HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes

Minesh Patel, Geraldo F. Oliveira, Onur Mutlu

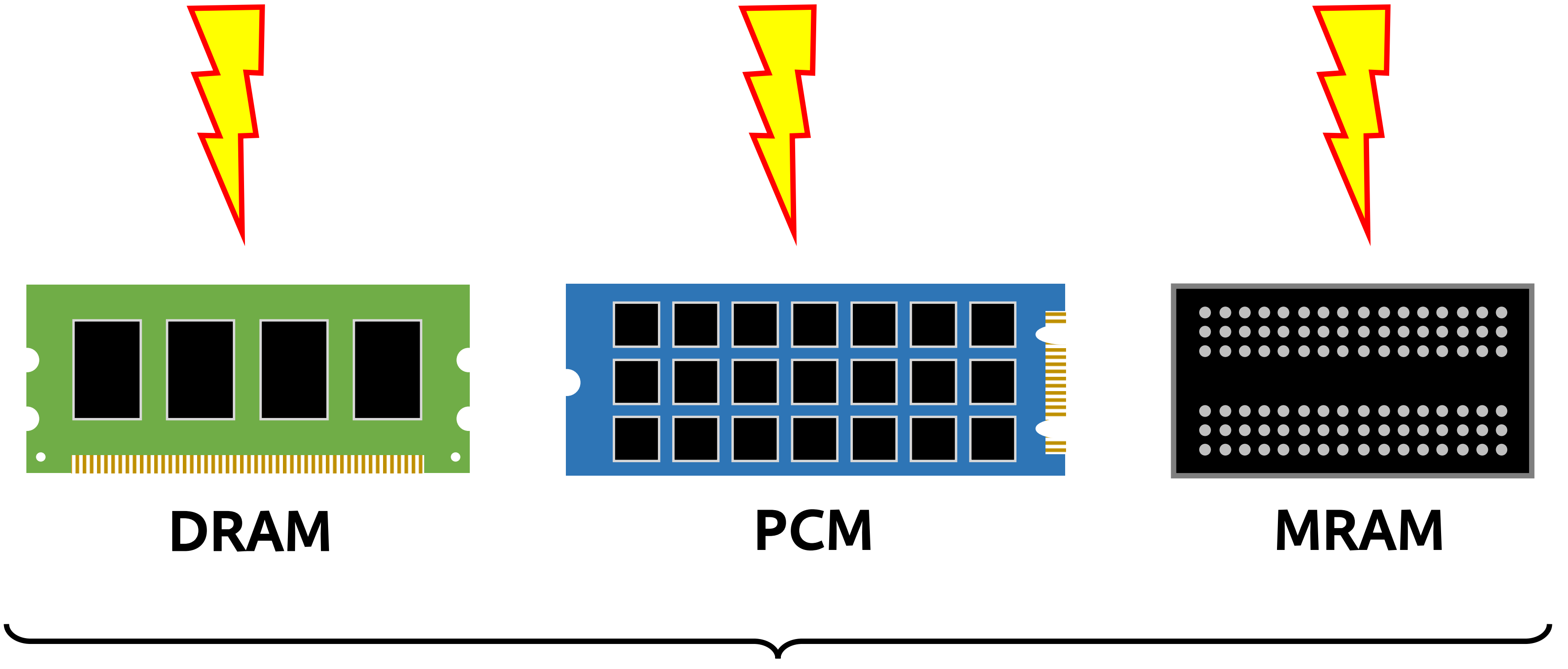
https://arxiv.org/abs/2109.12697

https://github.com/CMU-SAFARI/HARP

ETH zürich

SAFARI

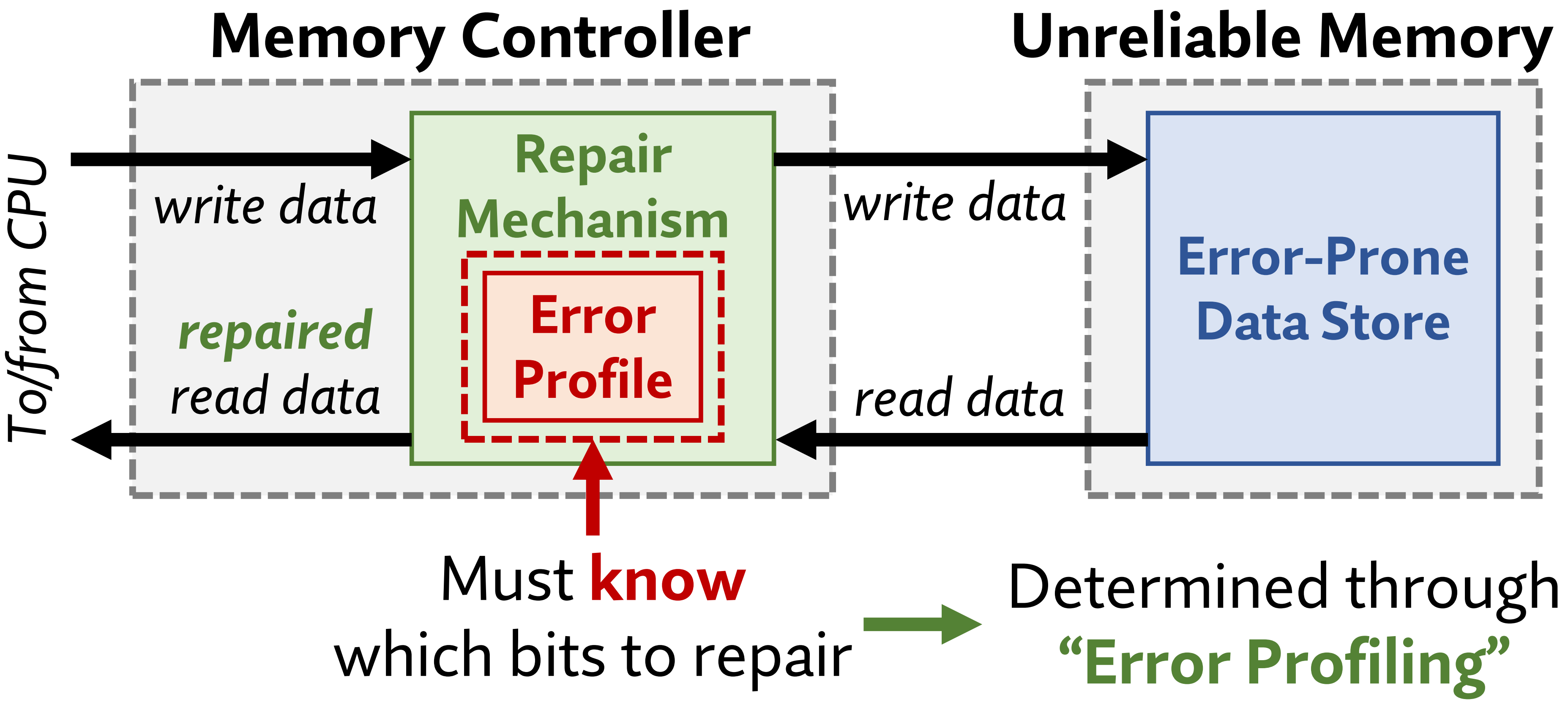
