Reprogrammable Redundancy for Cache V_{min} Reduction in a 28nm RISC-V Processor

Brian Zimmer, Pi-Feng Chiu, Borivoje Nikolić, and Krste Asanović

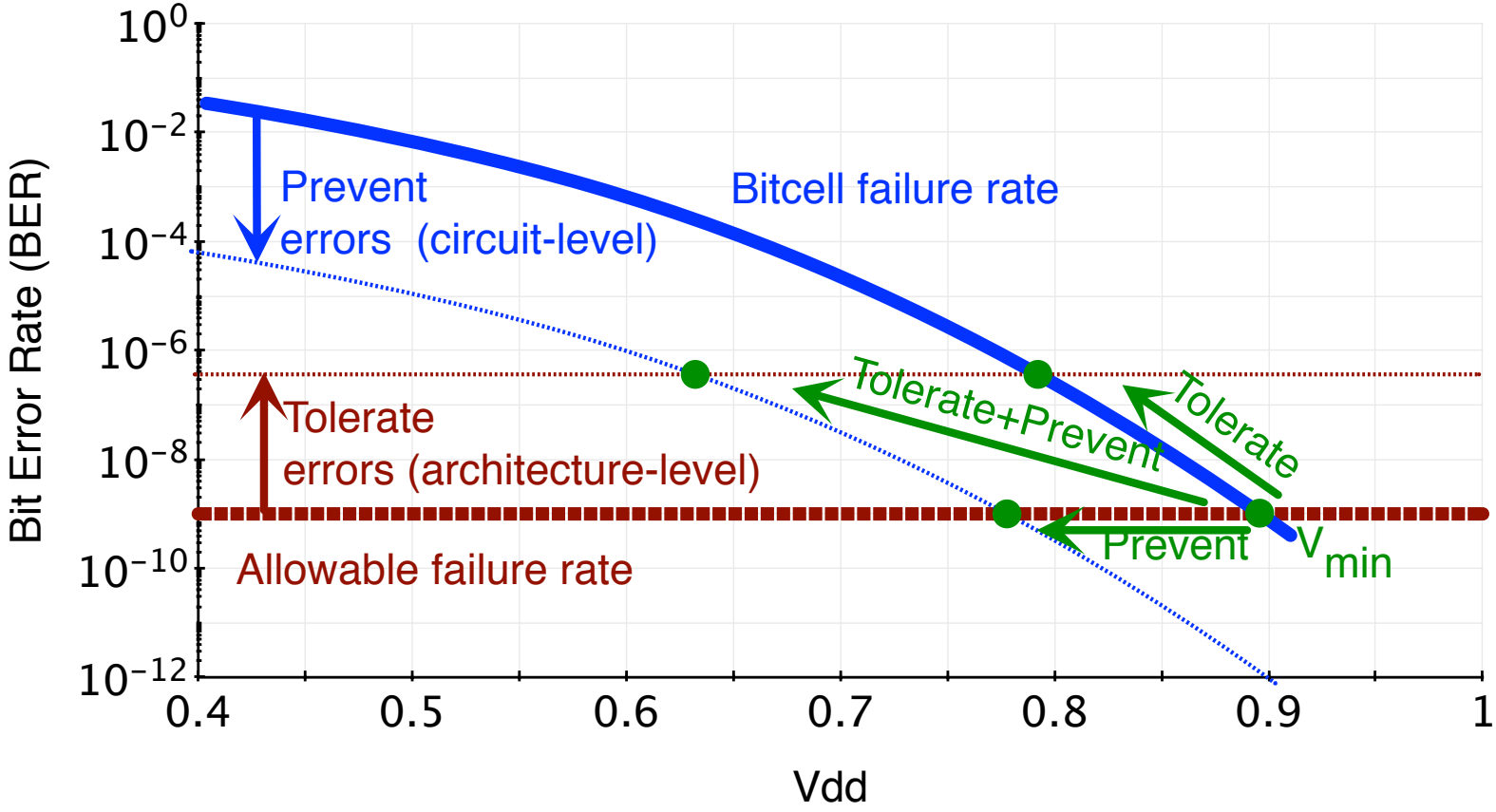
Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley

5th RISC-V Workshop, Nov. 29-30, 2016

Motivation

- **Voltage scaling is effective in reducing energy consumption**
- **SRAM limits minimum operating voltage (Vmin)**

Approach



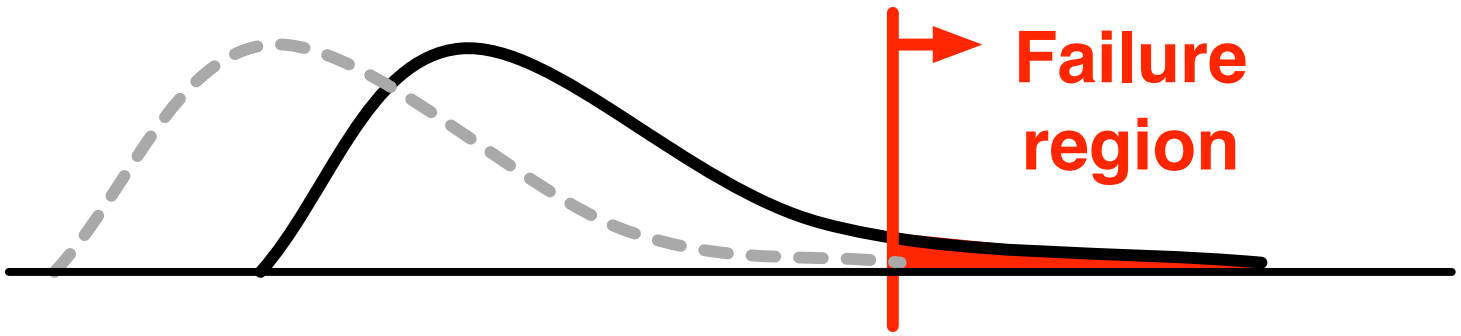
- **Instead of <u>preventing</u> errors, <u>tolerate</u> errors**
- **Significant Vmin reduction possible by tolerating ~1000s of errors per MB**

Related work

- Circuit-level (prevent)
  - Assist circuits (negative bitline, wordline underdrive, etc)
  - Cell upsizing
- Architecture-level (tolerate)
  - SECDED ECC
  - Fused column redundancy
  - Line disable

Chip Goals

- **Prove that SRAM Vmin can be effectively lowered by tolerating a reasonable number of failing bitcell**
- **Intuition: target tail of distribution**



