ENGINEERING
TEXAS A&M UNIVERSITY

HMMU: A Hardware-based Hybrid Memory Management Unit

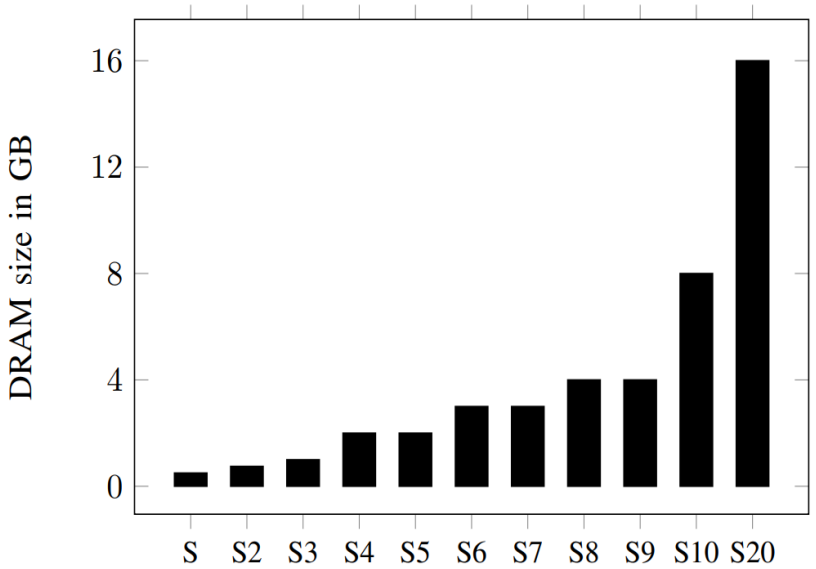
Fei Wen, Mian Qin, Paul Gratz, Narasimha Reddy

2021

Mobile Computing Platform

ENGINEERING
TEXAS A&M UNIVERSITY

- Ever-increasing memory footprint of mobile applications
- Dilemma:
  - Demand for larger memory.
  - DRAM needs constantly refresh stored data bits
  - Limited energy budget



Hybrid Memory Solution

- Non-volatile-memory (NVM)
  - Minimal standby power
  - Higher capacity density
  - Lower cost per GB
  - Long access latency, limited write endurance

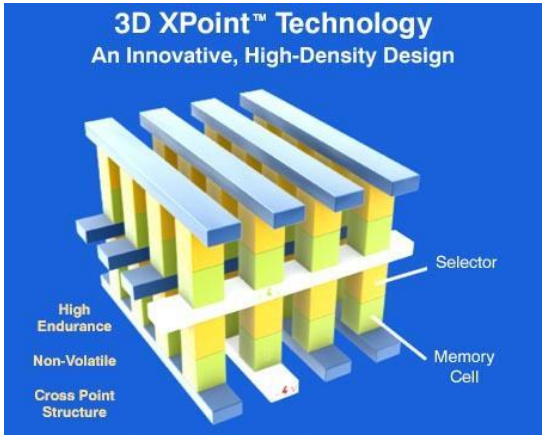
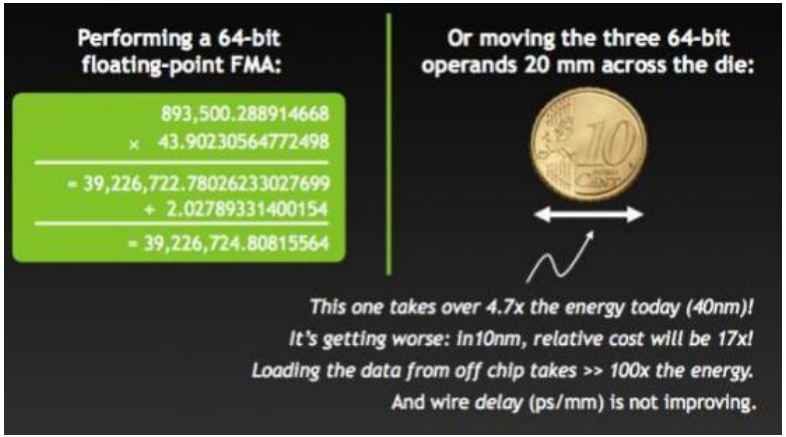


