FireSim Updates – FireAxe Multi FPGA FireSim Support

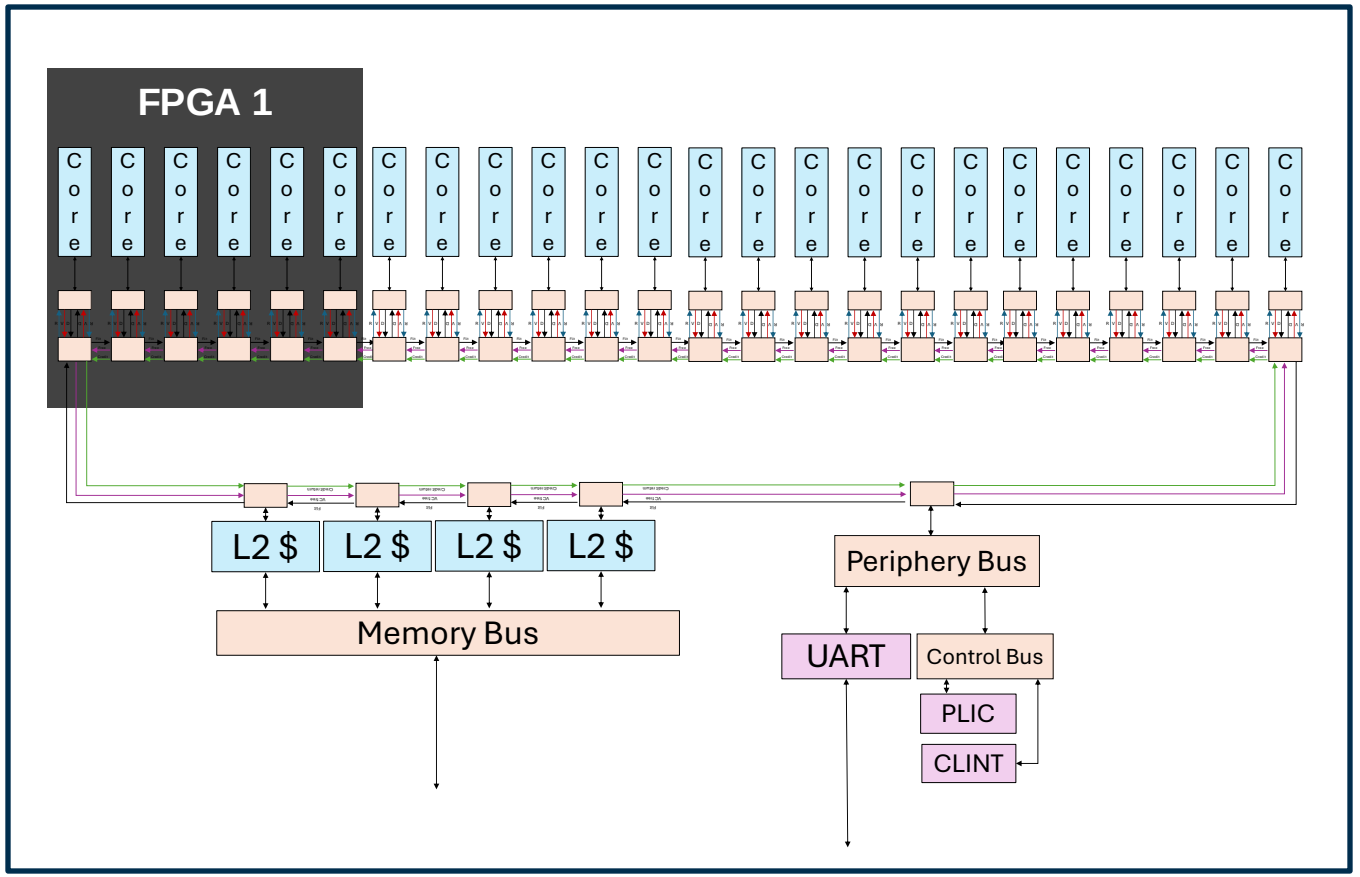
https://fires.im


@firesimproject

Speaker: Joonho Whangbo

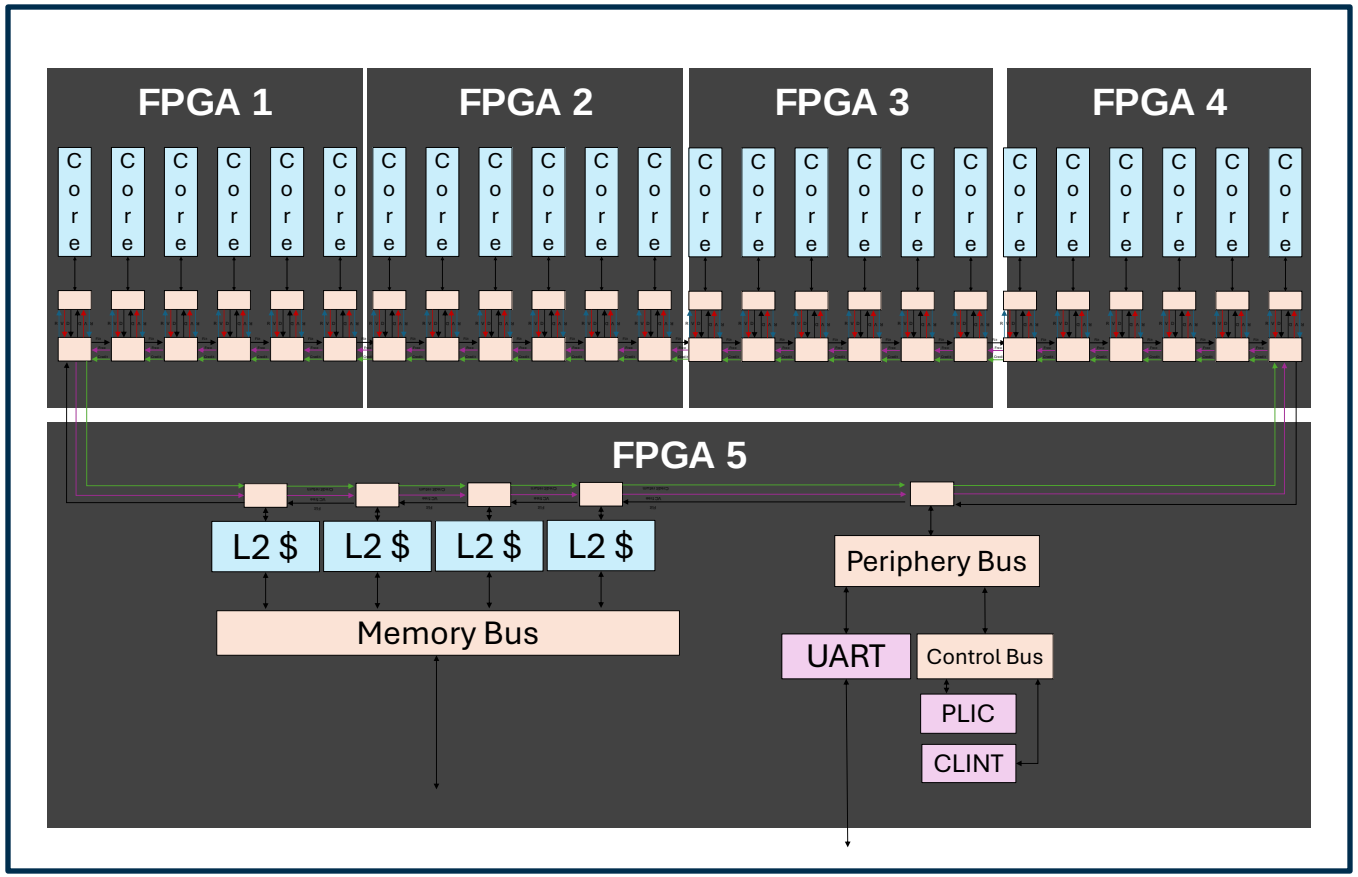

Berkeley Architecture Research

Large designs don’t fit on one FPGA



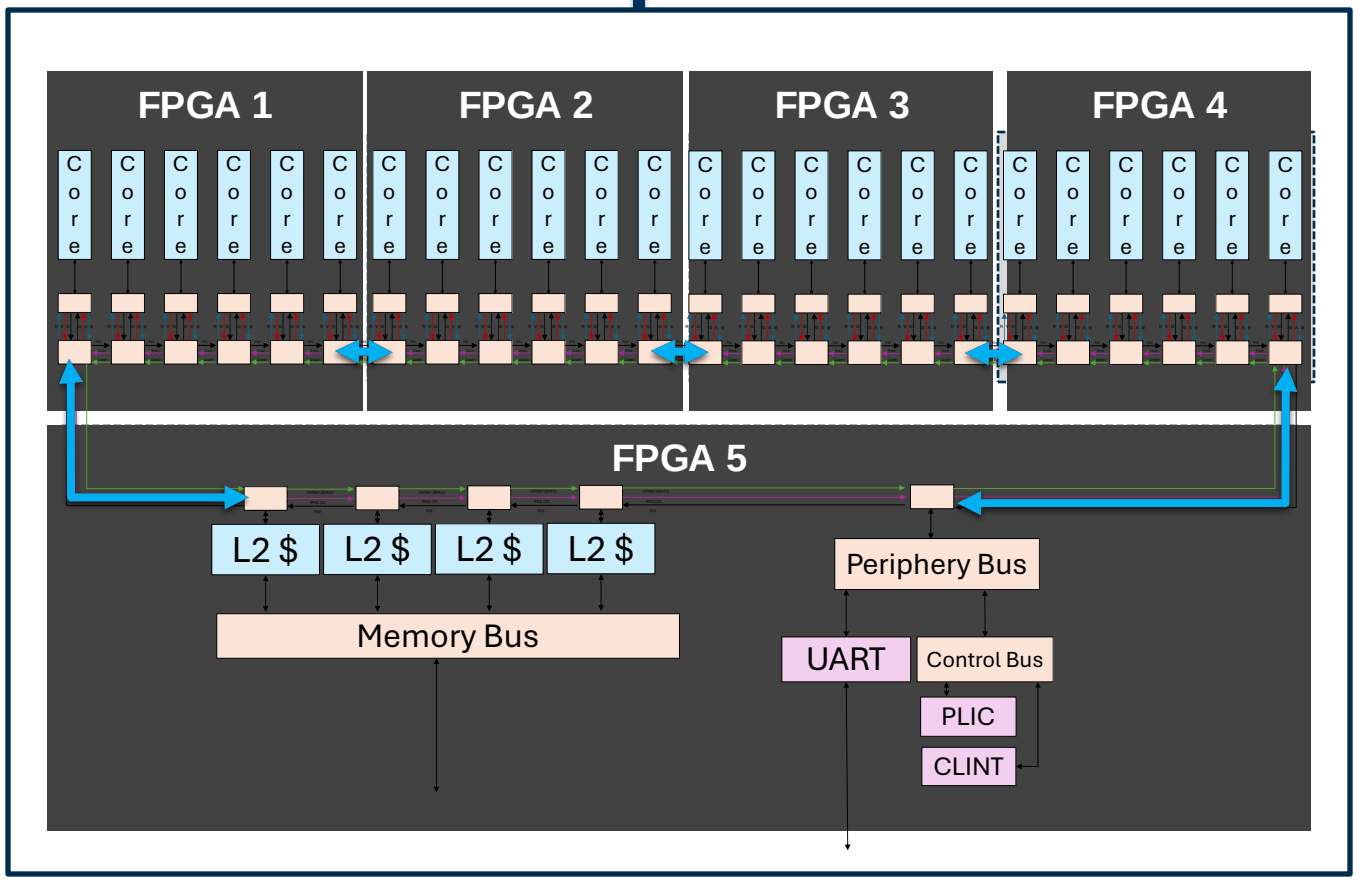
- Suppose we want to simulate an SoC with:
  - 24 Berkeley OoO machine (BOOM)
    - 3 wide out-of-order processor
    - **0.79 mm²**
  - Last level cache
    - 2MB capacity, 4 banks
  - Peripherals
- **> 18.7 mm²** ASIC area w/ commercial 16nm
- **SoC doesn’t fit in an FPGA**

Solution - Partition the design onto multiple FPGAs



- Partitioning onto multiple FPGAs: **higher simulation capacity**
- Hand partitioning is undesirable
- **Compiler:** Automate the partitioning process

FireRipper: FireAxe's compiler Input design / flags



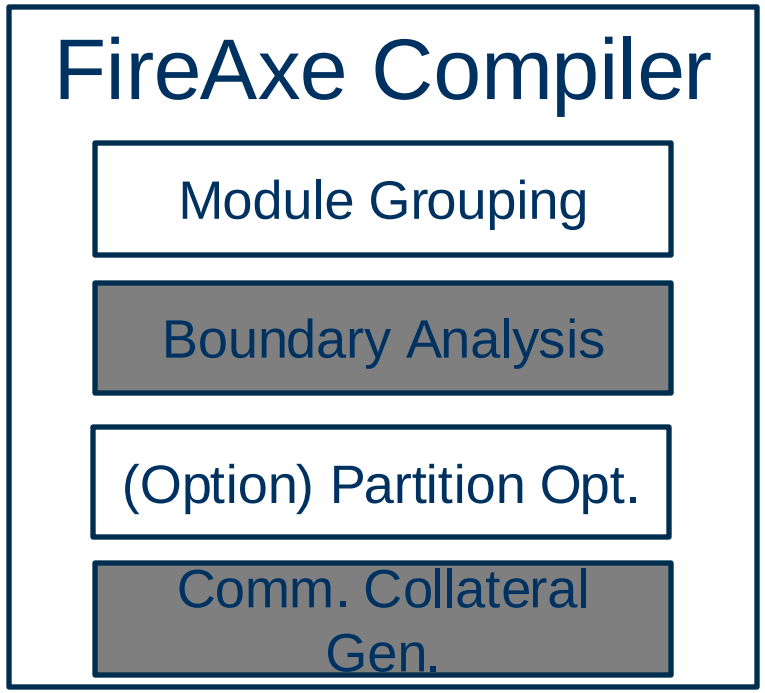
FireAxe Compiler

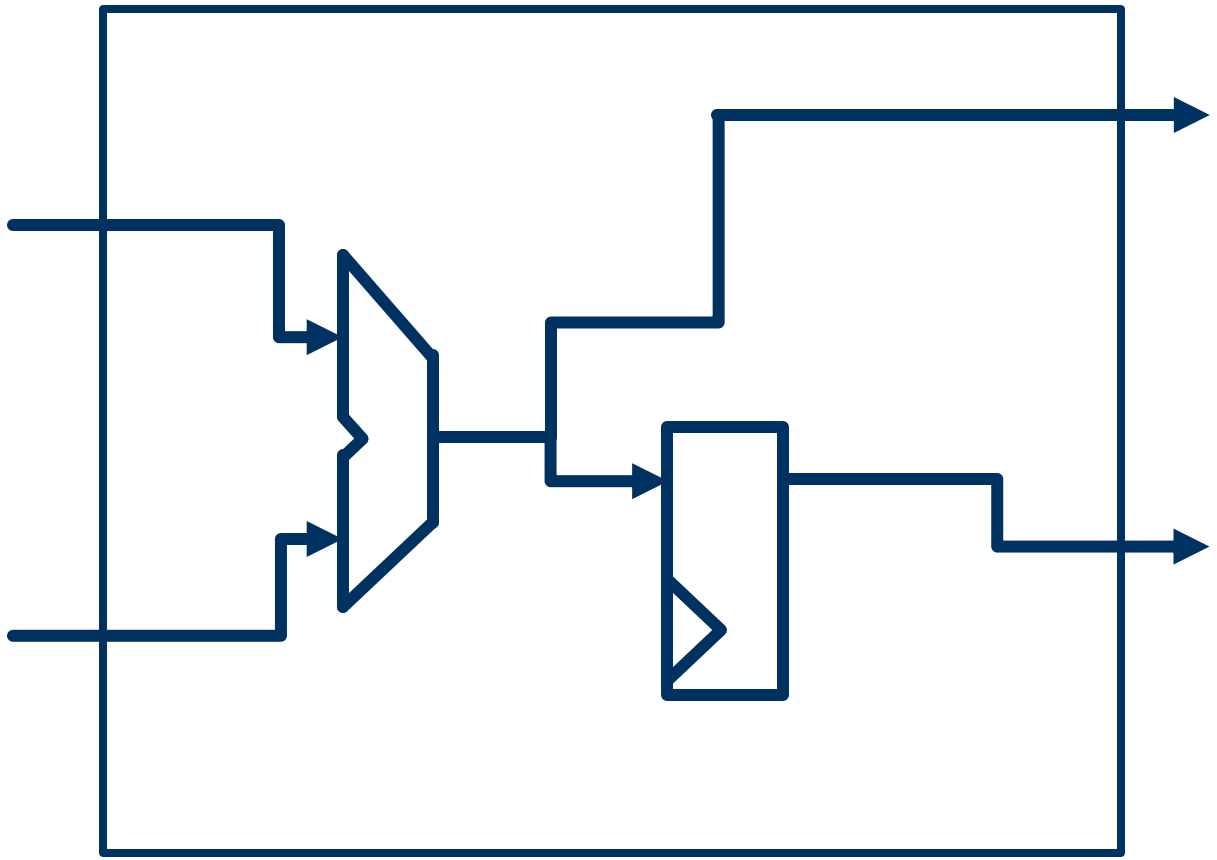
- Module Grouping
- Boundary Analysis
- (Option) Partition Opt.
- Comm. Collateral Gen.

