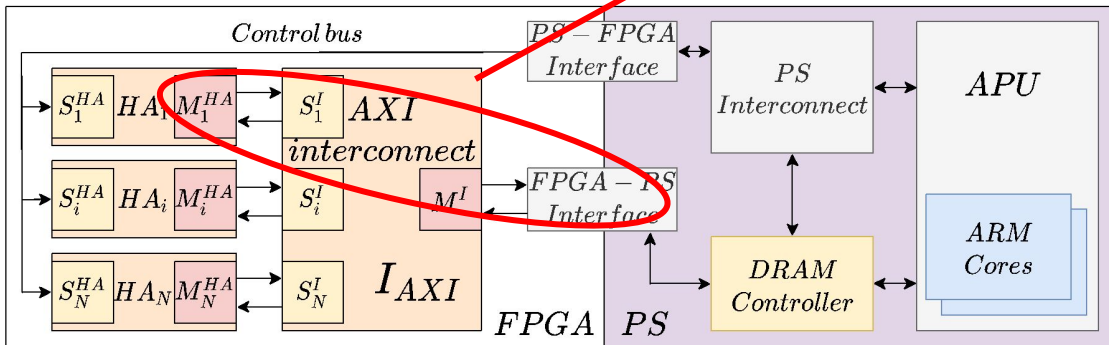


Example

The interconnect receives a write request from a manager initiator (e.g., sending the header to the interconnect for writing N words of data, with N ranging from 1 to the maximum burst length allowed).

Sample 4-word write transaction - CT switching

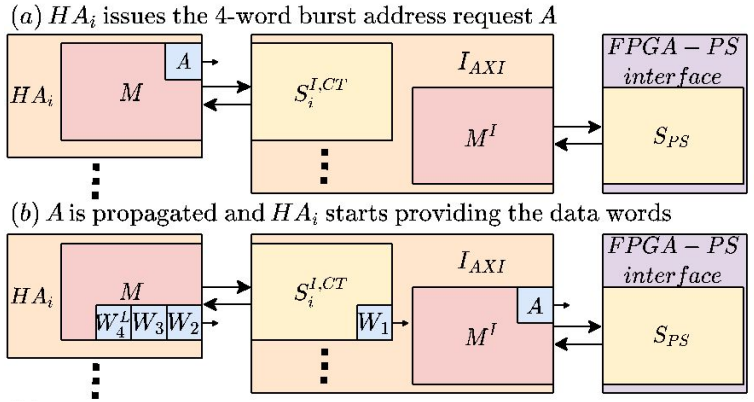


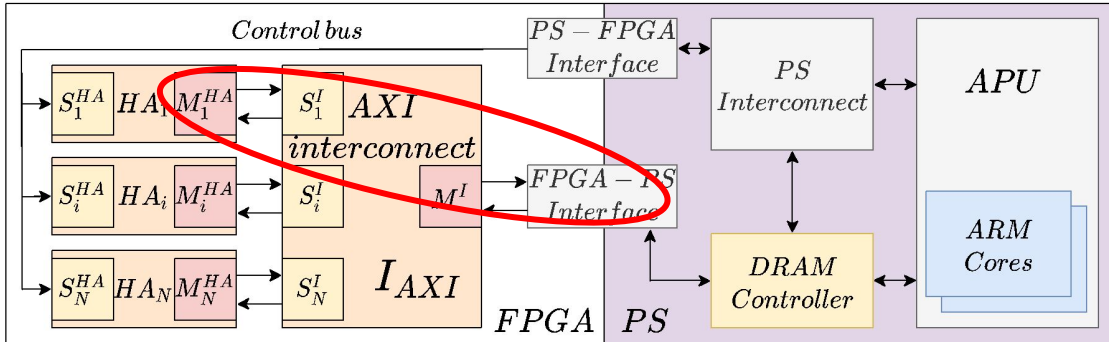


Example

The interconnect then speculatively propagates the request to the target peripheral through the shared bus before the corresponding data are received at the interconnect.

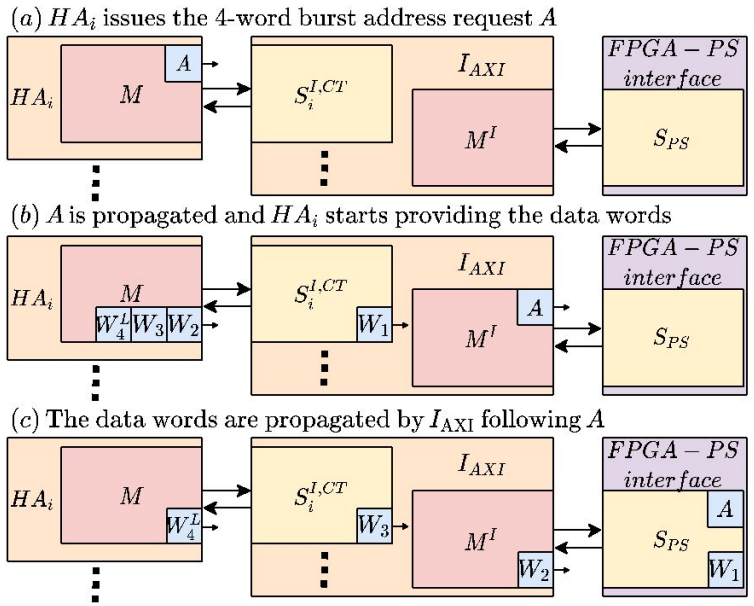
Sample 4-word write transaction - CT switching