RISC-V

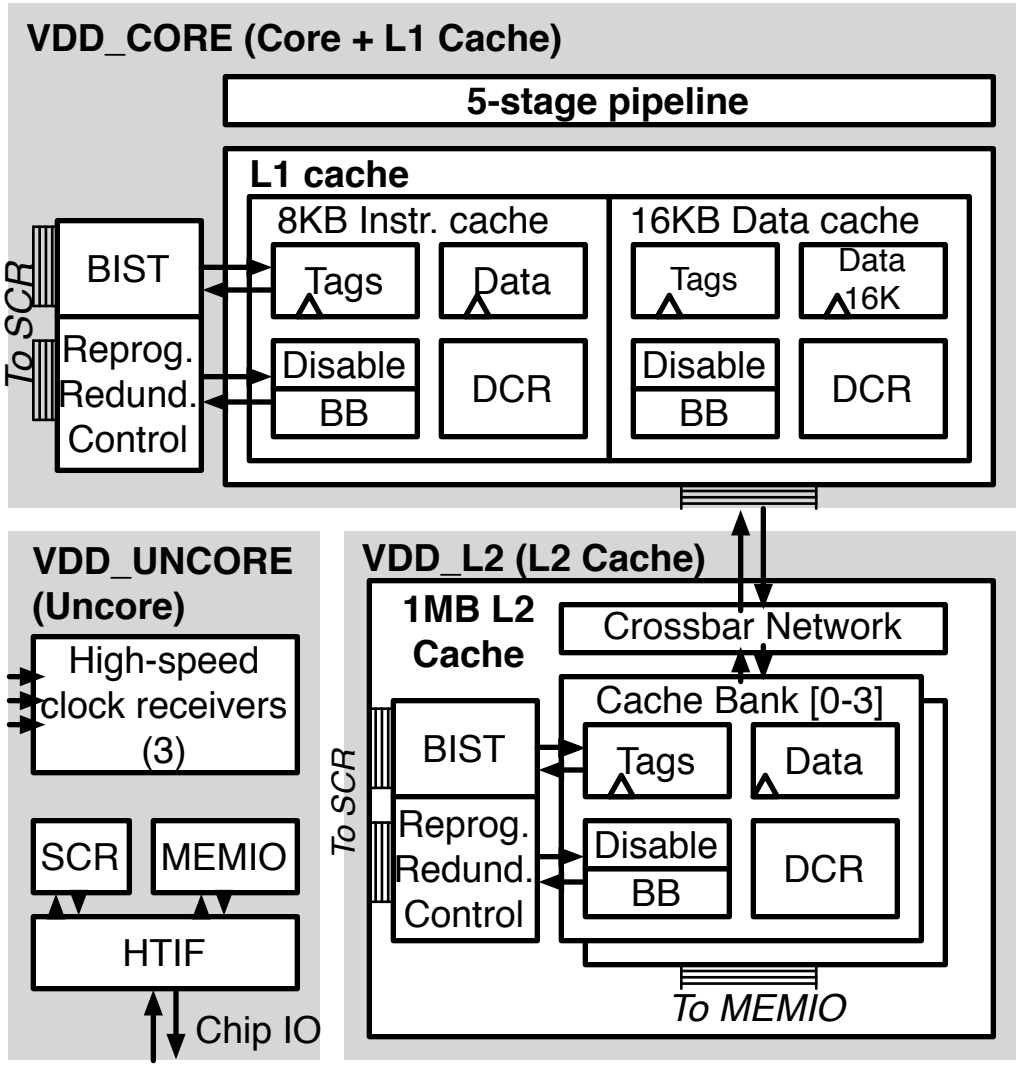
- Want real silicon evaluation
- Rocket chip generator perfect platform to build experiment on
  - Realistic SRAM usage/constraints
  - Software toolchain for testing
- Modified caches to add reprogrammable redundancy, ECC, and BIST

Implemented Techniques

- **Three techniques:**
  1. **Dynamic column redundancy (DCR): avoid single-bit errors in data SRAM**
  2. **Line disable (LD): avoid >=2 bit errors in data SRAM**
  3. **Bit bypass (BB): avoid all errors in tag SRAM**

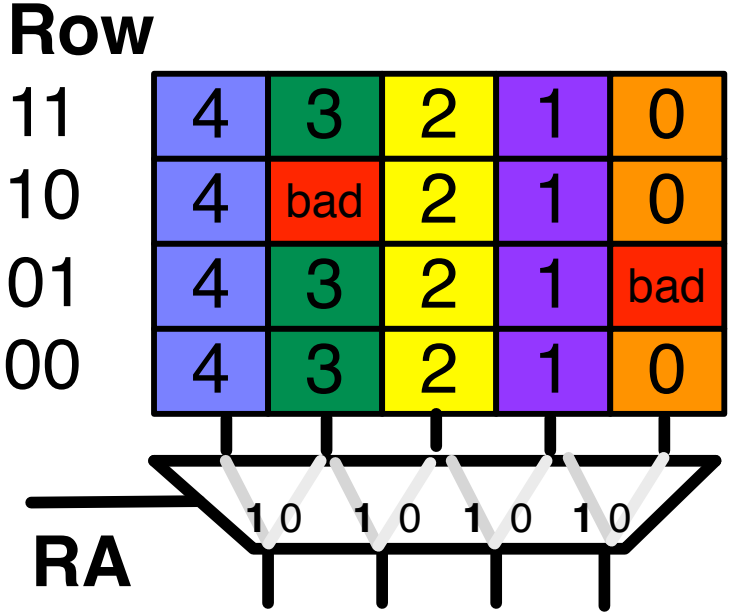
System Architecture

- **Target system: RISC-V "Rocket" scalar processor**
- **16KB L1 and 1MB L2 cache protected by DCR, BB, and LD**
- **ECC monitors SRAM to ensure all errors are avoided**



Dynamic Column Redundancy (DCR)

Structure:

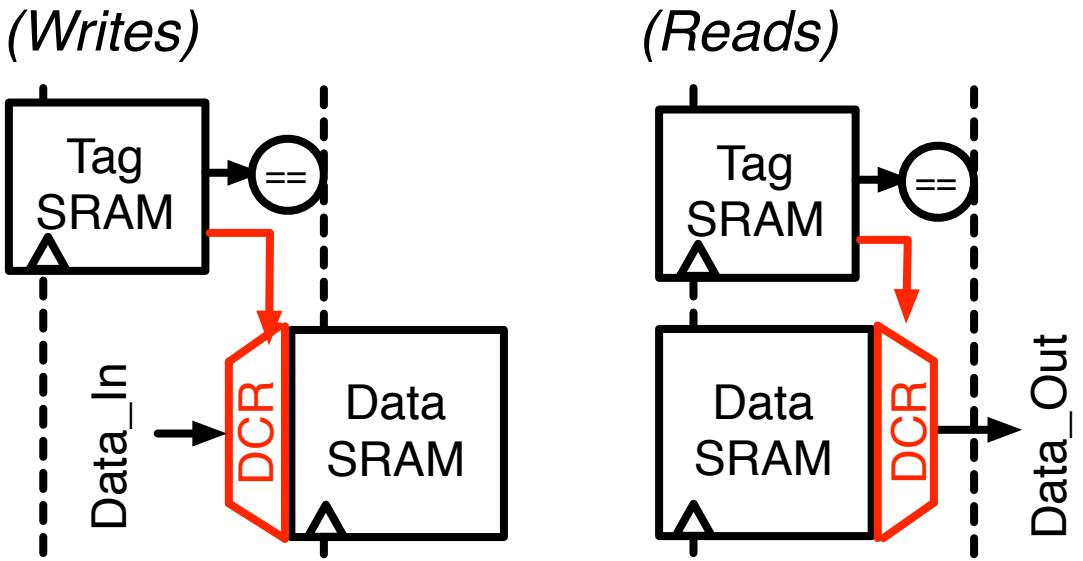


Example Accesses:

Row	RA	x
11	0000	3 2 1 0
10	1000	4 2 1 0
01	1111	4 3 2 1
00	0000	3 2 1 0

- **Traditional column redundancy, but different mux address per row**
- **Bits per RA is programmable**

DCR Implementation



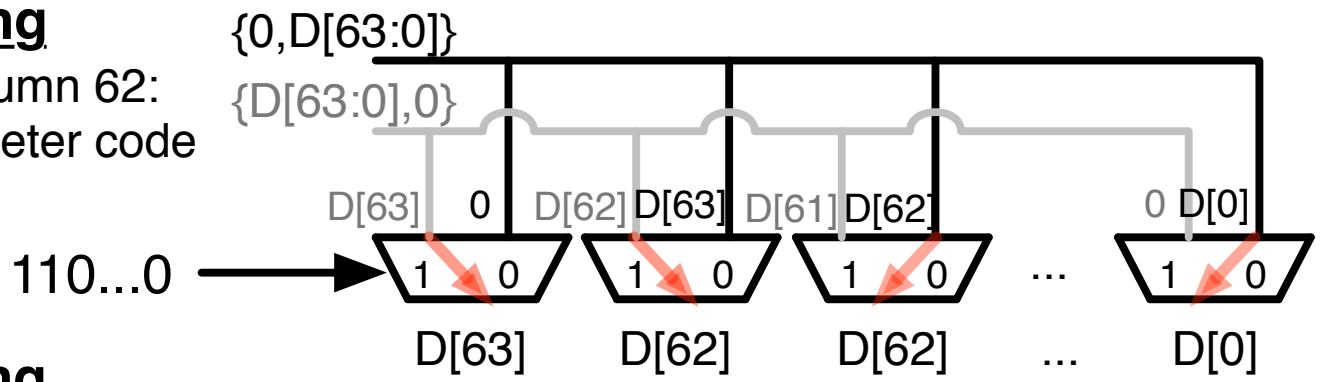
- **Redundancy address stored in tag array**
  - **Lookup is part of tag access and does not impact critical path**
- **Small timing overhead for shifting only**
- **Most area overhead inside tag array**

