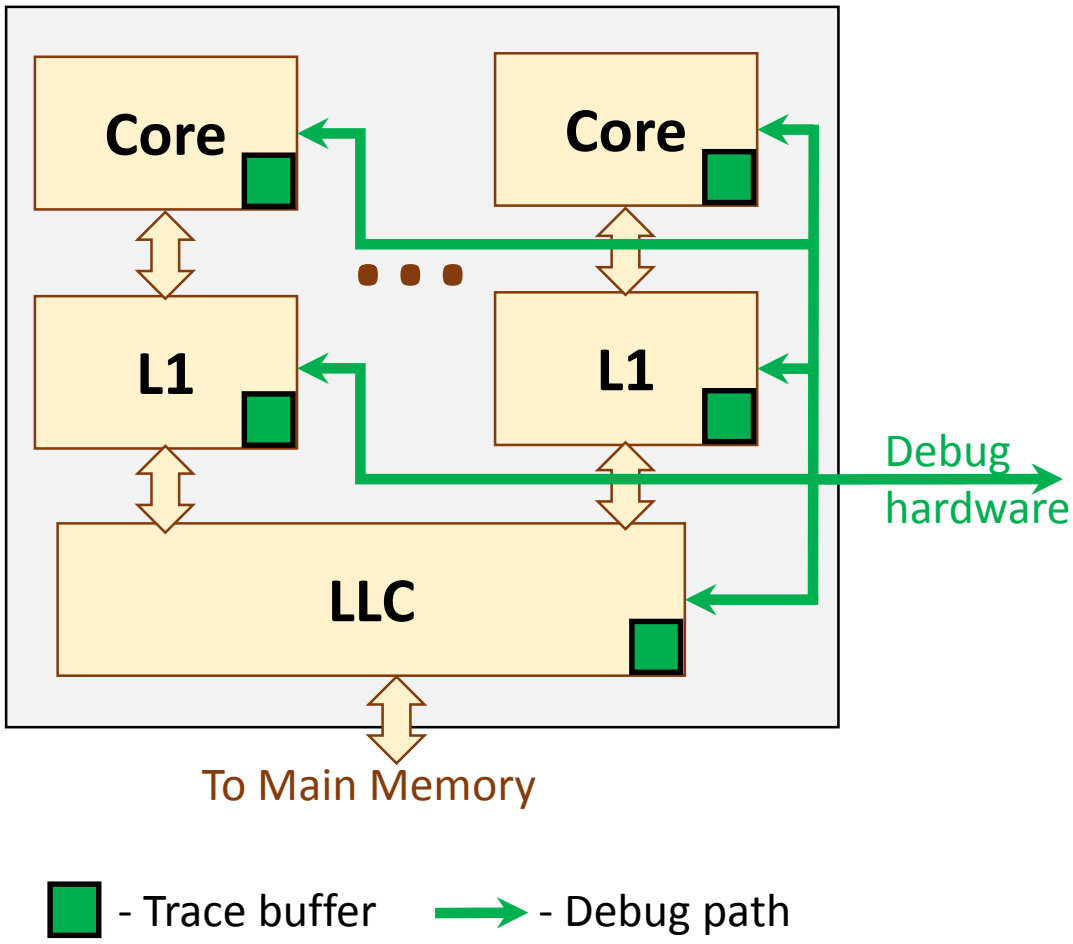
Extending Trace History Through Tapered Summaries in Post-silicon Validation

Sandeep Chandran†, Preeti Ranjan Panda†, Smruti R. Sarangi†, Deepak Chauhan‡, Sharad Kumar‡

† Department of Computer Science and Engineering, IIT Delhi, New Delhi, India

‡ Freescale Semiconductors Pvt Ltd, Noida, India

Motivation



Traditional Chips

- Increasing complexity
- Pre-silicon verification does not scale
- Higher likelihood of design bugs in first-silicon