Golden Gate / FPGA Mapping

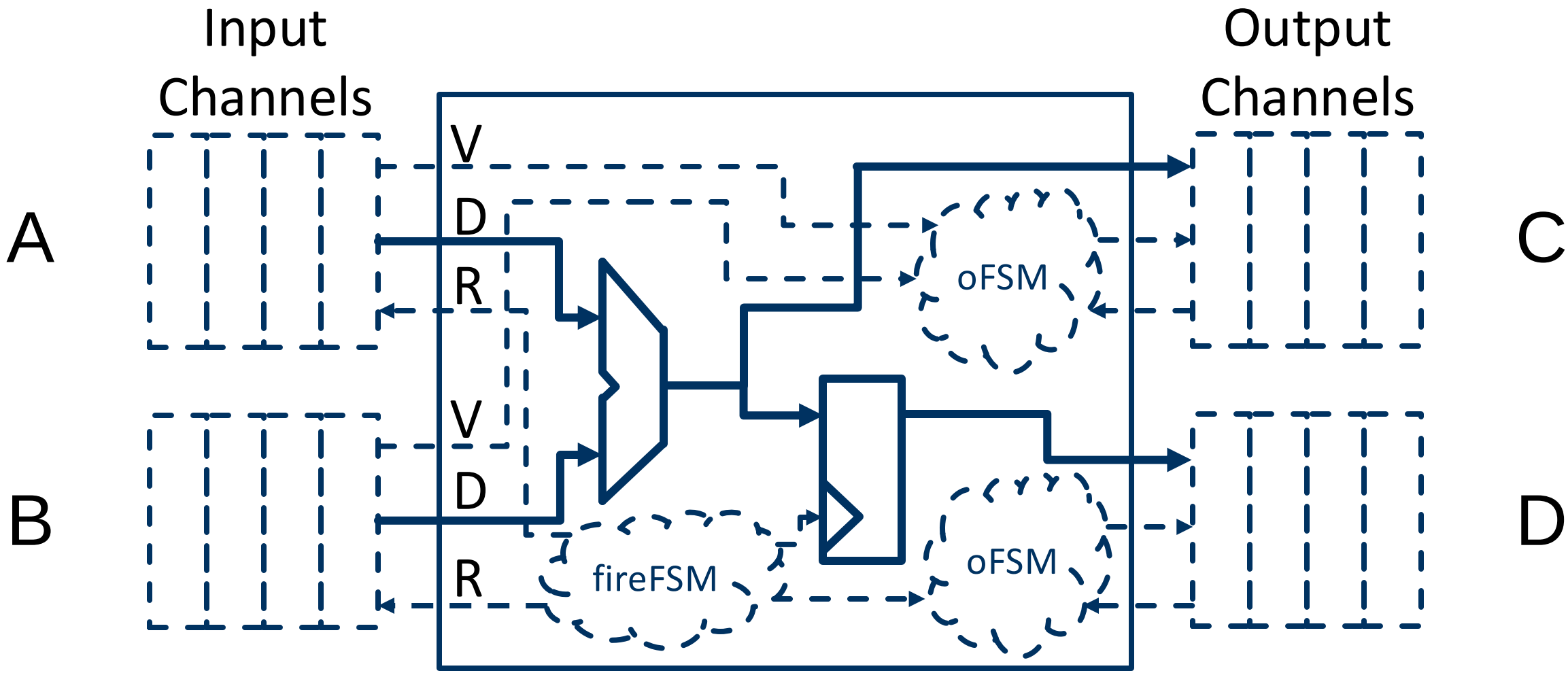
Partitioned FPGA accelerated RTL simulation

