Increasing Architectural Resilience to Small Delay Faults

Peter Deutsch, Vincent Ulitzsch

Sudhanva Gurumurthi (AMD), Vilas Sridharan (AMD)

Joel Emer (MIT), Mengjia Yan (MIT)

May 1st, 2024

Technische Universität Berlin

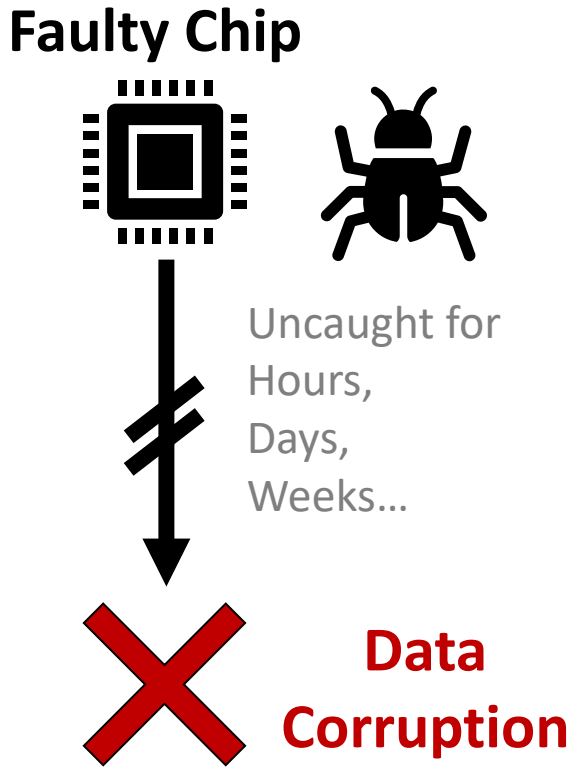
AMD

MIT CSAIL

compute. collaborate. create

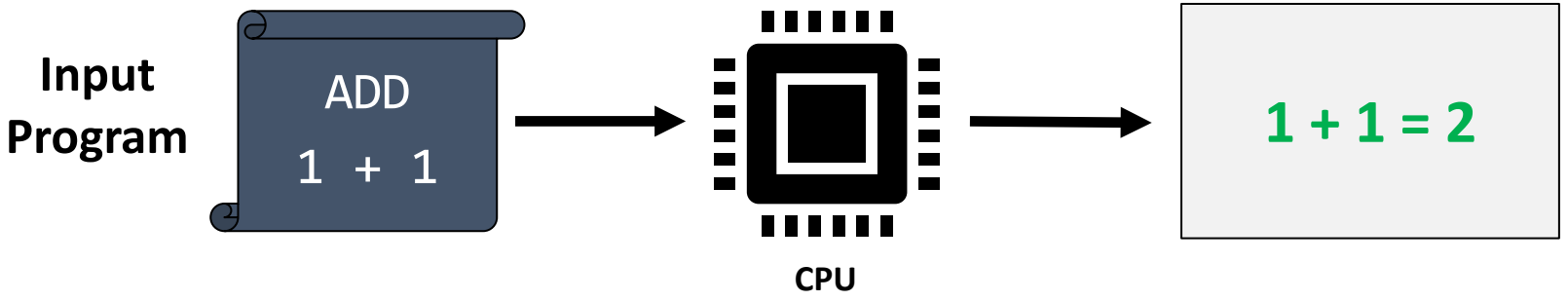
Motivation: Emerging Silent Data Corruptions (SDCs) indicate a reliability blind spot

- Cloud providers are experiencing frequent **silent data corruptions** due to unknown & undetected **chip defects**.
- Prior reliability work is limited to studying and mitigating the effects of **particle strikes**.
- We need a methodology to reason about a chip's vulnerability to emergent fault models like **small delay faults**.