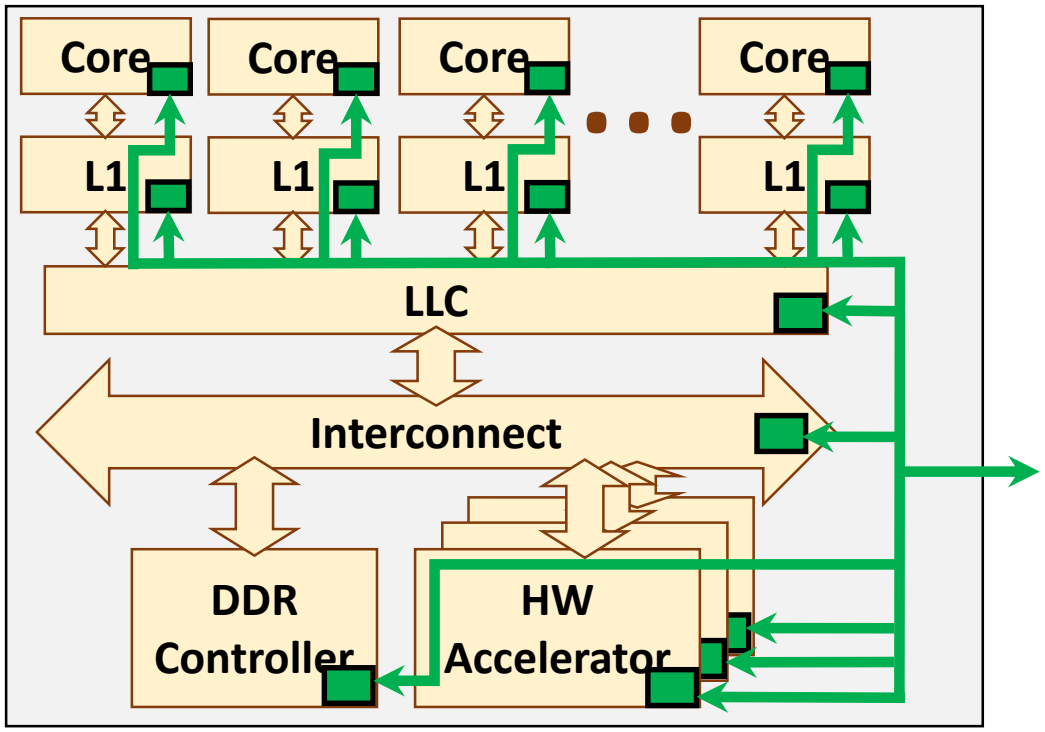
Solution

- Add hardware hooks into silicon that help debugging

Associated Problem

- Which hooks are most helpful?
- Balance visibility and area overhead

Motivation



Modern Chips

- Further increase in complexity due to tighter integration of modules

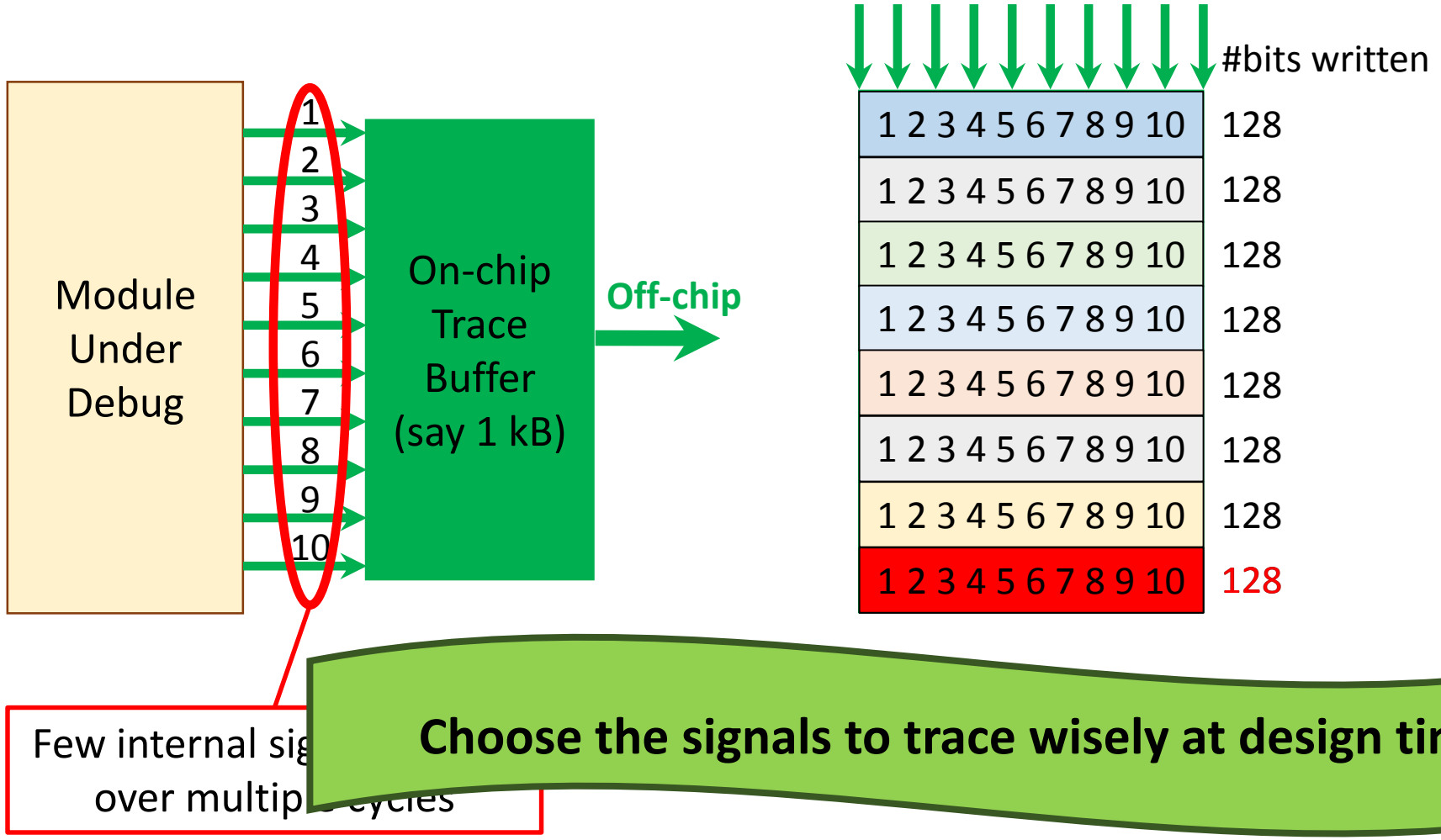
Even More Problems!

- increased inter-module interactions
- increased amount of traces to capture, transfer off-chip, and analyze

Traditional debug hardware does not scale!

