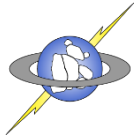
BankShot – Caching Slow Storage In Fast NVM

Meenakshi Sundaram Bhaskaran, Jian Xu, Steven Swanson

Non-volatile Systems Laboratory
Department of Computer Science and Engineering
University of California, San Diego

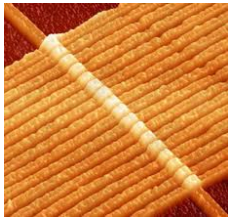
Overview

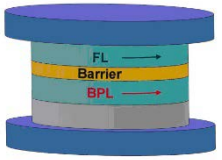
- Motivation
- System Overview
- Design Options
- Evaluation

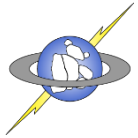


Technology Trend

	Memristor	PCM	STT-RAM	DRAM	Flash	HDD
Density (F2)	<4	4-16	20-60	6-12	1-4	2/3
Read Time (ns)	<10	10-50	10-35	10	100,000	$5\text{-}8\text{x}10^6$
Write (ns)	~10	50-500	10-90	10	100,000	$5\text{-}8\text{x}10^6$
Endurance	10^9	10^9	10^{16}	$>10^{16}$	10^4	10^4







Source: Yang, J Joshua, Dmitri B Strukov, and Duncan R Stewart. "Memristive Devices for Computing." *Nature nanotechnology* 8, no. 1 (2013): doi:10.1038/nnano.2012.240 , http://www.pdl.cmu.edu/SDI/2009/slides/Numonyx.pdf

Big Data

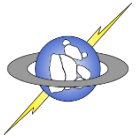
"World's Server processed 9 Zettabytes of data in 2008" – hmi.ucsd.edu

"Over half a petabyte of new data arrives in the warehouse every 24 hour" - Facebook

"Google processes over 20 petabyte of data per day" – arstechnica.com

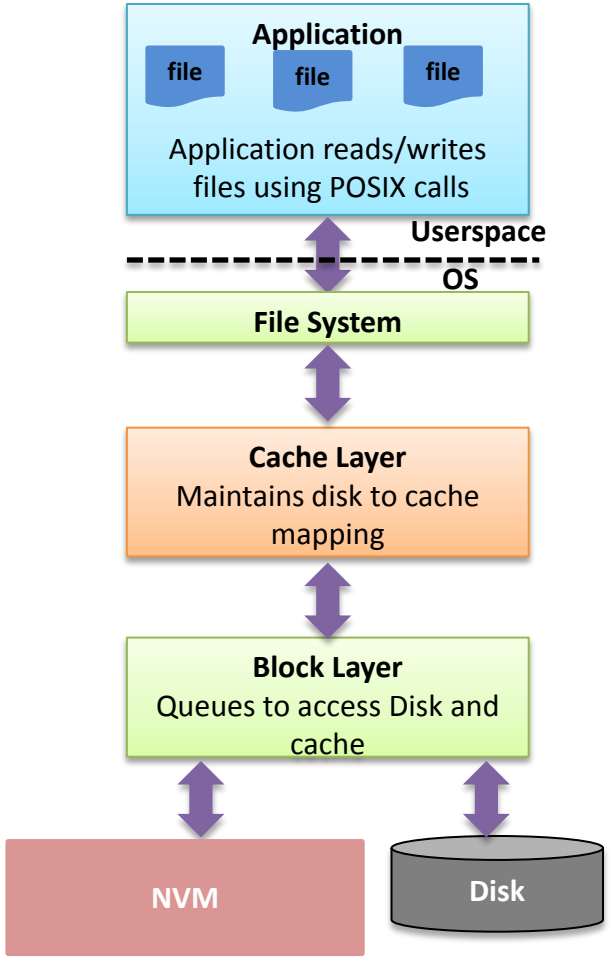
"82 per cent of traffic was focused on just 8 per cent of photos" - Facebook

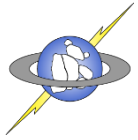
Emerging NVM are most suitable as a persistent cache for larger conventional storage



Cache Hierarchy System Design

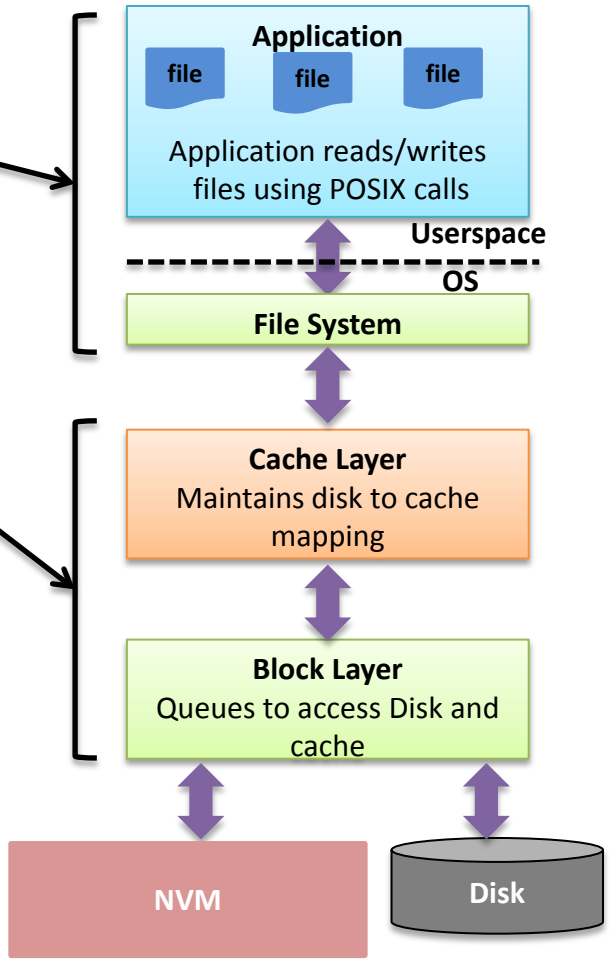
- A cache manager in kernel
  - lookup,
  - Allocation
  - eviction
- OS provides
  - Set protection
  - Mediate sharing