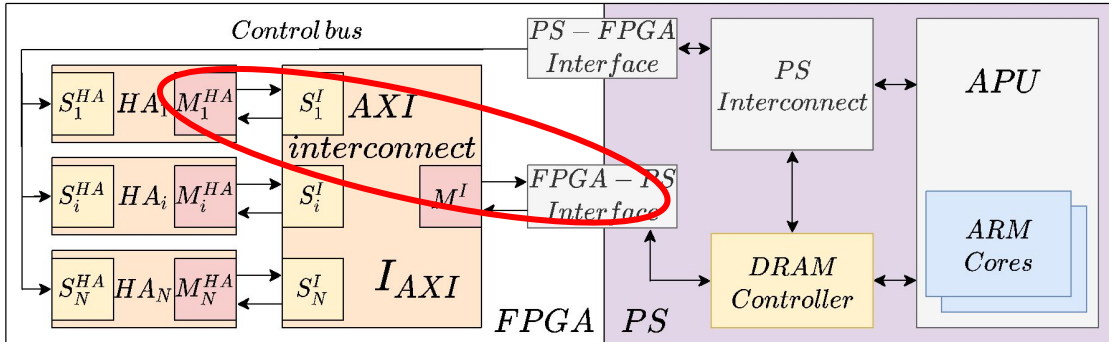


Example

Sample 4-word write transaction - CT switching



This speculatively reserves the bus to the manager transaction issuer, which is left free to delay its data provisioning phase. This reserves the whole bus path between the initiator and the target peripheral until all of the words associated with the request are provided by the manager.



Example

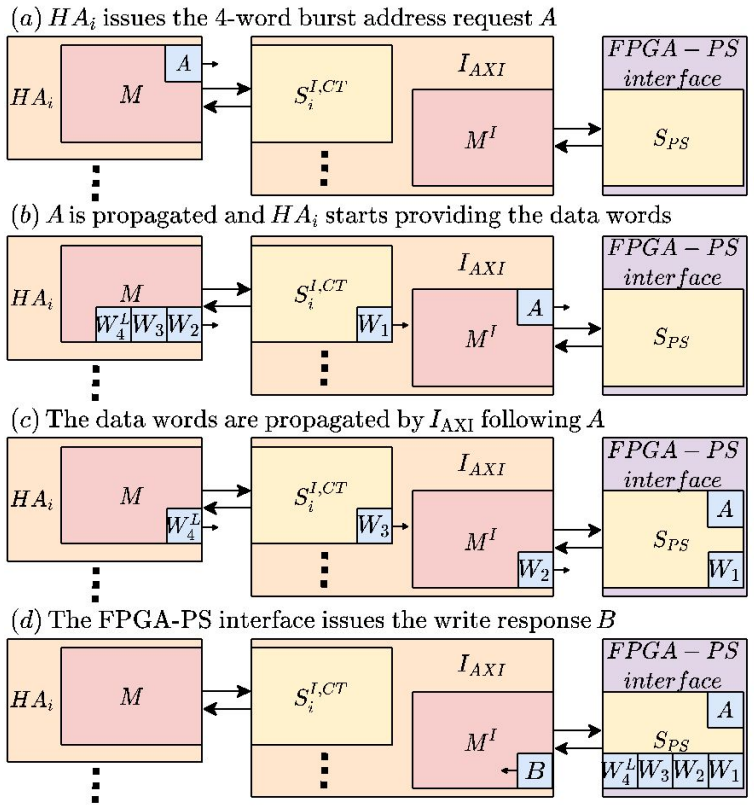
How stalls are generated?

Typical switching method: **Cut-through**

+

Protocol definition: write data must be propagated in the same order of the request for transactions

Sample 4-word write transaction - CT switching



Example

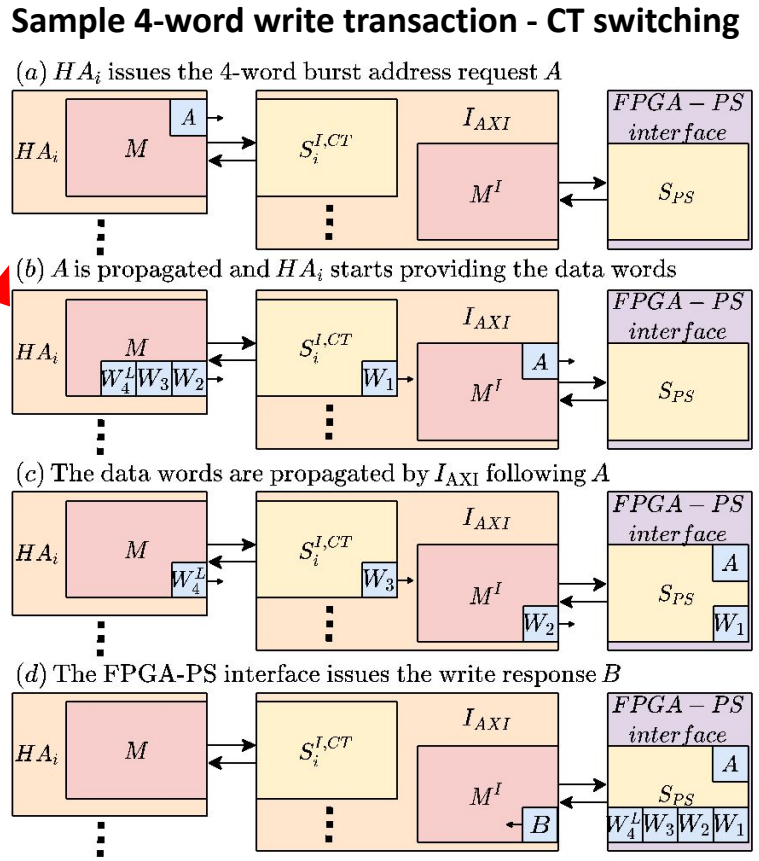
How stalls are generated?