Figure Source: newsroom.intel.com

Technology	DDR4	3Dxpoint
Read Latency	50ns	100ns
Write Latency	50ns	300ns
Read Energy	4.2nJ	1.28nJ
Write Energy	3.5nJ	8.7nJ
Background Power	30mW/GB	~ 0

A common problem: Energy Consumption vs. Big Data



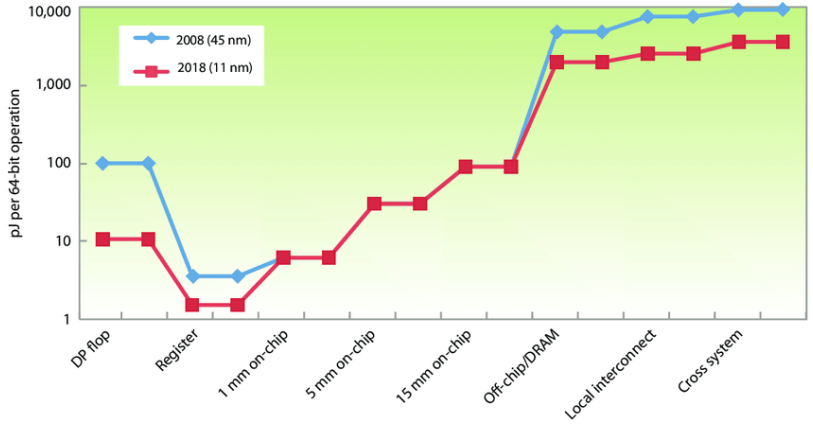


Figure Source: Nvidia Public

- Data movement overtakes computation as the dominant power consumption.
- Big question : How to move data smart and efficiently (and avoid movement at all)?
- For mobile computing: how to efficiently migrate data between memories?

Introduction

ENGINEERING
TEXAS A&M UNIVERSITY

- The question I'm trying to answer:

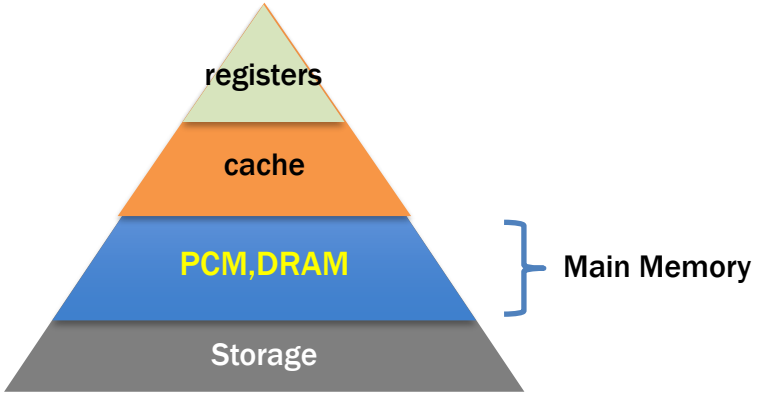
Can we architect the mobile memory system to satisfy growing demands of memory capacity.

  - Dilemma of growing DRAM size and limited energy budget
  - Growth of mobile application memory footprint
  - Data movement now consumes more power than computation.
- Solutions
  - Mobile computing memory system: improve data placement and migration between different memory devices

Chapter 1:
Hardware Memory Management for future Hybrid Memory Systems

Memory Hierarchy

- Emergence of new memory technology
  - PCM, memristor, 3D Xpoint, etc.
  - Higher storage density
  - Lower energy consumption
  - Lower cost than DRAM
  - Longer access latency