- Trace buffer → - Debug path

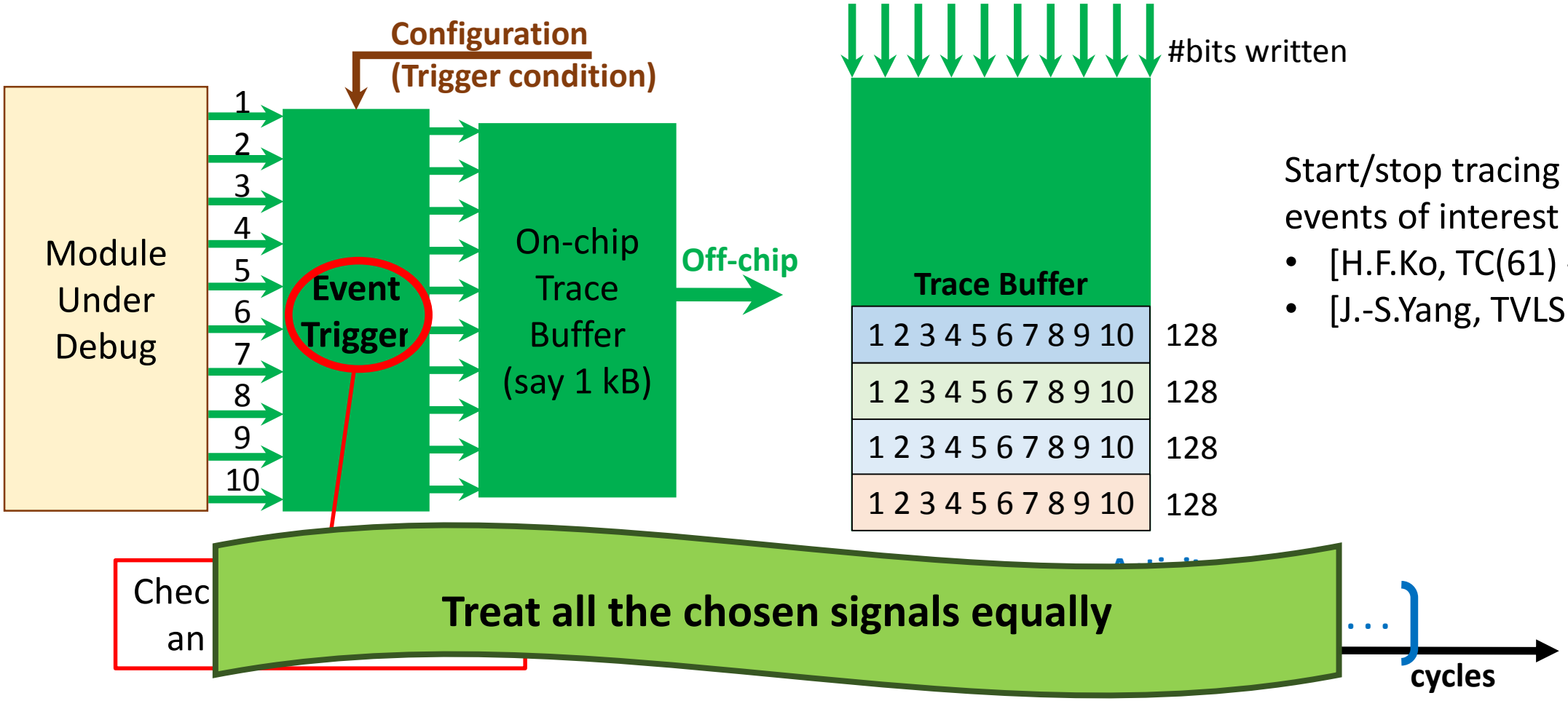
State-of-the-art



Automatically identify signals to trace

- [Q.Xu, ASPDAC-10]
- [S.Ma, ICCAD-15]

State-of-the-art



Start/stop tracing when events of interest occur

- [H.F.Ko, TC(61) - 12]
- [J.-S.Yang, TVLSI(21) - 13]