Typical switching method: **Cut-through**

+

Protocol definition: write data must be propagated in the same order of the request for transactions

(b) The hardware modules are trusted to provide (quickly) all of the data words after submitting a request for transaction



Example

How stalls are generated?

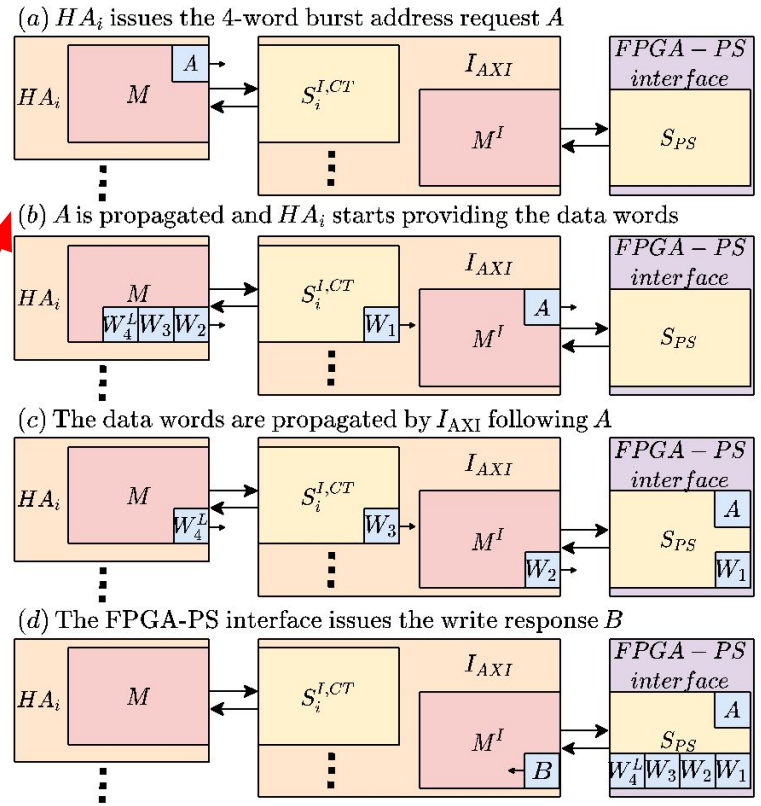
Typical switching method: **Cut-through**

+

Protocol definition: write data must be propagated in the same order of the request for transactions

(b) The hardware modules are trusted to provide (quickly) all of the data words after submitting a request for transaction

Sample 4-word write transaction - CT switching



Example: AMBA AXI specification

A5.2.2 Write data ordering

A Manager must issue write data in the same order that it issues the transaction addresses.

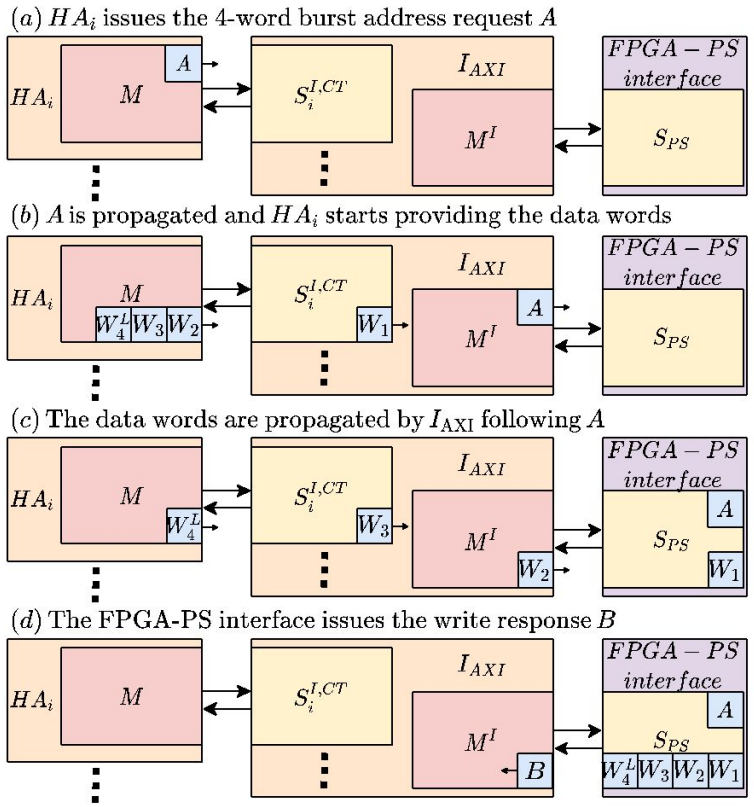
An interconnect that combines write transactions from different Managers must ensure that it forwards the write data in address order.

The interleaving of write data with different IDs was permitted in AXI3, but is deprecated in AXI4 and later. See the AMBA AXI and ACE Protocol Specification issue F specification for more details on write data interleaving.