DCR Details

Example: 64 bit word (D[63:0]), avoid bit at index 62

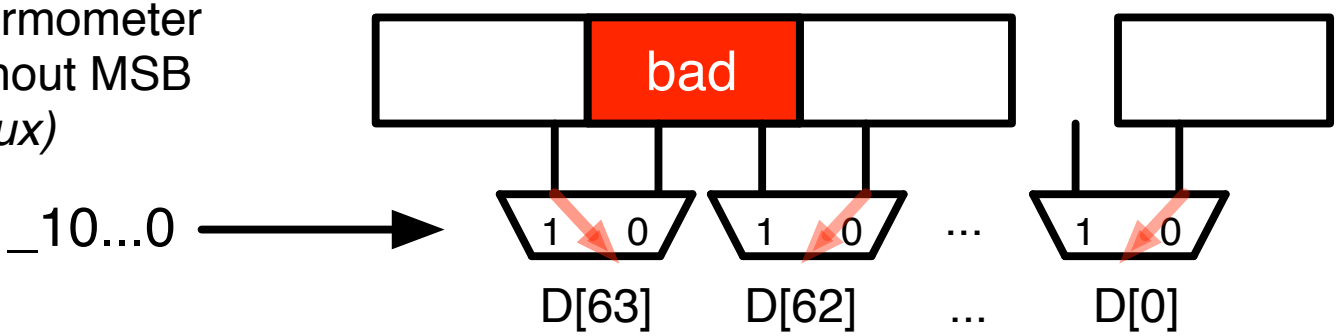
Encoding

Avoid column 62:
Thermometer code



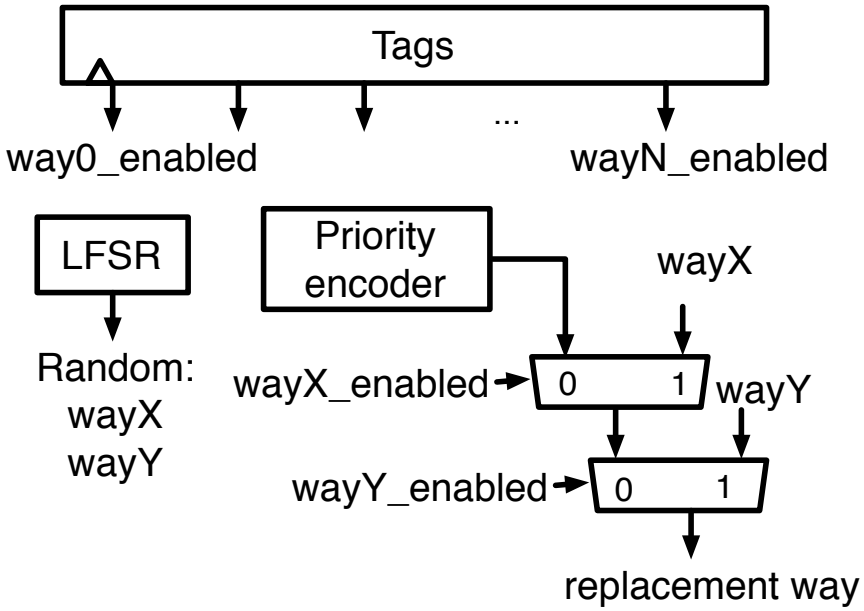
Decoding

Same thermometer code, without MSB
(1 less mux)



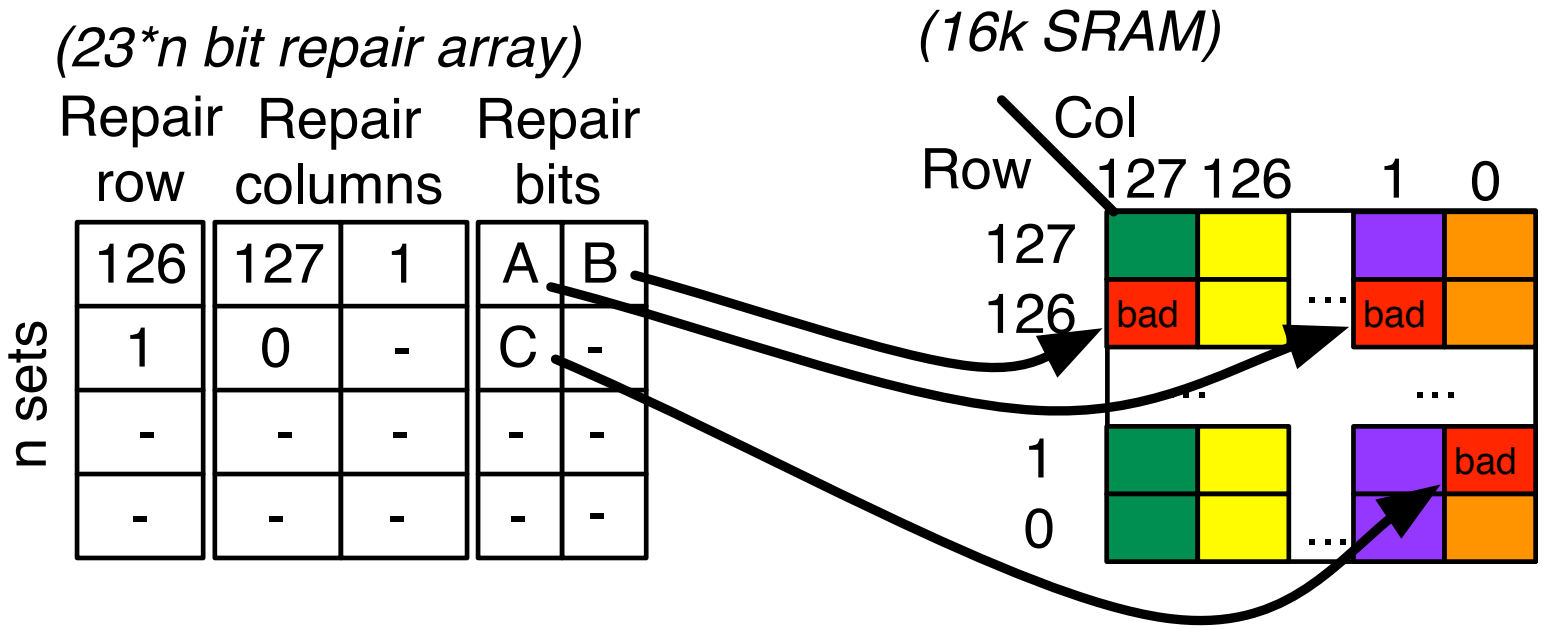
- **Single 2:1 mux on critical path**

Line Disable (LD)



- **Multibit failures limit Vmin, but extending DCR to handle multibit faults is too expensive**
- **Multibit failures are rare, can pay large cost to avoid -> disable an entire way**

Bit Bypass (BB)