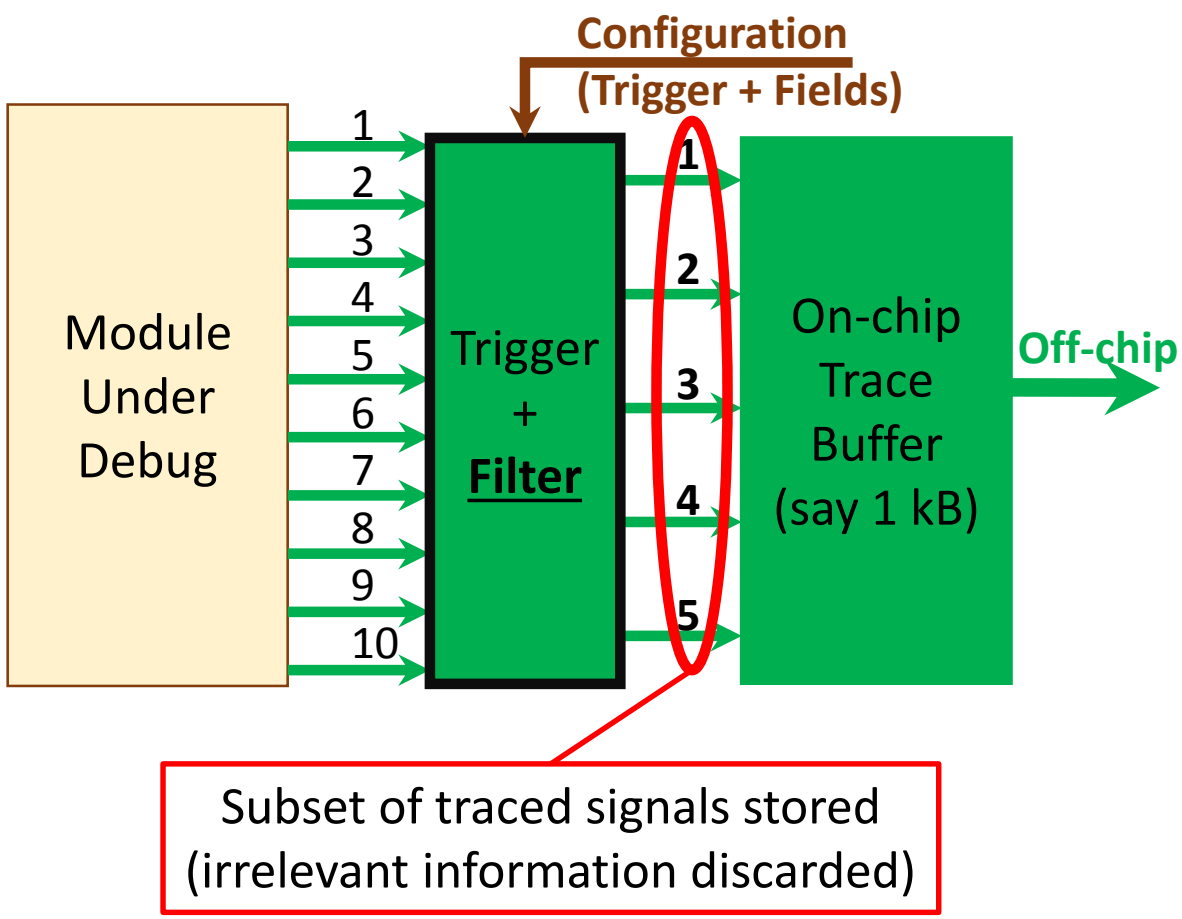
Observations

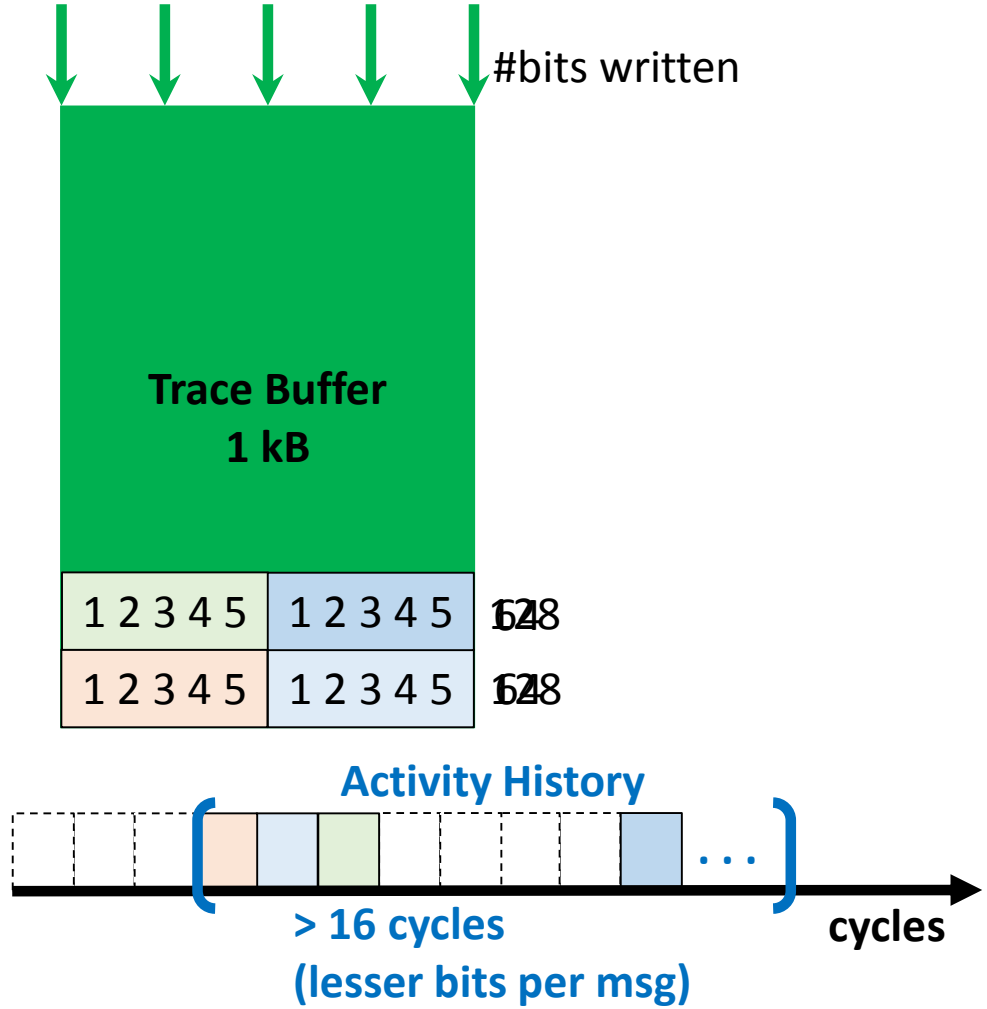
- **Sometimes, a subset of signals is more important than others**
- Depends on:
  - Debug scenario
    - Example: Signals within interconnect irrelevant when debugging pipeline
  - Time of occurrence
    - Just prior to the observed error – All information is important
    - Further in the past – Sequence leading to the error is important