Memory Errors



All suffer **worsening error rates** with continual technology scaling

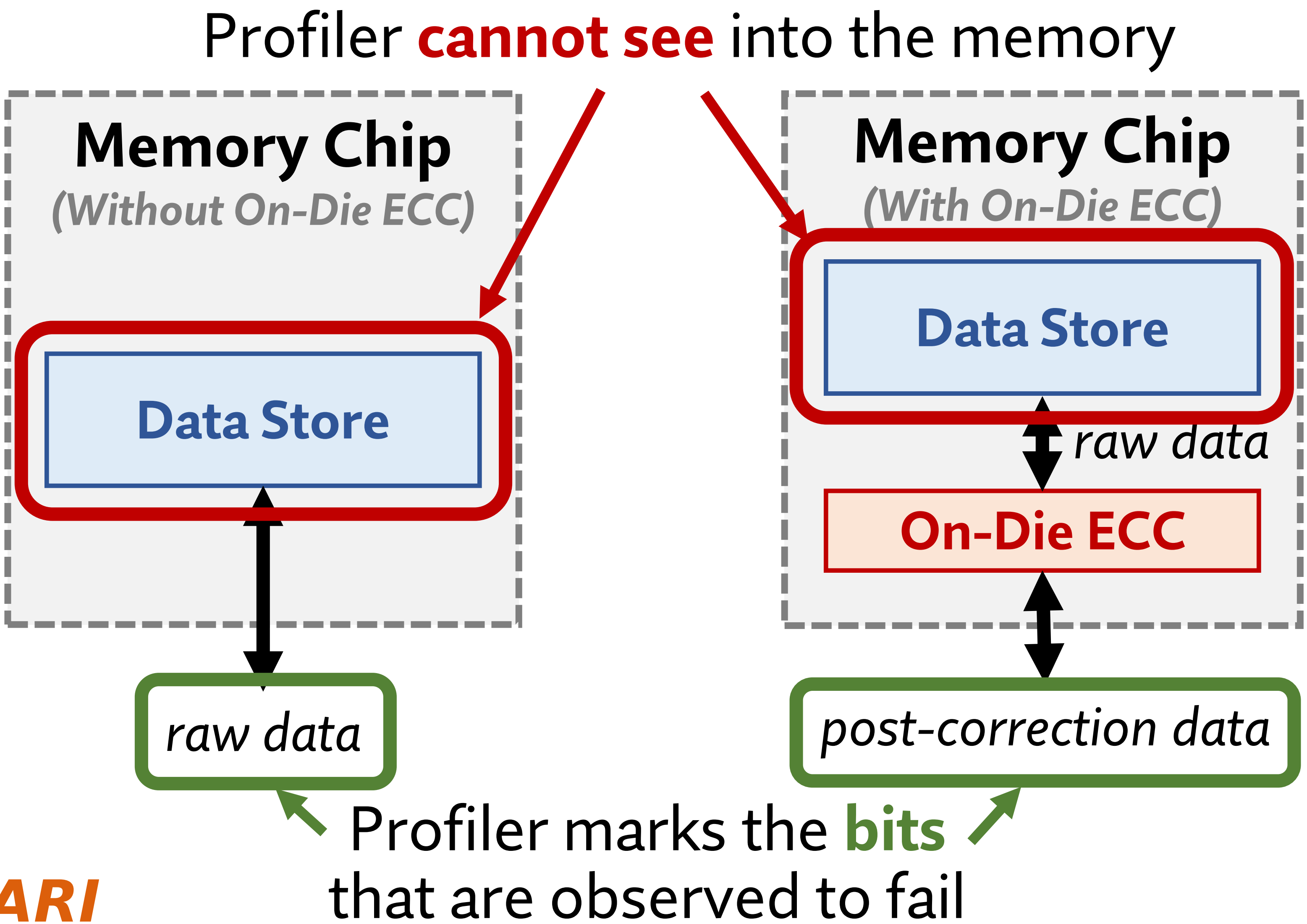
Memory Repair Mechanisms

- **Repair mechanisms** combat high memory error rates
  - **Identify** and **repair** any bits that are **at-risk of error**