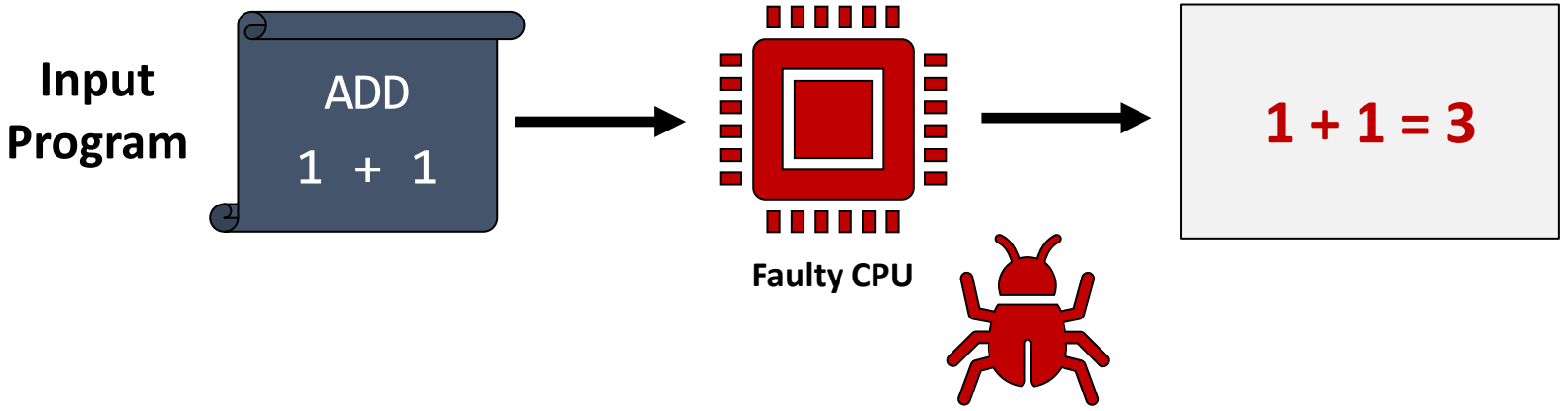


Our Research: We leverage architectural insights to better protect chips against small delay faults *early in the design process.*

CPUs can miscompute, resulting in Silent Data Corruptions (SDCs)



Silent data corruptions occur when faulty CPUs silently miscompute, resulting in incorrect results



Silent data corruptions already inhibit at-scale workloads

SDCs are pervasive in High Performance Computing…

In High-End Computing clusters, the time spent recovering from SDCs can exceed 65% of all computation [1]!

168-HOUR JOB, 5 YEAR MTBF

# Nodes	work	checkpt	recomp.	restart
100	96%	1%	3%	0%
1,000	92%	7%	1%	0%
10,000	75%	15%	6%	4%
100,000	35%	20%	10%	35%

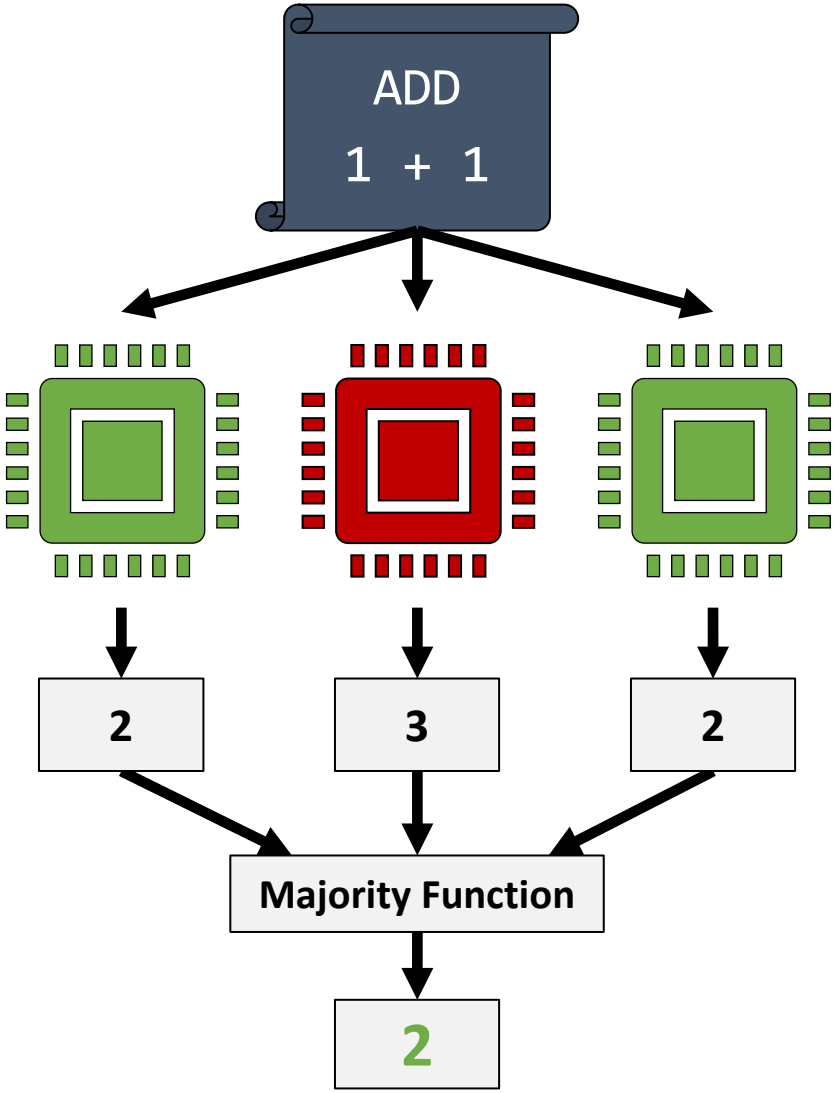
And in ML workloads, where correctness is often hard to determine!