Observations

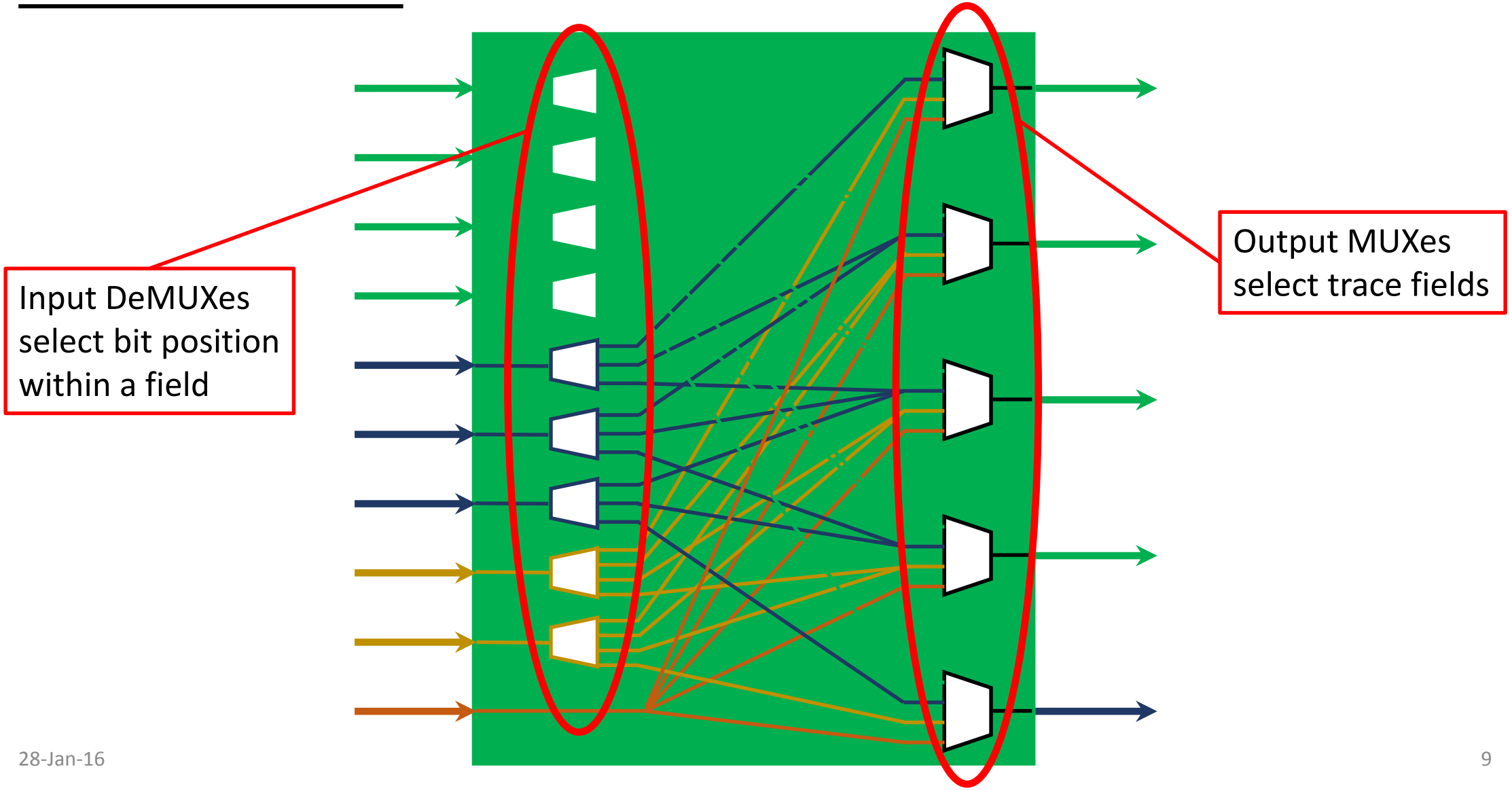
- Opportunity:
  - Tapered Storage
    - All signals for few cycles prior to observed erroneous behavior
    - Only identifying information further into the past (rest discarded)
  - Extend activity history using tapered storage
    - Space saved by discarding irrelevant information
- Challenges:
  - Select a subset of traced signals based on debug scenario
  - Efficiently store detailed traces and summaries simultaneously
  - Decide when to generate summaries of detailed traces