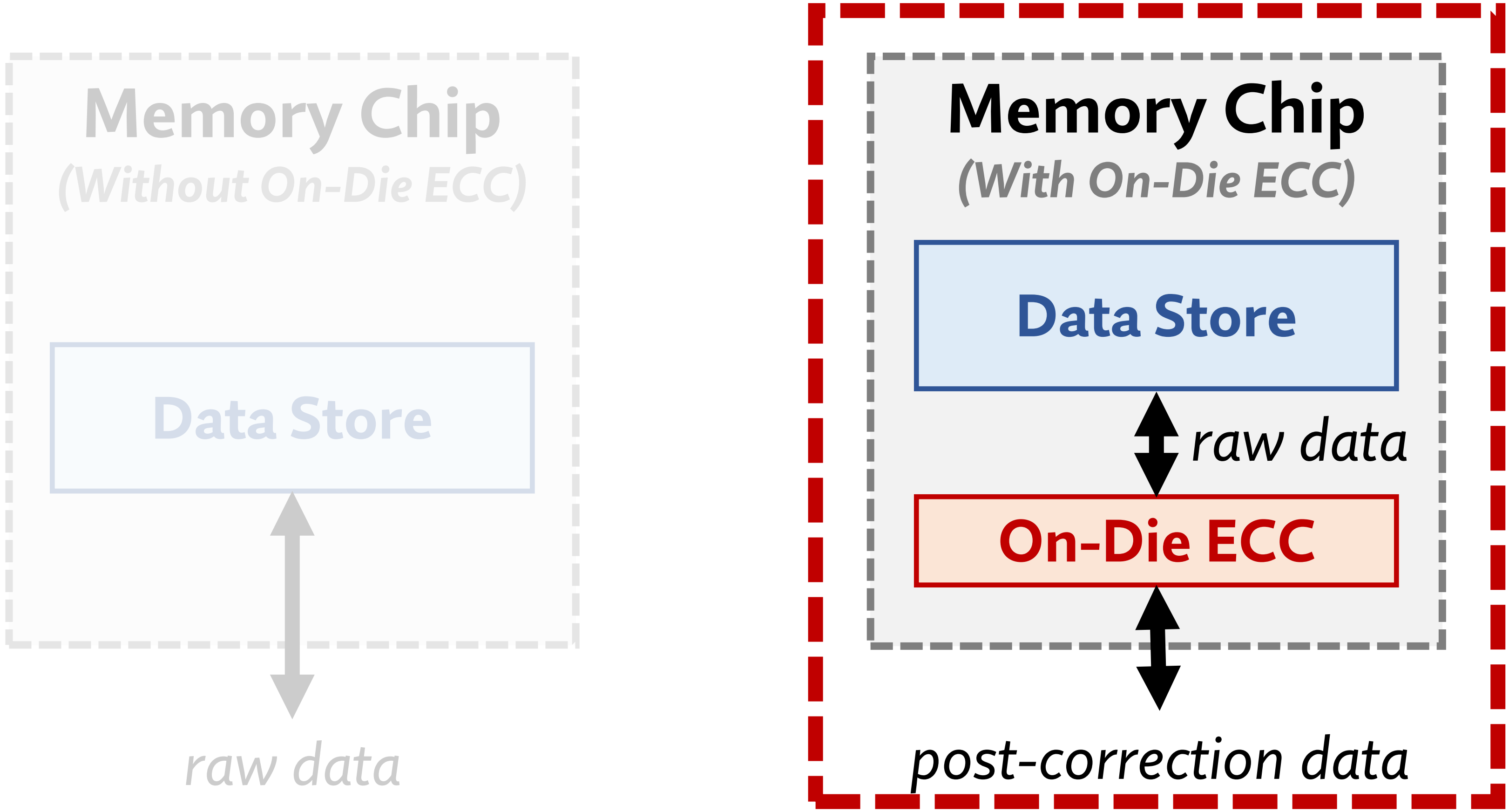
Profiling a Memory Chip

- **Profiler's goal:** identify all bits that are **at risk of error**



Profiling a Memory Chip

- **Profiler's goal:** identify all bits that are **at risk of error**



Q: How does on-die ECC affect error profiling?