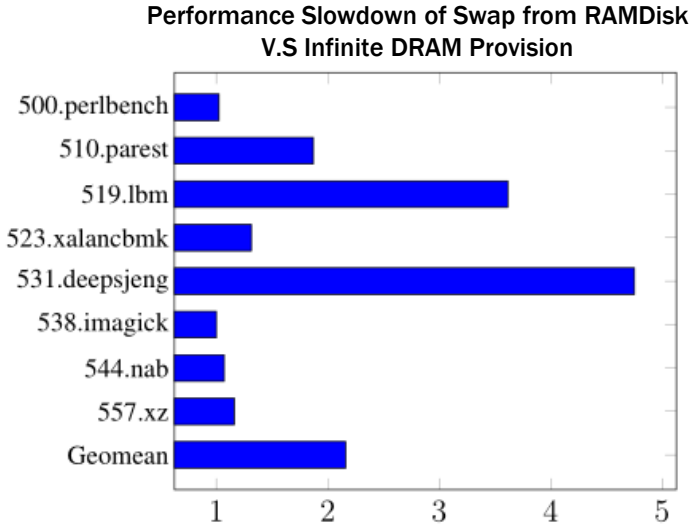


Technology	HDD	FLASH	3Dxpoint	PCM	DRAM
Read Latency	5ms	$100\mu s$	$7-8\mu s$	10-50ns	50ns
Write Latency	5ms	$100\mu s$	$18-30\mu s$	50-500ns	50ns
Energy per bit	1-10mJ	0.00002pJ	0.025pJ	2-15pJ	0.005
Endurance (Cycles)	$> 10^{15}$	10^4	10^7	10^9	$> 10^{16}$
$ per GB	0.025-0.5	0.25-0.83	1.7-2.4	N/A	5.3-7.8

New Memory Technologies Require Hierarchy Rearchitecting

ENGINEERING
TEXAS A&M UNIVERSITY

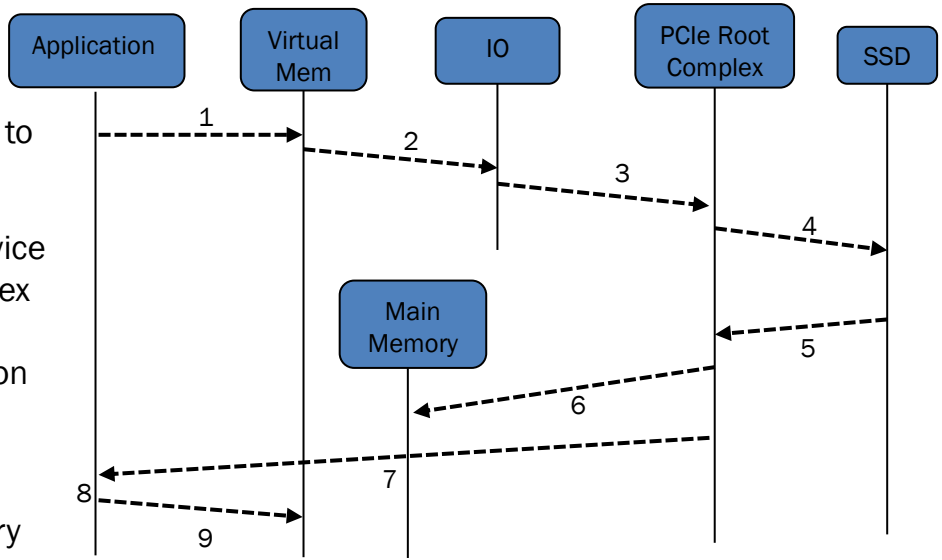
- Current utilization of new memory technology
  - Emerging NVM SSD to improve storage system
  - Limited impact on memory pressure for data intensive apps.
  - OS management increases page fault penalty.



Page fault penalty analysis

ENGINEERING
TEXAS A&M UNIVERSITY

1. Application generates a request to virtual memory system
2. Page missed, virtual memory forwards the request to I/O
3. I/O generates request to PCIe root complex
4. PCIe root complex relays request to actual SSD device
5. SSD returns the requested data to PCIe root complex
6. PCIe root writes data back to the main memory
7. PCIe root raises an interrupt notifying the application
8. The application reissues request to the virtual memory
9. The page is hit at in virtual memory this time
10. Data is obtained in the corresponding main memory



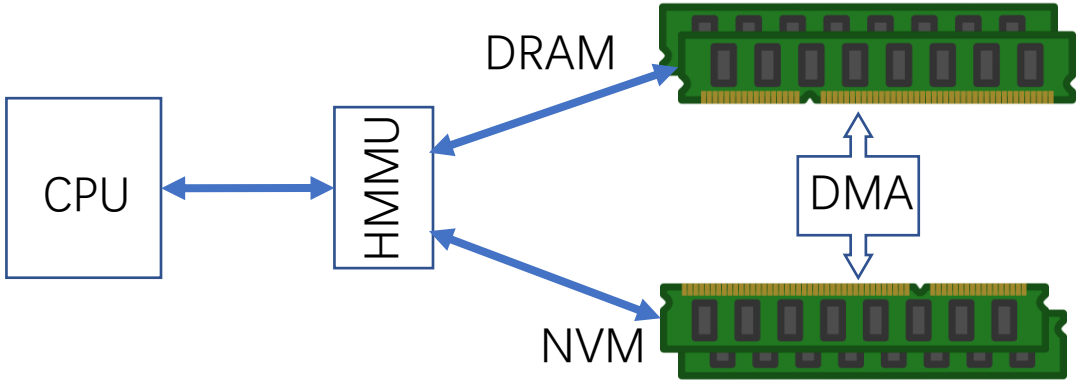
As the device speed of NVM improves, the ratio of OS processing time becomes an even larger portion of page fault penalty.