Filtering

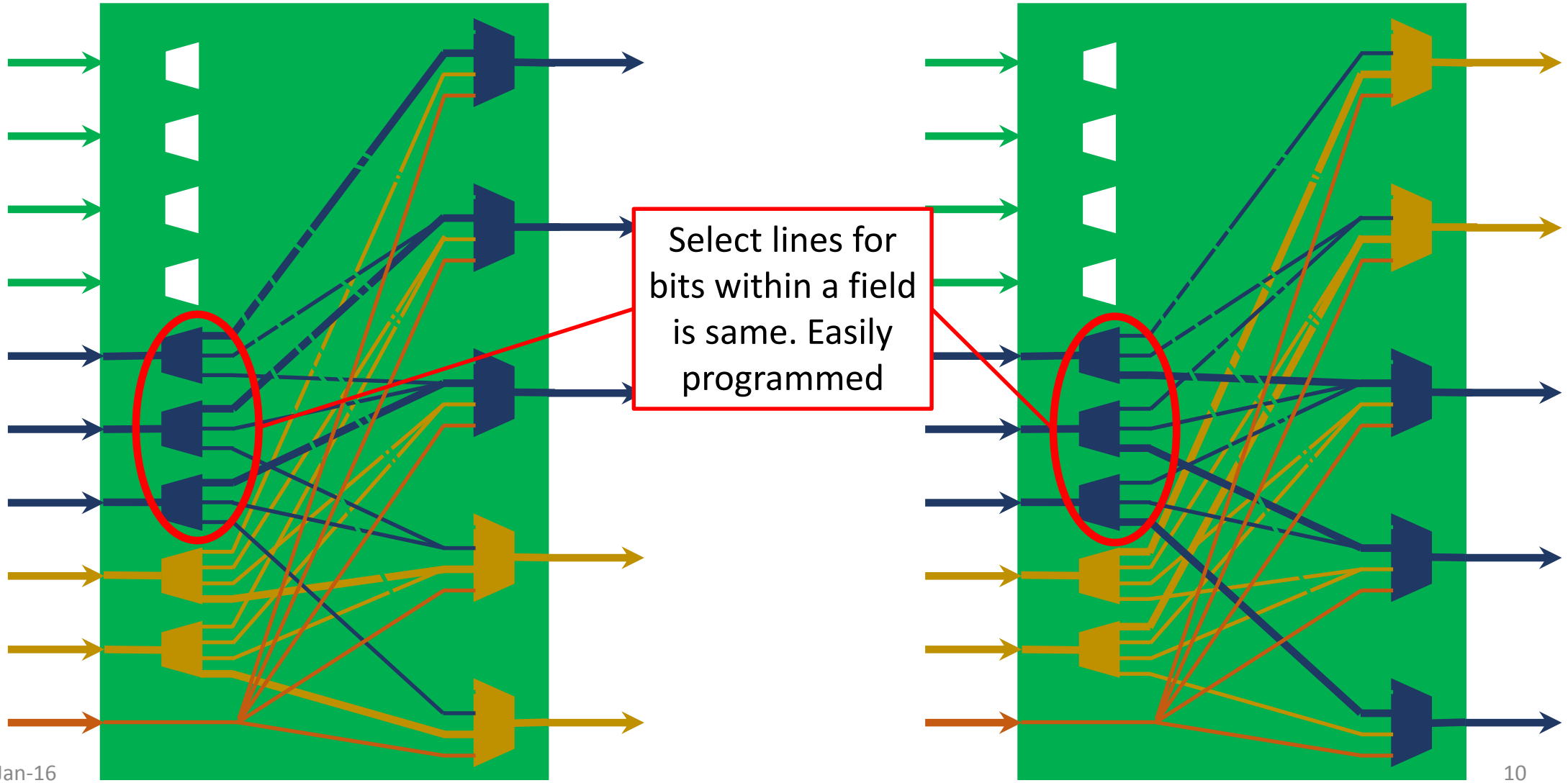

Configuration
(Trigger + Fields)
Module Under Debug
1
2
3
4
5
6
7
8
9
10
Trigger + Filter
1
2
3
4
5
On-chip Trace Buffer (say 1 kB)
Off-chip
Subset of traced signals stored
(irrelevant information discarded)


#bits written
Trace Buffer
1 kB
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
Activity History
> 16 cycles
(lesser bits per msg)
cycles