Profiling a Memory Chip

- **Profiler's goal:** identify all bits that are **at risk of error**

Memory Chip
(Without On-Die ECC)

raw data

Memory Chip
(With On-Die ECC)

On-Die ECC

post-correction data

Q: How does on-die ECC affect error profiling?

Goal: understand and **address** the challenges that on-die ECC introduces for error profiling

Challenges Introduced by On-Die ECC

1. **Exponentially increases the at-risk bits**

   A **small set** of raw bit errors generates a **combinatorially larger** set of at-risk bits

2. **Harder to identify each at-risk bit**

   At-risk bits are exposed only when **specific raw bit error patterns** occur

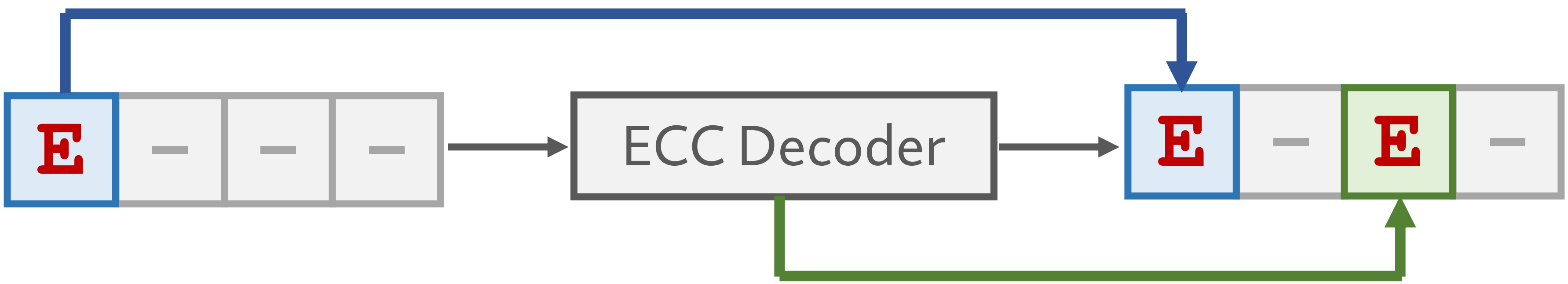
3. **Interferes with data patterns**

   Data patterns must consider **combinations of raw bits** instead of just individual bits alone

Key Observation: Two Sources of Errors

1 Direct error

Due to errors in the **memory chip**



2 Indirect error

Artifact of the on-die ECC algorithm

Upper-bounded by the ECC algorithm

Key Observation: Two Sources of Errors

1 **Direct error**

Due to errors in the **memory chip**

Key Idea: decouple profiling for **direct** and **indirect** errors

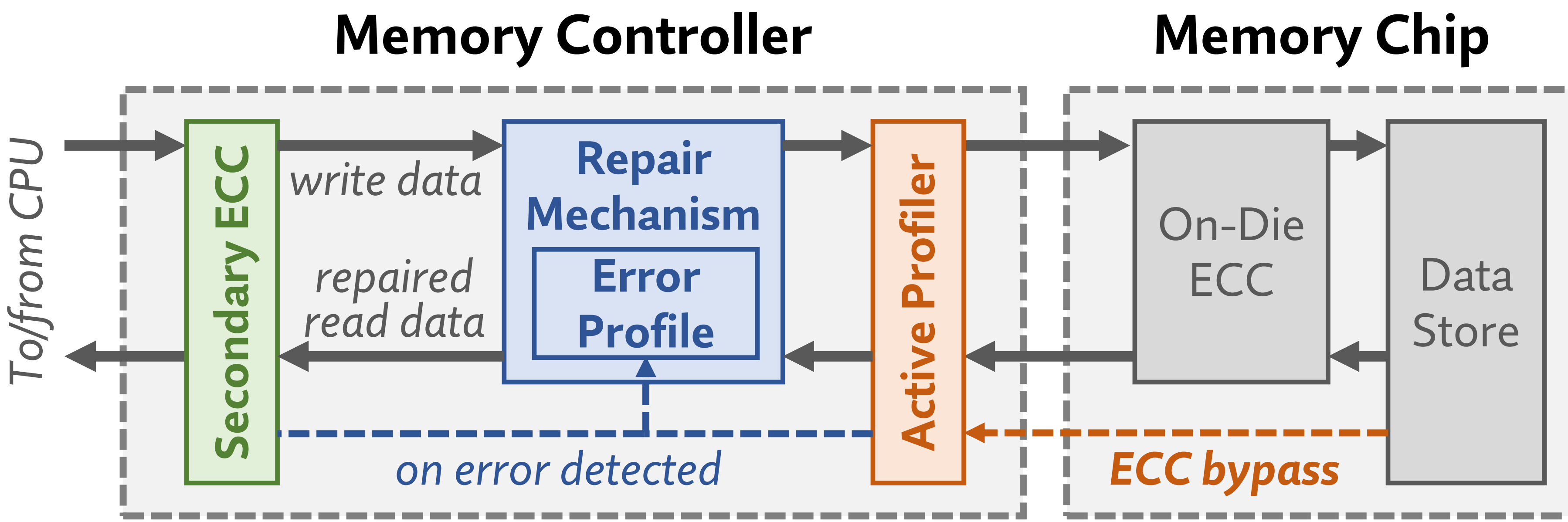
2 **Indirect error**

Artifact of the on-die ECC algorithm

Upper-bounded by the ECC algorithm

Hybrid Active-Reactive Profiling (HARP)