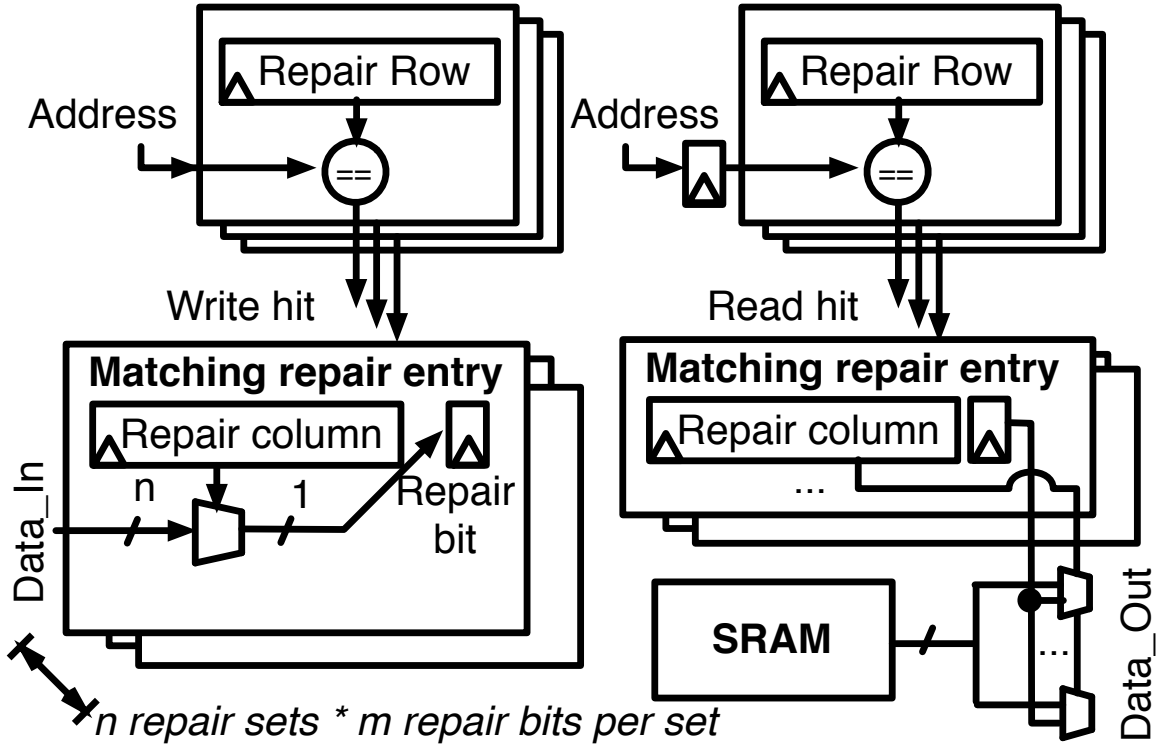


- **Tag arrays cannot have faults**
- **Use standard cell flip-flops to avoid failing bitcells**

BB Implementation

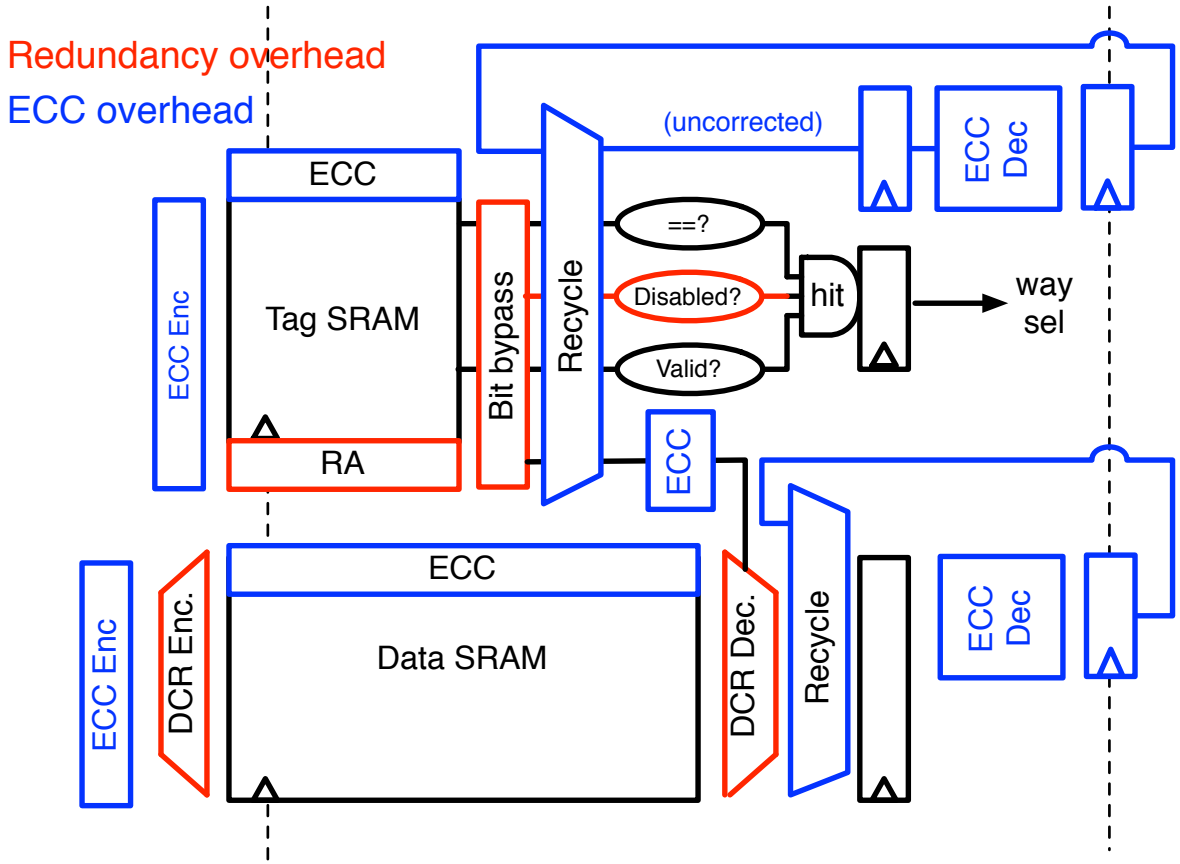
Writes: If writing to an address with a bad bit, store the correct bit value

Reads: If reading from an address with a bad bit, replace output with the correct value



L1 tags: Repair up to 7 addresses (n) with 2 failing bits each (m)
L2 tags: Repair up to 22 addresses (n) with 2 failing bits each (m)