ML training and inference at scale can often be stymied by SDCs, leading to poor accuracy and NaN results [2].

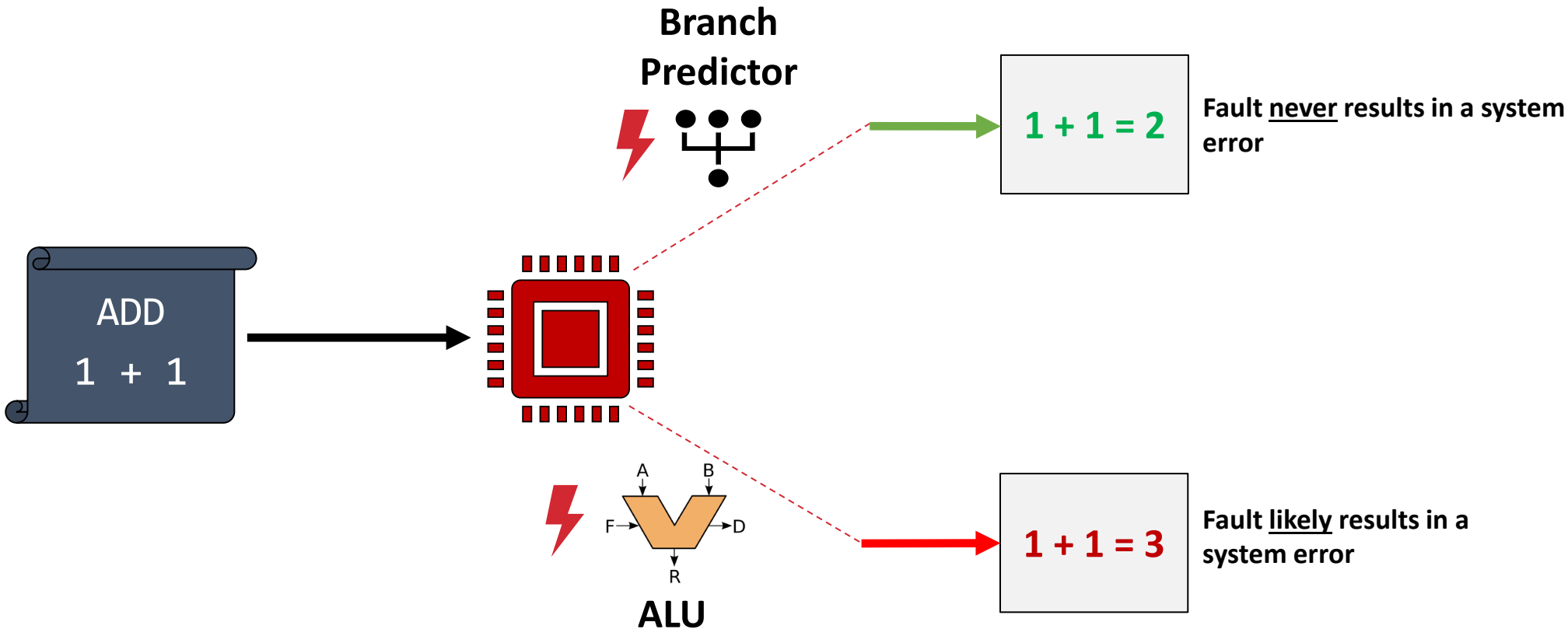
[1] Fiala et al.; Detection and Correction of Silent Data Corruption for Large-Scale High-Performance Computing; SC12
[2] Elsen et al.; The Adventure of the Errant Hardware; http://adept.ai/blog/sherlock-sdc

We can naïvely increase the reliability of chips using redundancy

- One strategy to improve reliability is **redundancy via duplication**.
- **Naïve Idea:** Three copies of the chip vote on the correct solution.
- **Problem:** It's not feasible to duplicate/triplicate the chip!