Boundary analysis & comm. collateral generation

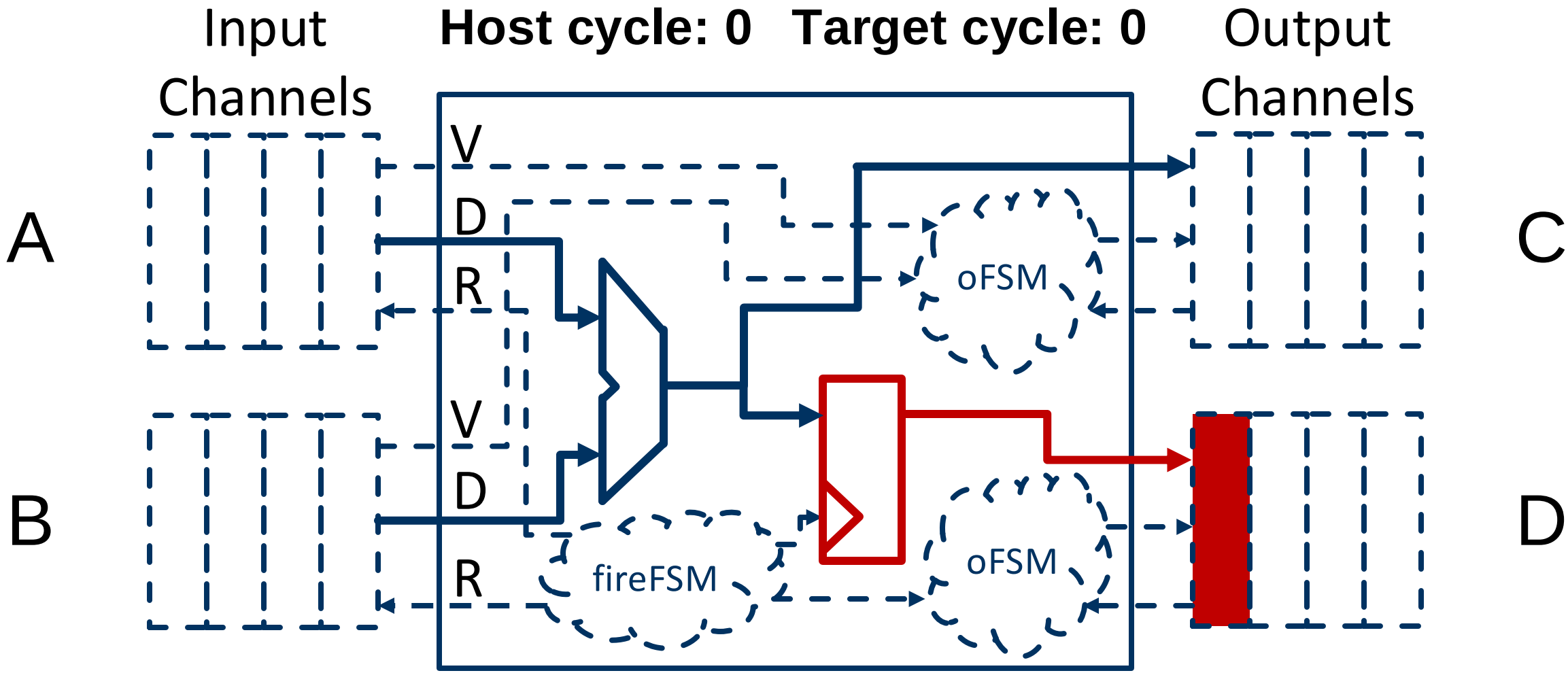




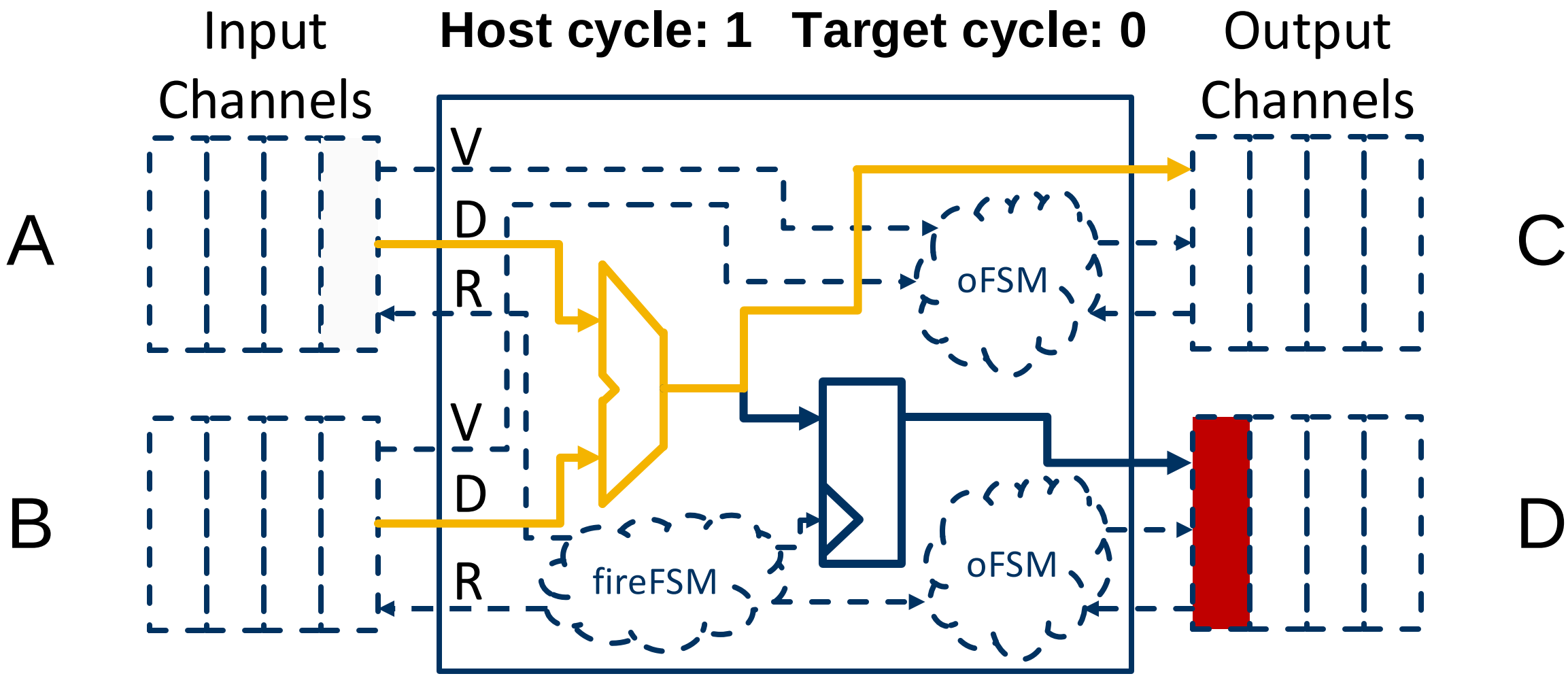
LI-BDN – decouple the notion of time



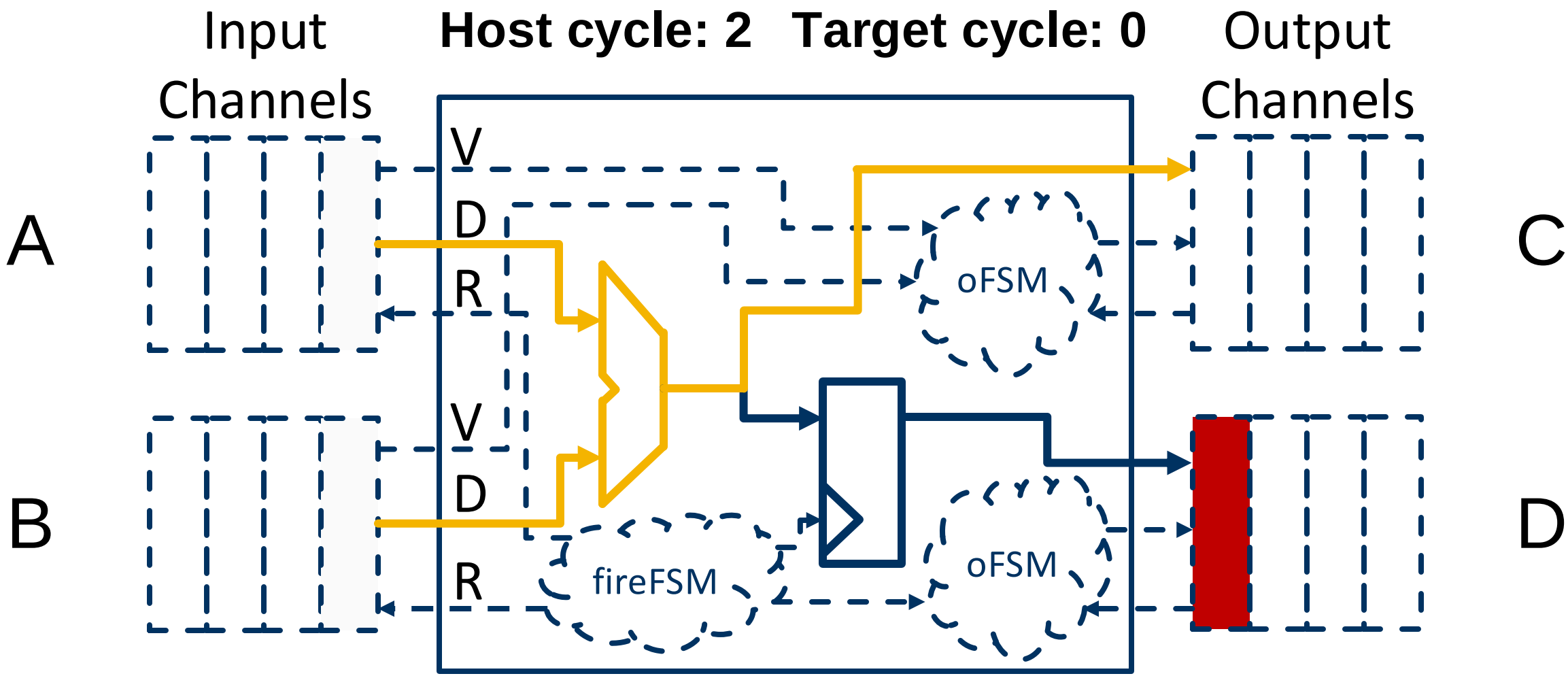
LI-BDN – decouple the notion of time



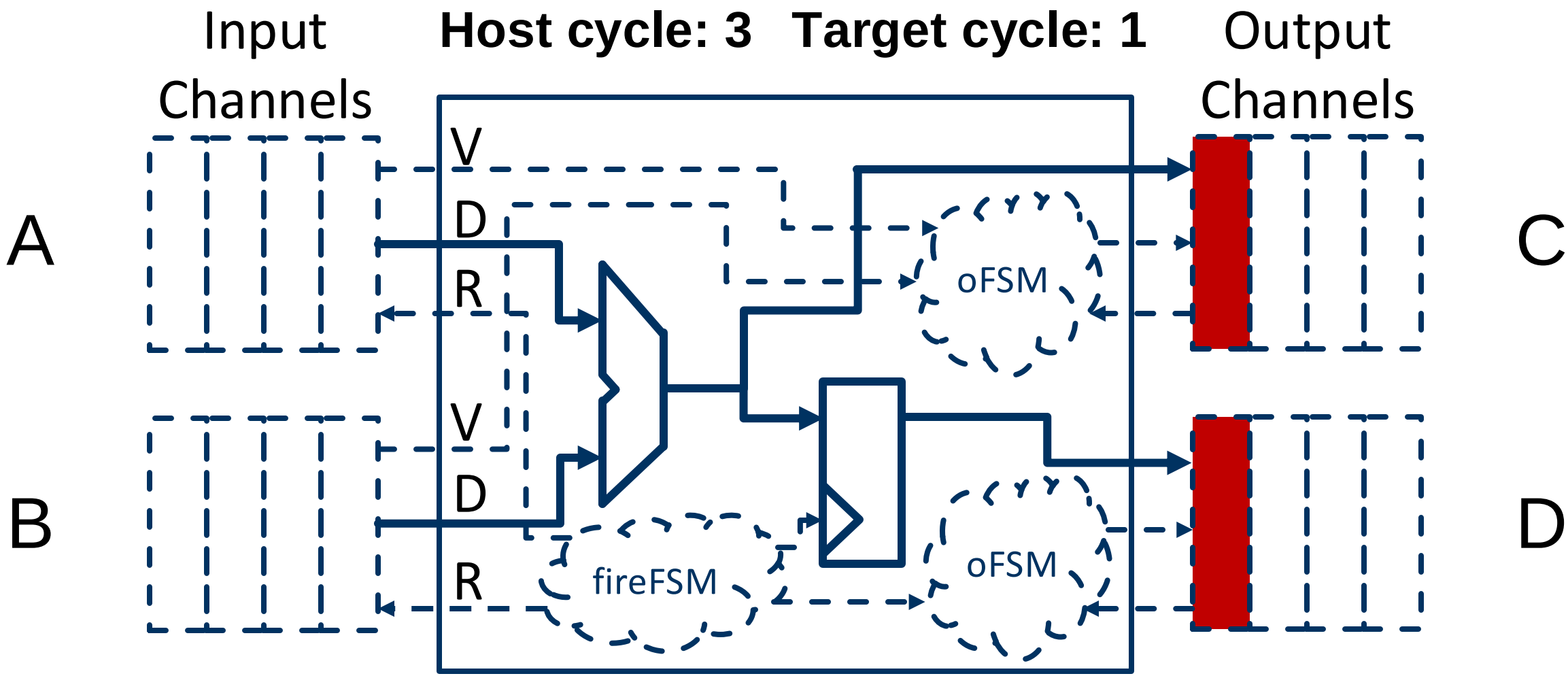
LI-BDN – decouple the notion of time



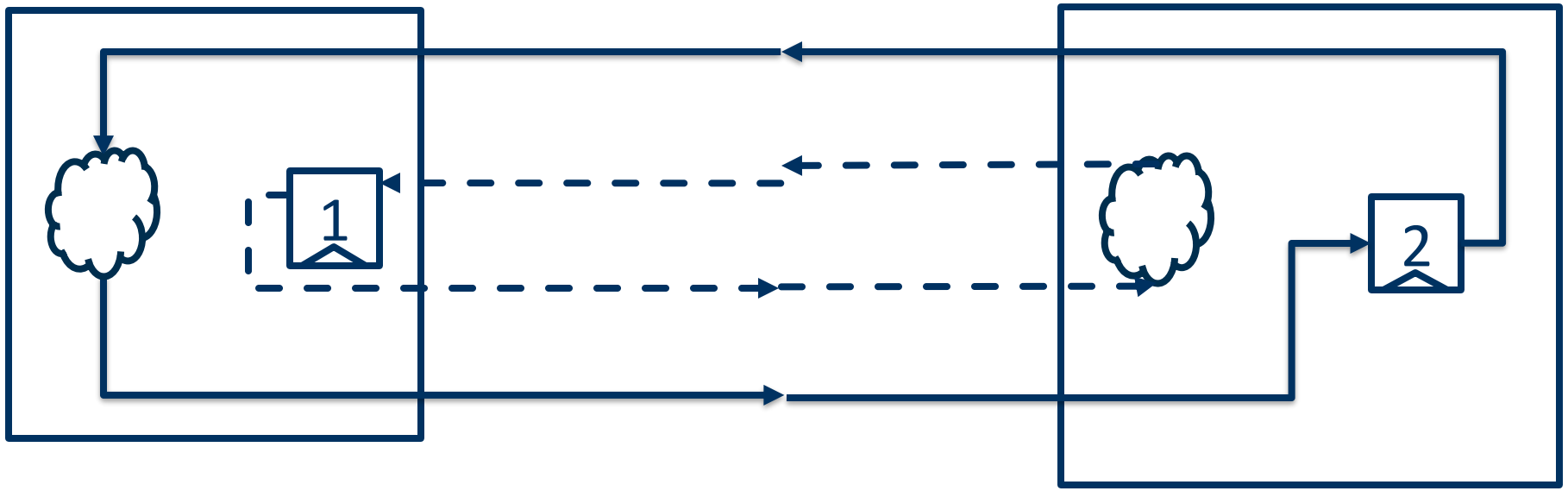
LI-BDN – decouple the notion of time



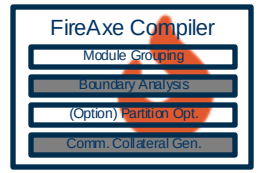
LI-BDN – decouple the notion of time

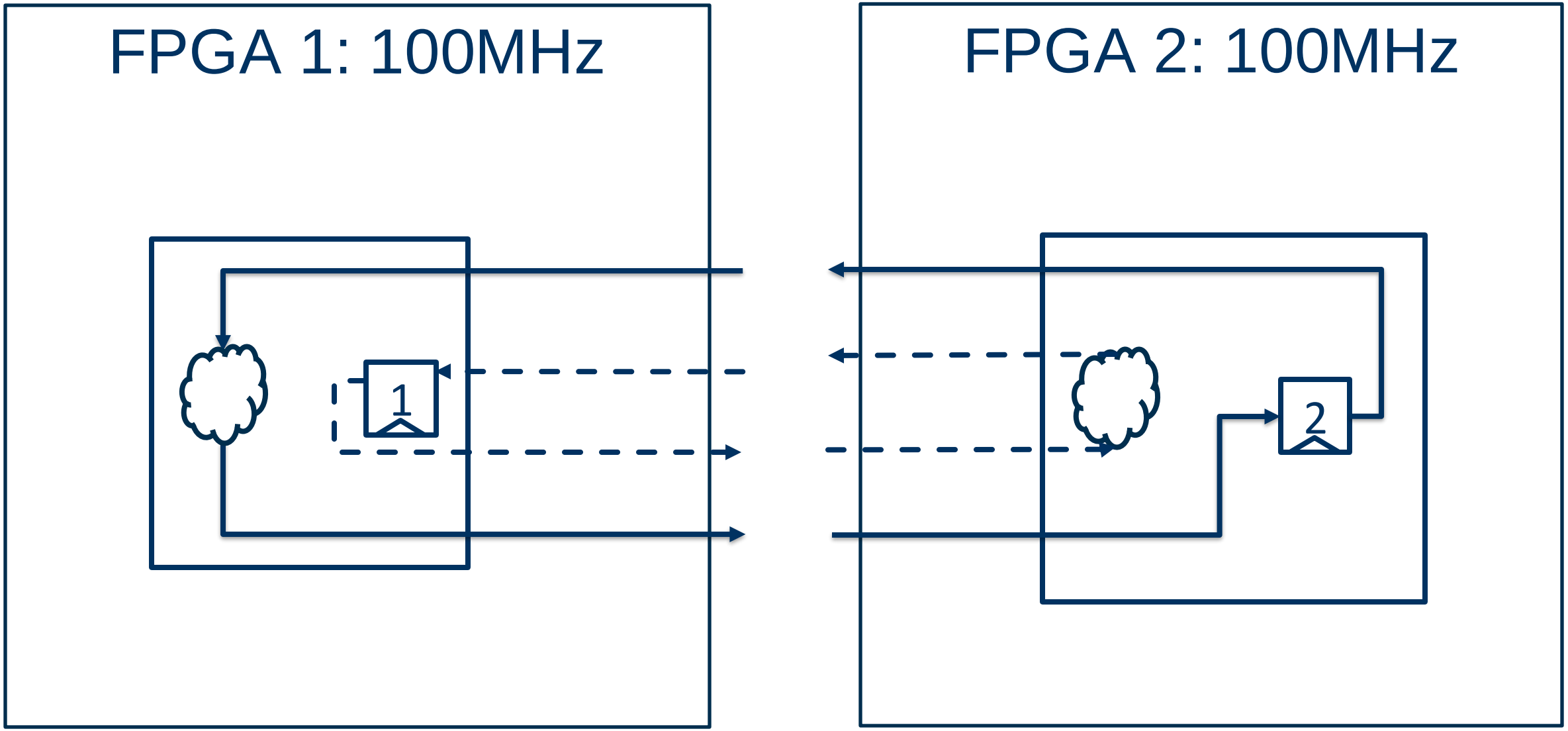


Boundary analysis – Comb. Logic

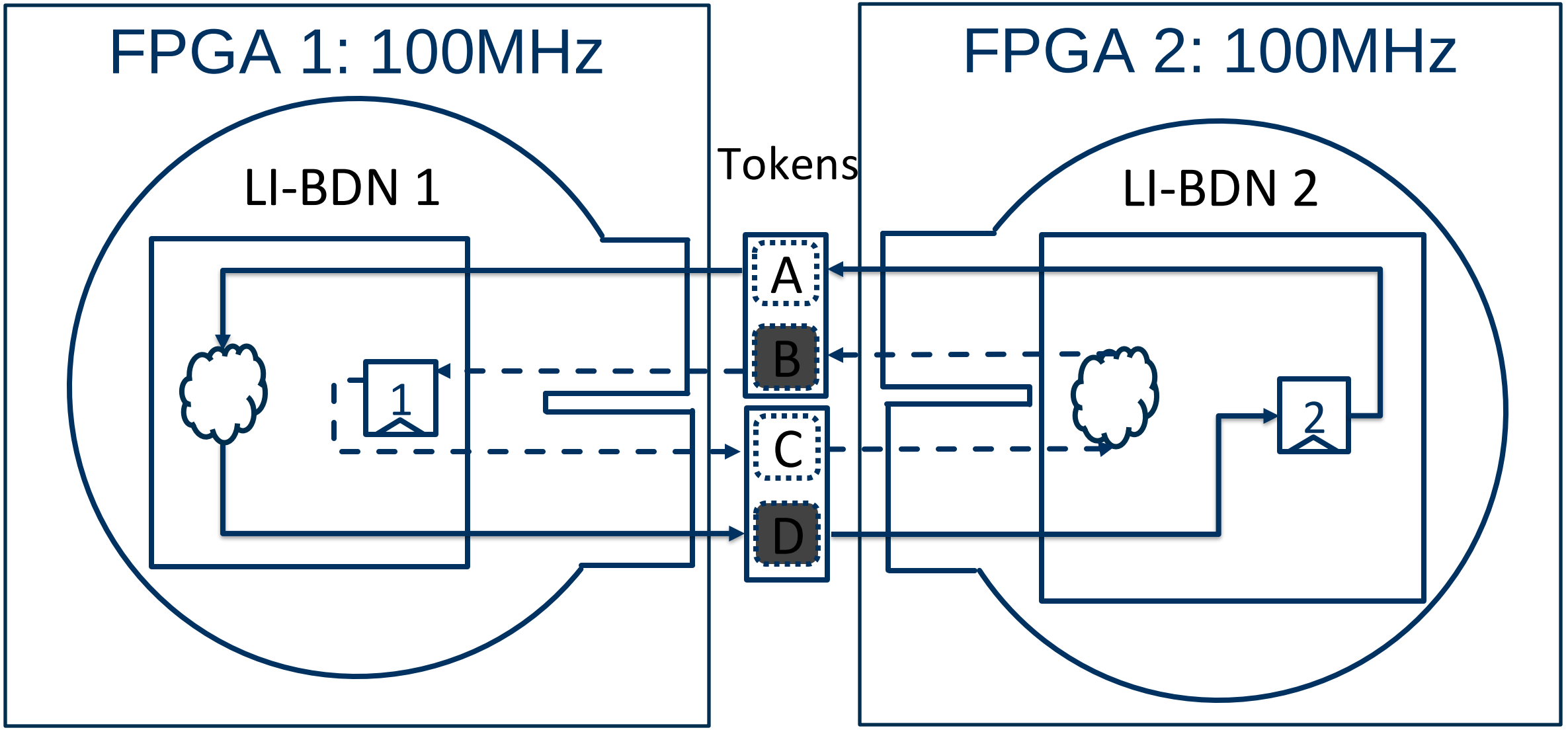


Boundary analysis – Comb. Logic