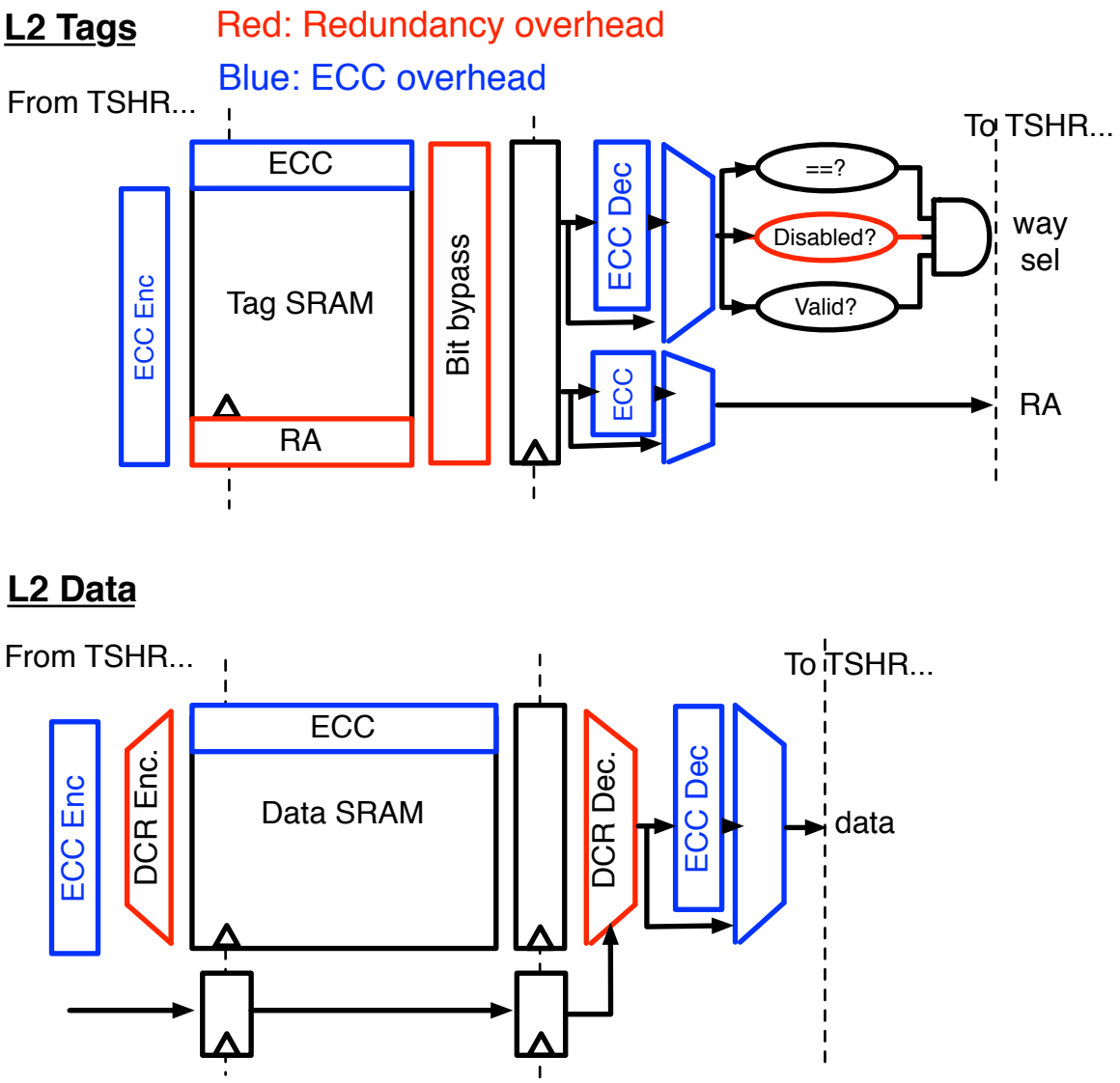
L1 Data Cache Pipeline



- **RR easy to add, but ECC difficult to add**
  - **ECC decoding is pipelined**
  - **If error detected, operation is recycled**

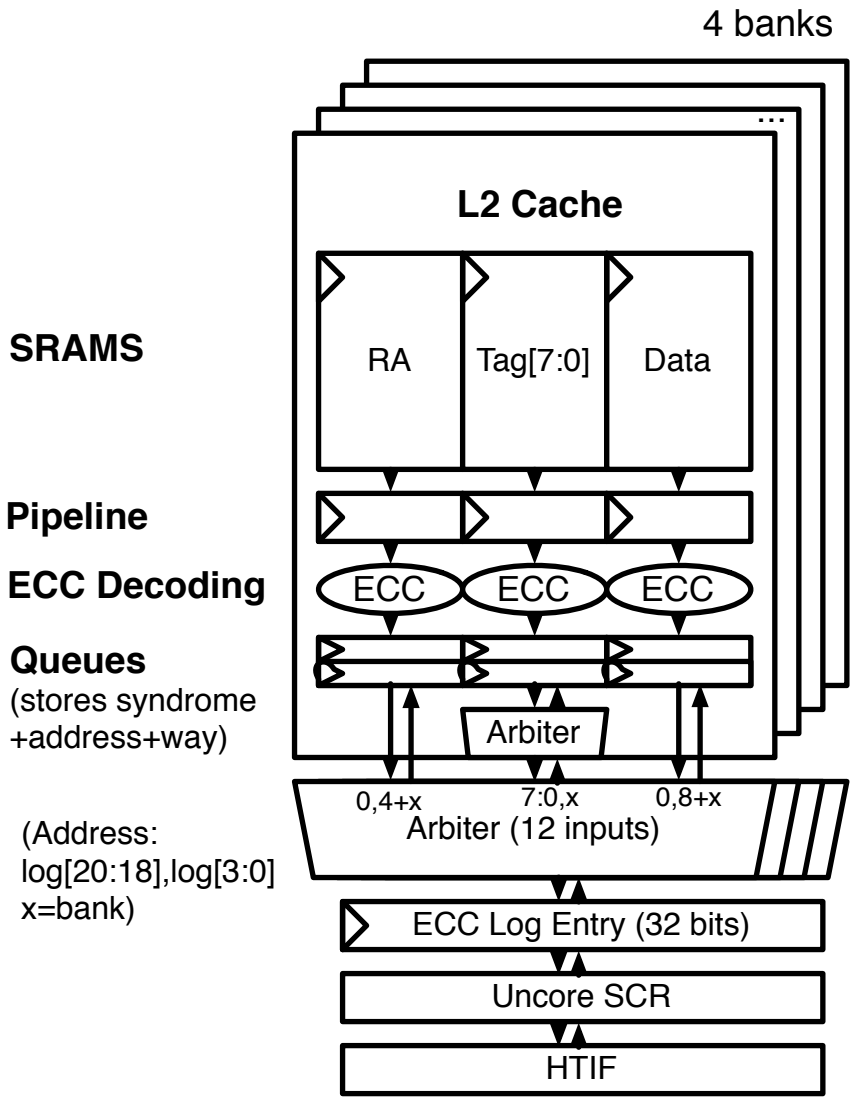
L2 Cache Pipeline

- **Adding latency by pipelining ECC isn't a problem**
- **Serial tag and data access**

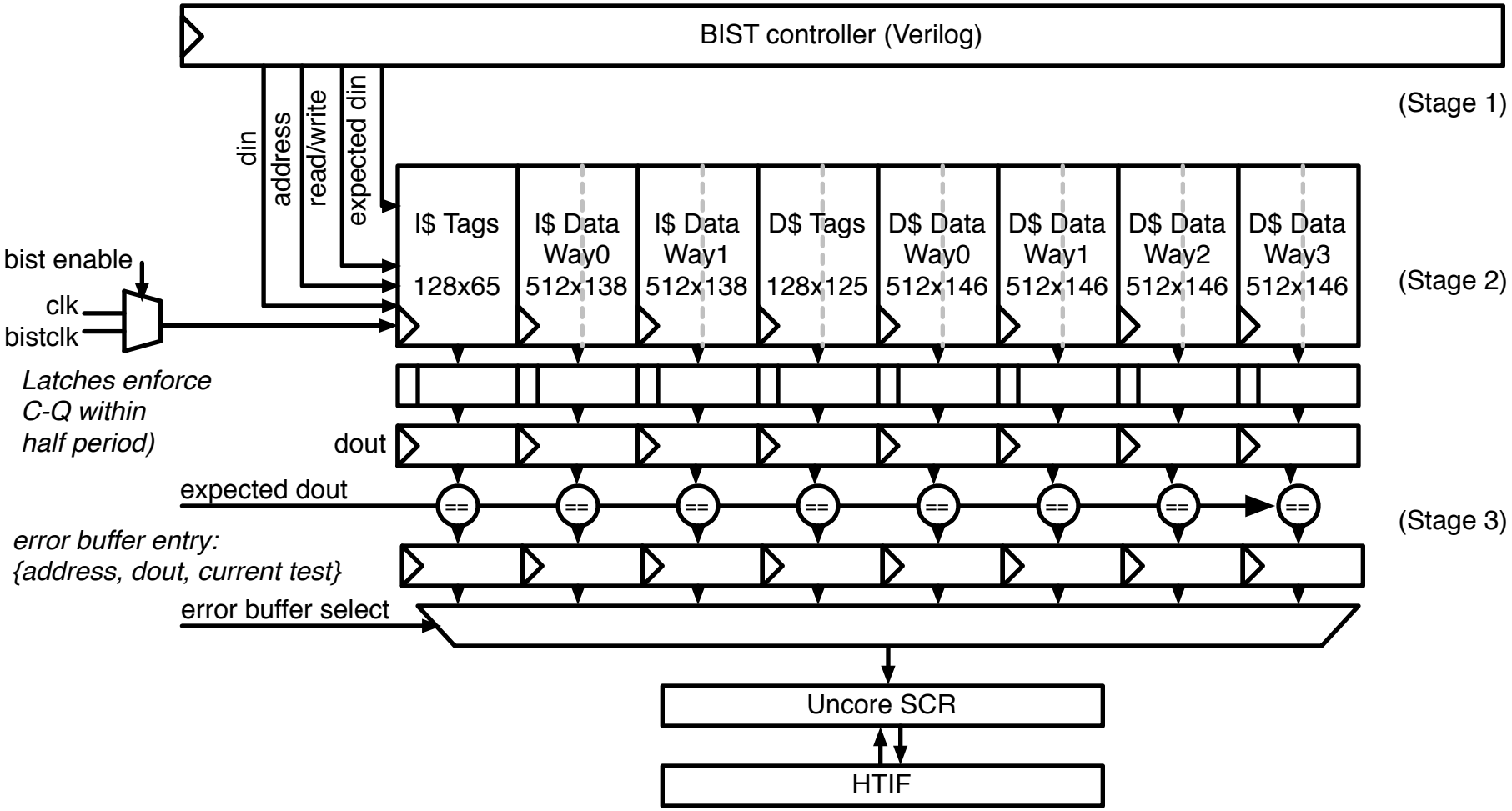


ECC logging

- **Queue ECC errors**
- **Arbitrate from every SRAM array**
- **Read through HTIF**