1 **Active Profiling**
Quickly identifies **direct errors**



2 **Reactive Profiling**
Safely identifies the **indirect errors**

Hybrid Active-Reactive Profiling (HARP)

1 **Active Profiling**

Quickly identifies **direct errors**

HARP **reduces** the problem of profiling **with on-die ECC** to profiling **without on-die ECC**

2 **Reactive Profiling**

Safely identifies the **indirect errors**

Evaluating HARP

- We evaluate HARP using **Monte-Carlo simulation**
  - Enables **accurately measuring** coverage (using a SAT solver)
  - 1,036,980 total ECC words
    - Across 2769 randomly-generated (71, 64) and (136, 128) ECC codes
    - ≈14 CPU-years (20 days on 256 cores) of simulation time

- Artifacts are **open-sourced**

Artifacts Evaluated Functional V1.1 · Artifacts Available V1.1 · Results Reproduced V1.1

DOI 10.5281/zenodo.5148592

https://github.com/CMU-SAFARI/HARP

Evaluation Comparison Points

- We evaluate HARP's error **coverage** and **speed** relative to **two baseline profiling algorithms:**

1. **Naive**: round-based profiling that **ignores** on-die ECC
   - Each round uses different data patterns (e.g., random data)
   - Profiler marks observed errors as at-risk bits

2. **BEEP** *[Patel+,MICRO'20]*: **knows** the exact on-die ECC implementation (i.e., its parity-check matrix)
   - Same overall round-based strategy as Naive
   - Data patterns designed using the known parity-check matrix

Evaluation Comparison Points

- We evaluate HARP's error **coverage** and **speed** relative

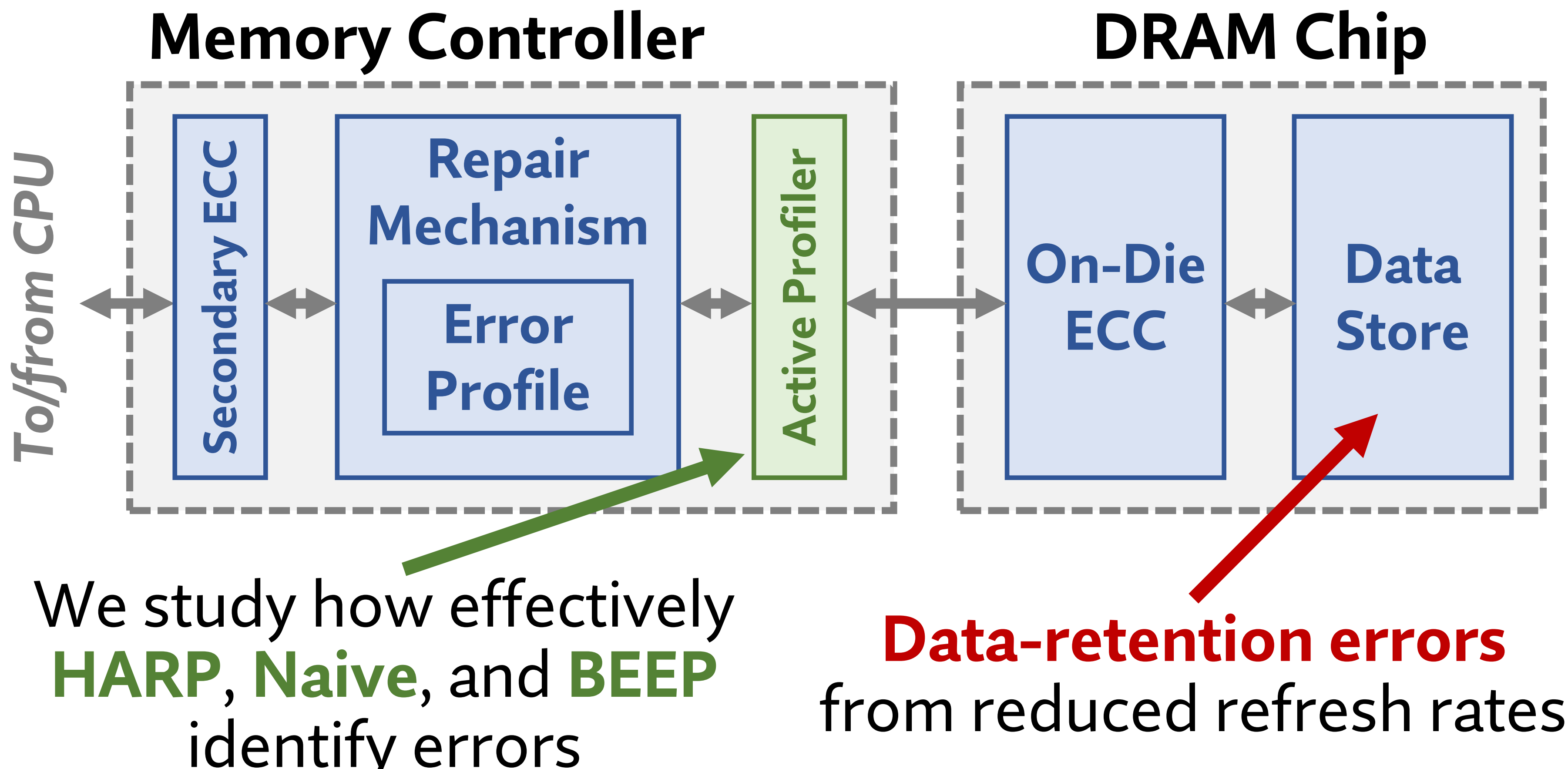
HARP **overcomes**
all three profiling challenges

