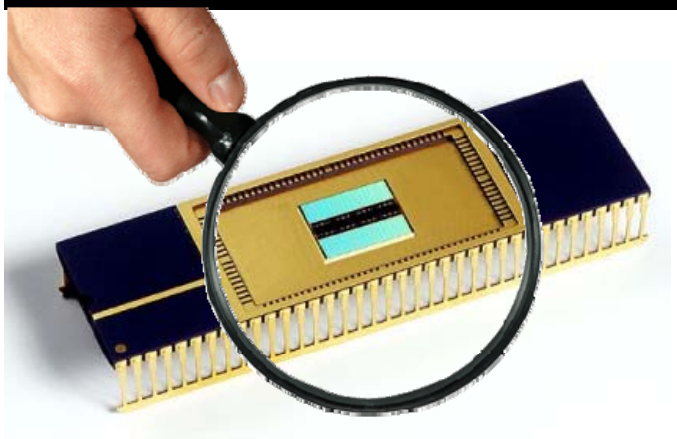


PRISM: Zooming in Persistent RAM Storage Behavior

Juyoung Jung and Dr. Sangyeun Cho

Dept. of Computer Science
University of Pittsburgh

pittCAST

Contents

- Introduction
- Background
- PRISM
- Preliminary Results
- Summary

Technical Challenge

HDD

- Slow and
- Power hungry

Fundamental solution?

Alternative storage medium

Technical Challenge

NAND flash based SSD

+ much faster
+ better energy
+ smaller …

- erase-before-write
- write cycles
- unbalanced rw
- scalability …

Emerging Persistent RAMs Technologies

	Latency			Program energy	Endurance	Density
	Read	Write	Erase			
NAND flash	25us	200us	1.5ms	10nJ	10^4~10^6	4~5F^2
PCM	20ns	100ns	N/A	100pJ	10^8~10^9	5F^2
STT-RAM	10ns	10ns	N/A	0.02pJ	10^{15}	4F^2
RRAM	10ns	20ns	N/A	2pJ	10^6	6F^2
FeRAM	75ns	50ns	N/A	2pJ	10^{13}	6F^2

PRAMs Win !

Write endurance

Read/write imbalance

Latency

Energy consumption

Technological maturity

Our Contribution

- There is little work to evaluate Persistent RAM based Storage Device (PSD)
  - Research environment not well prepared

- Present an efficient tool to study the impact of PRAM storage device built with totally different physical properties

PRISM

- **P**e**R**s**I**stent RAM **S**torage **M**onitor

- Study Persistent RAM (PRAM) storage behavior
  - Potentials of new **byte addressable storage** device
  - PRAMs' **challenges** as storage media

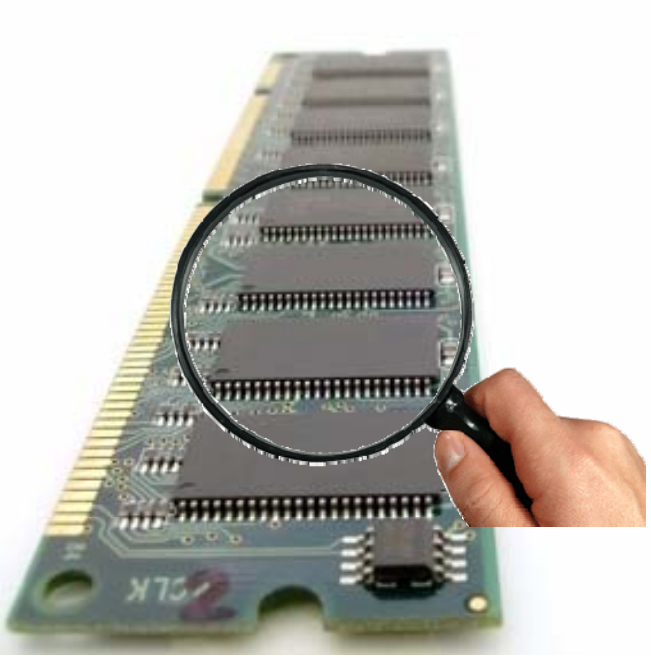
- Guide PSD (**P**RAM **S**torage **D**evice) design
  - Measure detailed storage activities (SW/HW)