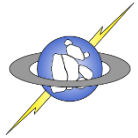


SW & OS Overhead – 4KB Cache hit

- Kernel + FS: 10.7us
- The Cache layer + Meta data persist: 21.2us
- PCM: 8us

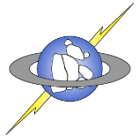


Software overhead obscures performance benefits of using Emerging NVM as Cache



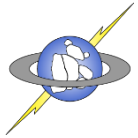
Overview

- Motivation
- Bankshot System Overview
- Design Options
- Evaluation



Bankshot – Caching System for Emerging NVM

- No OS overhead on cache hits
  - Cache hits serviced by user-space
- Write back caching
  - Specialized hardware to recover from crash
- File System Awareness
  - File extents and not disk blocks.
- Efficient write back
  - Asynchronous Write back optimized for backing store
- Data Usage Tracking
  - Hardware support for tracking cache usage



Bankshot System Stack

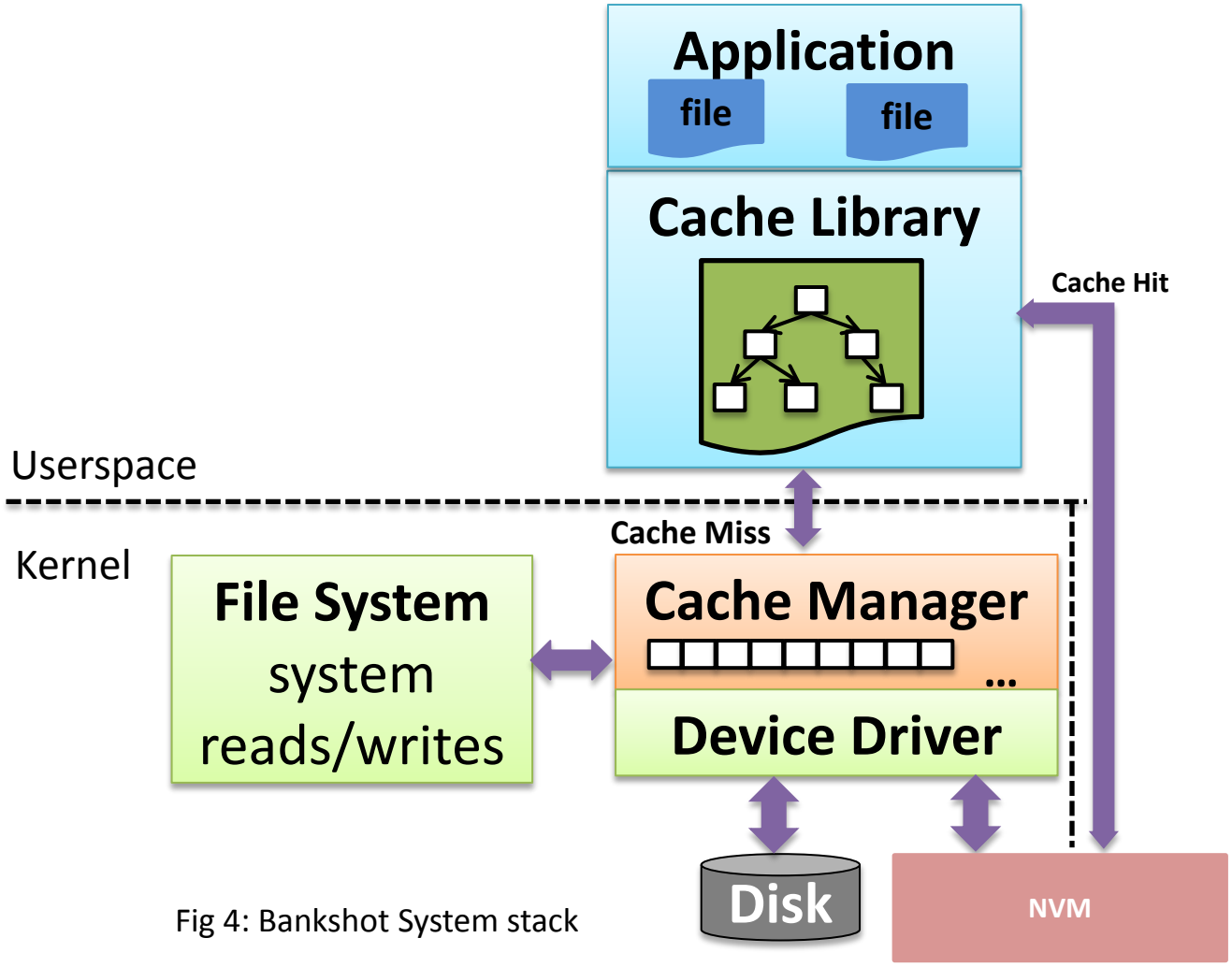
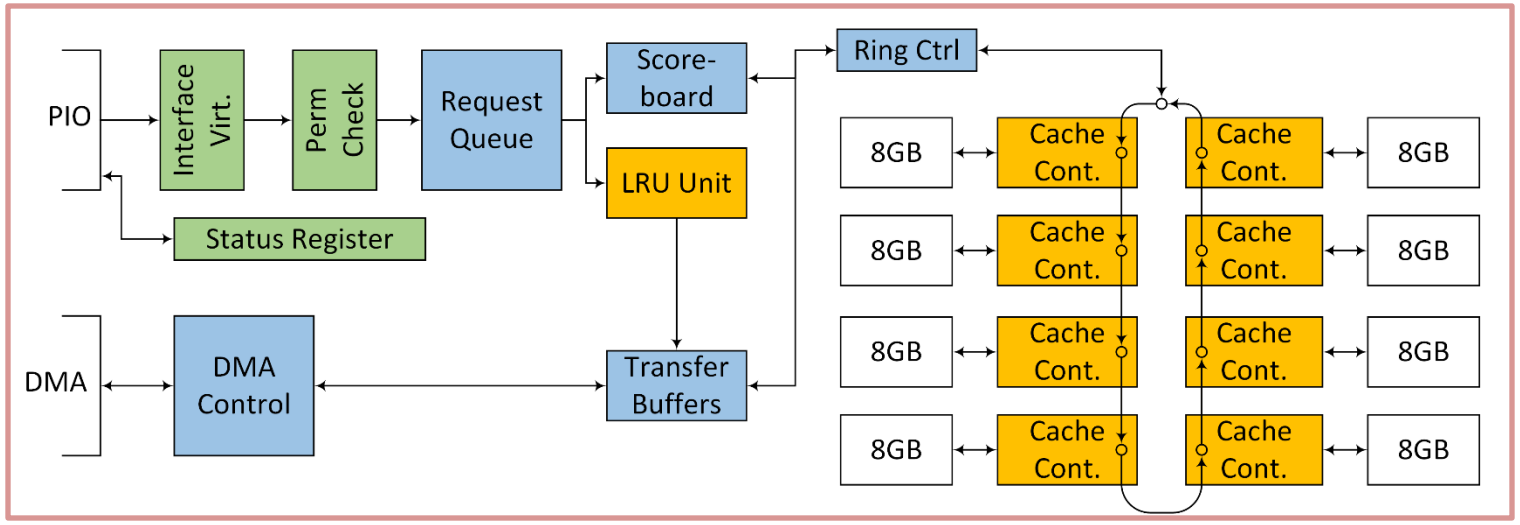
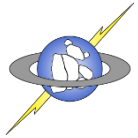