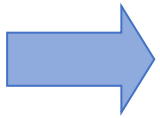
Not every bit-flip will result in a deviation from the program's expected output



We can focus our protection efforts to relevant parts of the chip!

Architectural Vulnerability Factor (AVF) [1]

How can we identify structures particularly vulnerable to **particle strikes**?



We can quantify a structure's **Architectural Vulnerability Factor (AVF):**

What fraction of time will a bit-flip in that structure result in architecturally incorrect execution?

Key Benefit: We can acquire reliability insights **early in the design** process.

[1] Mukherjee et al.; A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor; MICRO '03

Environmental, aging, and manufacturing conditions can cause chip failure

Errors can occur due to….

Particle Strikes

- Cosmic particles can strike state elements on the chip, flipping bits.
- **Focus of original AVF work!**

Wear-out effects

- Wires and transistors age over time, introducing additional delays.

Manufacturing defects

- Uncaught lithographic errors and poor via connections can result in undefined behaviour.

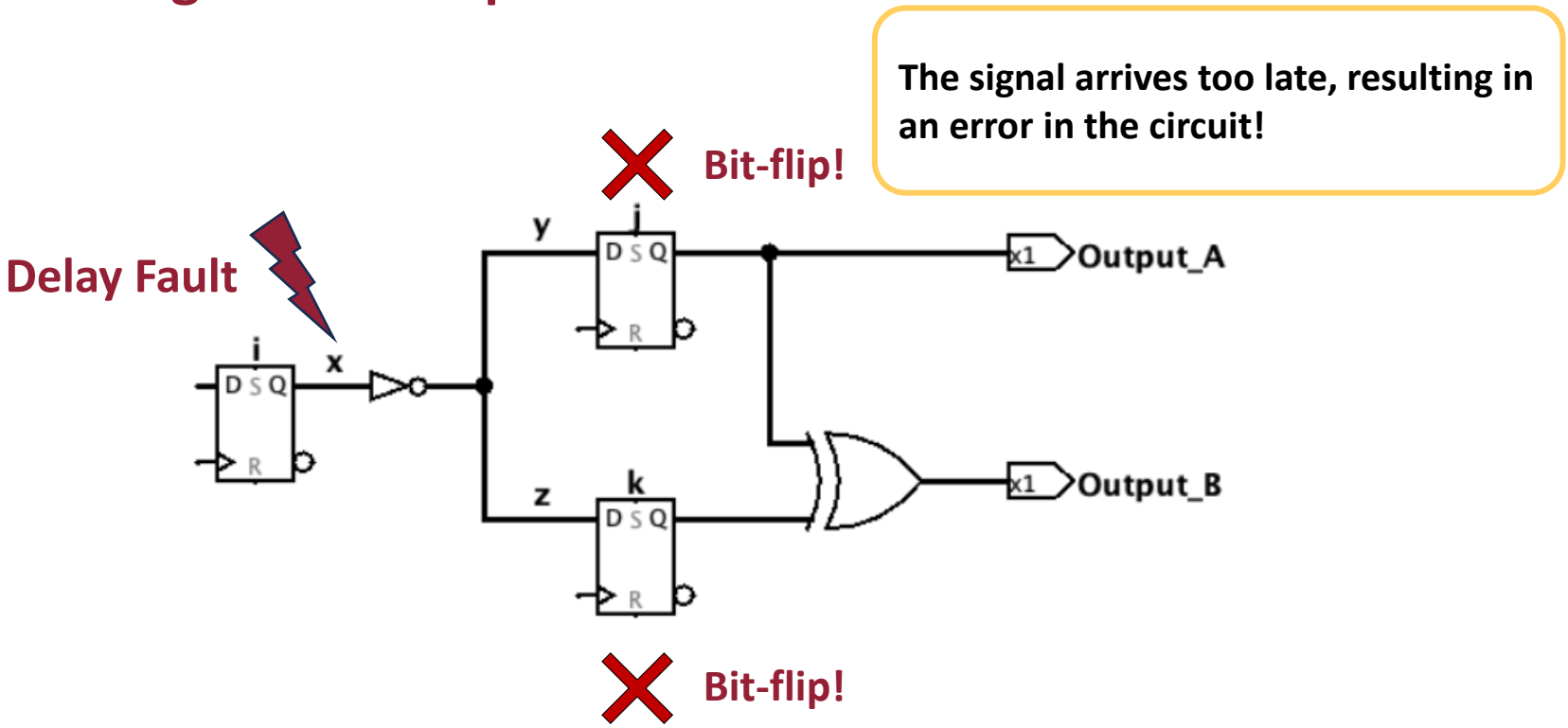
Prior work has been focused on particle strikes