PRISM

HDD or SSD

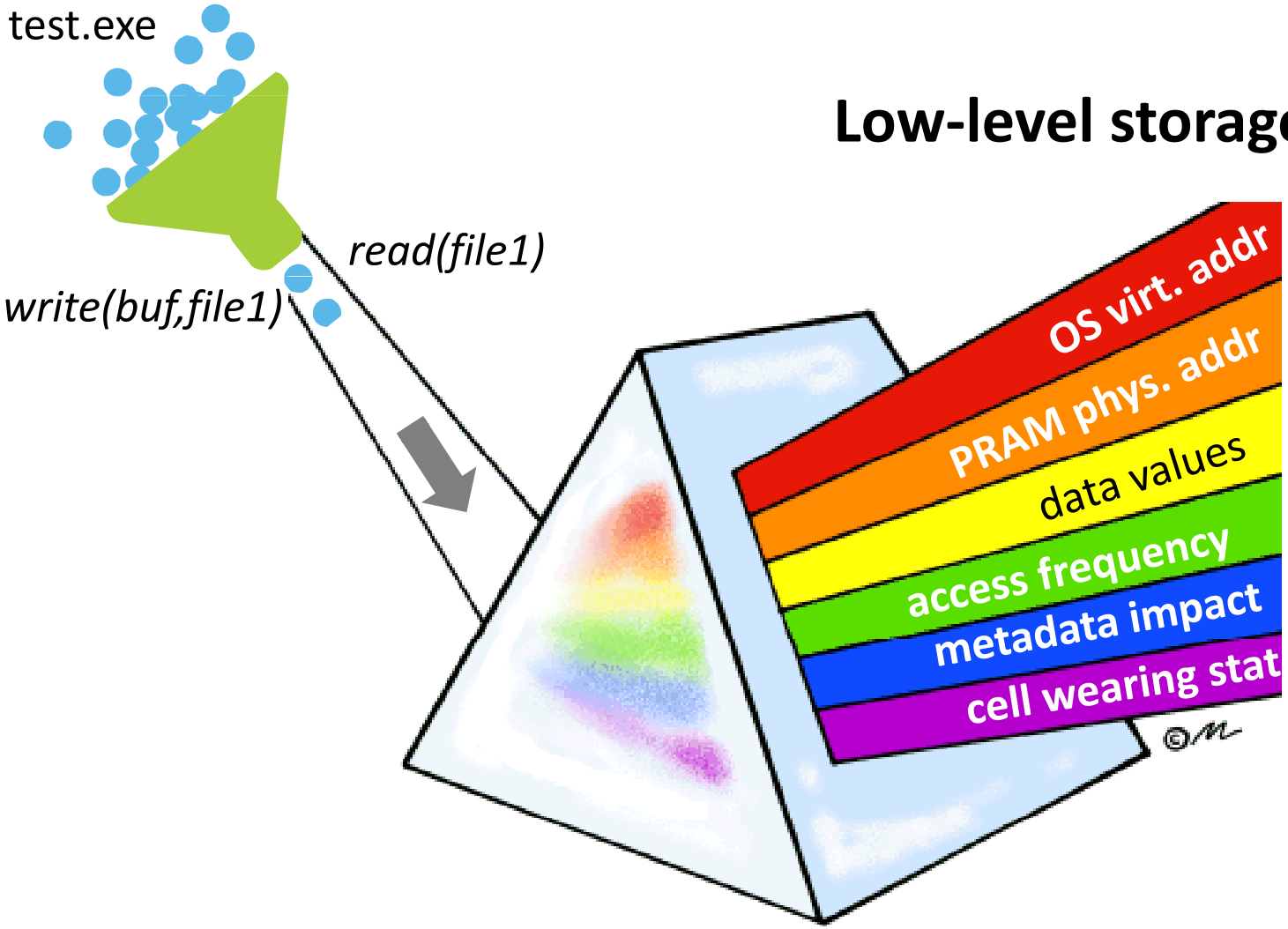
Block devices

PSD



Non-Block devices

PRISM



Low-level storage behavior

address mapping

wear leveling

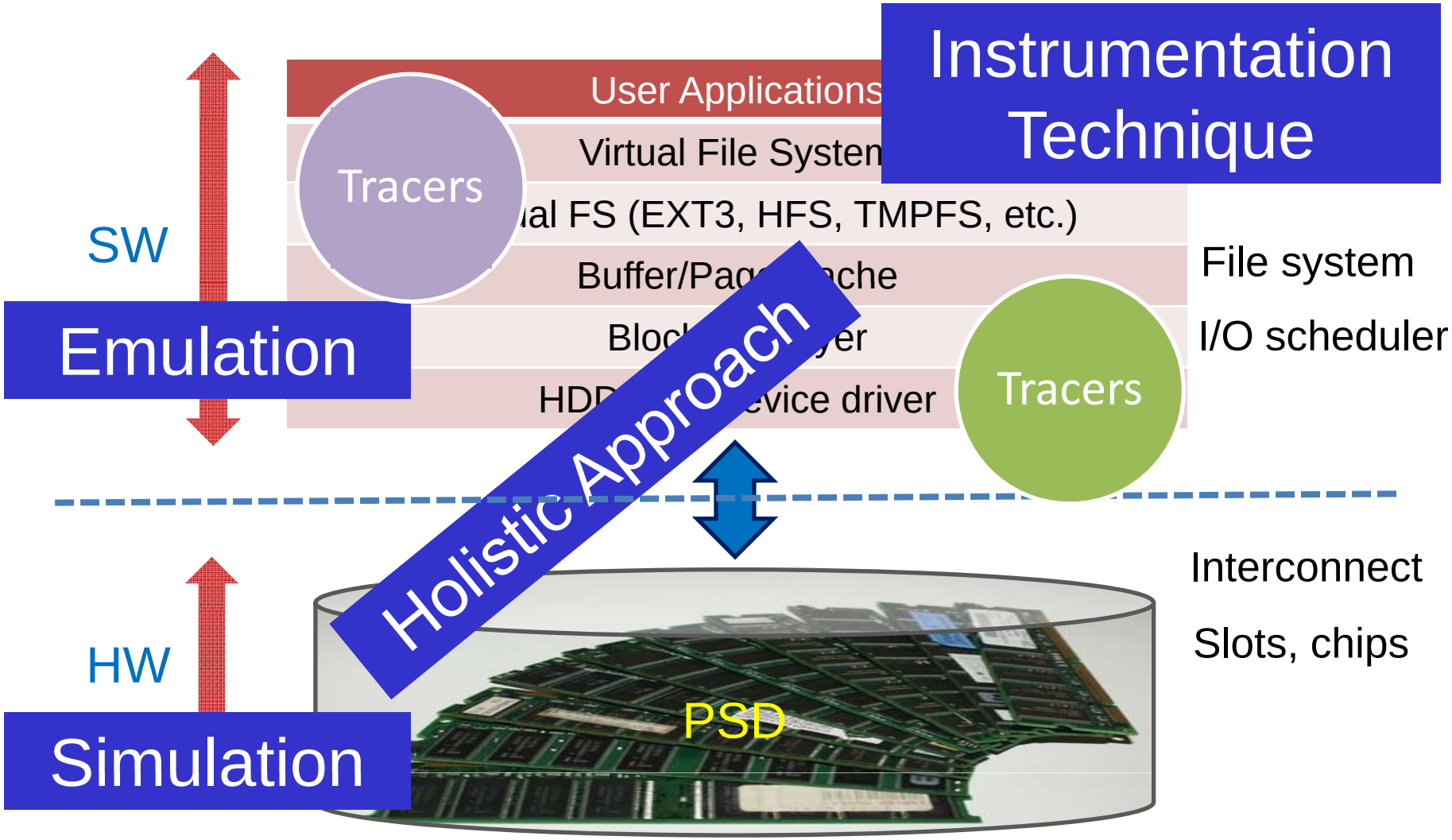
bit masking

exploit parallelism

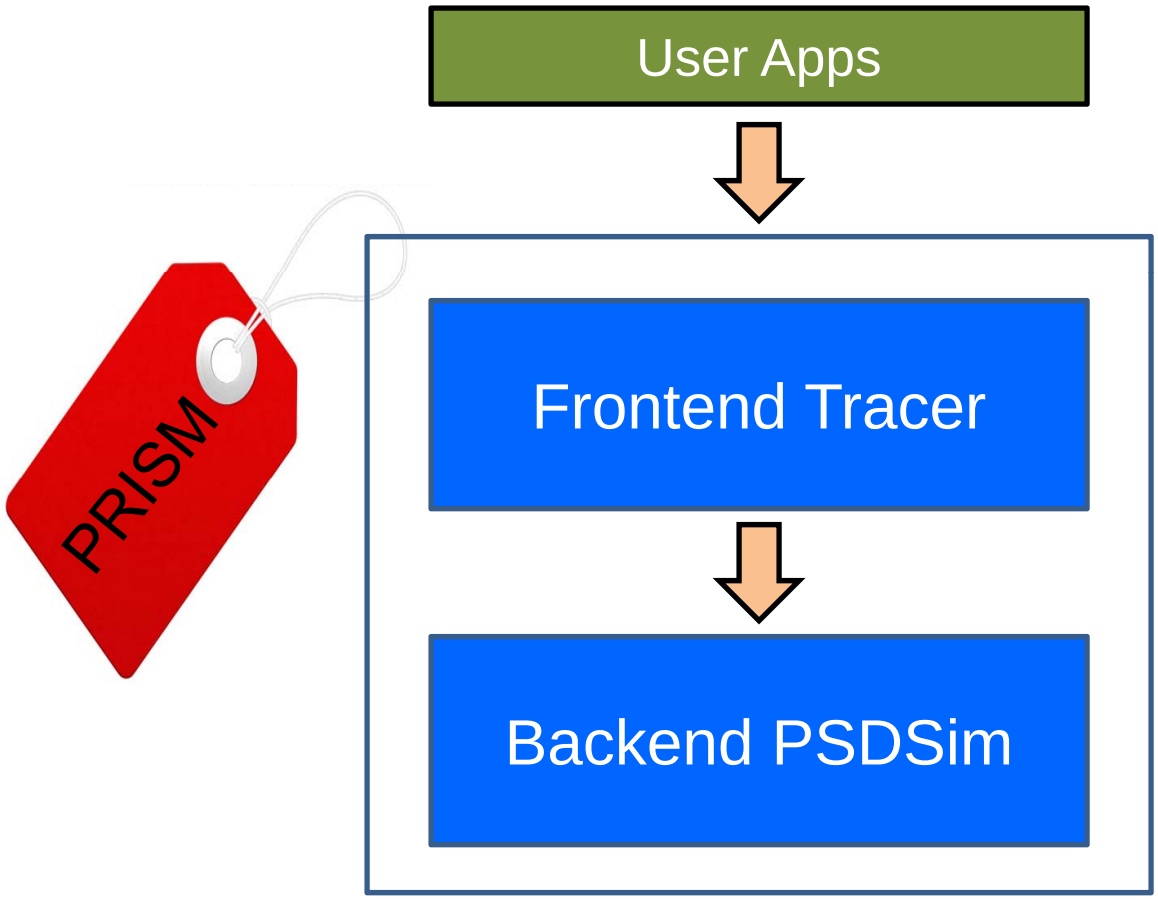
resource conflicts

etc.