Related Works

ENGINEERING
TEXAS A&M UNIVERSITY

- Use DRAM as cache for NVM[D. Jevdjic et al. MICRO-2014][M. K. Qureshi et al. IEEE-2014]
  - Pros:
    - only needs small changes to existing architecture and software
    - Managed by hardware, no operating system intervention.
  - Cons:
    - Hard to manage such a large number of tags efficiently
    - Less memory is visible to OS than the actual resource that we have.
    - Loss of the parallel accesses to two memory devices
- Use two different memories in a single flat address space [C. Su et al, ACM-2015][J. Sim et al, IEEE-2014][V. Fedorov et al, MEMSYS-2017]
  - Pros: More memory capacity is visible/utilizable to OS
  - Cons:
    - Needs OS involvement in page placement and management.
    - Page table and TLB update is quite costly

Hybrid Memory Management Unit (HMMU)



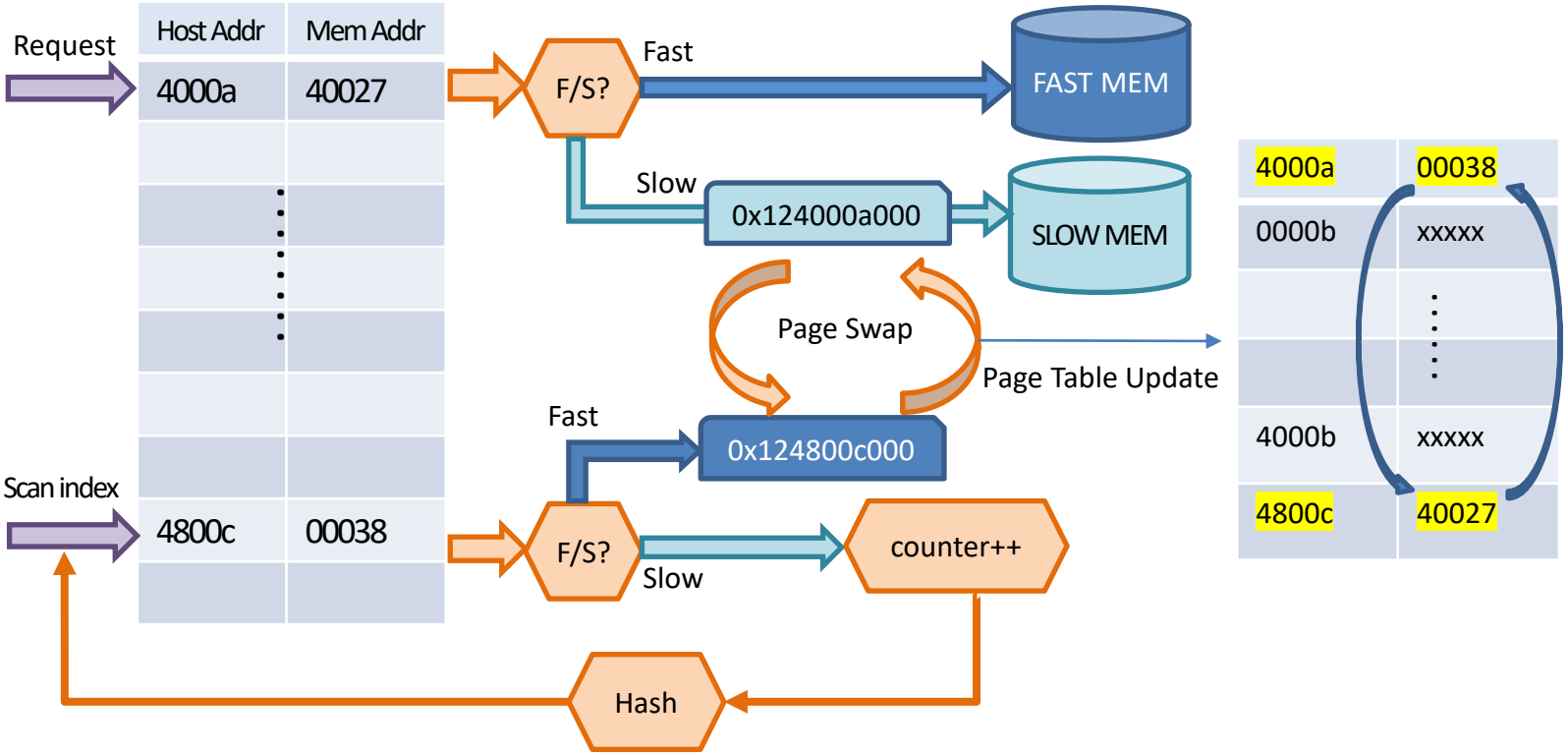
- Best of both worlds: Parallel DRAM and NVM memory
  - The large datasets stored in NVM memory - high capacity
  - Recently and frequently used data in DRAM - fast access
- Data allocation and migration between two tiers of memory managed by hardware
  - After the data is obtained from NVM on demand, the DMA engine prefetches the next a few lines to DRAM
  - The page movement engine archives the unused data from DRAM to NVM periodically
- More efficient than traditional OS, workload is offloaded to the background