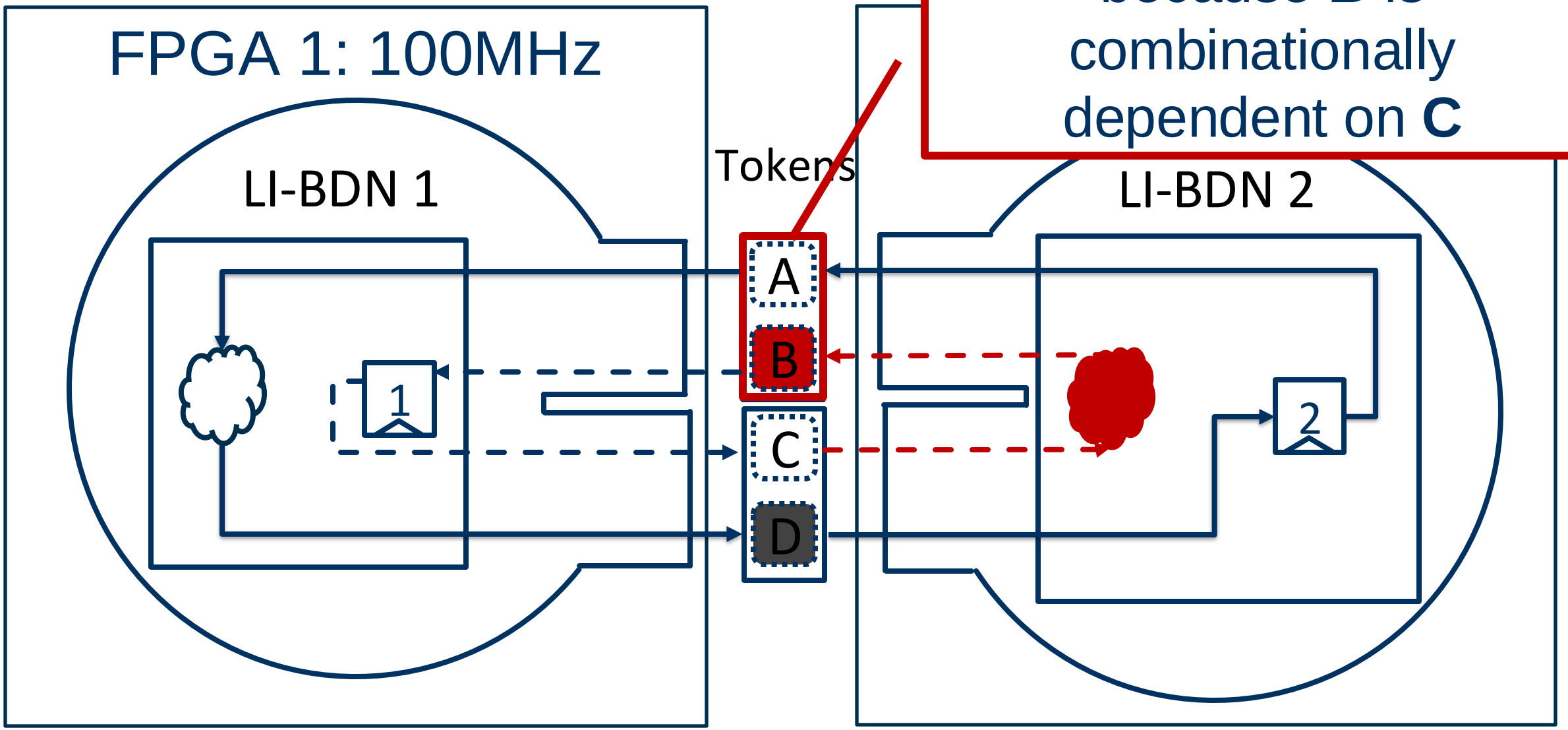




Boundary analysis – Comb. Logic

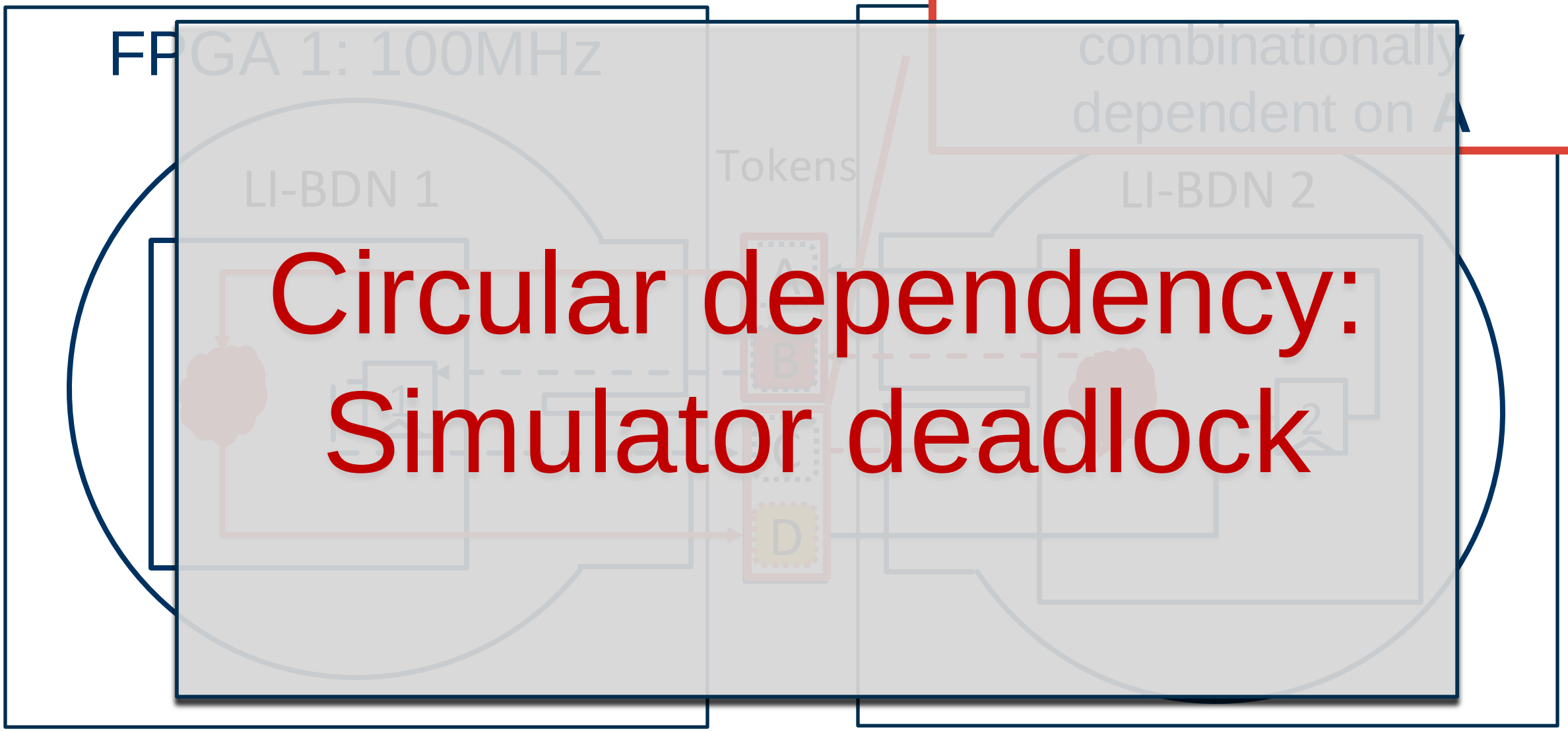


Boundary analysis – Comb.



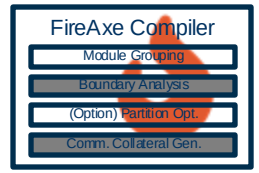
Boundary analysis – Comb.

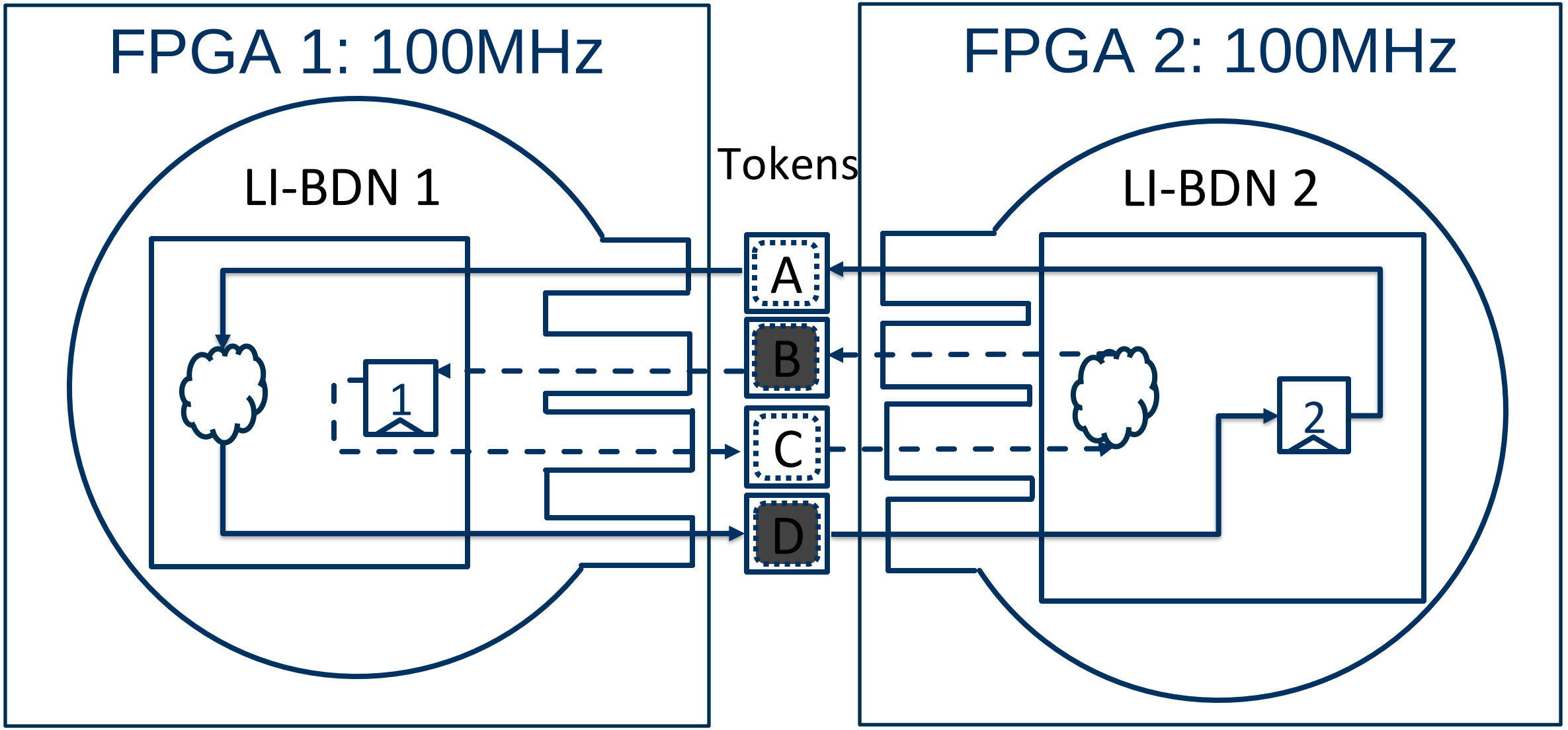
Token cannot be sent because **D** is combinationally dependent on **A**



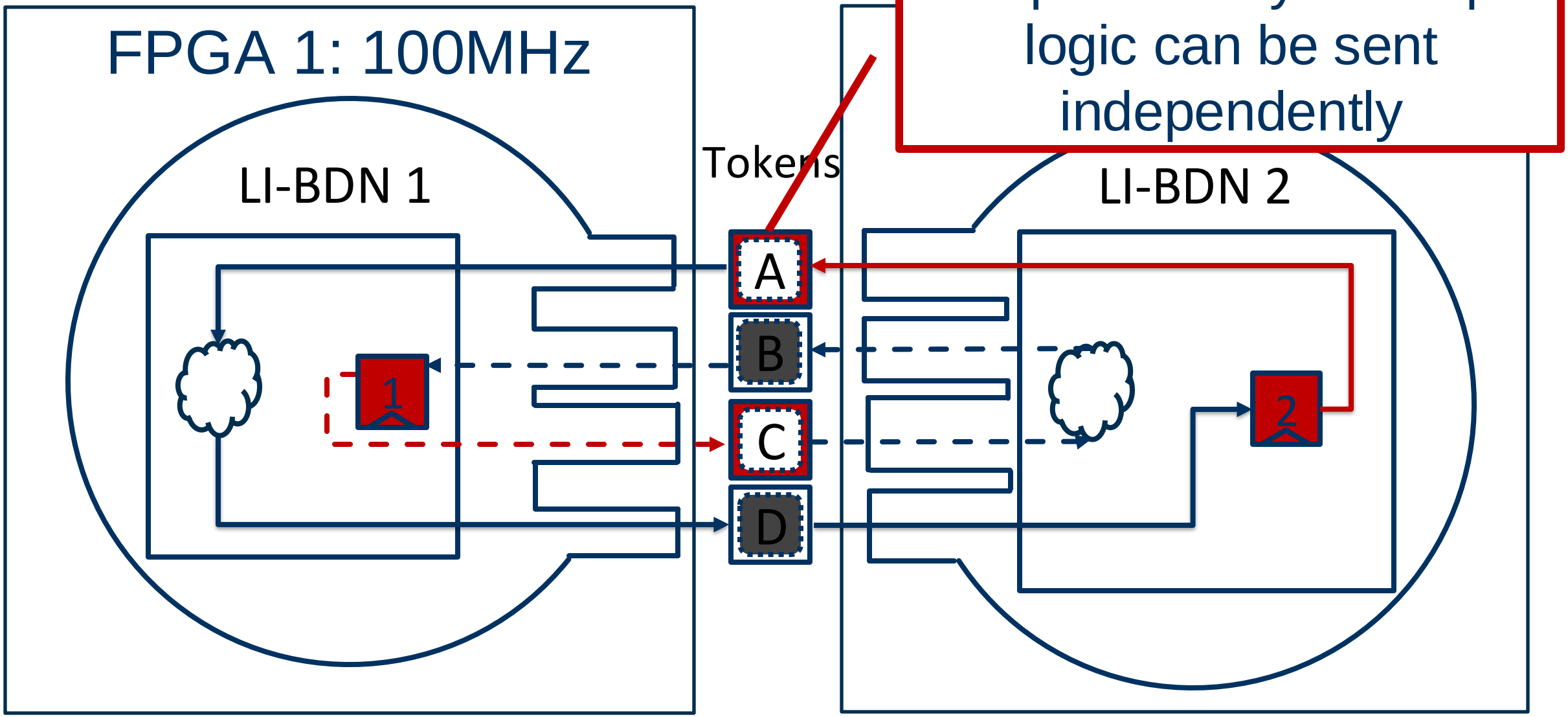
Circular dependency: Simulator deadlock

Preventing deadlocks by splitting the channels





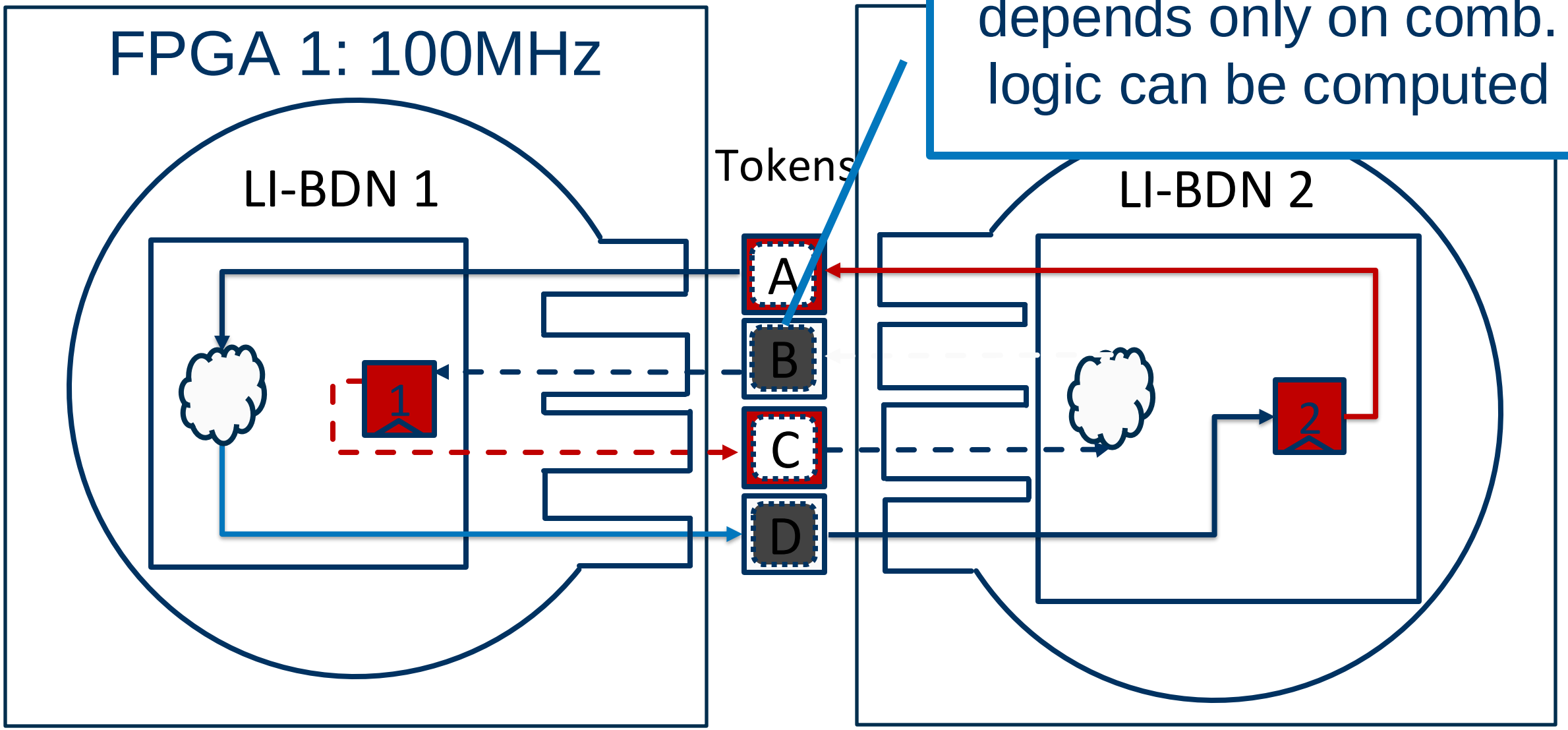
Preventing deadlocks by split



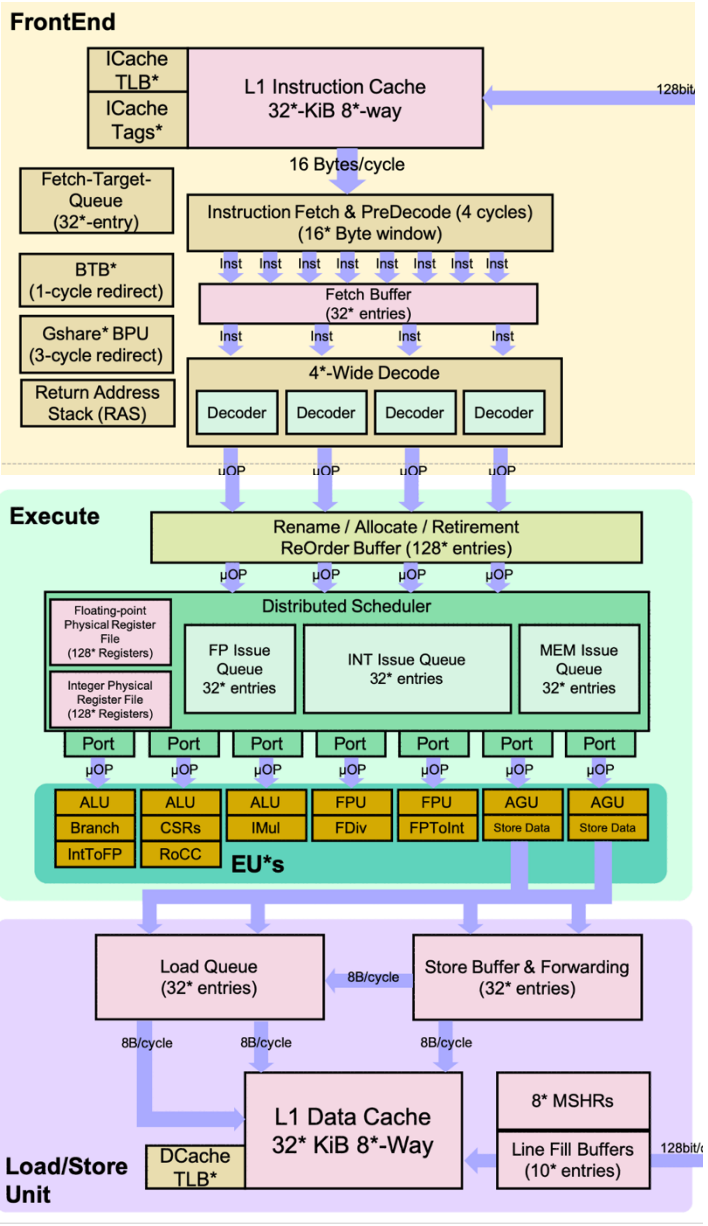
Output tokens that depends only on seq. logic can be sent independently