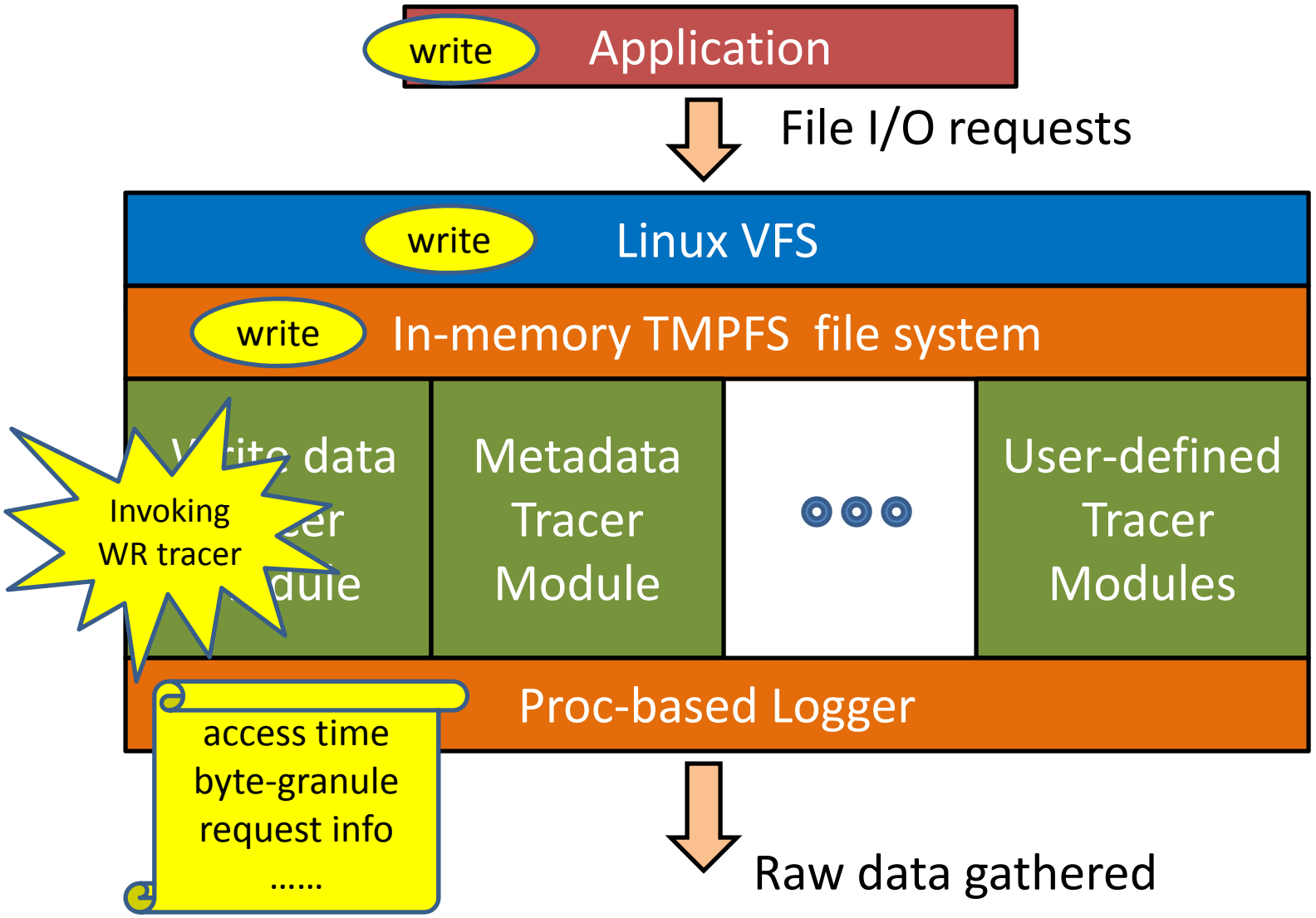
PRISM Implementation Approach



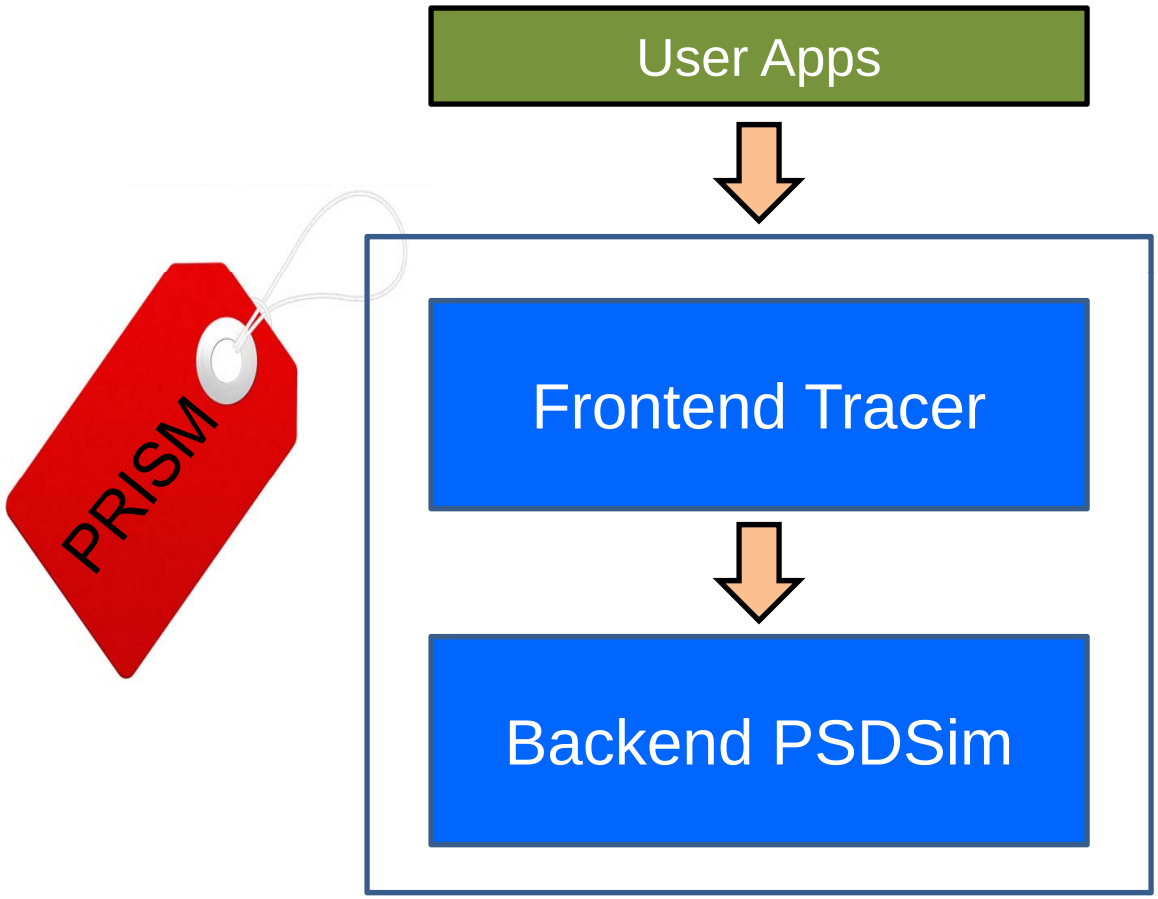
PRISM Components



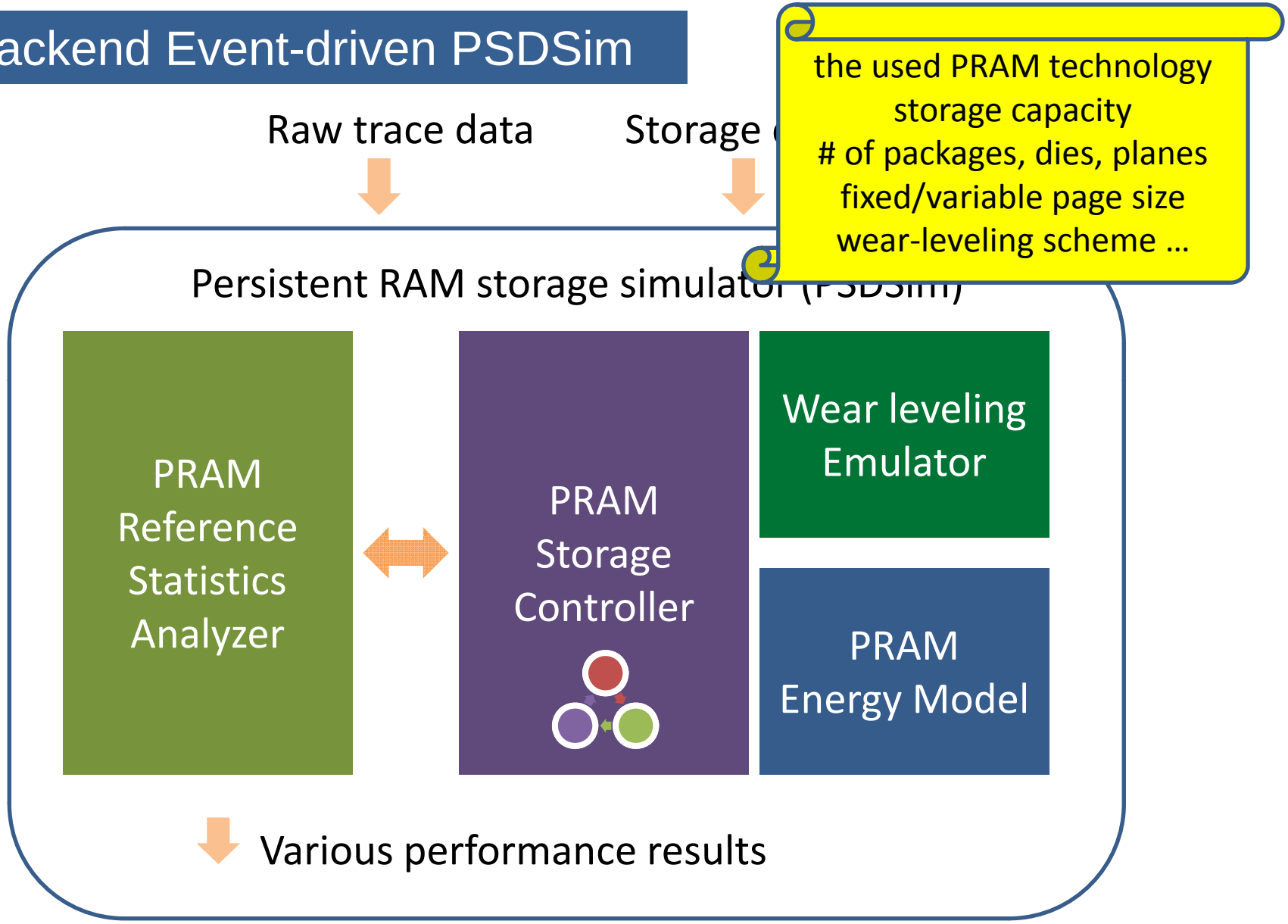
Frontend Tracer



PRISM Components



Backend Event-driven PSDSim



Preliminary Results of Case Study

- Objective: **enhancing PRAM write endurance**
- Experimental Environment

Components	Specification
CPU	Intel Core Duo 1.66 GHz (32-bit)
L2 Cache	2 MB
Main Memory	1 GB SDRAM
Operating System	Linux 2.6.24
File System	Tmpfs 10 GB capacity
Workload	TPC-C 2-hours measurement