Summary of Demonstrative Example

Cut-Through Write

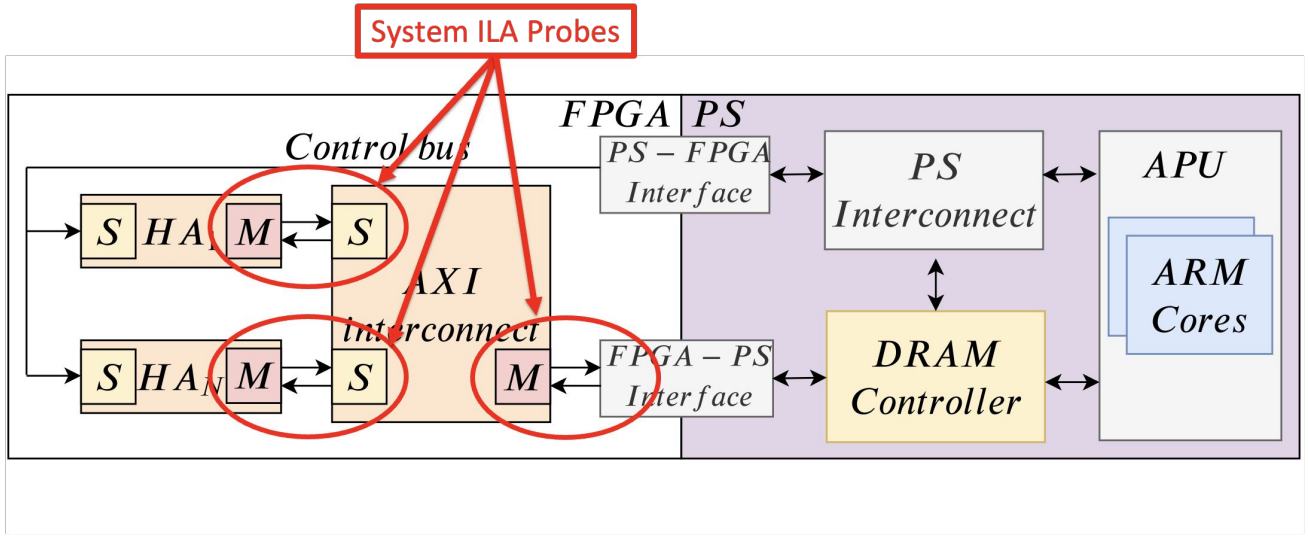
- Manager issues write address A (burst).
- Interconnect forwards A to shared port before data W arrive.
- Shared port becomes booked for A; other managers blocked.
- Manager delays W → prolonged stall; bus/path occupied.
- AXI lacks a native abort for such stalled transactions.