Fig 4: Bankshot System stack

Bankshot NVM

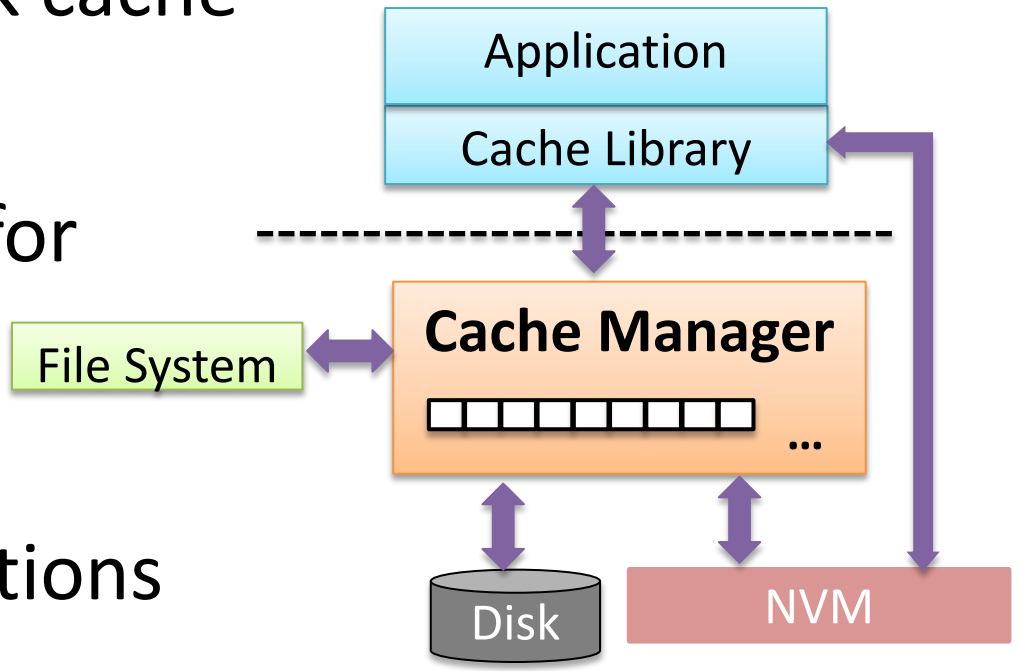


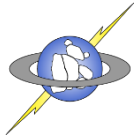
- Virtualized interface & permission check
  - Provides Safe User space access
- Controller & LRU Unit
  - Cache Specific logic
- Request Queues & DMA
  - Efficient data path