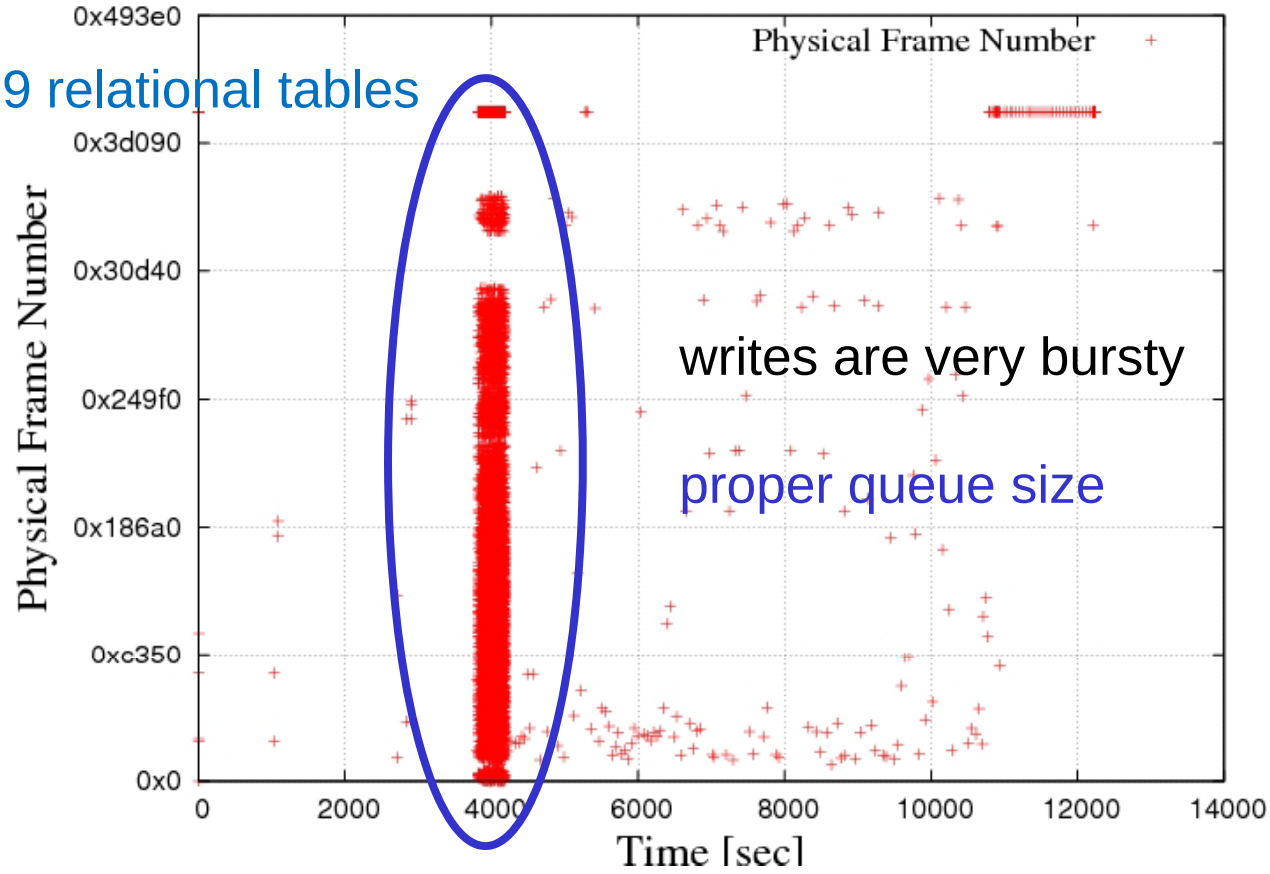
Possible Questions from Designers

- How our workloads generate writes to storage?
  1. Temporal behavior of write request pattern
  2. The most dominant data size written
  3. Virtual address space used by OS
  4. File metadata update pattern
- What about hardware resource access pattern?
  1. Resource utilized efficiently
  2. Do we need consider wear-leveling?