Preventing deadlocks by splitt

Output tokens that depends only on comb. logic can be computed

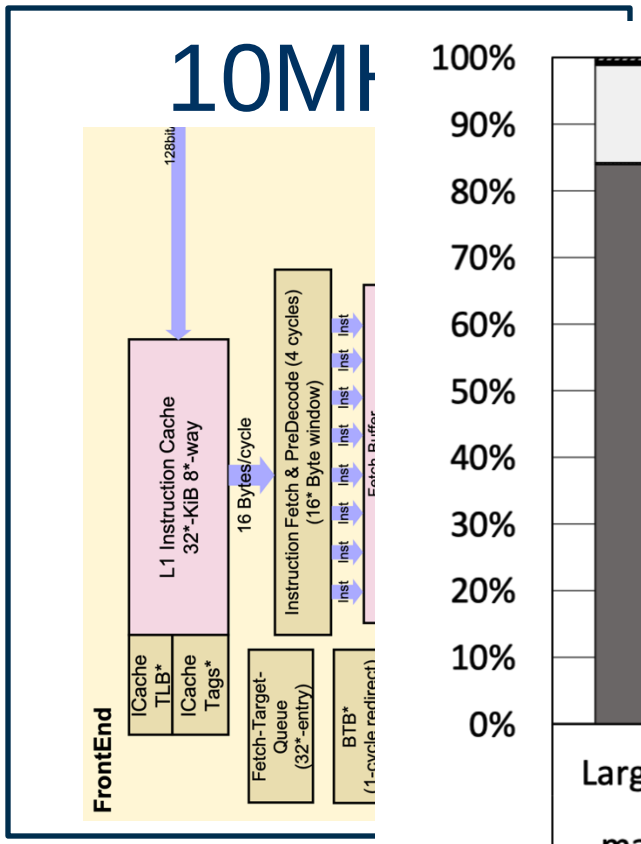


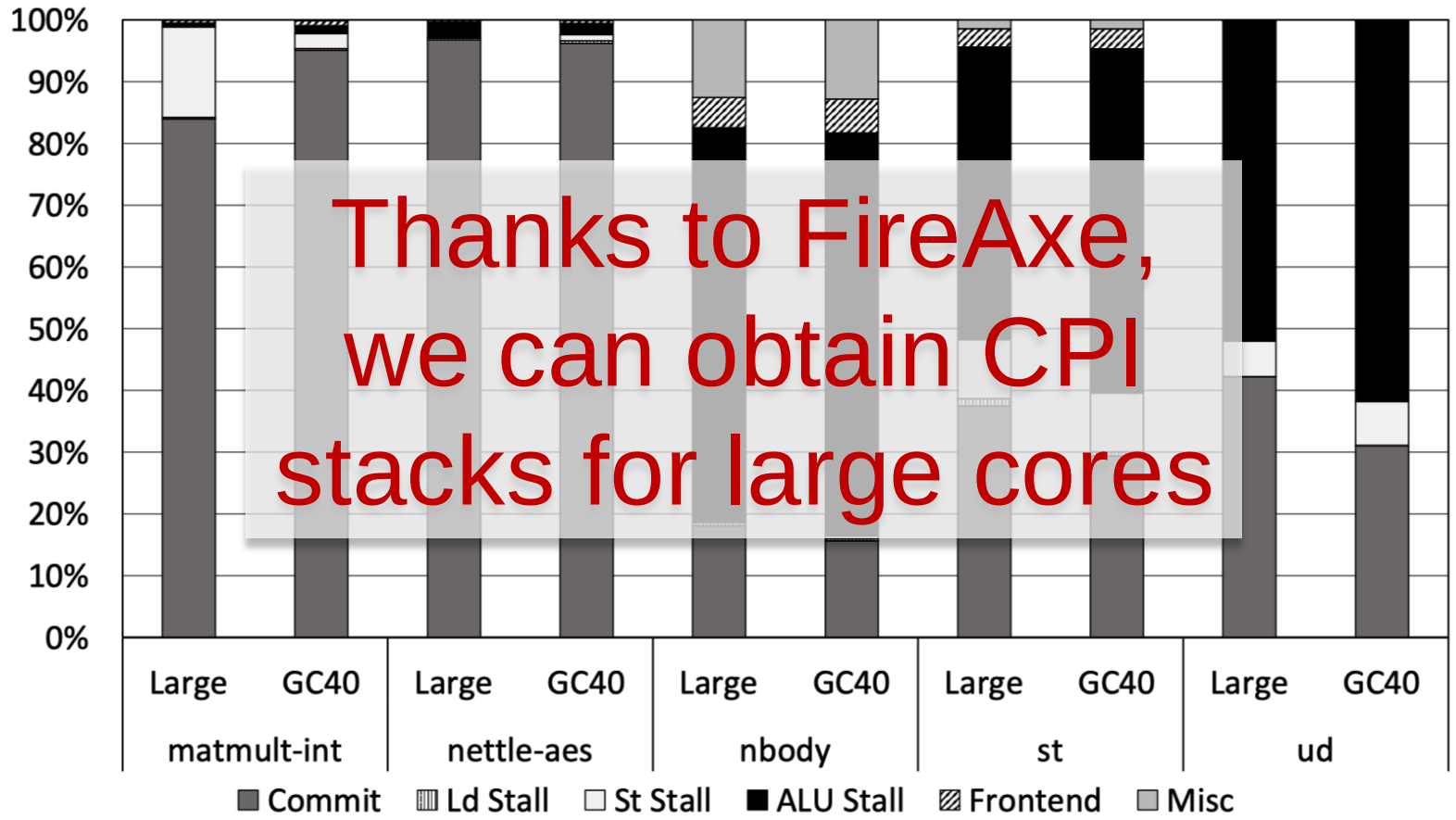
Case Study: Partitioning a large OoO Core



- Larger variant of BOOM
  - 6 wide issue
  - 216 ROB entries
  - 115 I-phys reg / 132 F-phys reg
  - 76 Ld queue entries
  - 45 St queue entries
- 1.56 mm^2 in commercial 16nm tech
- Over **7000 bits** going through the partition boundary

Case Study: Partitioning a large OoO Core

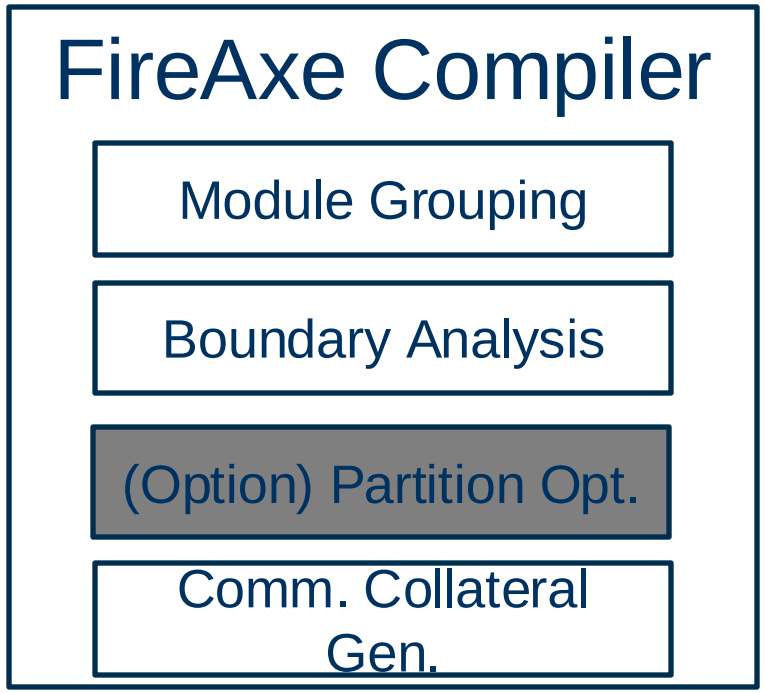




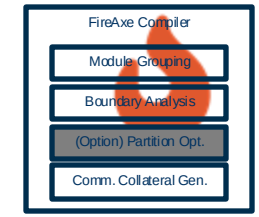
ıchmarks
h)

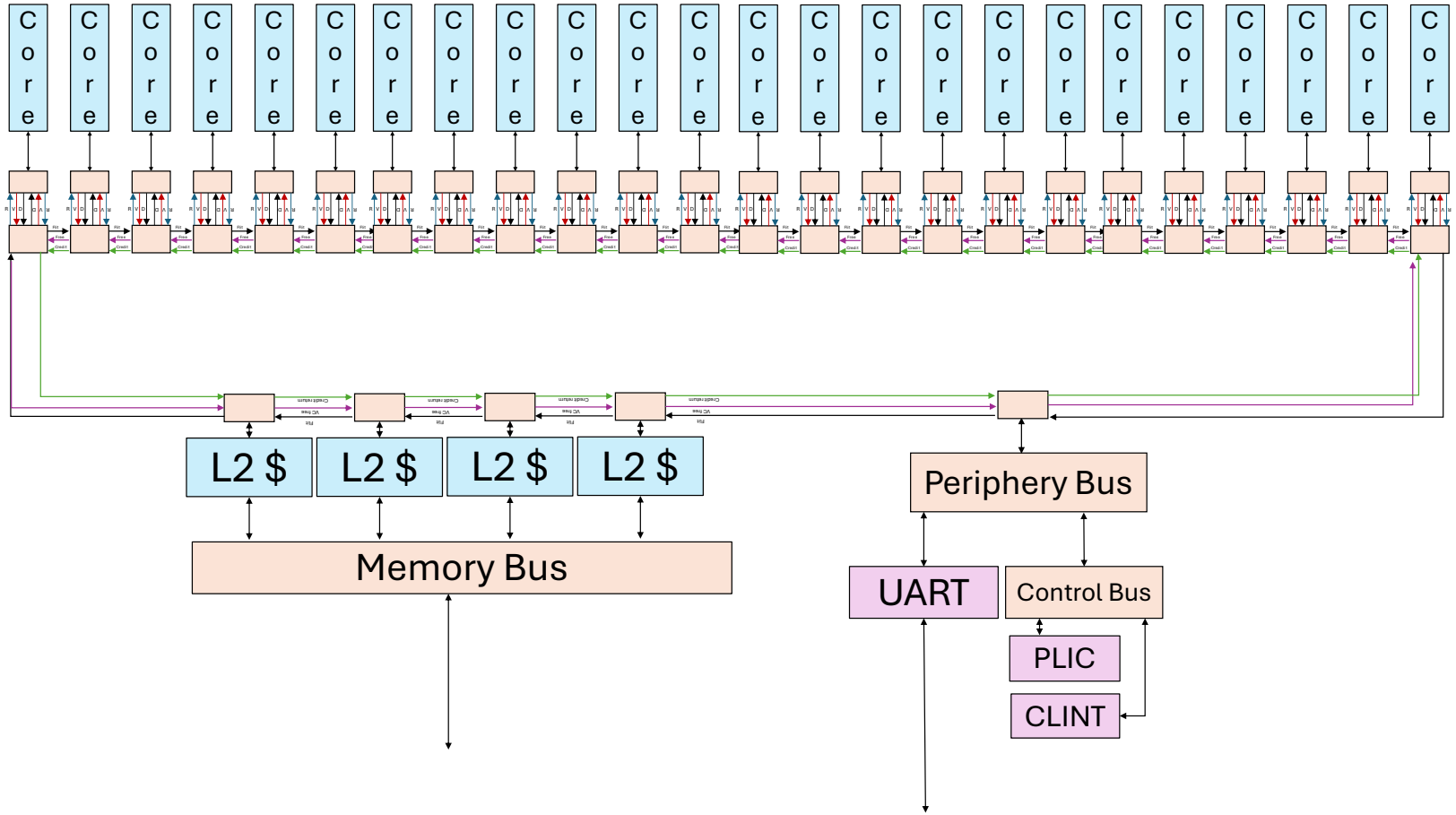
Thanks to FireAxe, we can obtain CPI stacks for large cores