Hybrid memory management unit design

ENGINEERING
TEXAS A&M UNIVERSITY

- Phase 1: First simple, hardware-only replacement policy
  - Simple “random” victim page selection
- Phase 2: Multi-granularity memory management
  - Adapt to granularity of locality
  - Reserve small partition of fast memory for high locality chunks of slow pages
  - Migrate full page when sufficient fragments have been touched
- Phase 3: Speculative management and prefetching
  - Explore options for prefetching and speculative cache management

Phase 1 architecture



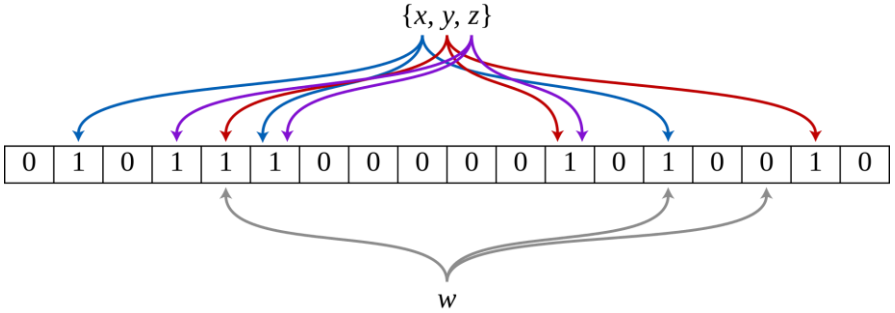
Phase 1: Counter-based Page Relocation Policy

ENGINEERING
TEXAS A&M UNIVERSITY

- How to search for the victim page on DRAM for data swap?
  - Internal page redirection table. Each entry has a host address and actual memory (advice) address.
  - Need to search a fast page and record both its reference address and actual memory address
  - Any reference address can be redirected to any physical page. This is actually a 512k (2GB/4KB)fully associative search.
  - This search needs to be done in limited cycles.

- Possible solutions
  - Keep and maintain a LRU list of pages.
  - Surprisingly, a counter-based pseudo random page search works just fine.[T. Johnson et al. VLDB][E. J. O'Neil et al. ACM]
- Reasoning: The counter incremented monotonically, the chance is pretty low to hit a recently relocated fast page again.
- Use bloom filter to protect the recently accessed pages from being replaced.

Bloom Filter



- A space-efficient data structure
- Quickly test whether an element belongs to a predefined set
- $k = \frac{m}{n} \ln 2$ k: number of hash functions, m:vector length, n:number of elements
- After modification, it supports remove operation.
- For efficiency, we only tracked the last 128 pages.
- Safety: no false negative errors, only false positive errors.

Phase 2: Multi-level Granularity Memory Management

ENGINEERING
TEXAS A&M UNIVERSITY

- Motivation:
  - Relocating 4k pages consumes substantial time and DRAM bandwidth
  - If only one or few blocks are hit, it's inefficient to move the whole 4K data.
- We propose:
  - Manage the memory data in various and flexible granularity, from page level to cache block level.
  - If a page is needed for limited times, only relocated the specific demanded blocks of data.