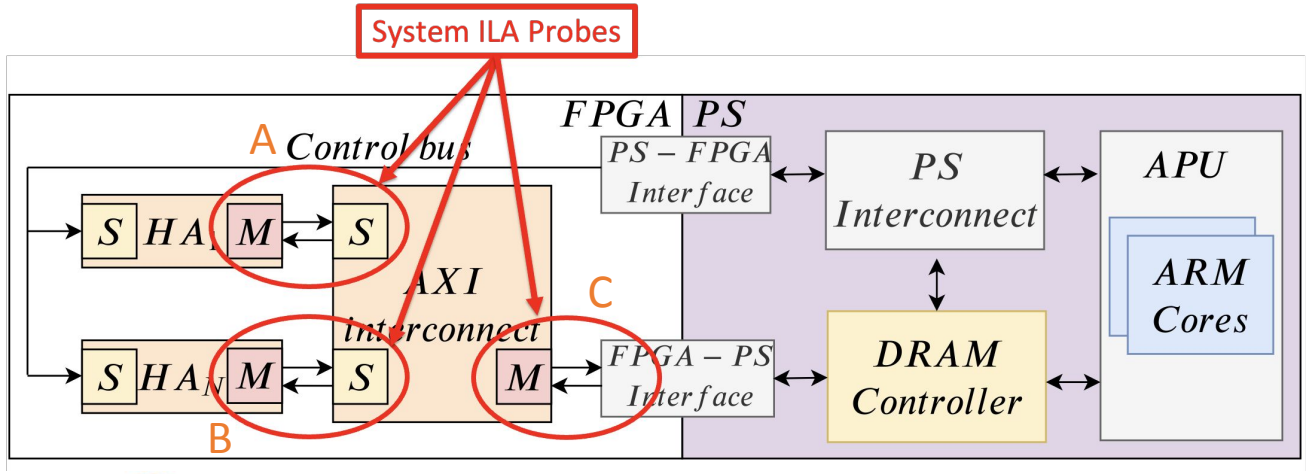
Where a DoS is generated

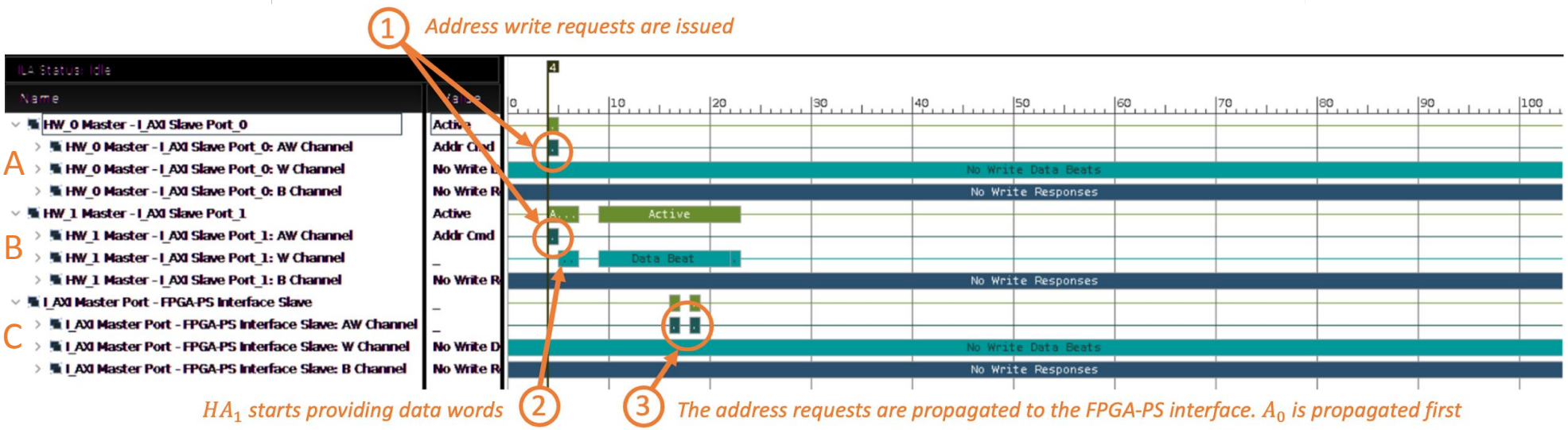
- Speculative reservation without bounded data timing.
- Single misbehaving manager blocks shared link for all.
- Stalls propagate through cut-through paths.
- Lack of protocol-level abort of bus transactions