Optional partitioning optimizations



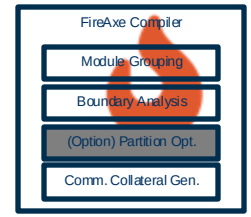
Optional partitioning optimizations

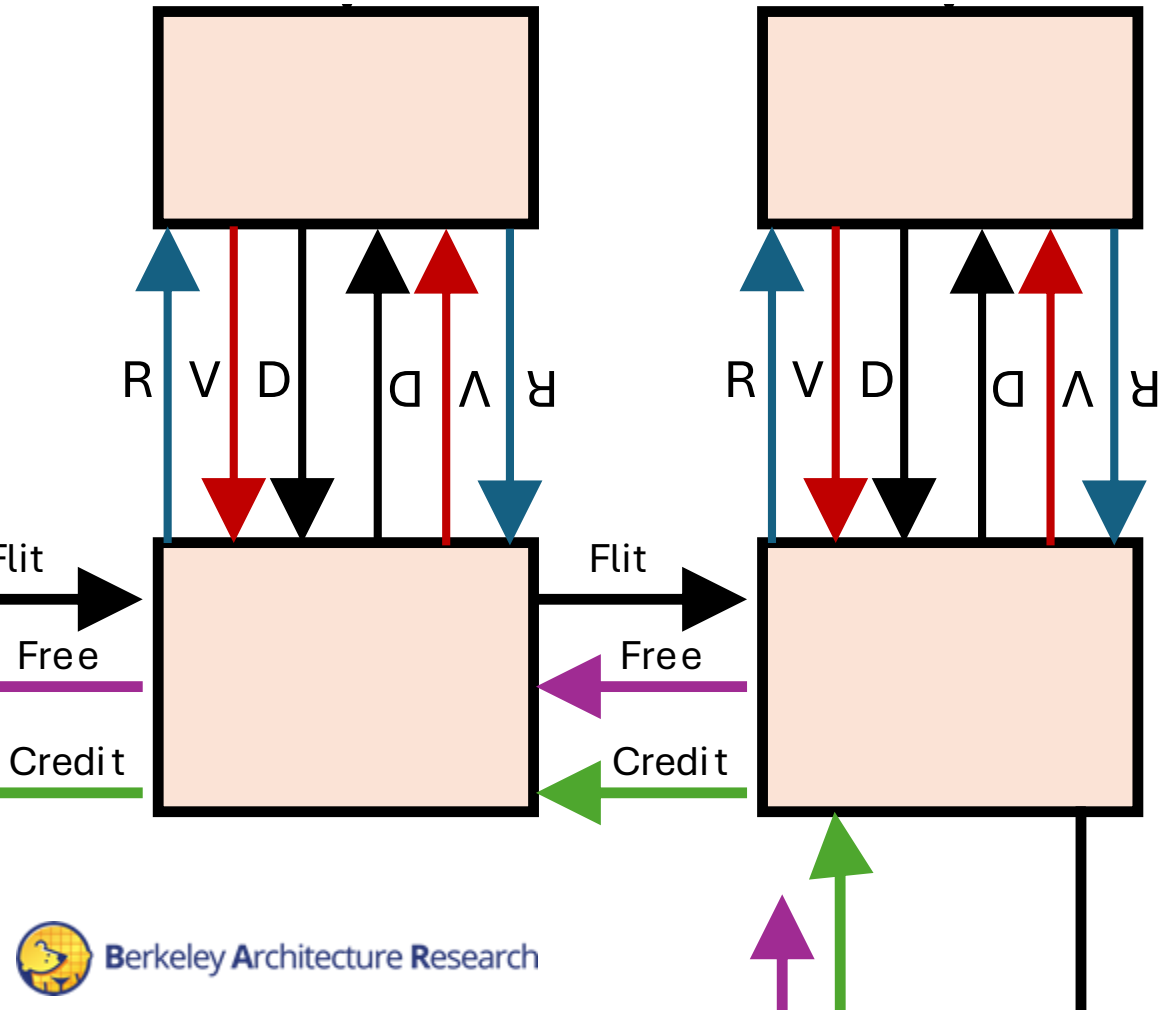




- Use microarchitectural semantics to improve partitioned simulation performance

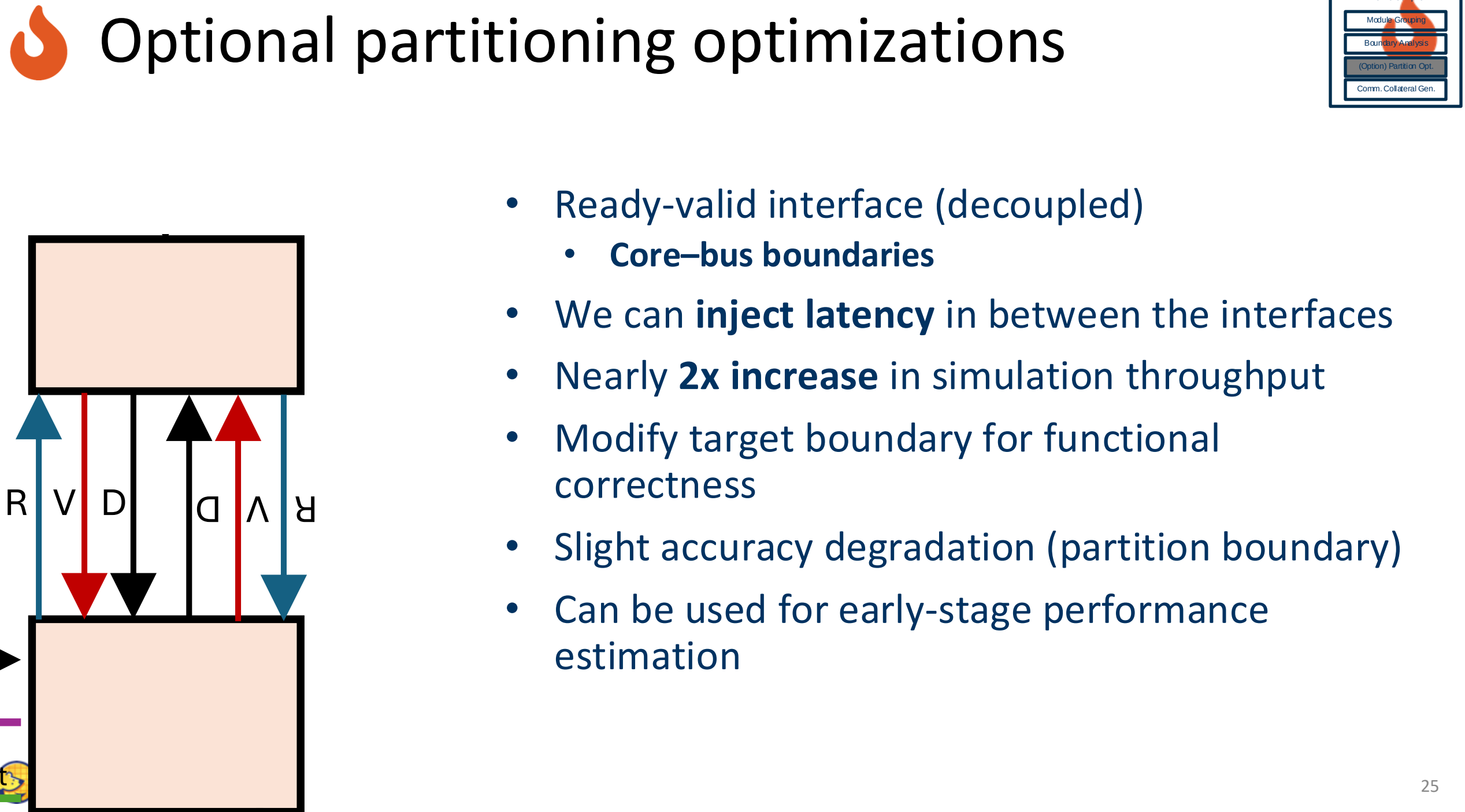
Optional partitioning optimizations





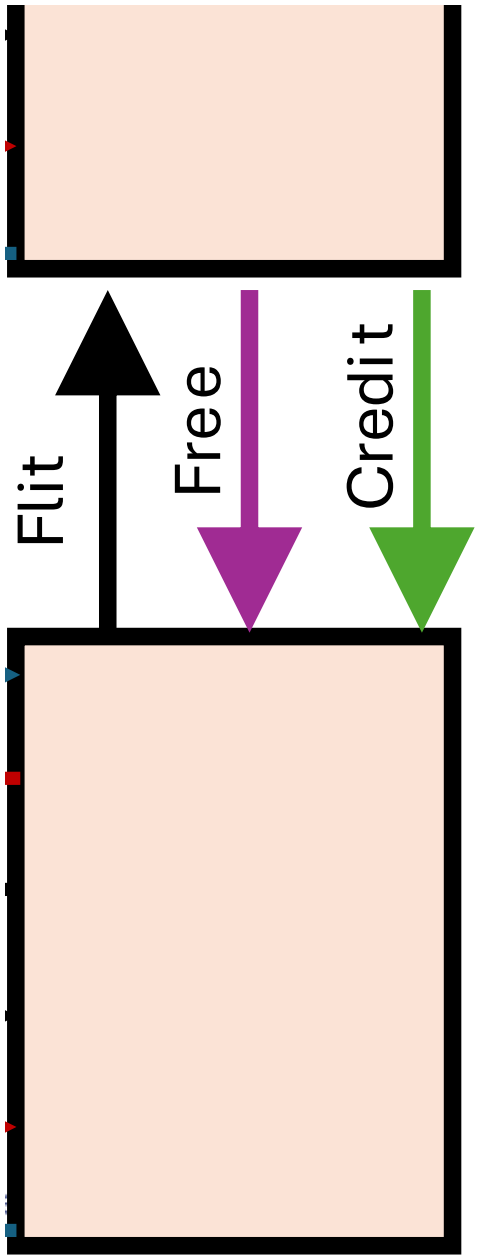
- Latency insensitive boundaries
  - Latency sensitive components cannot scale over a certain degree

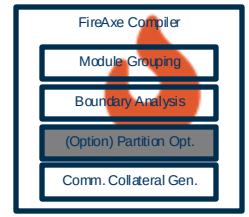
Optional partitioning optimizations



- Ready-valid interface (decoupled)
  - **Core–bus boundaries**
- We can **inject latency** in between the interfaces
- Nearly **2x increase** in simulation throughput
- Modify target boundary for functional correctness
- Slight accuracy degradation (partition boundary)
- Can be used for early-stage performance estimation

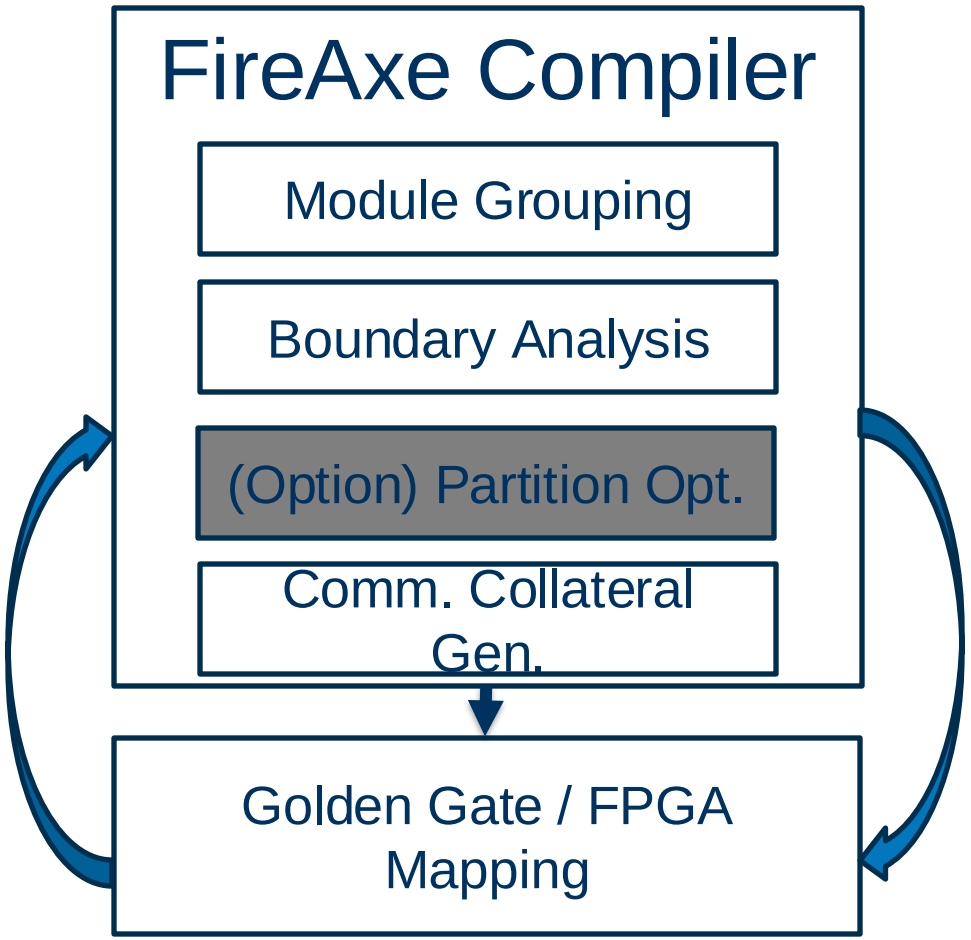
Optional partitioning optimizations



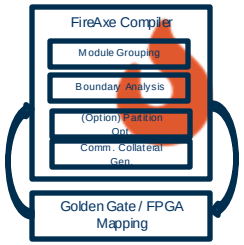


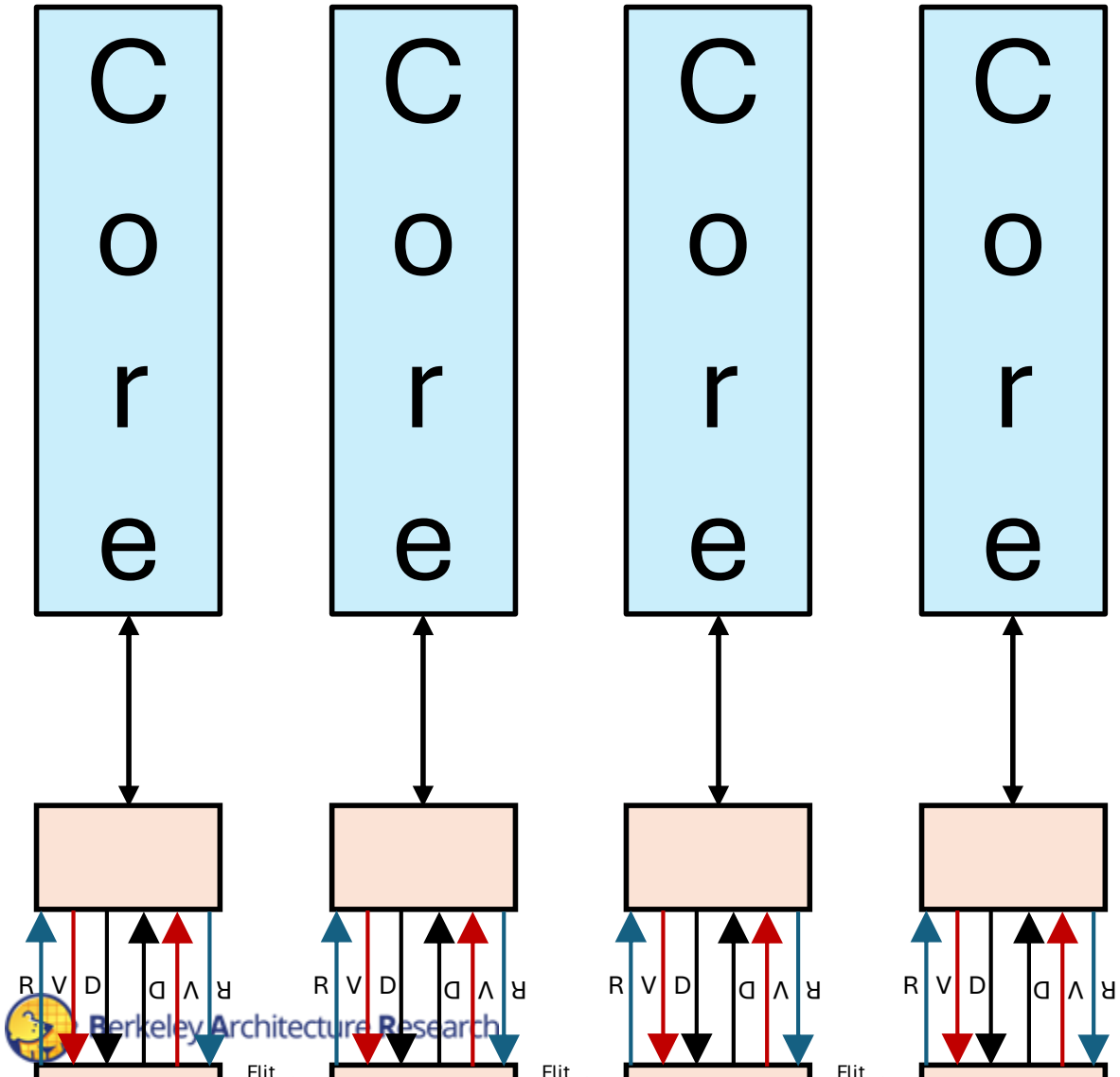
- Credit based interfaces
  - **NoC router node boundaries**
- **No target boundary modifications**
  - Latency-insensitive
  - No comb-deps
- **Narrow** partition boundary
- Map SoC topology onto FPGA topology