Cache Manager

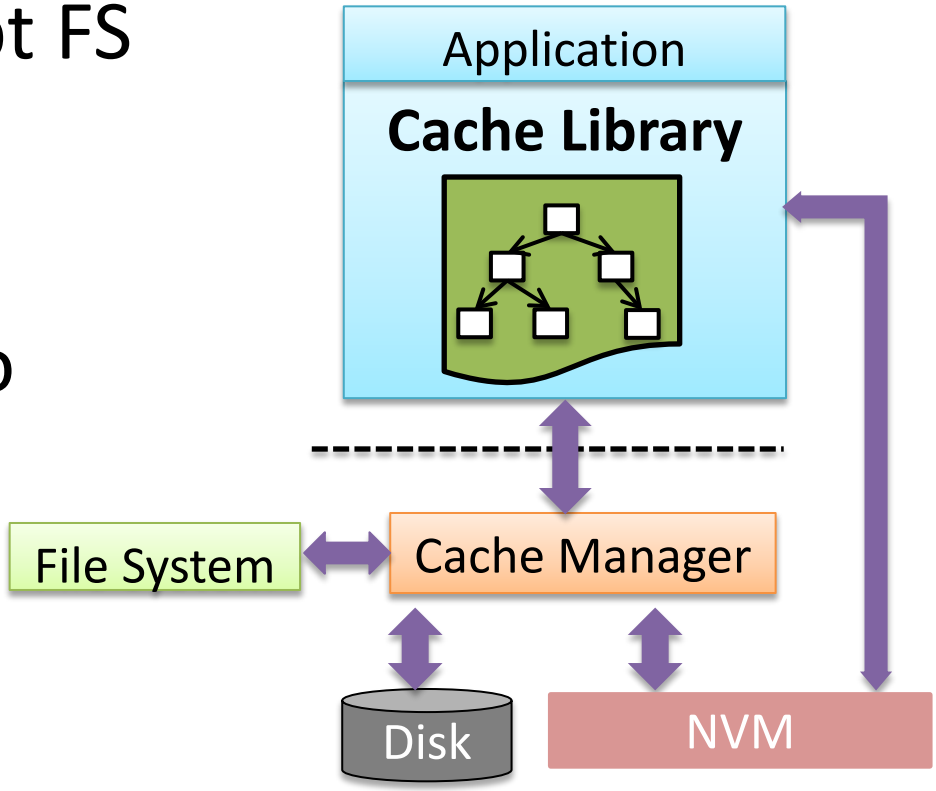
- Exposes a write back cache
- Service miss
- Queries file system for
  - File extents
  - File permissions
- Performs cache evictions
- Writes back dirty data

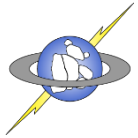




Cache Library

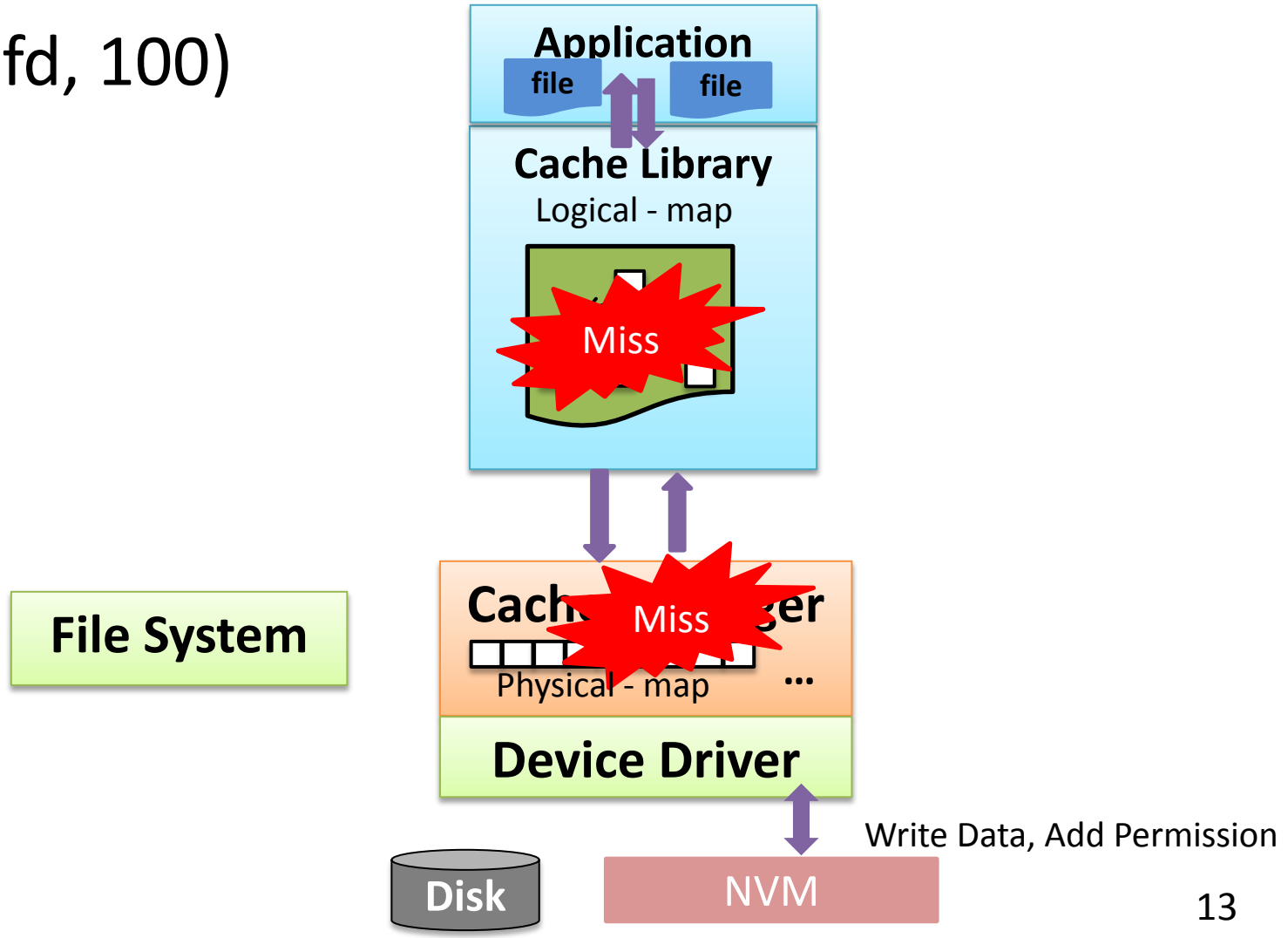
- Transparently intercept FS calls
- POSIX compatibility
- Translate file offsets to cache locations
- Issue and complete hardware requests