Block-level (fragment) cache-like management

ENGINEERING
TEXAS A&M UNIVERSITY

- A small partition of DRAM reserved for sub-page block data with high localities, under cache-like management
  - 4-way associate cache Pseudo LRU replacement policy
  - Cache insertion upon first touch to a slow-mem page
- Allocation policy
  - Examine several sub-page vs. full page allocation policies.

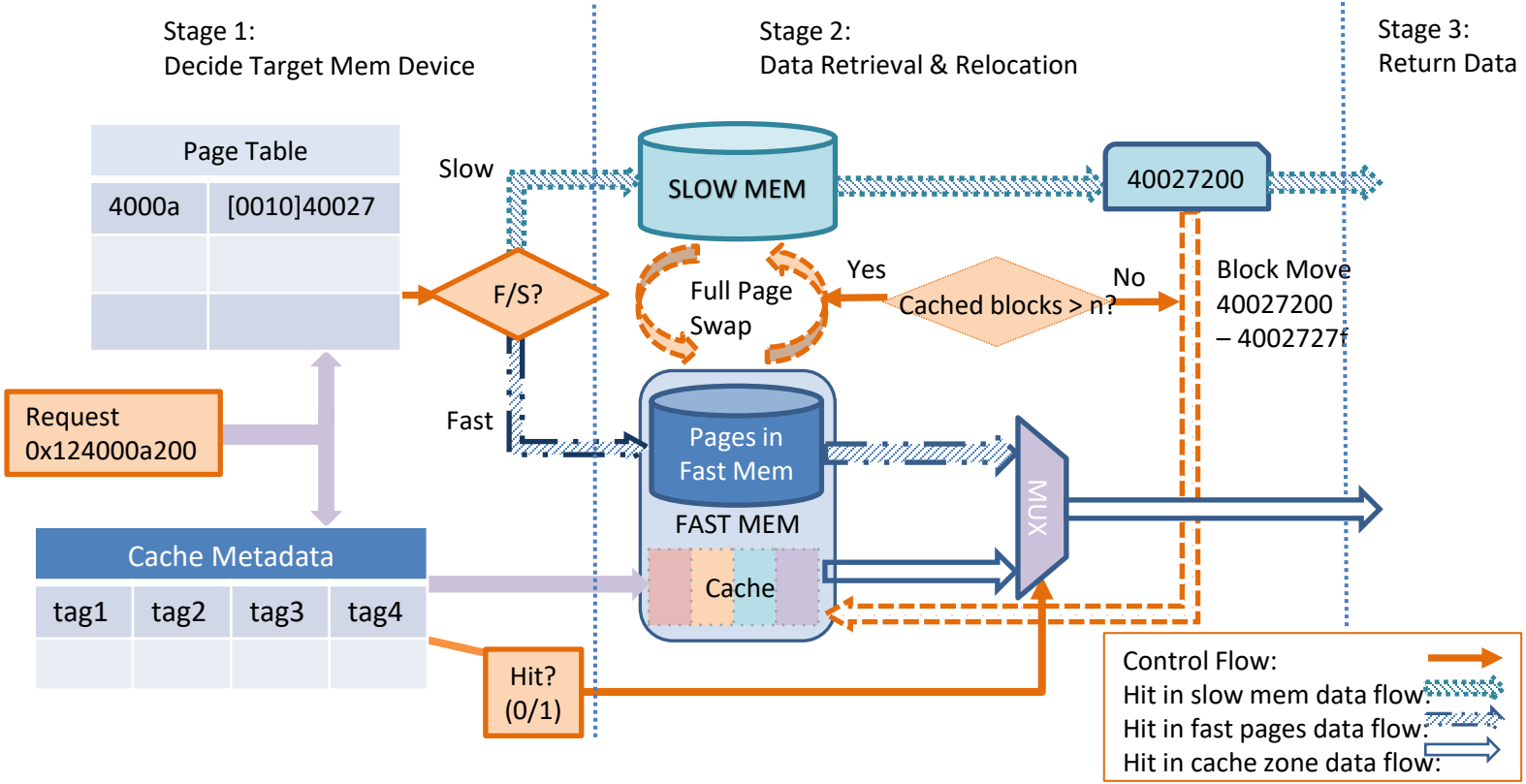
Sub-page vs. full page allocation

ENGINEERING
TEXAS A&M UNIVERSITY

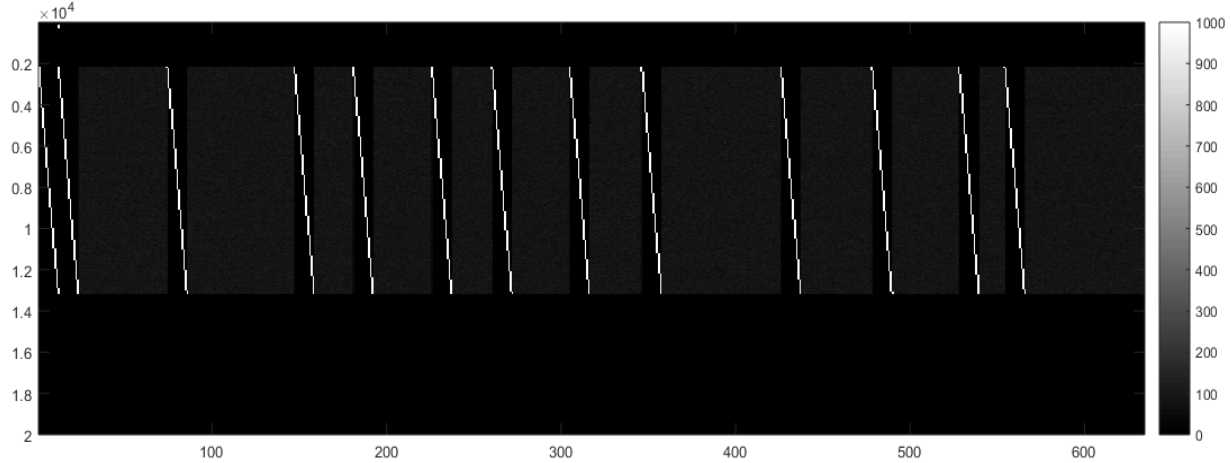
- Caching for pages w/ few blocks touched, page movement for fully utilized pages
- Switch to full page movement when blocks touched passes a threshold.
  - A bitmap for each page table entry.
  - Each bit indicates whether a miss occurred to the corresponding section of that page.
- Alternative policy: counts the number of cached blocks of the page