Partitioning & FPGA resource opt. synergy



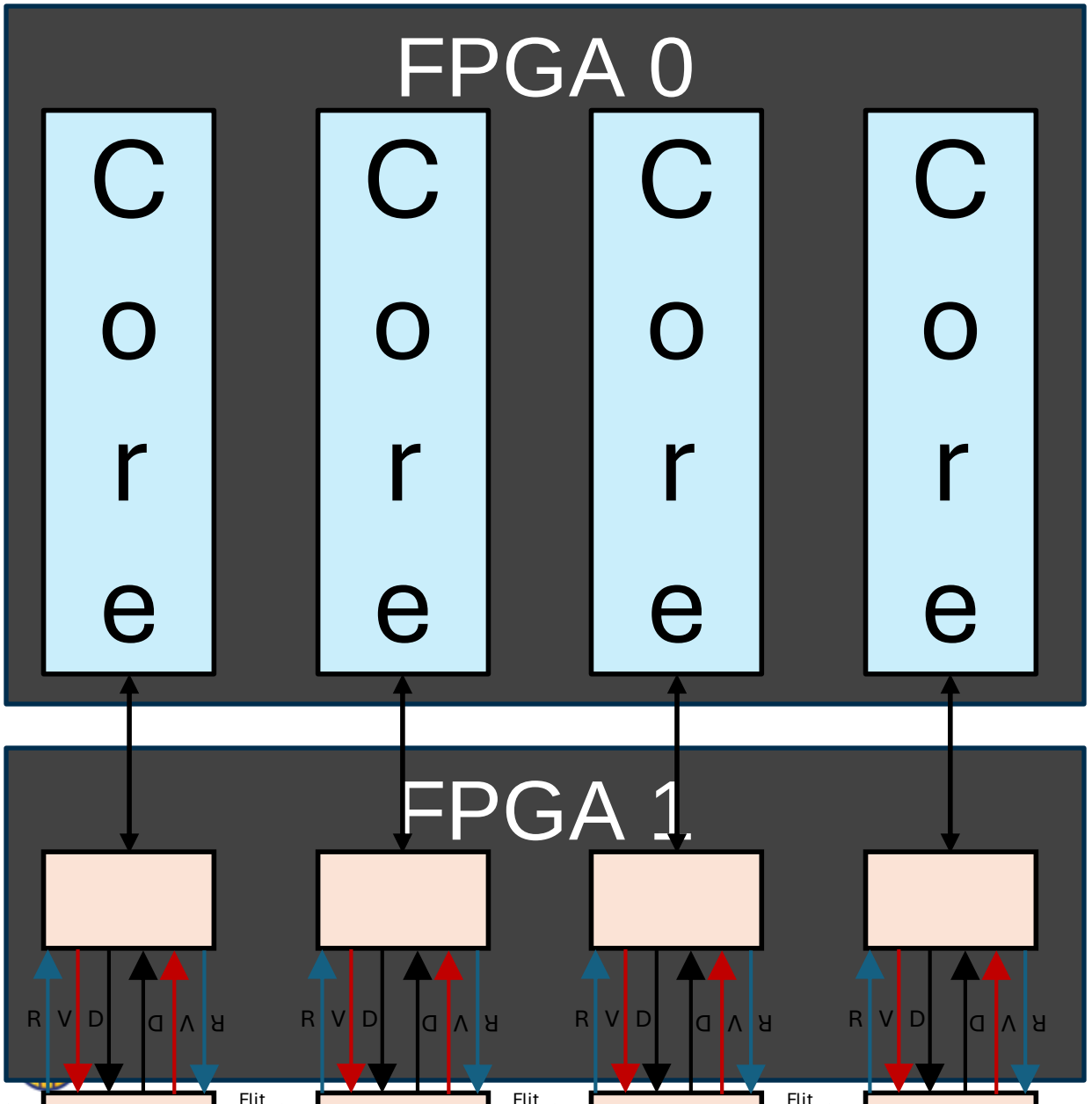
Partitioning & FPGA resource opt. synergy

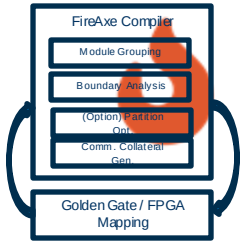




- Sometimes, modules are stamped out multiple times (e.g. cores)
- FireSim can employ simulator level multithreading to save FPGA resources

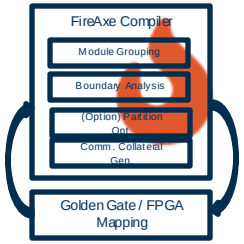
Partitioning & FPGA resource opt. synergy

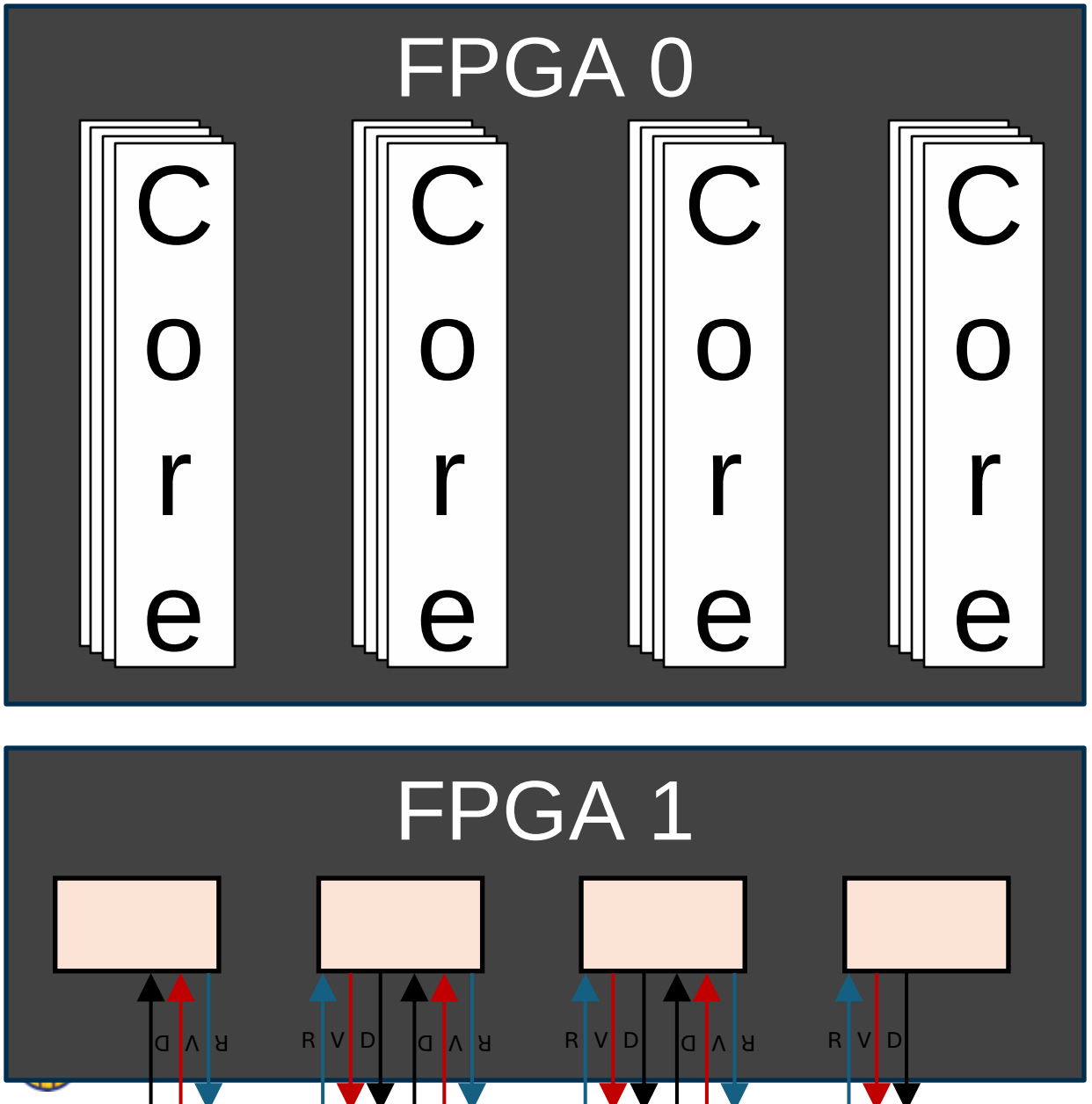




- Example partitioned simulation
- FPGA resource consumption is proportional to the number of cores

Partitioning & FPGA resource opt. synergy





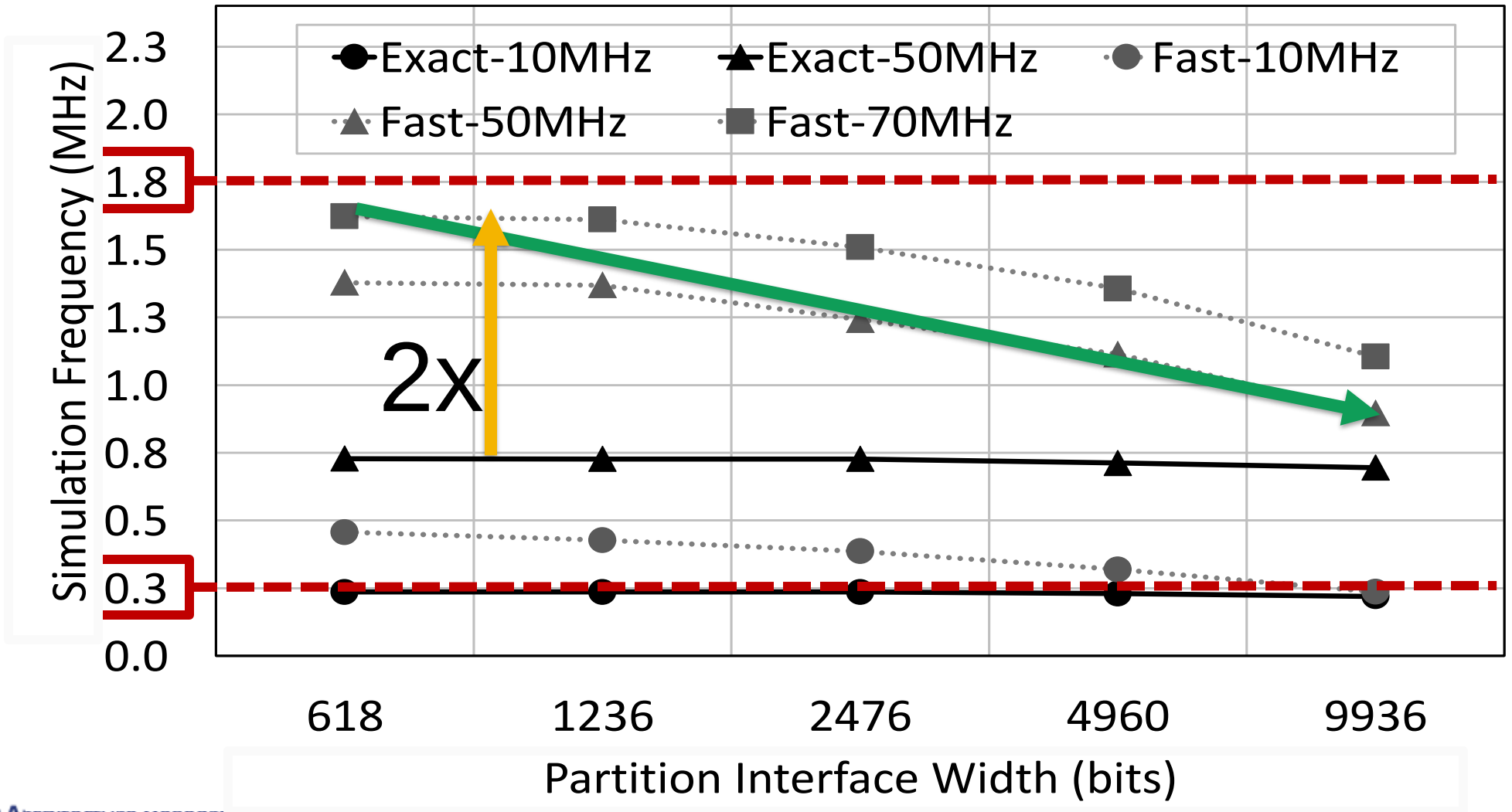
- FPGA resource optimization
  - Can share combinational logic and only replicate sequential logic
  - Time / FPGA resource tradeoff
- To simulate 1 target cycle
  - 4 host-FPGA cycles
  - 10~50 **inter-FPGA communication** cycles
- Overhead of multithreading hidden due to inter-FPGA link latency!

Supported FPGA platforms

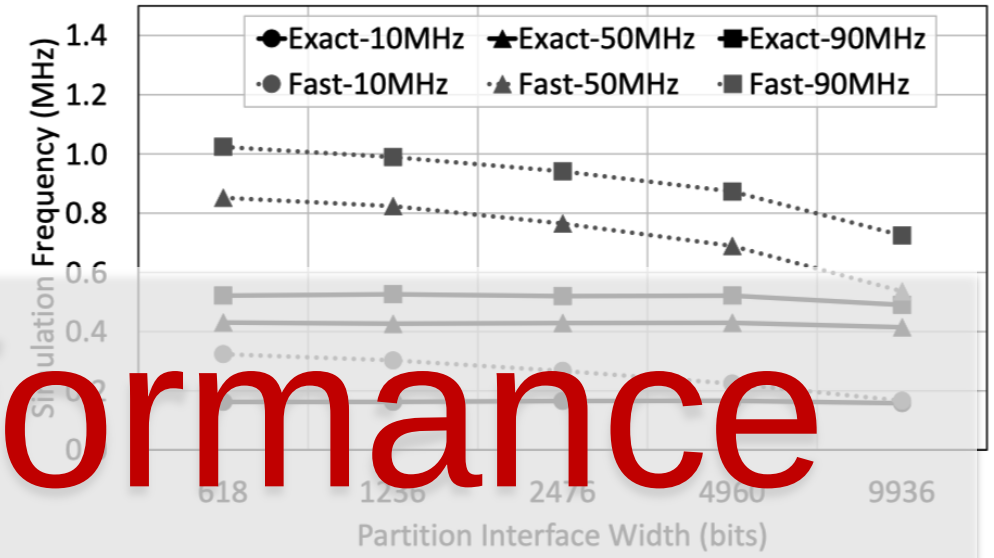
- Cloud EC2 F1 instances
  - Direct peer to peer PCIe FPGA communication scheme
- Local FPGAs
  - Xilinx U250s connected via cheap QSFP direct attach cables
  - 2x simulation performance vs EC2 F1 instances due to direct links

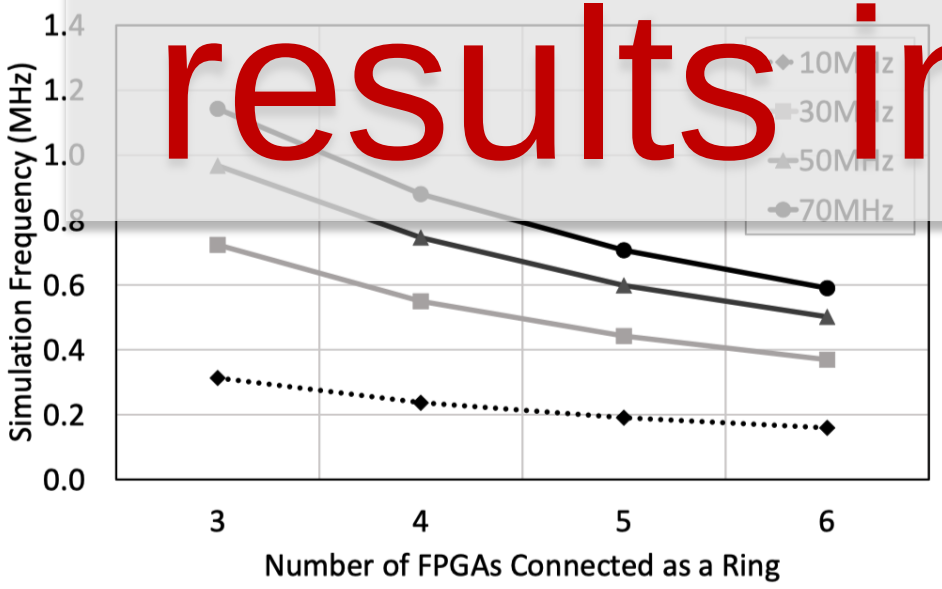
Passive Option

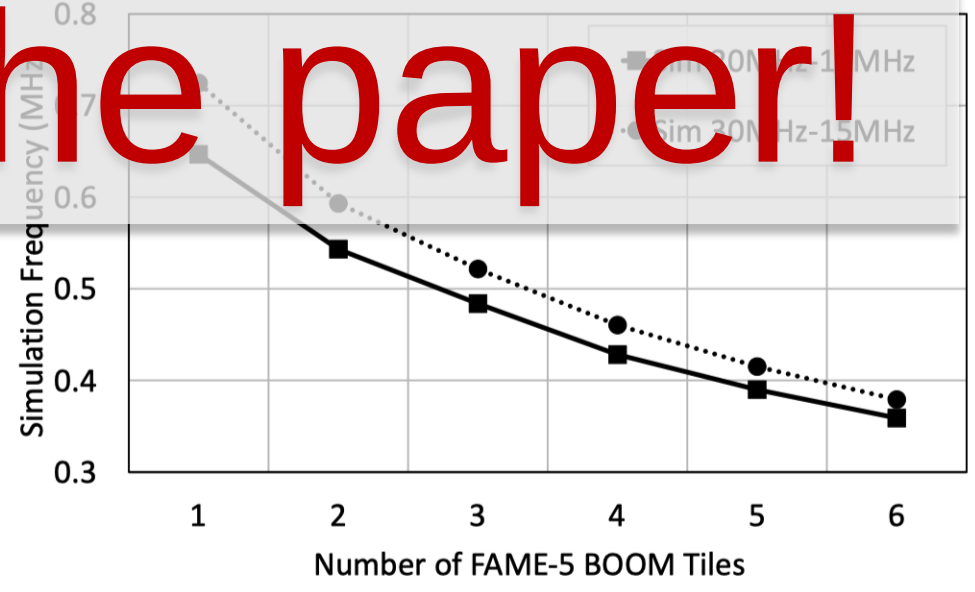
Performance characteristics – 2 FPGA on-premises