Large amount of prior work studies errors caused by particle strikes…

Existing work makes two primary assumptions…	 **Uniform Fault Model**	▪ Each state-element/flip-flop has an **equally likely chance** of having an error.
	 **Flip-Flop Point of Strike Model**	▪ Bit flips are assumed to **occur at state elements**. ▪ Particle strikes on logic/wires are assumed to not be significant due to masking effects.

Our Research: Utilizing AVF to reason about delay faults, an emergent fault model.

Delay faults have emerged as an important fault mode



- Recent findings [1,2,3] point to **marginal timing defects** as an emergent reason for chip failures.
- A delay induced on a circuit element's signal may result in a **setup violation**, resulting in an error.

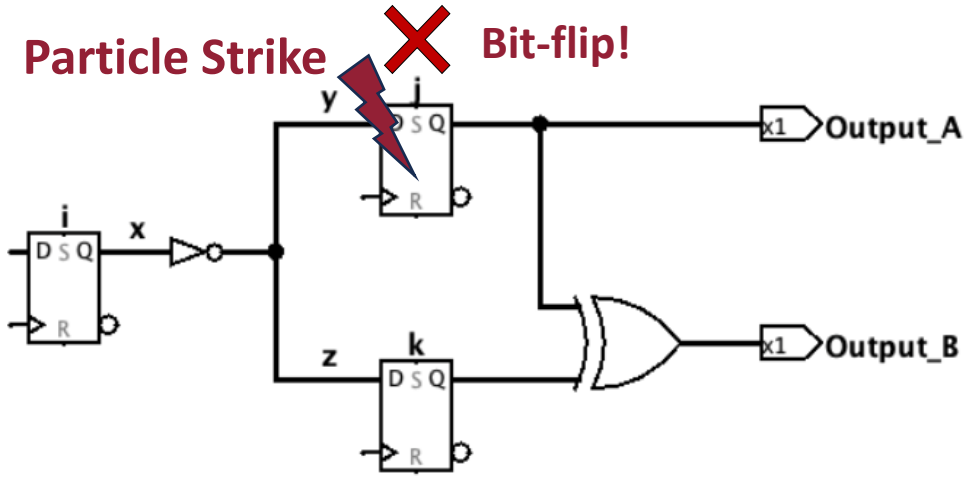
[1] https://semiengineering.com/screening-for-silent-data-errors/
[2] https://www.sigarch.org/silent-data-corruption-at-scale/
[3] Singh et al.; Silent Data Errors: Sources, Detection, and Modeling; VTS '23

Using AVF to study delay faults is challenging

Goal: We want to utilize AVF to reason about small delay fault models, but there are **key differences** to prior work…