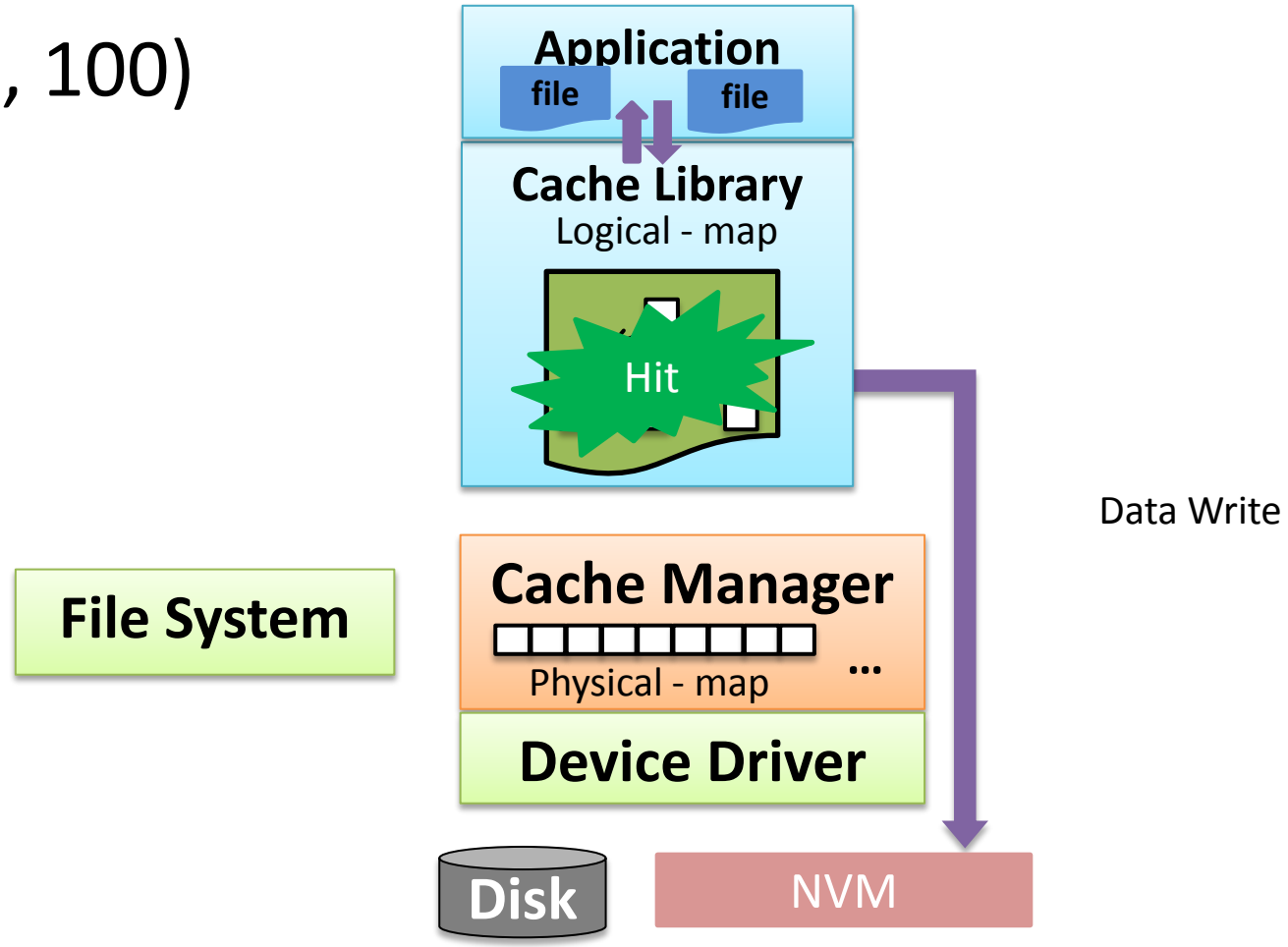
Cache Miss - Write

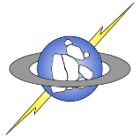
- Write (fd, 100)



Cache Hit

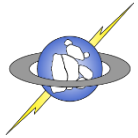
- Write (fd, 100)





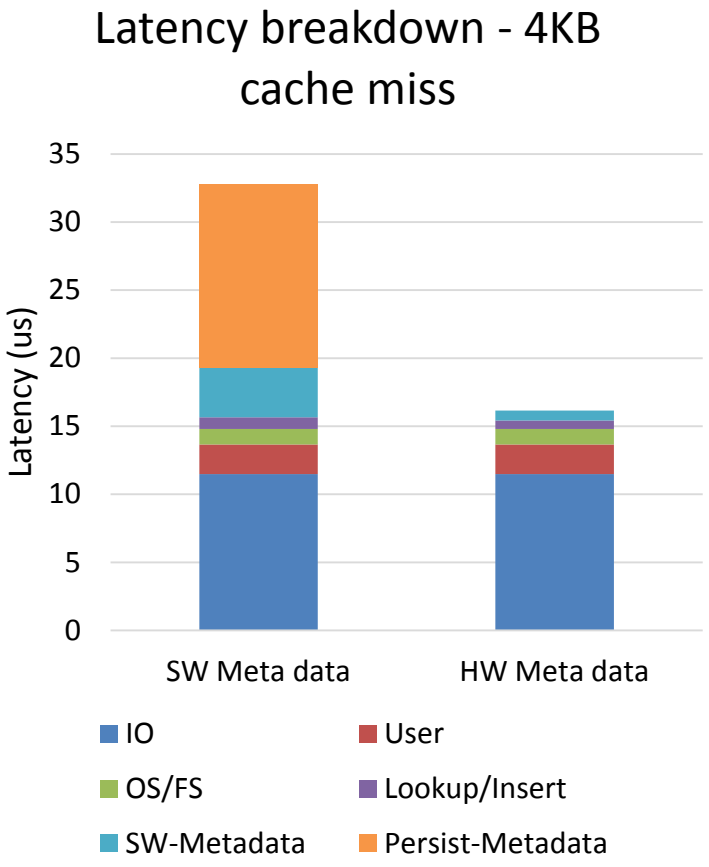
Overview

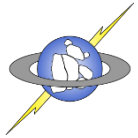
- Motivation
- System Overview
- Design Options
  - Recovery
  - Dirty Data Tracking
  - Replacement Policy
- Evaluation