TPC-C File update pattern

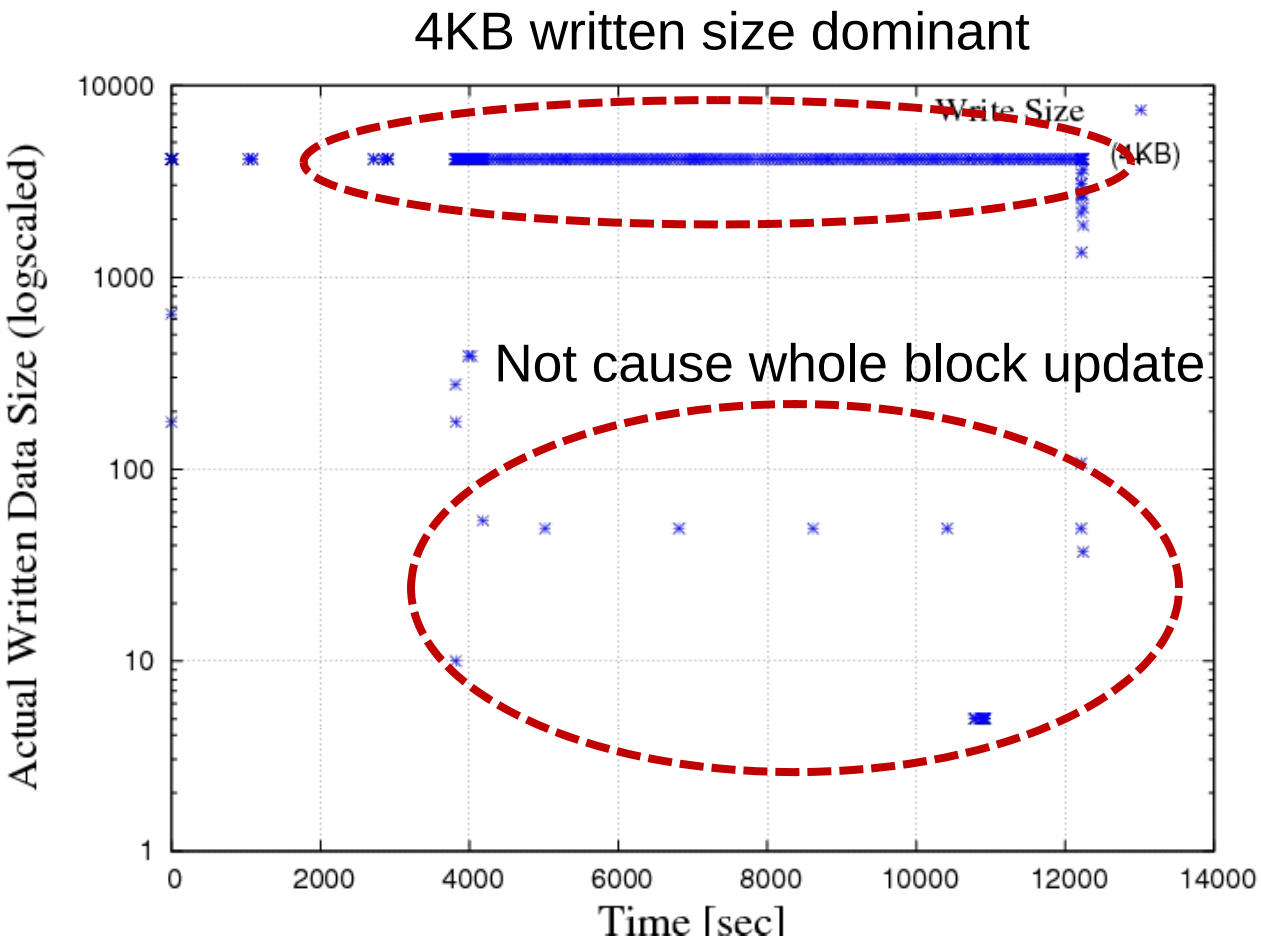
Temporal behavior of writes

Populating 9 relational tables

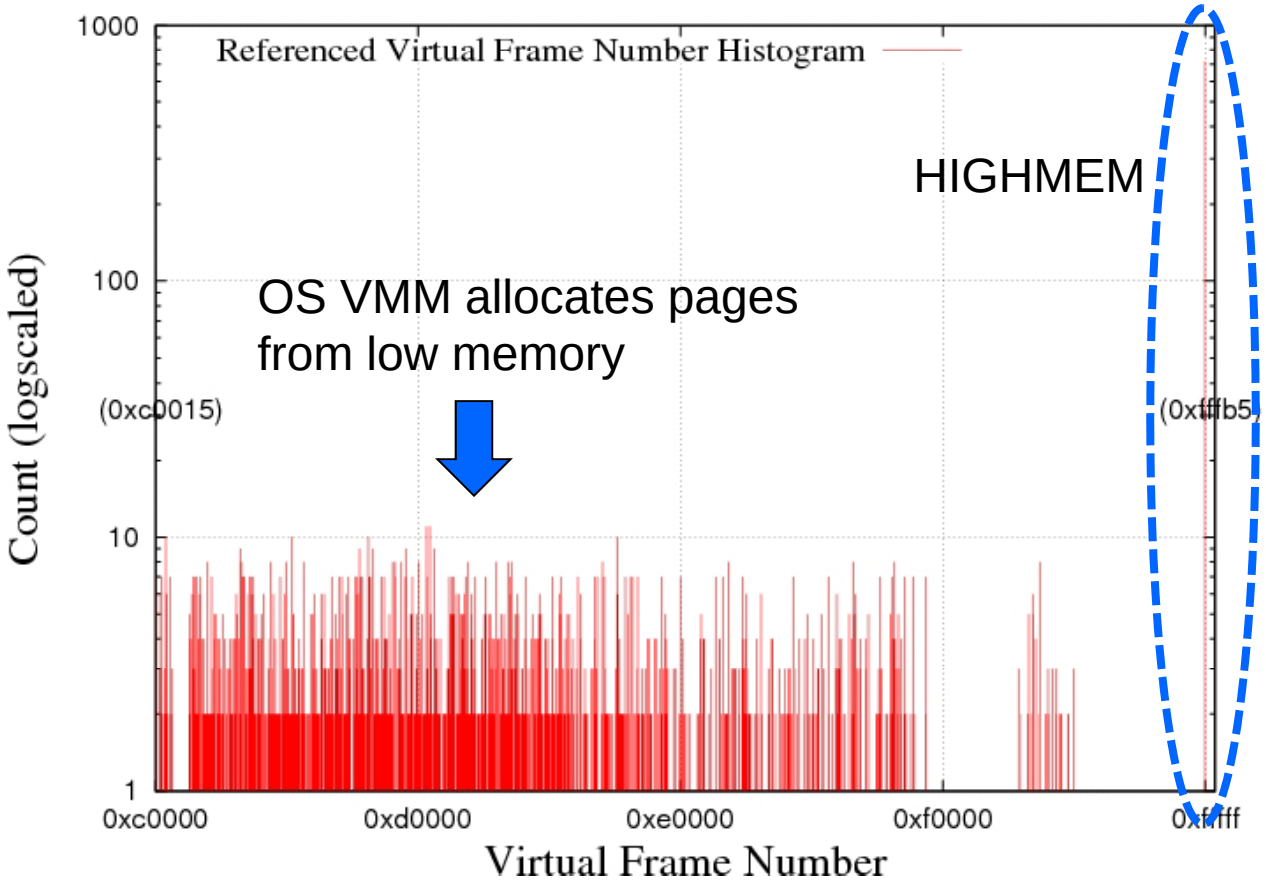


(Wall-clock time)