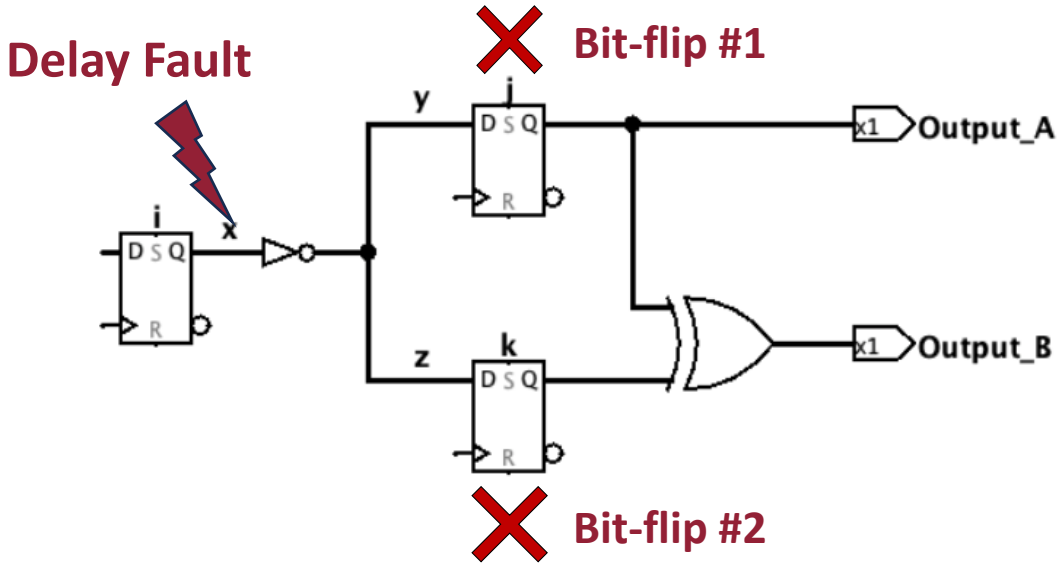
Challenge 1: The point of fault is no longer the point of error

Particle Strike Model



The particle **directly strikes** the state element, which subsequently has an error.