- Profiler marks observed errors as at-risk bits

HARP performs **20.6- to 62.1% faster**
than the best-performing baseline

Case Study: DRAM Data-Retention

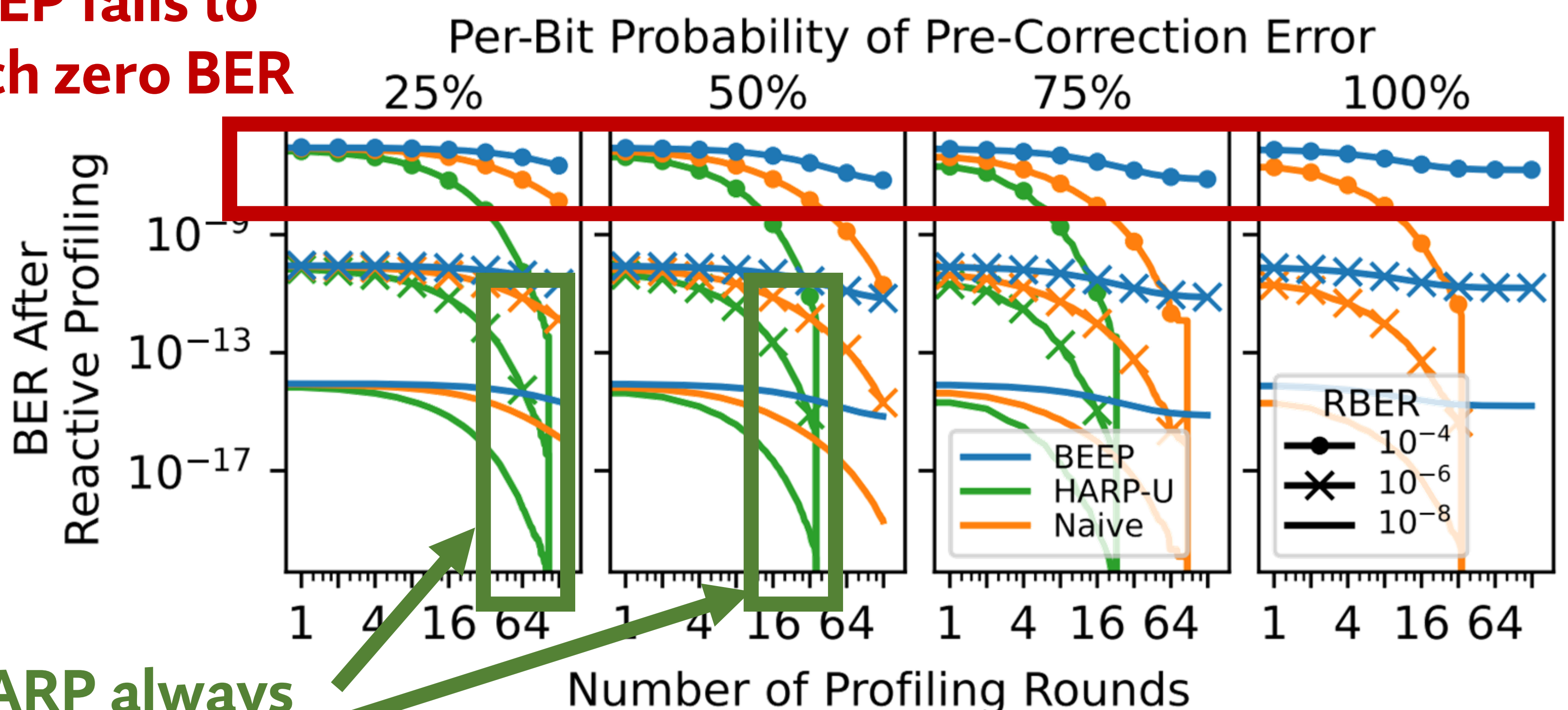
- We consider a system that uses an **ideal repair mechanism** to safely **reduce the DRAM refresh rate**