Cache Recovery

- Persist cache map for durability
- Software:
  - Requires extra IO
- Hardware:
  - HW stores map atomically





Dirty Data Tracking - Cache Organization

- Fully associative
- Divided into 4KB blocks
- 2048 Blocks = 1 Chunk
- chunks map to file
  - Allows Sharing
  - Units of protection

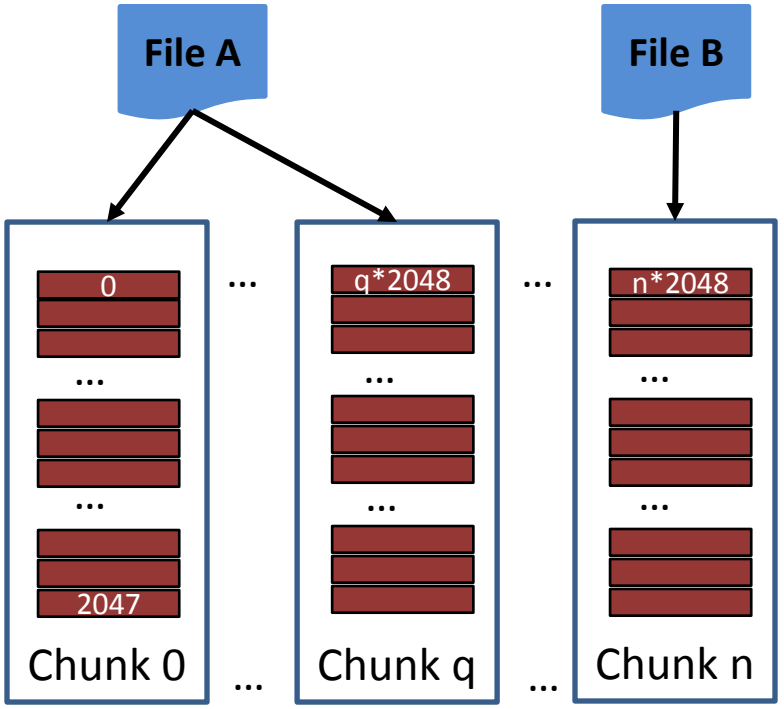
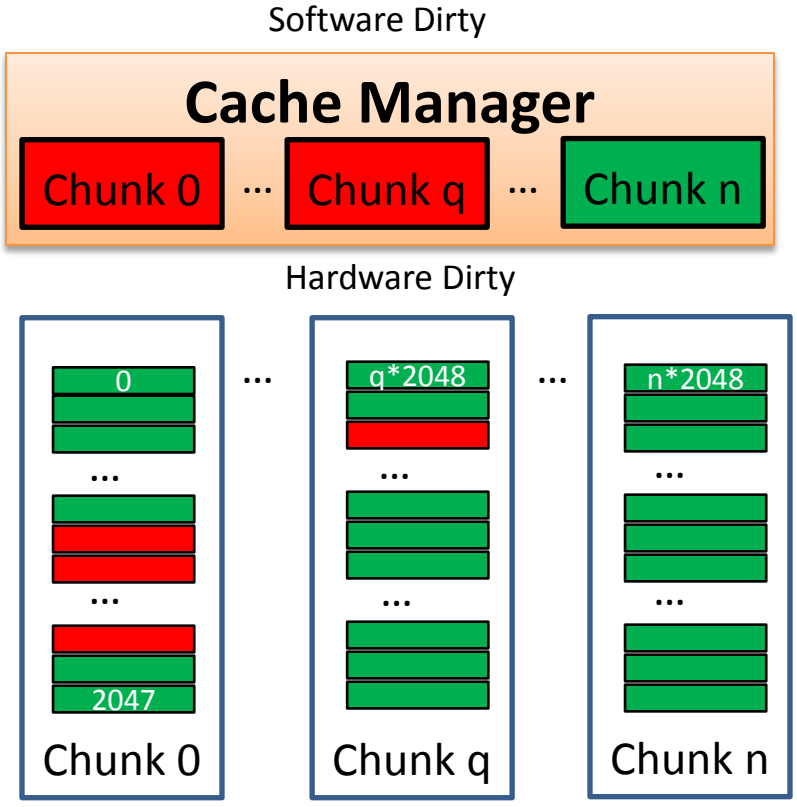
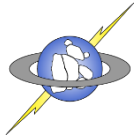