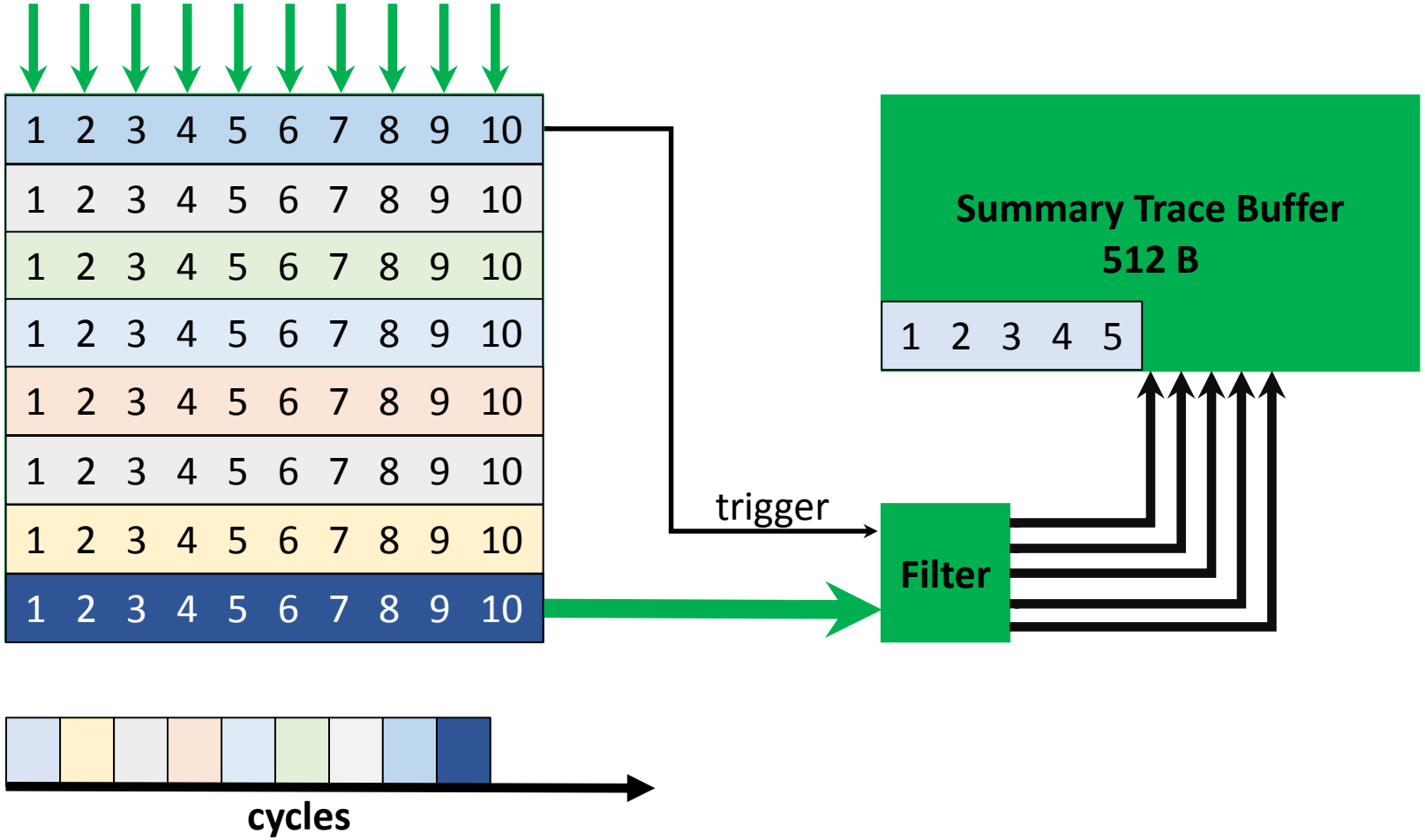
Online Filter



Online Filter

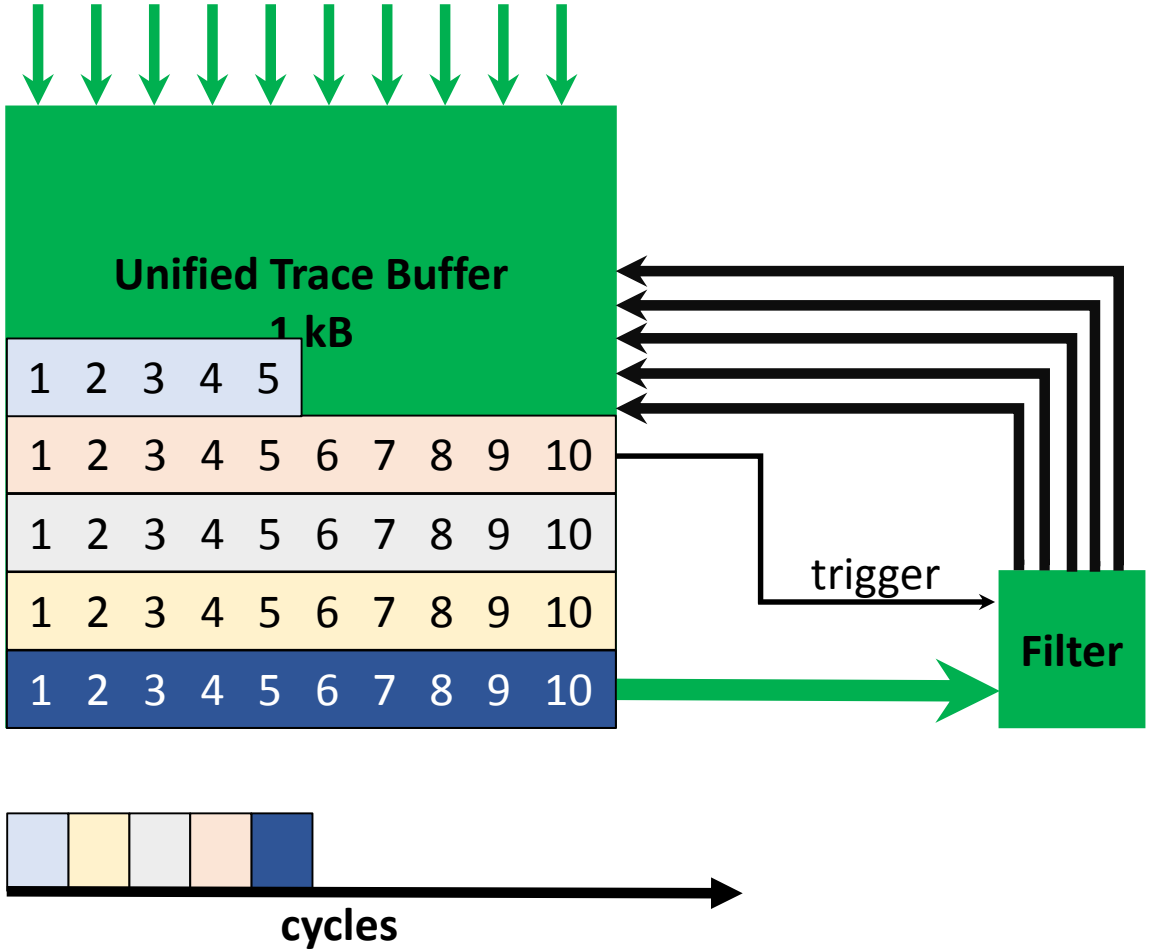


Storage Architecture (Split)