Case Study: DRAM Data-Retention

- We measure the end-to-end **bit error rate (BER)** for each of the profilers

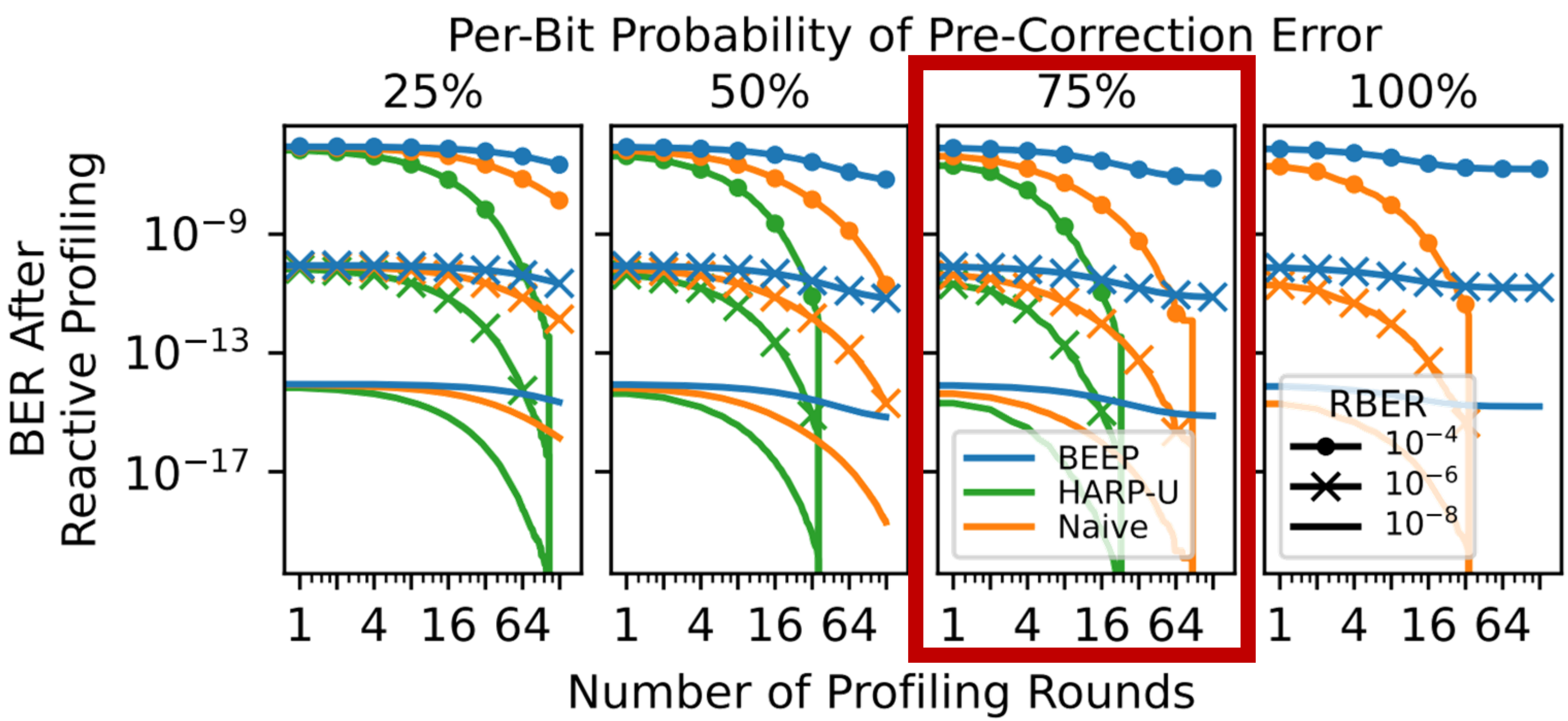
BEEP fails to reach zero BER



HARP always reaches zero BER

Case Study: DRAM Data-Retention

HARP reaches zero BER **3.7x faster** than the best-performing baseline



Other Information in the Paper

- **Detailed analysis of on-die ECC**
  - How on-die ECC introduces statistical dependence between post-correction errors
  - Differences between direct and indirect errors
  - Error profiling challenges introduced by on-die ECC

- **Discussion about HARP's design decisions**

- **More evaluation results**
  - Coverage of direct and indirect errors
  - Analysis of profiler bootstrapping
  - Case study on the end-to-end memory bit error rate (BER)

- **Detailed artifact description**

Other Information in the Paper

HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes

Minesh Patel
ETH Zürich

Geraldo F. Oliveira
ETH Zürich

Onur Mutlu
ETH Zürich

ABSTRACT

Aggressive storage density scaling in modern main memories causes increasing error rates that are addressed using error-mitigation techniques. State-of-the-art techniques for addressing high error rates identify and repair bits that are at risk of error from within the memory controller. Unfortunately, modern main memory chips internally use on-die error correcting codes (on-die ECC) that obfuscate the memory controller's view of errors, complicating the process of identifying at-risk bits (i.e., error profiling).