Fig 5: Cache organization

Dirty Data Tracking

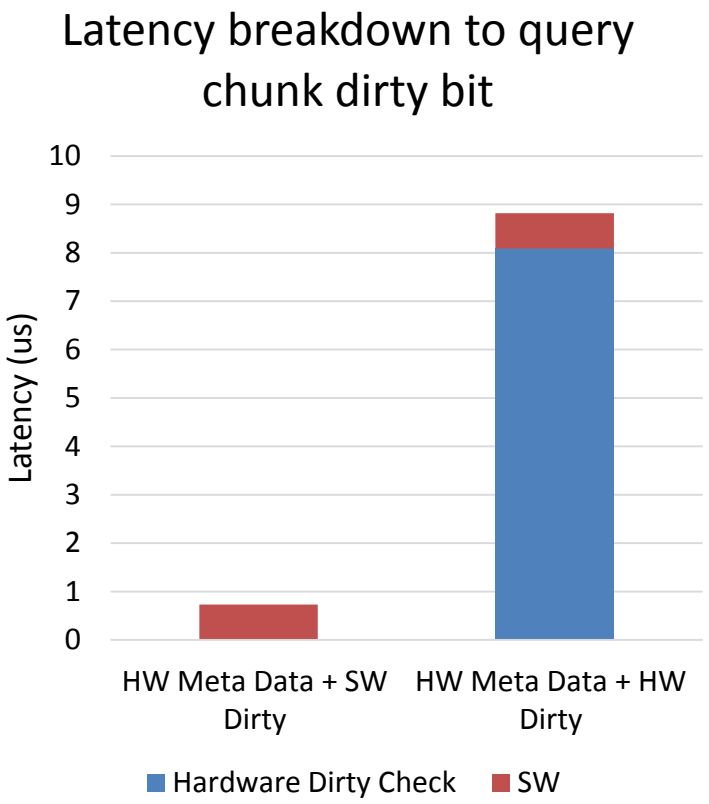
- Software
  - Per chunk tracking
  - Set on write to a block in chunk
- Hardware
  - Per 4KB block tracking
  - Set on application write
  - Queried by cache manager
- Hybrid
  - Combines hardware & software dirty

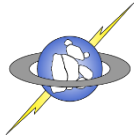




Dirty Data Tracking

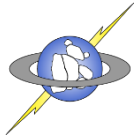
- HW Dirty
  - 8us to read dirty bit for a chunk
- Hybrid scheme
  - IO to NVM only for dirty chunks
  - Disk IO to write back takes several ms





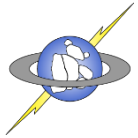
Replacement Policies

- FIFO
  - Simple software only scheme
  - Chunks evicted in order of allocation
- LRU
  - Hardware tracks recency of access to chunks
  - Cache manager queries HW for LRU chunk
  - Improves hit ratio
  - IO takes only 5us



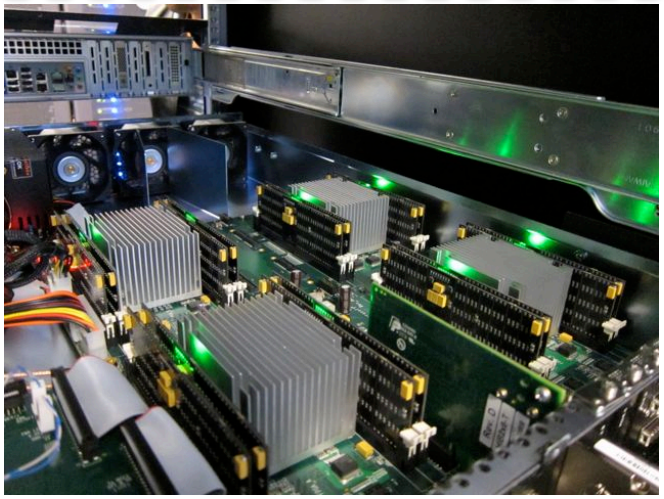
Overview

- Motivation
- System Overview
- Design Options
- Evaluation



Bankshot Prototype

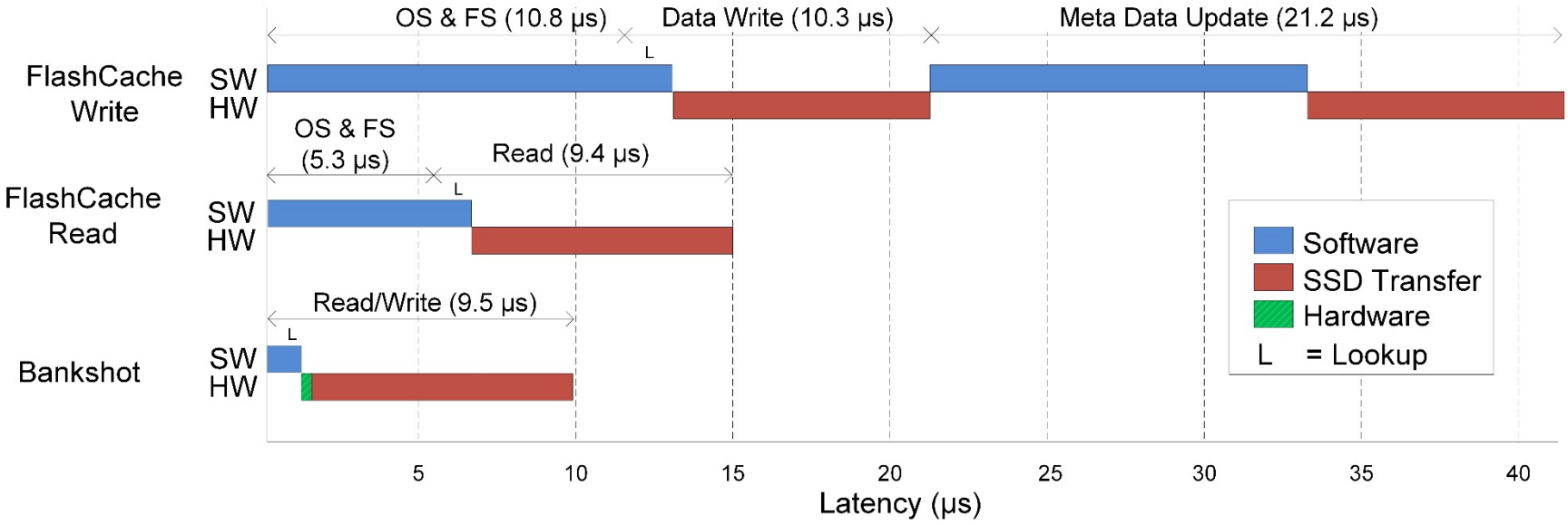
- BEE3 FPGA platform
- PCIe 1.1 x8, 2GB/s full duplex
- 64GB of emulated PCM
- Intel x86-64
- Linux kernel 2.6.32