BIST Interface



- **SRAMs tested in parallel**

Comparison to prior art

Technique	Maximum BER	Protects tags?	Area Overhead		Total Cache Overhead
			Tags	Data	
Proposed (BB +DCR+LD) §	9.8×10^{-5}	Yes	BB: 15% DCR: 6% LD: 4%	DCR: 0.7%	2.2%
Extreme Redundancy**	2.4×10^{-5}	Yes	SC: 6.4% Flops: 0.5%	SC: 6.4% Flops: 0.5%	6.6%
Line Disable § [3]	1.8×10^{-5}	No	4% #	-	0.2% #
ECC† [4]	1.3×10^{-6}	Yes	7% #	7%	6.7% #
Static Redundancy* [2]	4.4×10^{-7}	Yes	SC: 1% # Fuses: 0.12%	SC: 1% # Fuses: 0.12%	1.1% #
Nominal	1.1×10^{-10}	No	-	-	-

§ Up to 1% disabled **1 column repair/KB and 1 row repair/32KB

†1 repair/128 bit *10 repairs/MB

Estimate (not reported), fuse=100× bitcell, flop=10× bitcell

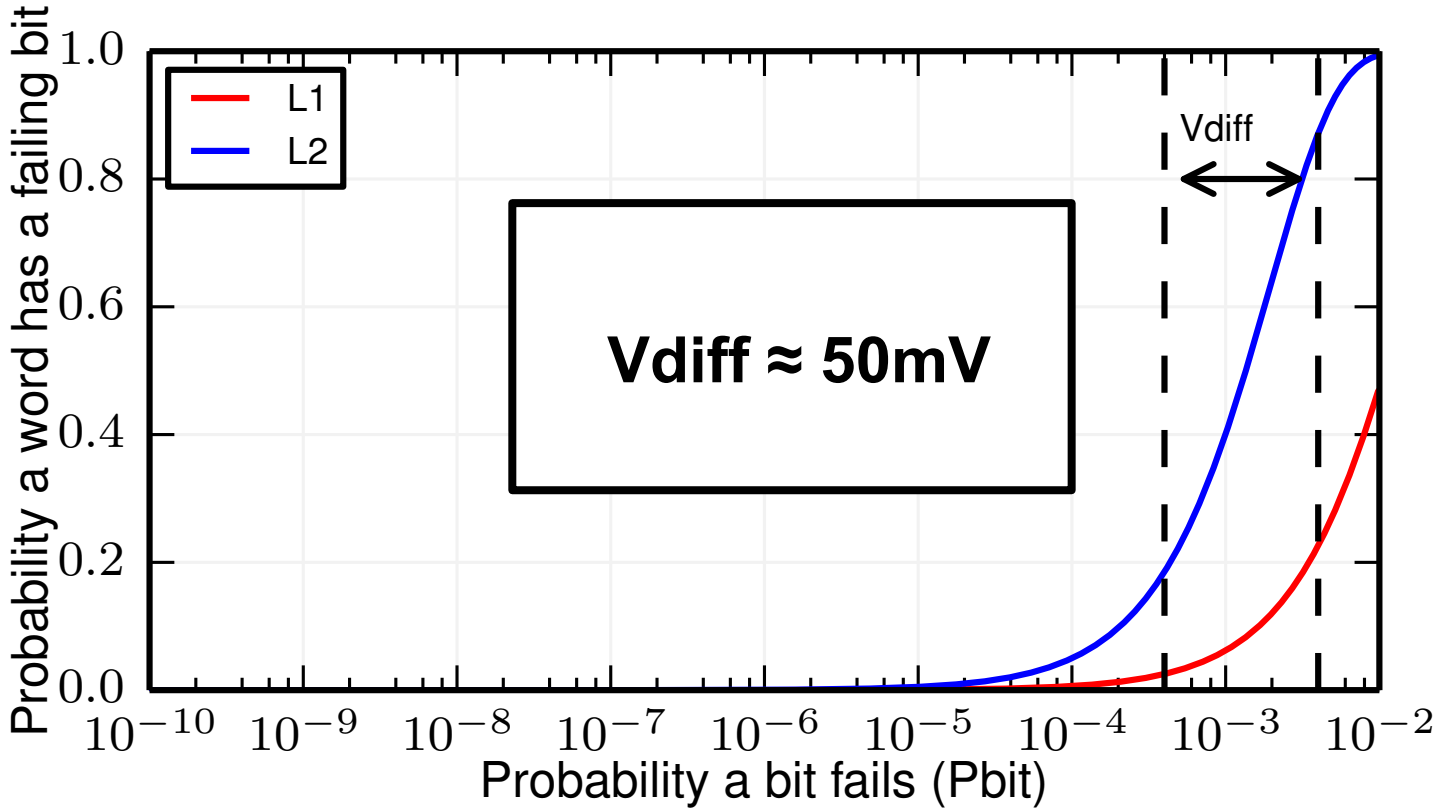
[2] Huang JSSC 2013 [3] Chang JSSC 2007 [4] Sawant ISSCC 2011

Area Overhead Breakdown

Area Overhead		Total Cache Overhead
Tags	Data	
BB: 15% DCR: 6% LD: 4%	DCR: 0.7%	2.2%

- **Data array (L2): 137 bits + 1 redundant column**
- **Tag array (L2): 216 bits + 8 bits (LD) + 13 (DCR)**
- **BB: 15% larger than tags for additional standard cells**
- **Tag is 6% of L2, data is 90% of L2**

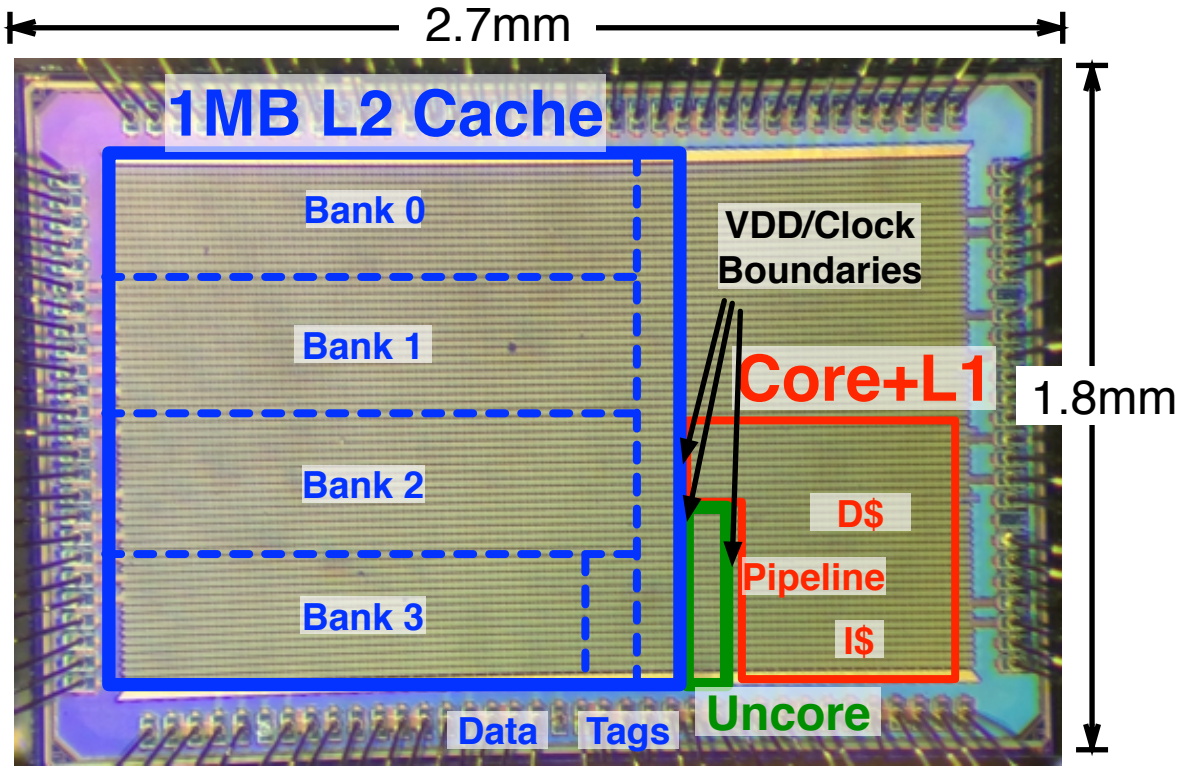
Optimal BER



- **Techniques are intentionally optimized to tolerate BER of 1×10^{-4}**
- **In 1MB cache, ≈1000 errors**