Demonstration on COTS FPGA

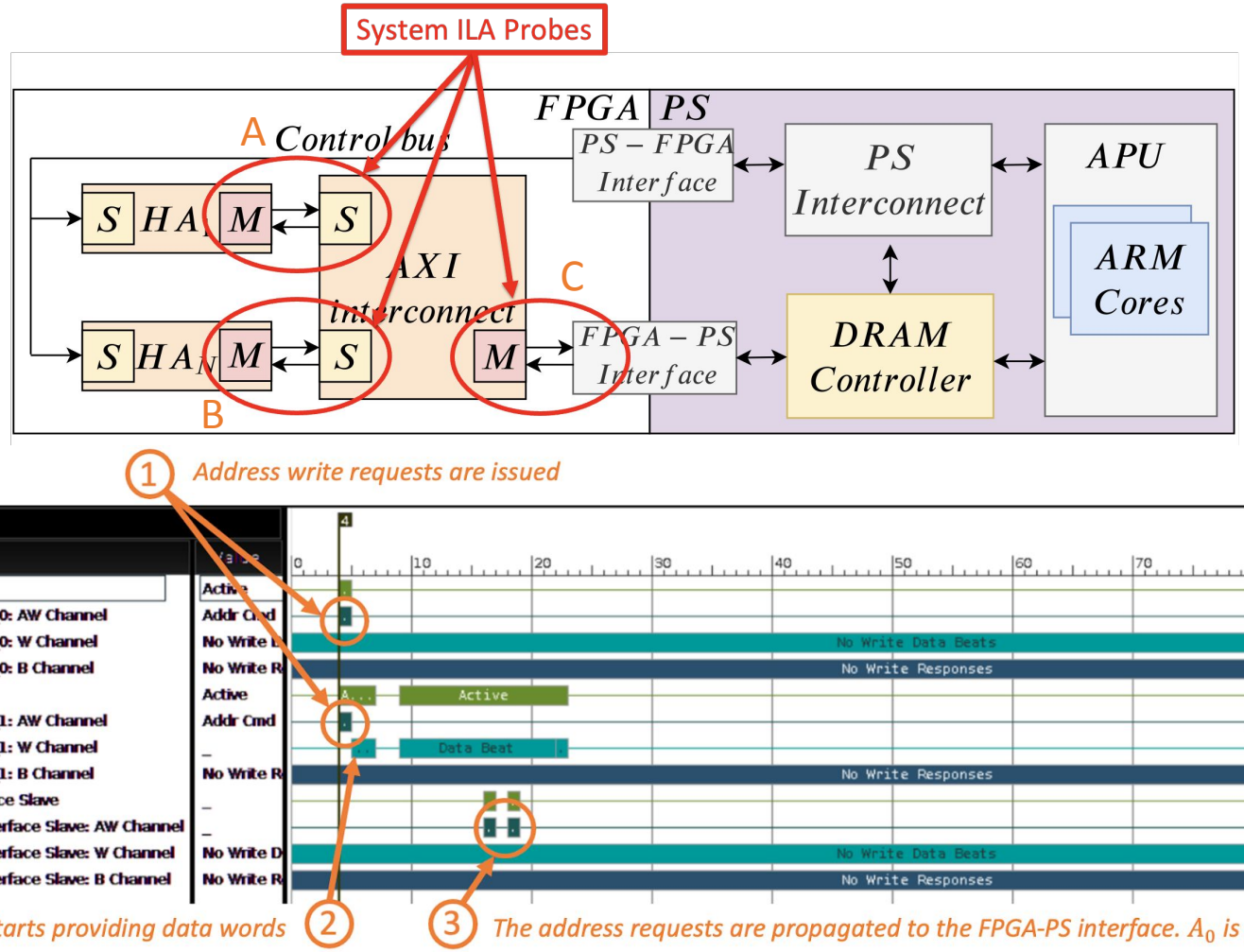


Demonstration on COTS FPGA





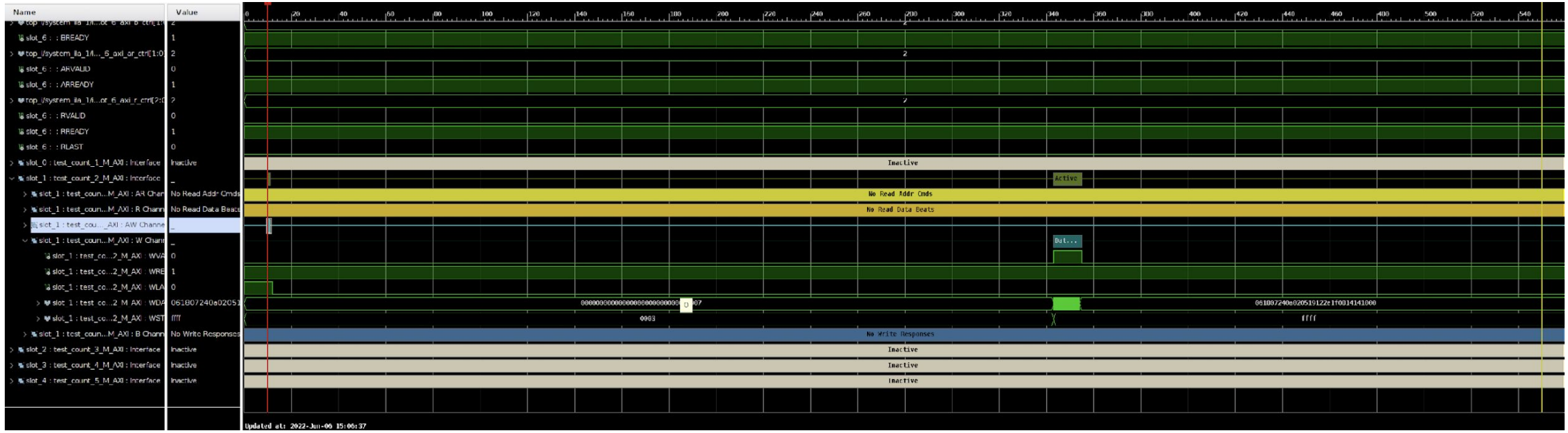
Demonstration on COTS FPGA



One misbehaving module cause denial of service in accessing the shared DRAM memory

Demonstration - commercial manager IP

Real waveform track collected from an execution of the Xilinx DPU core (v 3.3) on a ZYNQ Ultrascale+ platform (ZCU102)



Demonstration - commercial manager IP

The DPU implements this behaviour for speculative bus access. This behaviour is fully compliant with the protocol standard

Demonstration - commercial manager IP

The DPU implements this behaviour for speculative bus access. This behaviour is fully compliant with the protocol standard

If the DPU shares the bus with other hardware modules, these last cannot access the shared peripheral while the bus is stalled by the DPU core

Modes of Introduction

Where it gets introduced

- Phase: Architecture & Design.

  Unsafe switching (e.g., cut-through) with multi-channel standards.

  Trust in manager modules; lack of bounded timing for data after address.

- Risk increases with third-party interconnect IP when switching policy is undocumented.

Platforms & Technologies dependencies

- Languages/OS: Not language- or OS-specific.
- Architectures: SoC; examples include ARM-based interconnects.
- Technology: Bus/Interface, Communication, Network-on-Chip.
- Class: System-on-Chip; prevalence depends on interconnect choice.

AXI is a representative example, not exclusive.

Impact

- Scope: Availability.
- Denial-of-Service on shared resources (memory, I/O).
- System-wide delays; missed deadlines in real-time domains.
- Possible need for interconnect reset; degraded reliability.

Implications particularly in critical systems (automotive, avionics, space, medical, robotics, etc.).

Detection Methods

Architecture/Design Review

- Switching policy (cut-through).
- Availability protection on shared resources.
- Availability protection on the untrusted managers.

Verification

- Properties to prove bounded stalls to shared resources.
- Temporal properties for progress/completion.

Simulation / Emulation

- Inject delayed/withheld data from manager side.
- Observe stall propagation in simulation.
- Some VIP may not trigger this (as it is compliant with the standard)

Mitigations & Trade-offs

Store-and-Forward

- Legacy solution: Buffer all data before booking shared resource.
- Prevents propagation of upstream stalls to the rest of the system.
- Trade-offs: added considerable latency and area.

Cut-and-Forward

- Split bursts into bounded chunks (C).
- Book shared port per chunk only when data is already buffered.
- Balances latency/area vs availability; configurable C.
- Tradeoff: shorter transactions - might impact average performance

Monitor & Recovery

- Detect long stalls at the manager or peripheral level.
- Abort/complete stalled transactions according to the standard when stalls are detected.
- Restore interconnect to usable state.
- Tradeoff: additional logic + analysis for configuration

Observed Examples & Relations

Observed Examples

- Observed in commercial AXI interconnect IPs
- Cut-through used as default in commercial interconnects (performance)

Related Weaknesses

- ChildOf: CWE-841 (per submission).
- Cross-reference with availability/DoS-centric hardware CWEs.