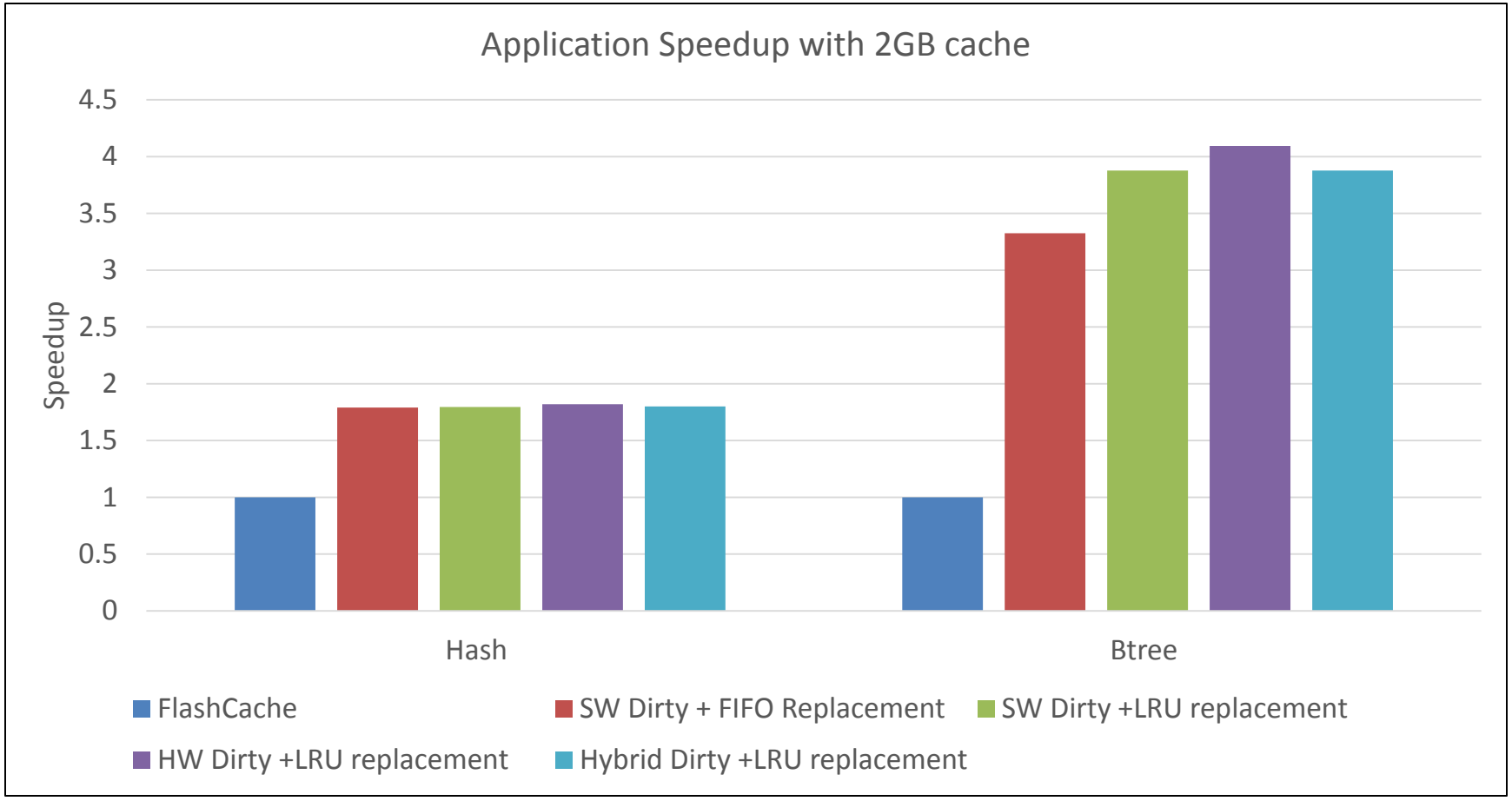
Adrian M. Caulfield, Arup De, Joel Coburn, Todor I. Mollow, Rajesh K. Gupta, and Steven Swanson. 2010. Moneta: A High-Performance Storage Array Architecture for Next-Generation, Non-volatile Memories. In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture* (MICRO '43). IEEE Computer Society, Washington, DC, USA, 385-395.
Adrian M. Caulfield, Todor I. Mollov, Louis Alex Eisner, Arup De, Joel Coburn, and Steven Swanson. 2012. Providing safe, user space access to fast, solid state disks. In *Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems* (ASPLOS XVII). ACM, New York, NY, USA, 387-400.

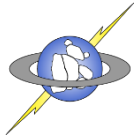
Evaluation – 4KB Cache hit latency

- Bankshot improves
  - Write latency by 4x
  - Read latency by 1.5x



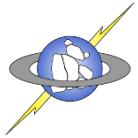
Application Performance





Conclusion

- Eliminate OS/FS interactions
- Reduces SW Overhead for hits to ~1us
- Intelligent hardware
  - ability to track dirty bit
  - Store mapping information durably
  - Virtualized interface



Thank You