Fabricated Prototype

- TSMC 28nm HPM
- 2.7mm x 1.8mm, wire-bonded
- RISC-V core + 1MB L2 Cache

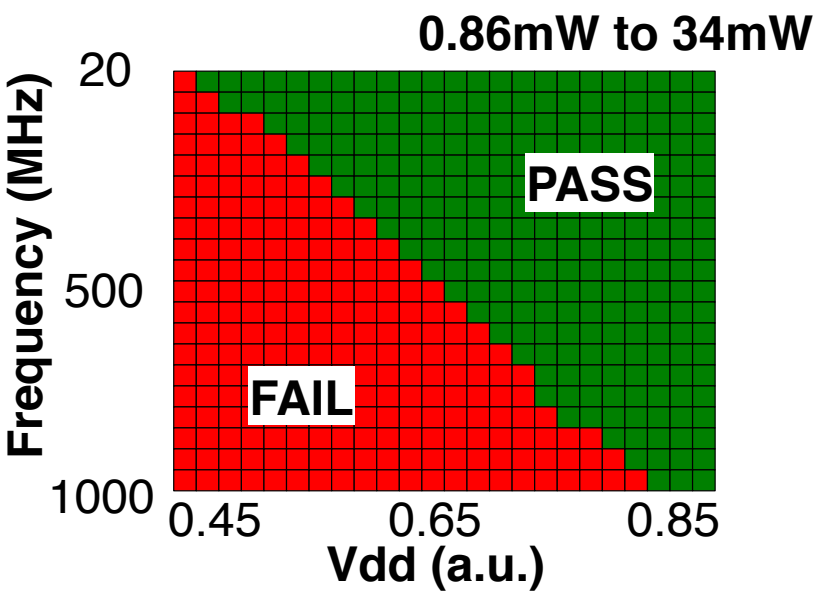


Design Flow

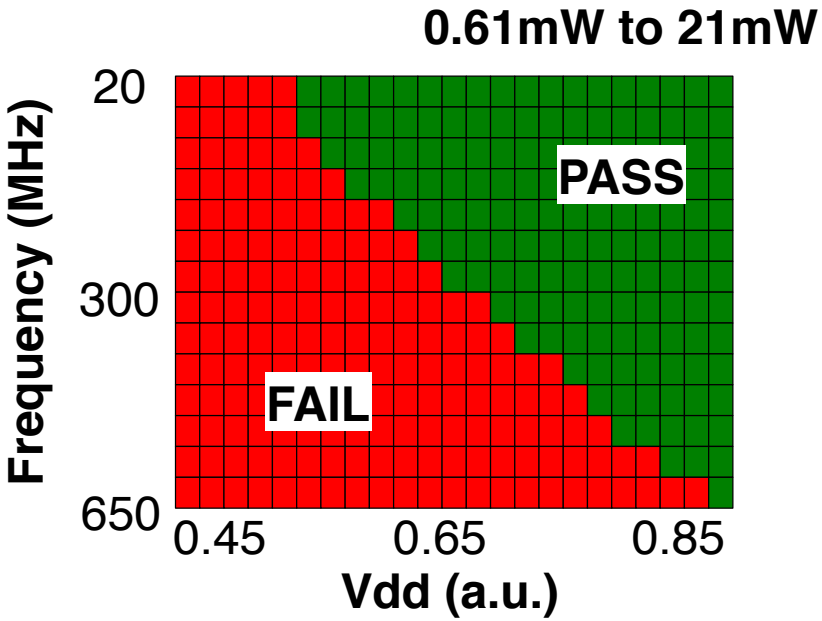
- TSMC provided standard cells and SRAM compiler
- Chisel + Verilog wrappers for BIST, multi-clock support
- Synthesis: Synopsys Design Compiler
- Place-and-Route: Synopsys IC Compiler
- Timing: Synopsys Primetime
- Simulation: Synopsys VCS