Recommendations & Checklists

Guidance for Architects & Integrators

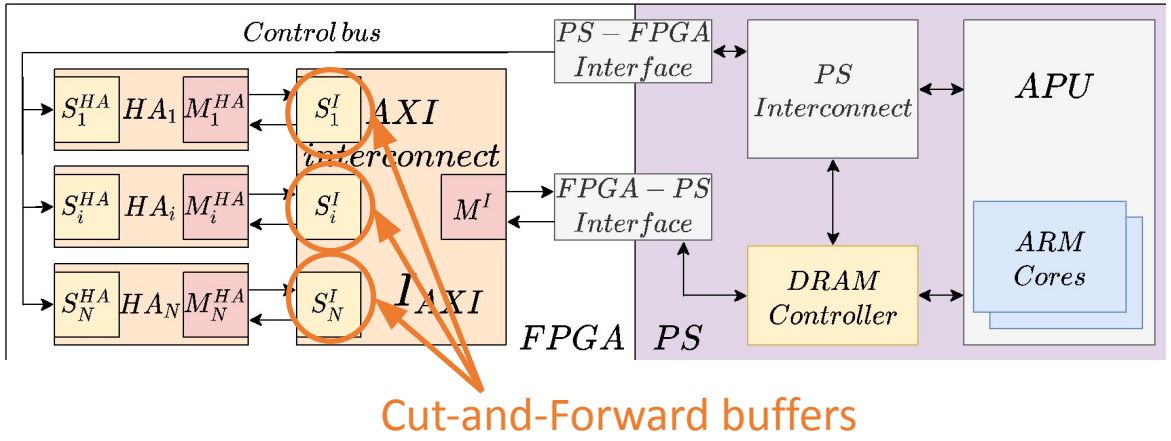
- Review switching policy from interconnect IP vendors.
- Enforce bounded timing from address to data on writes.
- Prefer S&F or C&F in critical scenarios or add monitors for recovery pathways.
- Verify IPs with properties and targeted stall tests.

The Cut and Forward (C&F) switching

What is Cut and Forward?

Novel **switching methodology** to be integrated in an AXI interconnect

AXI compliant – **fully transparent** to the **network** and resources (i.e., managers and peripherals)



Why Cut and Forward?

Can combine **low-latency** transactions **propagation** and **low area footprint** (Cut-Through) with **safe** and **secure** bus communications **for any workload** (Store-and-Forward)

Configurable on a target application through the C&F parameter C

How Cut and Forward works

C&F buffers are configured with the parameter C

C&F **waits for C data words** before **propagating** any address request (i.e., booking the shared bus)

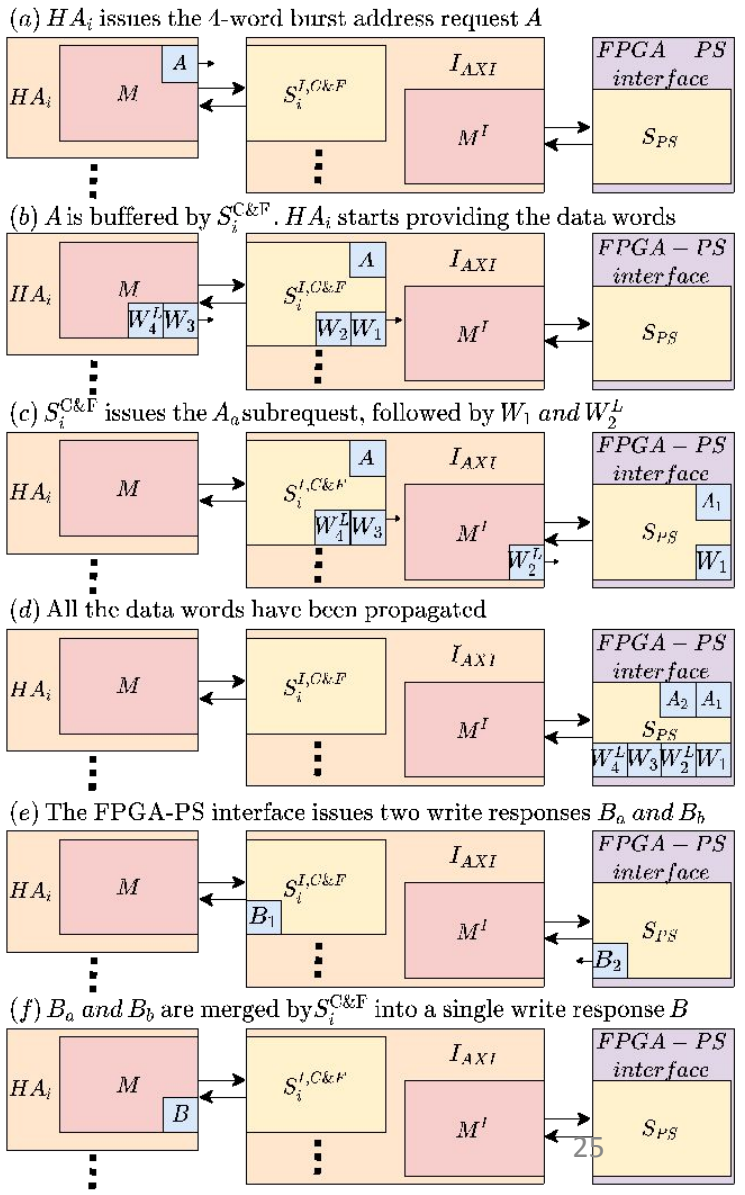
When C words **have been buffered**, C&F **generates** and **sends** a **subrequest for C words** (i.e., the bus is booked only for C words)