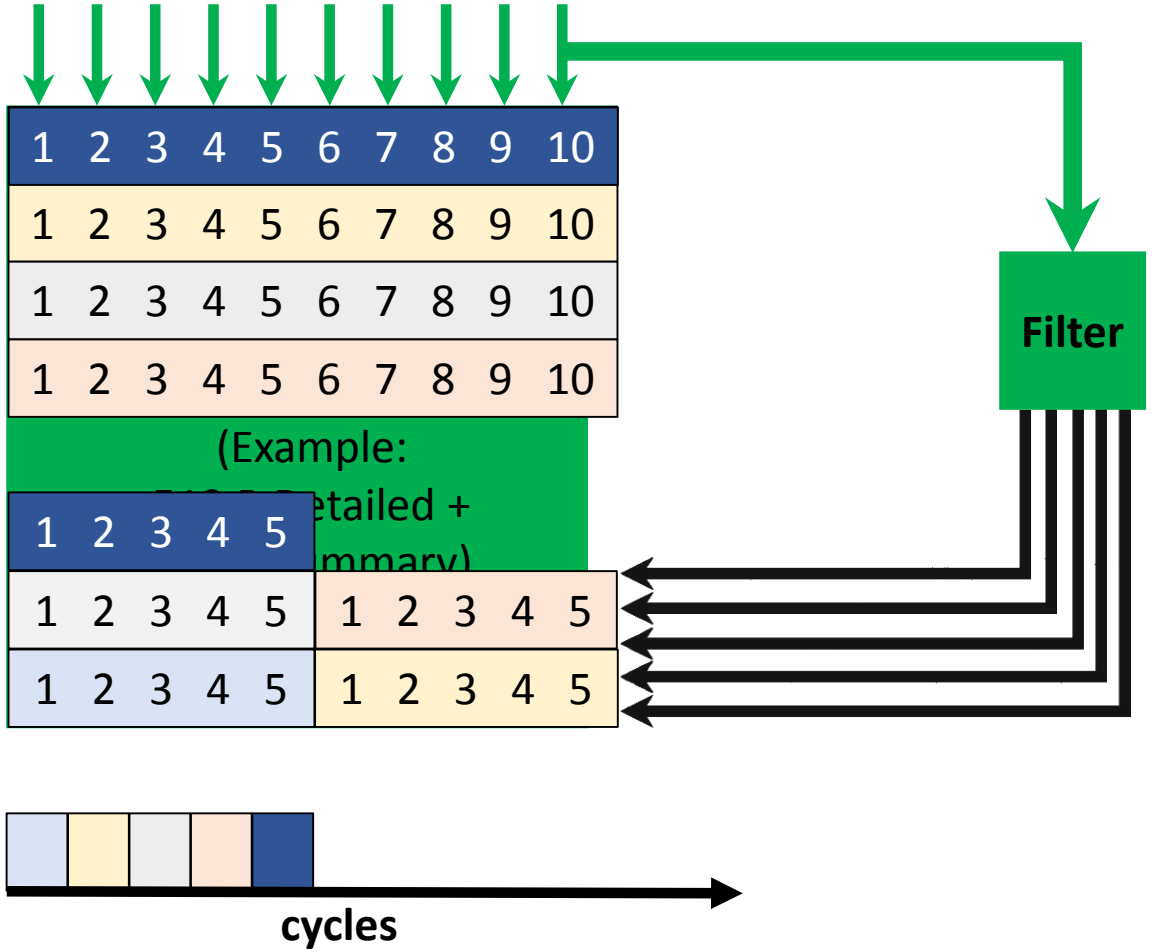
- Trace buffer sizes frozen at design time
- Fixed across debug scenarios

Storage Architecture (Unified)



- Flexible design
- Higher area overhead due to increase in the number of ports
  - 2 write ports
  - 1 read port

Storage Architecture (Overlapped)



- Flexible design
- Reduced number of ports
  - 1 write port
  - 1 r/w port
- Duplication of recent information
  - Present in detailed and summary trace buffers

Results

Setup

- Commercial LEON3 SoC (available under GPL)
- Insert existing state-of-the-art debug proposals
- Scenarios inspired by product erratas, industry

- CCI – Bug at the core-cache interface
- WIM – Bug deep inside the pipeline
- NAE, CSL – Bugs that lead to race condition

Buffer Loc	Trace Type	Detailed (#bits)	Primary (#bits)	Secondary (#bits)				Summary (#bits)				Area (mm^2)
				CCI	WIM	NAE	CSL	CCI	WIM	NAE	CSL	
Core	Instruction	128	32	-	-	-	-					0.0018
	Pipeline	151	18	32	3	32	-	114	85	114	83	0.017
	Cache	80	32	-	-	-	1					0.009
DSU	AHB	128	32	32	-	32	32	64	-	64	64	0.013

Generated summaries vary with debug scenario

- One filter per trace type
- Total area (incl trace buffer of 4kB) = 2.27% of 64kB cache

Results

Bug	# Stalls			History (cycles)		
	Detailed (4 kB)	Summary (3 kB)	Summary (2 kB)	Detailed (4 kB)	Summary (3 kB)	Summary (2 kB)
CCI	138	51	77	1332	3491	2374
WIM	94	36	54	1109	2085	1916
NAE	138	51	77	1332	3491	2374
CSL	138	52	78	1332	3534	2307

- Number of stalls reduced by 63%
- Transfer times reduced from 100.72s to 29.7s *

- 162% increase in activity history

* Uses a serial link operating at 115200 bps

Conclusions

- Proposed new debug methodology that uses combination of detailed trace messages and their summaries
  - Extends trace history by upto 162%
  - Minimal additional area – 2.27% of a 64kB cache (including 4kB trace buffer)
  - No assumptions about the type of error
  - Supports at-speed debugging of different classes of bugs like non-repeatable errors, bugs that have long suspect windows

Thank You!