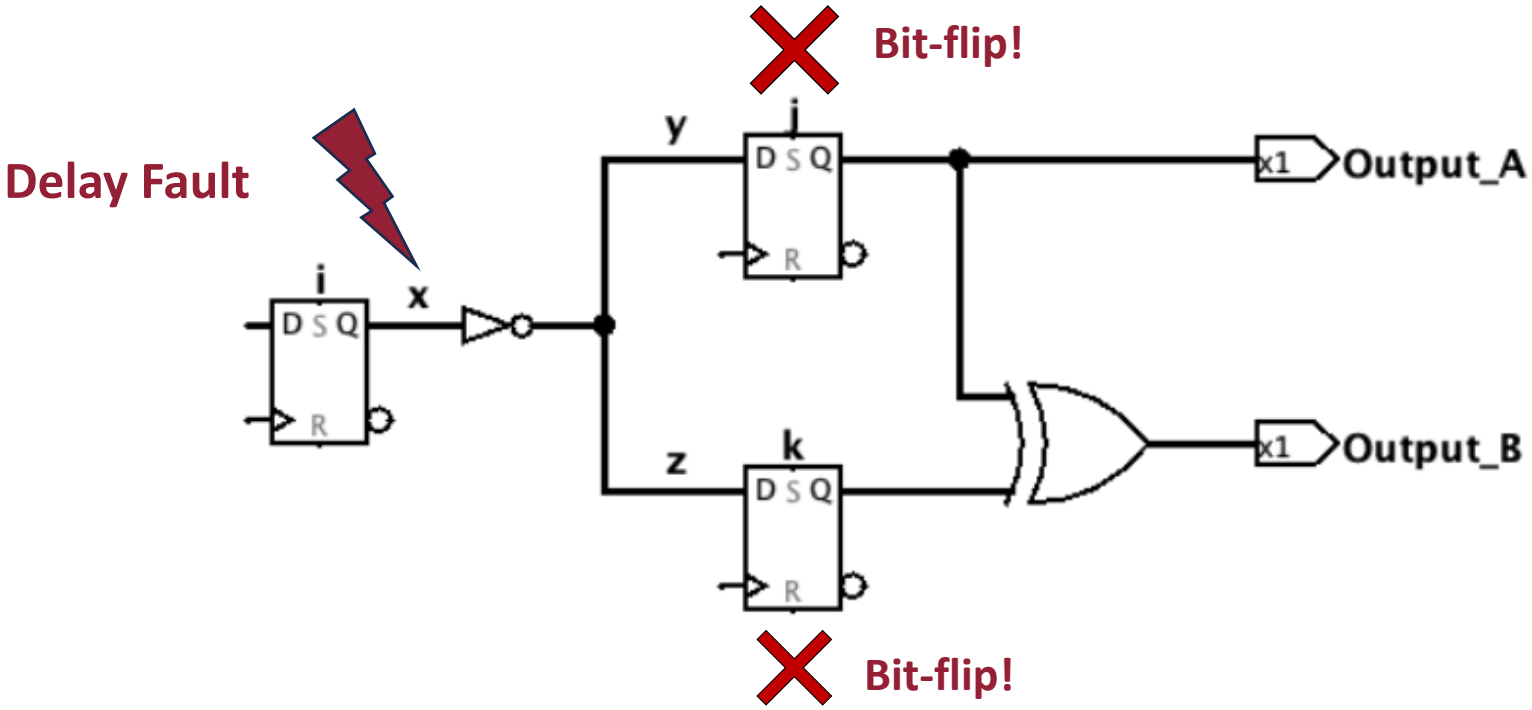
Delay Fault Model



The delay affects the **circuit element (e.g., wire)**, however the error is only observed at a downstream **state element**!

Key Observation #1: Rather than reasoning about the individual vulnerability of state elements, we need to reason about the vulnerability of generic circuit elements.

Challenge 2: Delay faults can result in multiple circuit errors



Key Observation #2: We need to reason about the combined interactions between circuit-level errors.

Our Contribution: An Architectural Vulnerability Factor for Small Delay Faults

We provide a methodology to derive and compute *DelayAVF*

DelayAVF:

The probability that a delay fault in a microarchitectural structure propagates to a program-visible error.

Deriving DelayAVF via studying the vulnerability of individual circuit elements

We reason about the impact of a fault through the lens of the affected element

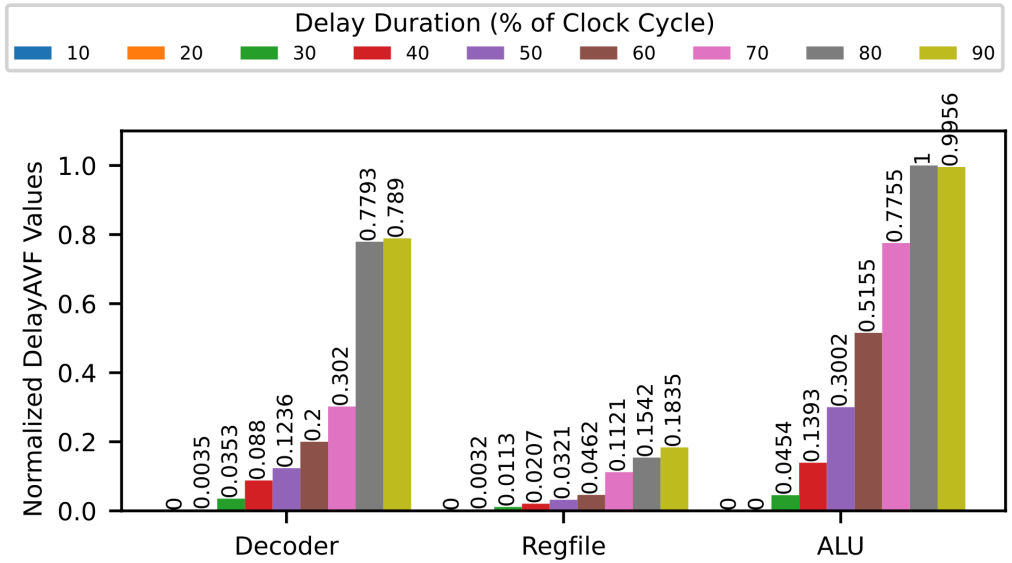
Delay ACEness	A circuit element *e* is *delayACE* in cycle *i* iff a fixed-length delay *d* results in a program-visible error.