To understand the problems that on-die ECC causes for error profiling, we analytically study how on-die ECC changes the way that memory errors appear outside of the memory chip (e.g., to the memory controller). We show that on-die ECC introduces statistical dependence between errors in different bit positions, raising three key challenges for practical and effective error profiling: on-die ECC (1) exponentially increases the number of at-risk bits the profiler must identify; (2) makes individual at-risk bits more difficult to identify; and (3) interferes with commonly-used memory data patterns that are designed to make at-risk bits easier to identify.

profiler impacts the system's overall bit error rate (BER) when using a repair mechanism to tolerate DRAM data-retention errors. We show that HARP identifies all errors faster than the best-performing baseline algorithm (e.g., by 3.7× for a raw per-bit error probability of 0.75). We conclude that HARP effectively overcomes the three error profiling challenges introduced by on-die ECC.

CCS CONCEPTS

• **Computer systems organization** → **Dependable and fault-tolerant systems and networks**; • **Hardware** → **Memory test and repair**.

KEYWORDS

On-Die ECC, DRAM, Memory Test, Repair, Error Profiling, Error Modeling, Memory Scaling, Reliability, Fault Tolerance

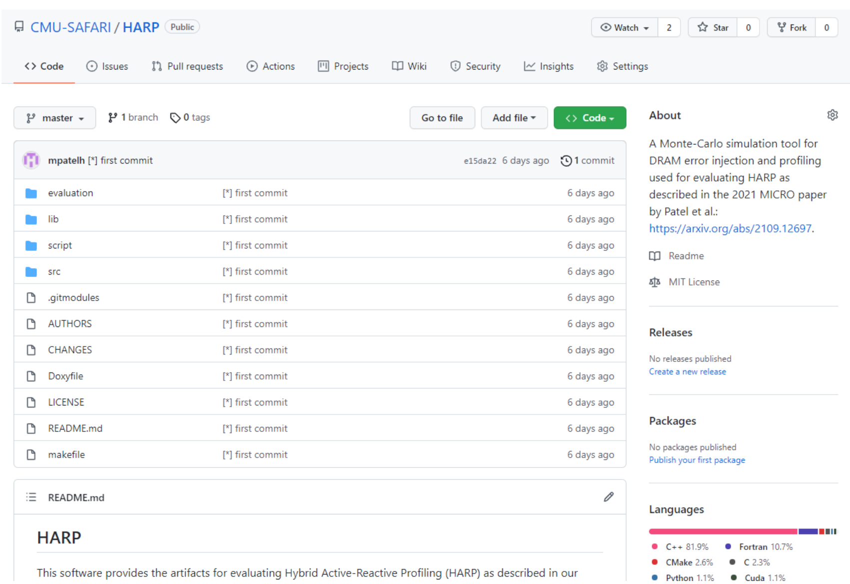
ACM Reference Format:
Minesh Patel, Geraldo F. Oliveira, and Onur Mutlu. 2021. HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes. In *Proceedings of the 54th Annual IEEE/ACM*

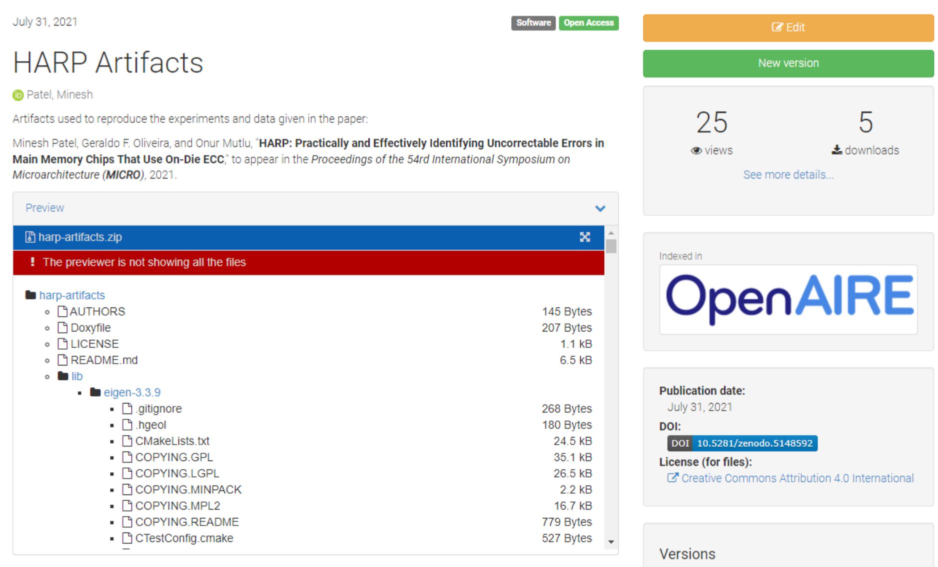
https://arxiv.org/abs/2109.12697

Artifacts are Open-Sourced

DOI 10.5281/zenodo.5148592

https://github.com/CMU-SAFARI/HARP





HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes

Minesh Patel, Geraldo F. Oliveira, Onur Mutlu

https://arxiv.org/abs/2109.12697

https://github.com/CMU-SAFARI/HARP

ETH zürich

SAFARI