Performance Characteristics

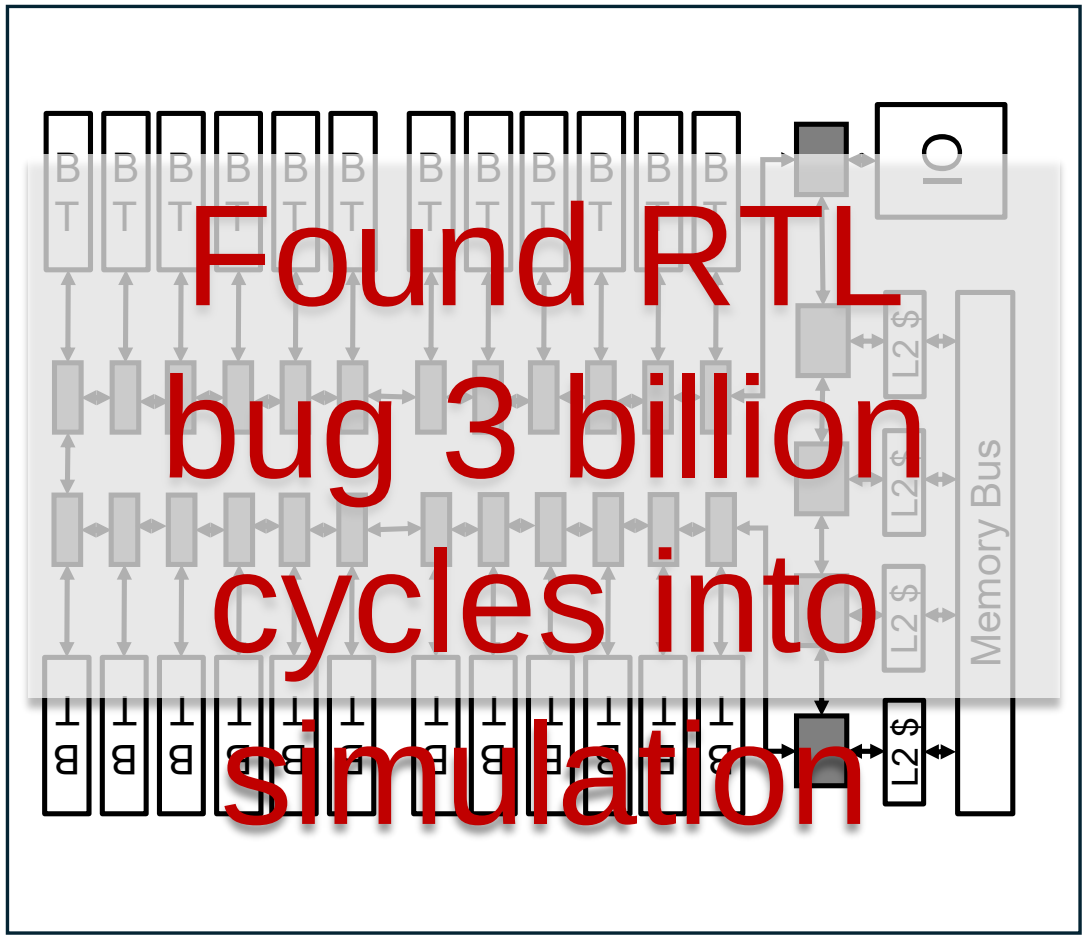




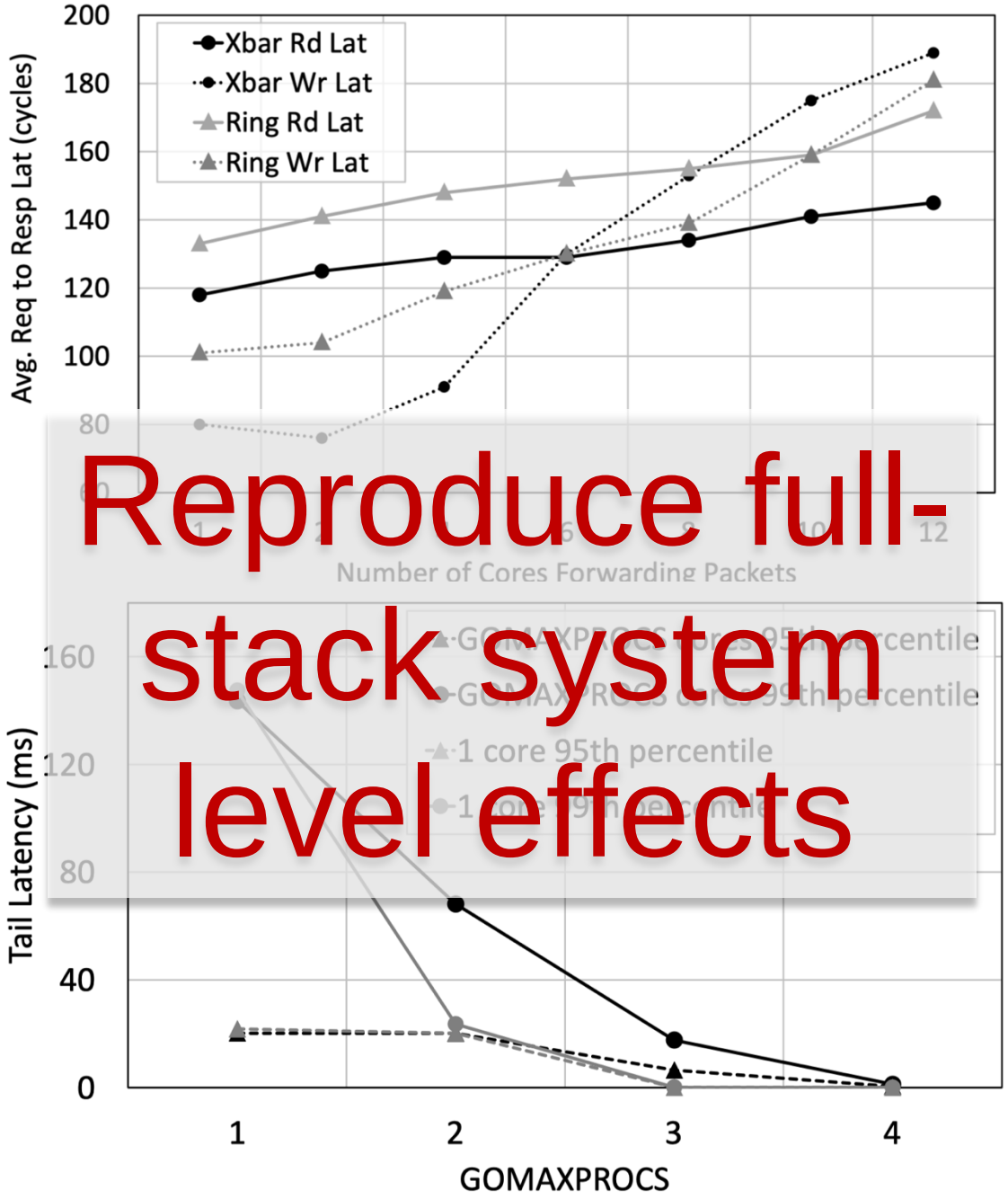


More performance results in the paper!

Case studies



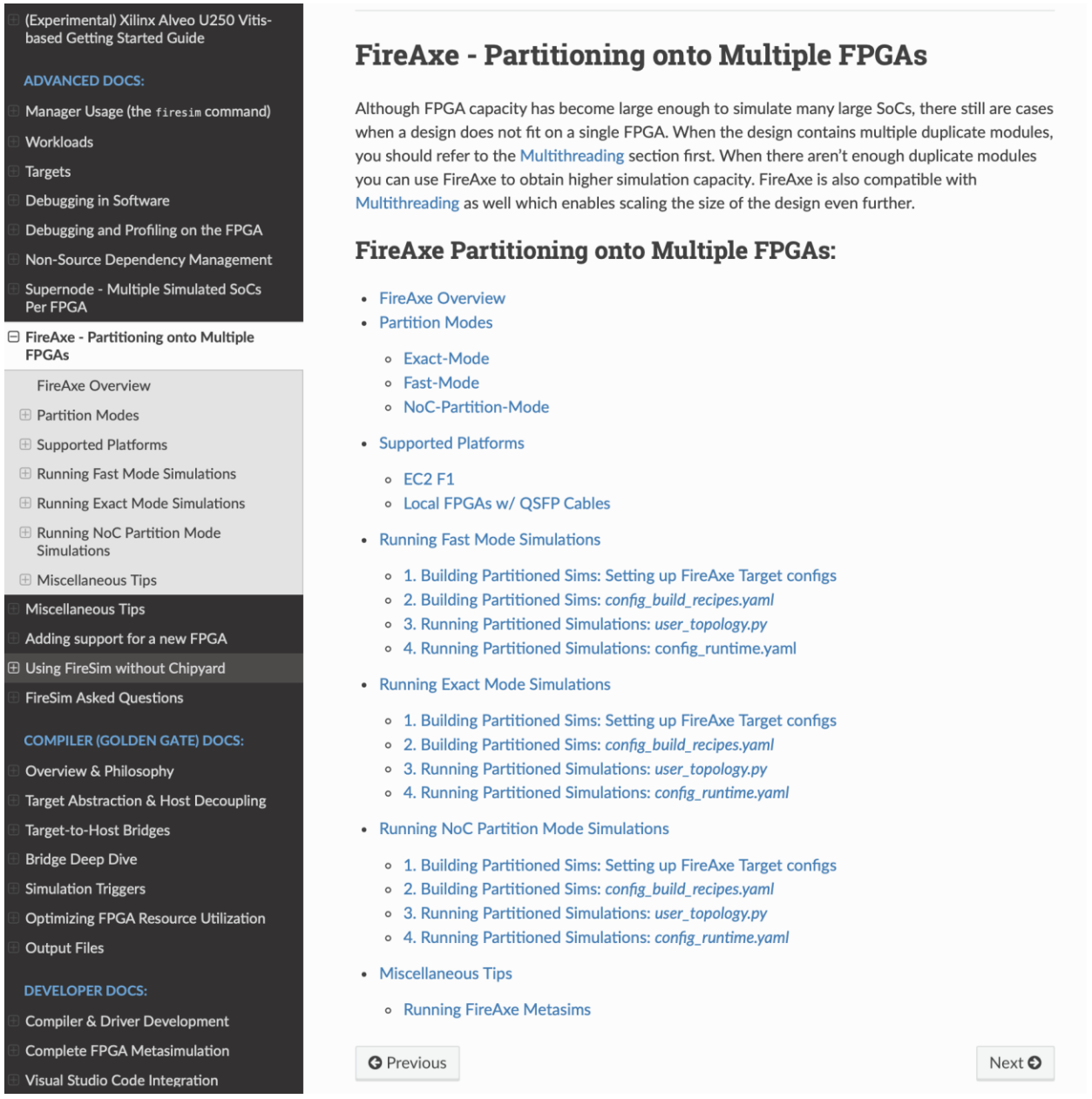
Found RTL bug 3 billion cycles into simulation



Reproduce full-stack system level effects

Extensive document support

- As all FireSim features, there is extensive documentation support
  - Setting up the F1/local FPGA instances
  - Commands & configuration
  - Examples provided for various partitioning topologies and optimization flags
  - Running FireAxe metasims for debugging



Conclusion

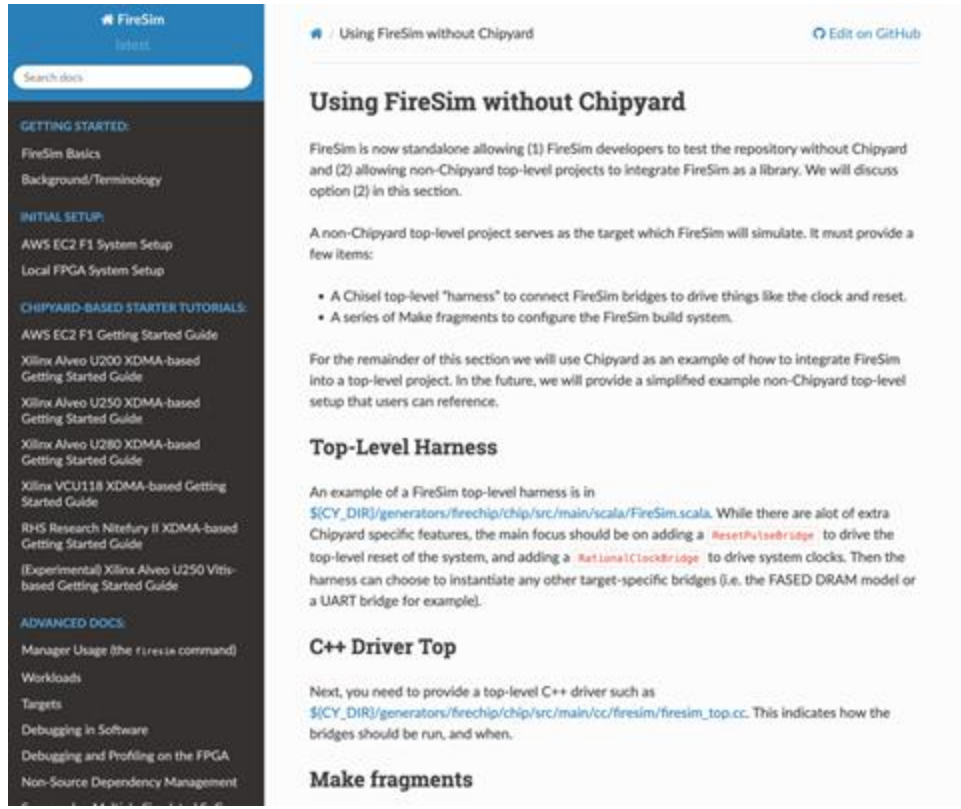
- FireAxe enables agile teams to *rapidly & accurately* model large-scale designs with minimal designer effort
  - **Dogfood-ed: Actively used in multiple ongoing projects at Berkeley**

- **Compiler automates the partitioning process**
  - Ensures functional correctness
  - Partitioning flexibility
  - Performance optimization knobs
  - Minimal code required for use

- Flexible & accessible: Cloud & on-premises FPGA support

Simulating non-Chipyard-based SoCs

- What about your own non-Chipyard design?
  - Isolated testing of single RTL component
  - Unique SoC top-level specific to your needs
  - Other unique usages
- FireSim now supports this!
  - Use FireSim like Verilator/VCS
    - FireSim is now a library decoupled from top-level
  - Cleaner API for target-specific bridges + harnesses
  - Use modern Chisel (and/or older Chisel versions)
- v2.0 release coming soon!
  - New docs on library usage + using new FPGAs
  - Examples on non-SoC top-levels



Join the community!

Questions?

Learn More:

Web: https://fires.im

Docs: https://docs.fires.im

GitHub: https://github.com/firesim/firesim

Mailing List: https://groups.google.com/forum/#!forum/firesim

@firesimproject

Email: joonho.whangbo@berkeley.edu

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0000849, by DARPA, Award Number HR0011-12-2-0016, and by NSF CCRI ENS Chipyard Award #2016662. Research was also partially funded by SLICE/ADEPT Lab industrial sponsors and affiliates Amazon, Apple, Google, Intel, Qualcomm, and Western Digital, and RISE Lab sponsor Amazon Web Services. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.