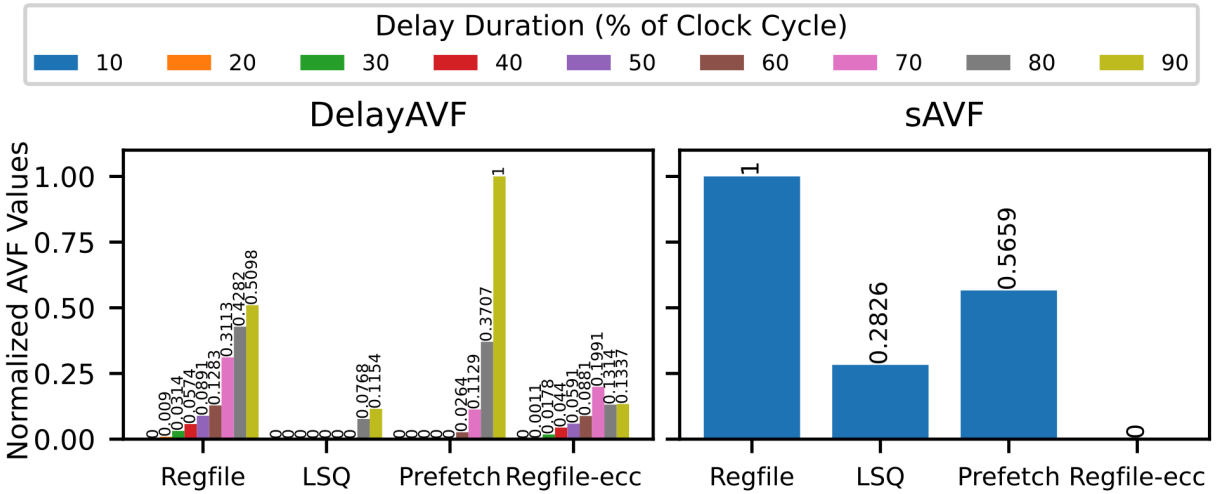
The DelayAVF of a structure can be formulated in terms of the ACEness of its elements

DelayAVF of a Structure	$$DelayAVF(H) = \sum_{\forall e \in C} \sum_{i=1}^{N} \frac{isDelayACE(e,i)}{N \cdot \lvert C \rvert}$$

This formulation enables us to tractably compute DelayAVF early in the design stage.

Experimental Results: DelayAVF Points to Vulnerable Structures in IBEX, a RISC-V Core





Observation #1

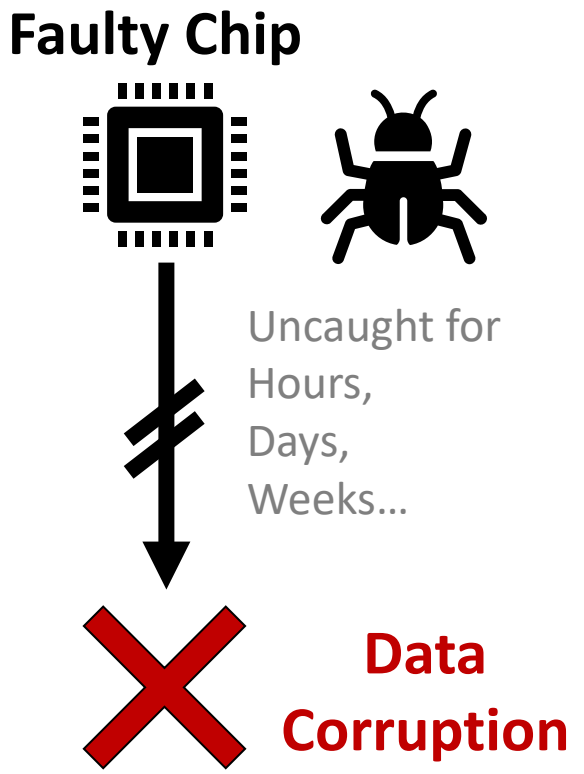
The vulnerability of microarchitectural structures to small delay faults can vary significantly!

Observation #2

Structures which are vulnerable to delay faults are not necessarily vulnerable to particle strikes.

Summary: A Methodology to Architecturally Reason About Small Delay Faults

- Small delay faults appear to be a driving factor of silent data corruptions experienced by cloud providers.
- **This work**: We developed a methodology to target expensive chip-level mitigations towards emerging fault models like small delay faults.



Our Research: We leverage architectural insights to better protect chips against small delay faults *early in the design process.*

Increasing Architectural Resilience to Small Delay Faults

Peter Deutsch, Vincent Ulitzsch

Sudhanva Gurumurthi (AMD), Vilas Sridharan (AMD)

Joel Emer (MIT), Mengjia Yan (MIT)

Questions/Comments?

pwd@mit.edu, vincent@sect.tu-berlin.de

Technische Universität Berlin

AMD

MIT CSAIL

compute. collaborate. create

Simulator Infrastructure