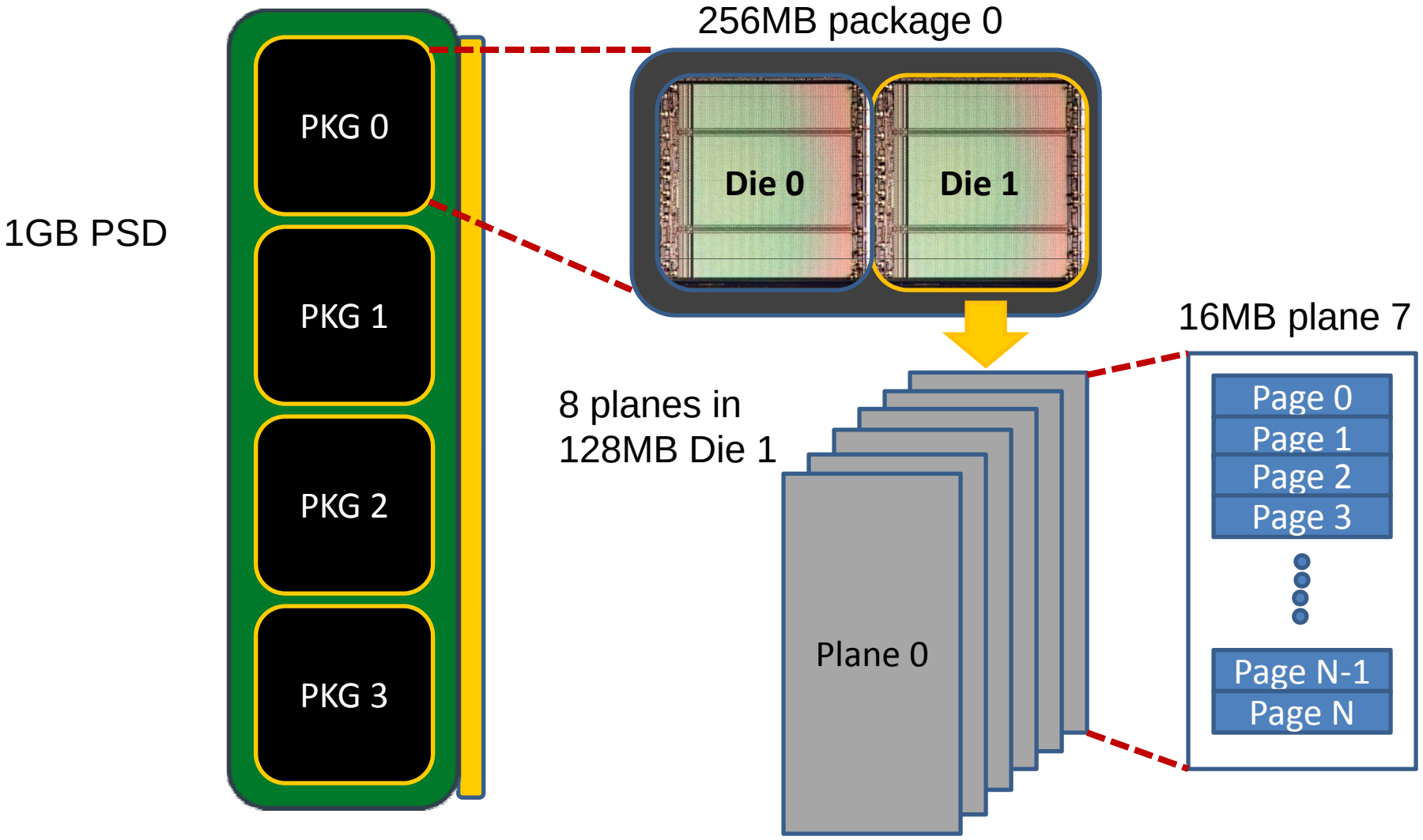
File Data Written Size



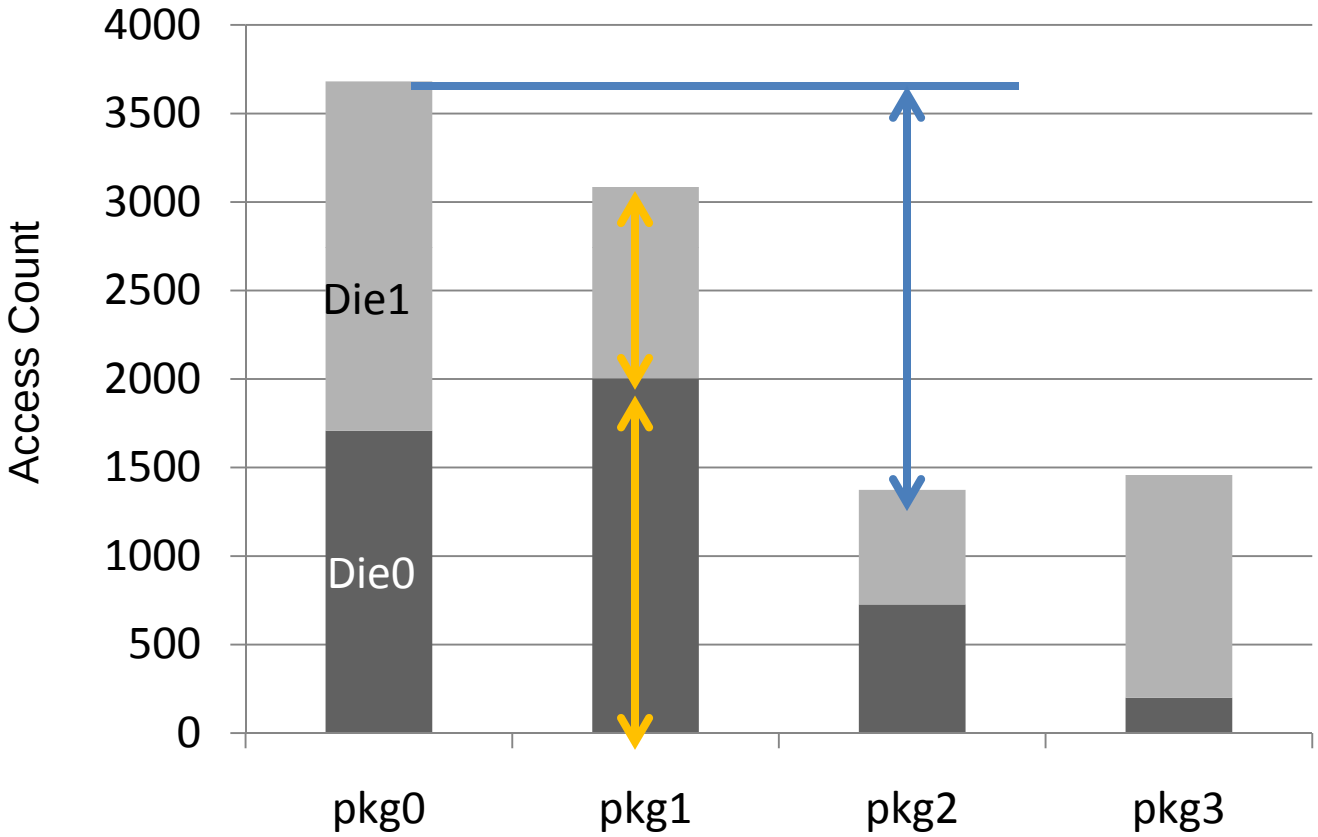
Virtual Page Reuse Pattern



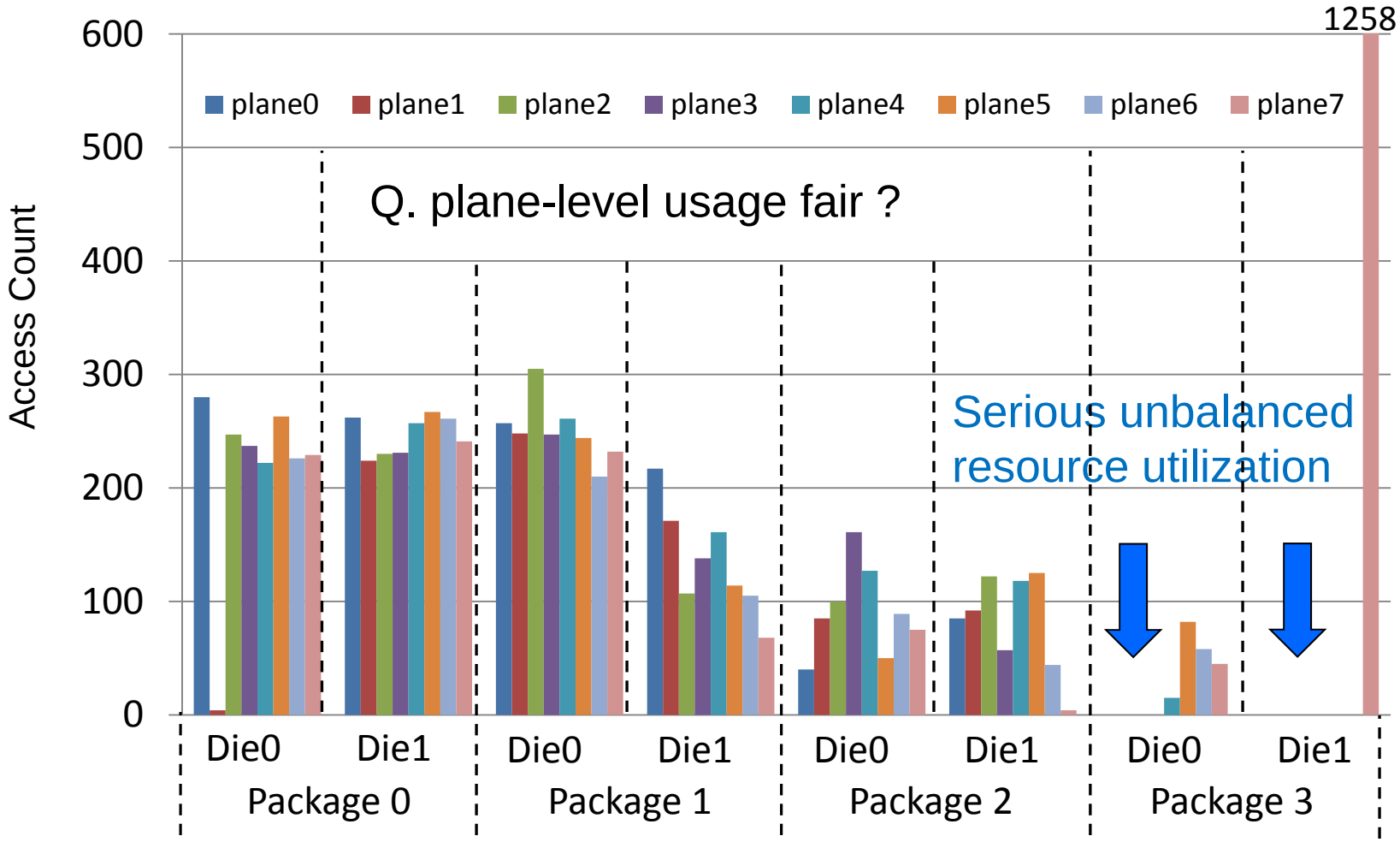
Example Hardware Organization



Hardware Resource Utilization

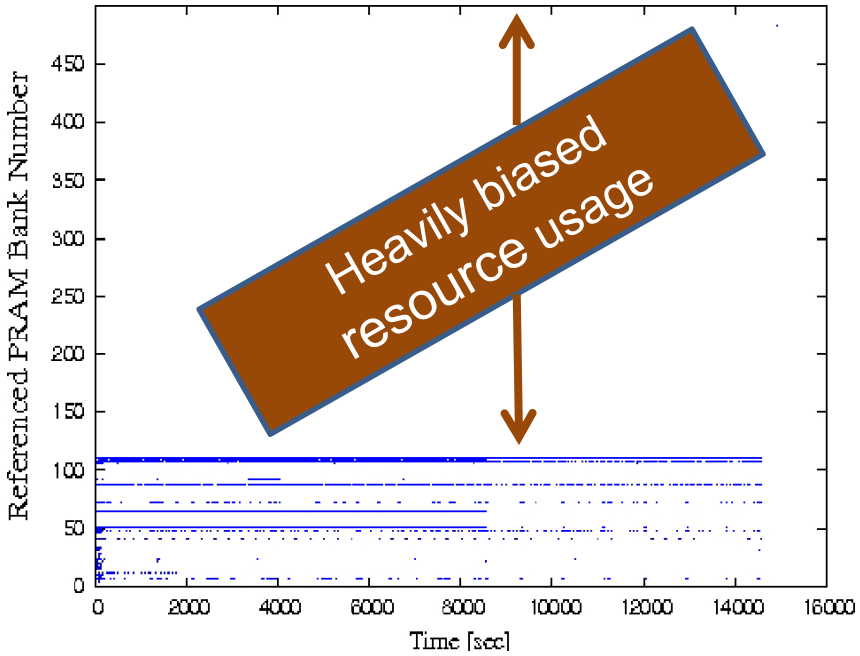


Storage-wide plane resource usage

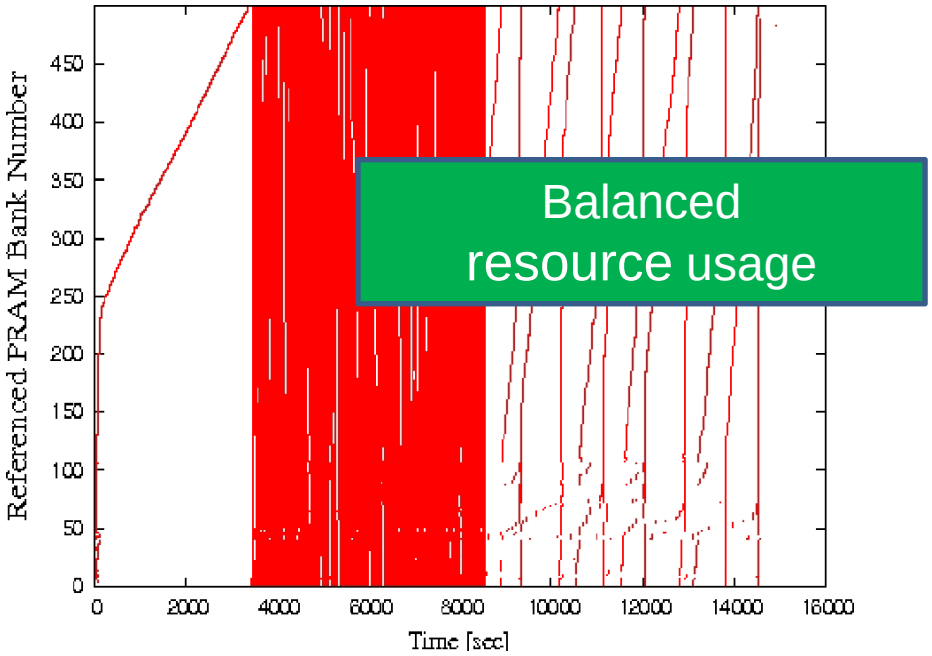