Measured Shmoo plot

L1 Cache + Core

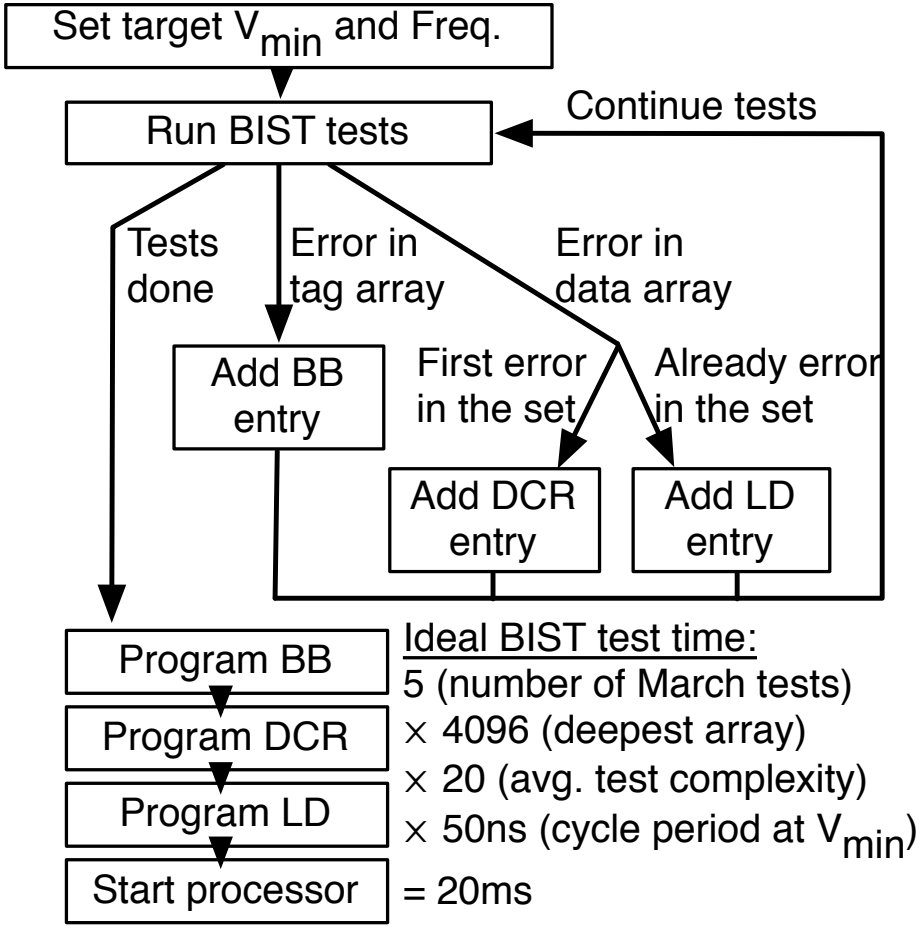


1MB L2 Cache



- **Processor core and L1 operate up to 1GHz and 30mW**

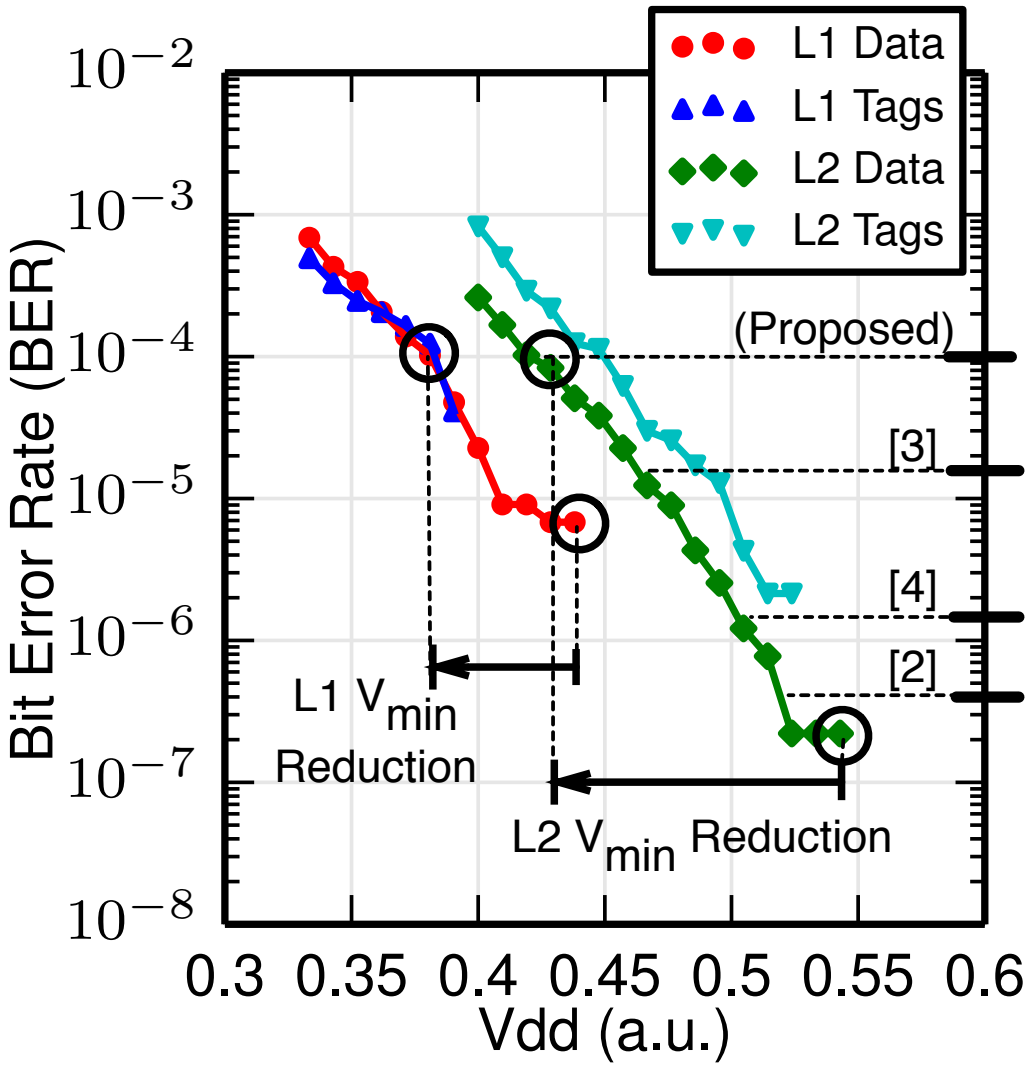
BIST Programming Algorithm



- **Standard BIST identifies fault locations**

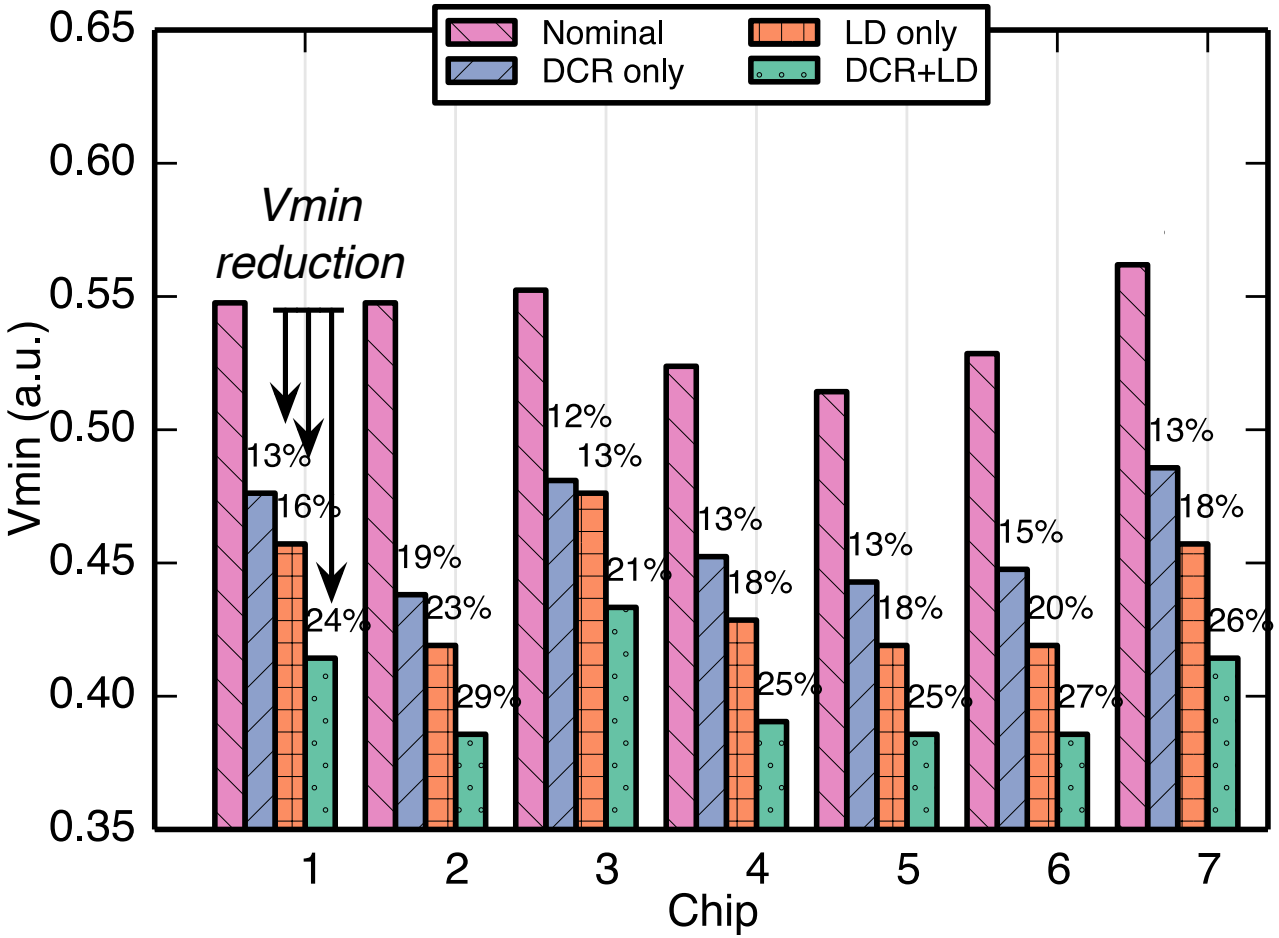
Measured SRAM Error Rate

- **Suite of March tests to identify errors**
- **As expected, high density bitcell in L2 has higher Vmin than high speed bitcell in L1**



[2] Huang JSSC 2013 [3] Chang JSSC 2007 [4] Sawant ISSCC 2011

Measured Vmin Reduction



- **Proposed techniques achieve 25% average Vmin reduction in L2 for 2% area overhead**

Summary

- Easier than assist techniques to adopt
  - Predictable Vmin reduction
  - Comparable effectiveness
- Requires power-up BIST or non-volatile memory

Conclusion

- DCR, LD, and BB are programmed by BIST to avoid failing bitcells
- Significant Vmin reduction for low overhead and no circuit changes
  - Can use in addition to assist techniques
- 28nm prototype processor shows
  - 25% Vmin reduction,
  - 49% power reduction
  - 2% area overhead

Acknowledgements

- Henry Cook
- Yunsup Lee
- Andrew Waterman
- James Dunn
- Brian Richards
- Stephen Twigg
- Scott Liao
- Jonathan Chang
- Work funded in part by BWRC, ASPIRE, DARPA PERFECT Award Number HR0011-12-2-0016, Intel ARO, and a fabrication donation by TSMC.