The **buffered C words** are following **propagated** by C&F with **no stalls**

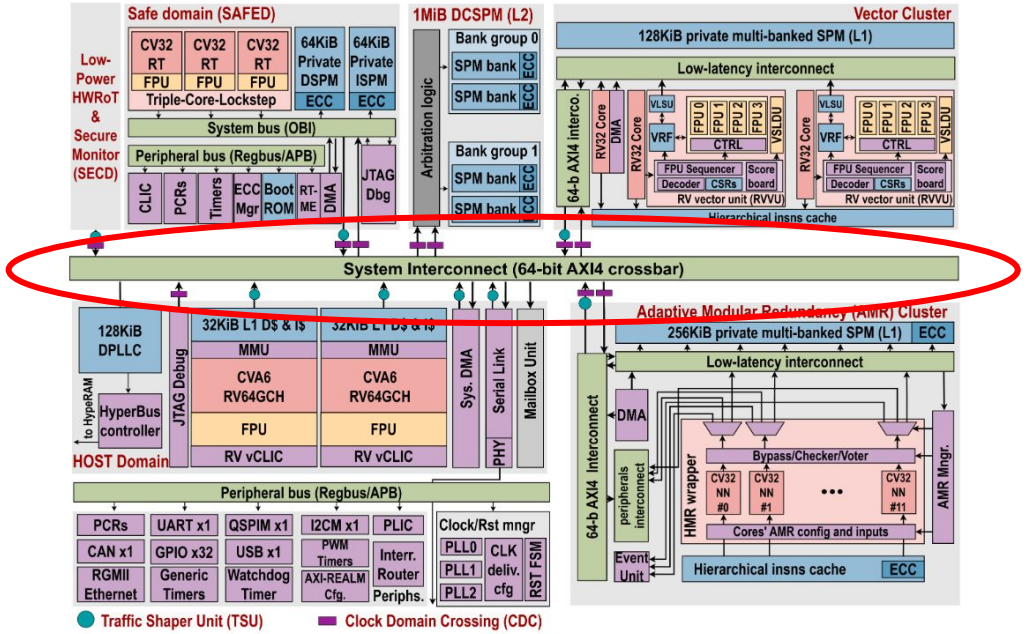
In parallel with the **propagation** of the C words, the **following** C words can be sampled by C&F. This introduce a **pipelining effect between data buffering (validation) and data propagation**

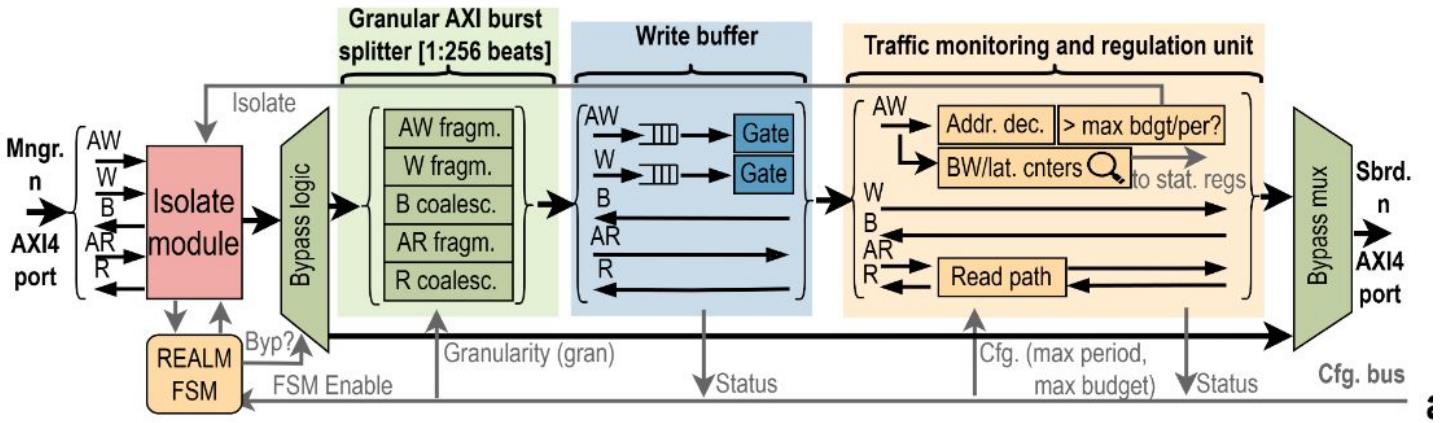
The **B responses** are **merged** by C&F – complain with the AXI standard

Sample 4-word write transaction – C&F switching configured with C=2



The Carfield SoC





A. Garofalo *et al*., "A Reliable, Time-Predictable Heterogeneous SoC for AI-Enhanced Mixed-Criticality Edge Applications," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, doi: 10.1109/TCSII.2025.3591225. keywords: {Hardware;Vectors;Artificial intelligence;Software;Bandwidth;Logic;Energy efficiency;Electric breakdown;Safety;Runtime;Heterogeneous mixed-critical SoC;time-predictable SoC;hardware acceleration;fault tolerant hardware},

References

F. Restuccia and R. Kastner, "Cut and Forward: Safe and Secure Communication for FPGA System on Chips," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41.

F. Restuccia, A. Biondi, M. Marinoni and G. Buttazzo, "Safely Preventing Unbounded Delays During Bus Transactions in FPGA-based SoC," 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Fayetteville, AR, USA

Meza, Andres, et al. "Safety verification of third-party hardware modules via information flow tracking." Proc. 1st Real-Time Intell. Edge Comput. Workshop (RAGE) Co-Located 59th Design Autom. Conf.(DAC).

T. Benz et al., "AXI-REALM: Safe, Modular and Lightweight Traffic Monitoring and Regulation for Heterogeneous Mixed-Criticality Systems," in IEEE Transactions on Computers.

A. Garofalo et al., "A Reliable, Time-Predictable Heterogeneous SoC for AI-Enhanced Mixed-Criticality Edge Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs.

frestuccia@ucsd.edu