Wear-leveling Effect

Before Wear-leveling

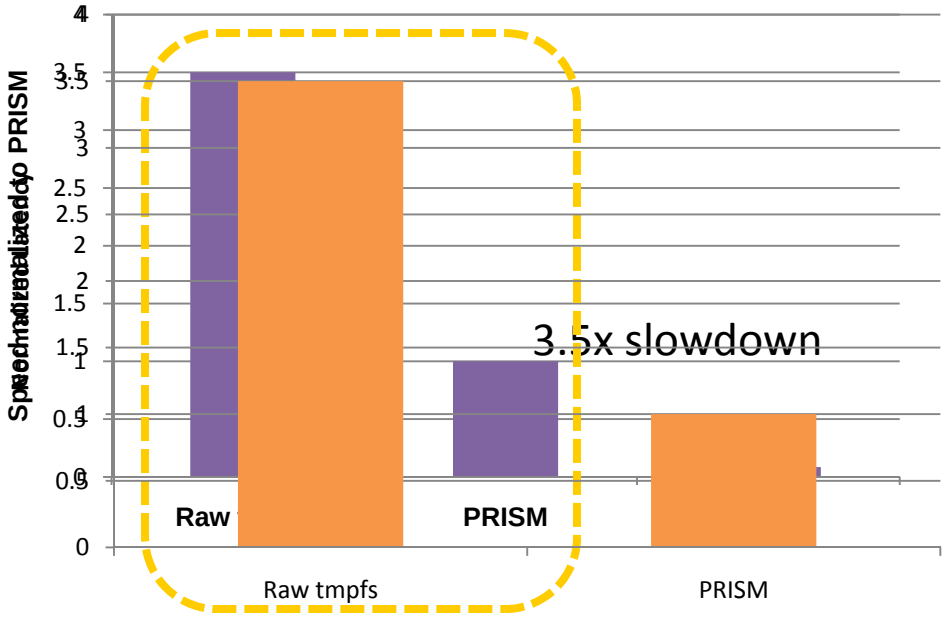


After RR Wear-leveling



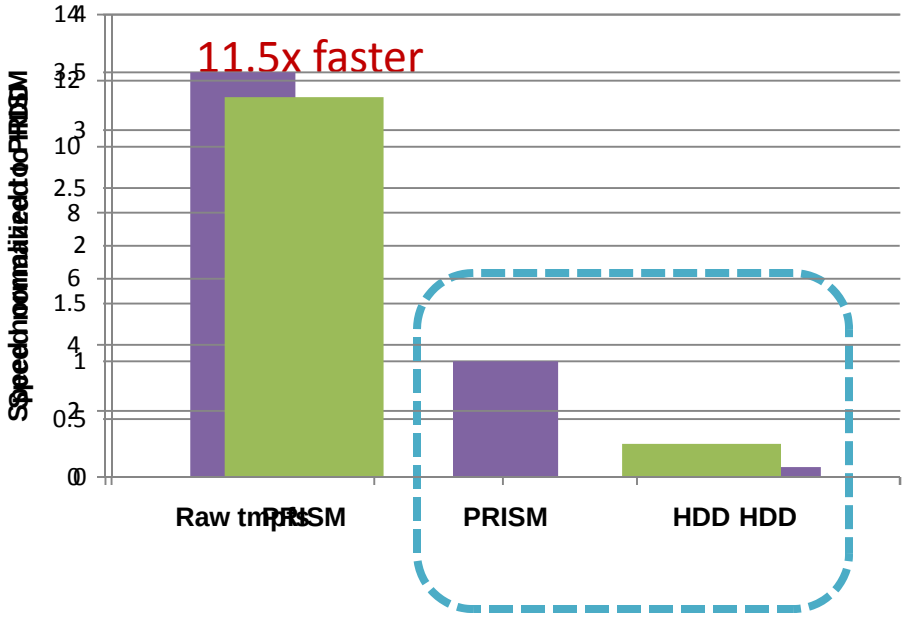
PRISM Overhead

- PRISM's I/O performance impact
  - We ran IOzone benchmark with 200MB (HDD-friendly) sequential file write operations



PRISM Overhead

- PRISM's I/O performance impact
  - We ran IOzone benchmark with 200MB (HDD-friendly) sequential file write operations



PRISM Summary

- Versatile tool to study PRAM storage system
  - Study interactions between storage level interface (programs/OS) and PRAM device accesses
  - Run a realistic workload fast
  - Extendible with user-defined tracer easily
  - Explore internal hardware organizations in detail

Thank you !