  - If >N bits set, trigger the process of whole page relocation from slow memory to the fast memory.
  - Otherwise we only relocate that particular touched block.

Page bitmap				HostAddr	Mem Addr
0	1	0	1	4000a	40027
1	0	0	0	4800c	00038

Hybrid Block/Page Relocation Scheme



Speculation and Pre-fetch

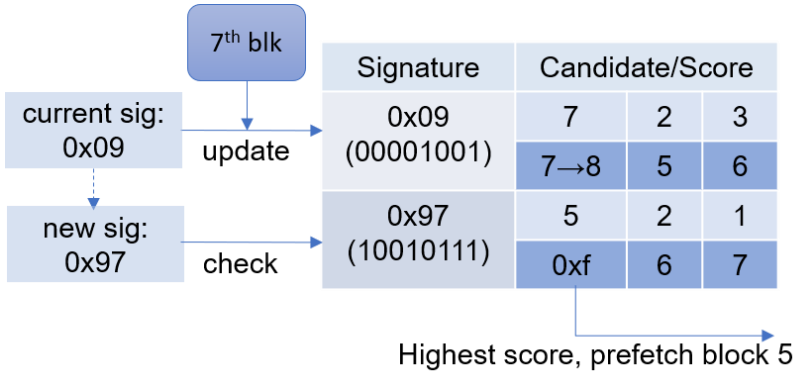


- PageMove did not perform that well with 531.deepjseng and 519.lbm
- Profiling results shows they have large working set but low spatial locality
- Prior works [J. Kim et al. ASPLOS '17][P. Michaud, HPCA'2016]indicate lightweight algorithms can predict future demand.

Signature Based Prefetch with Throttling

ENGINEERING
TEXAS A&M UNIVERSITY

- Signature based prefetch
- Light-weight meatadata, low hardware overheads.
- Monitor the prefetch accuracy online.
- Throttle the prefetch activities when accuracy drops to below certain level.

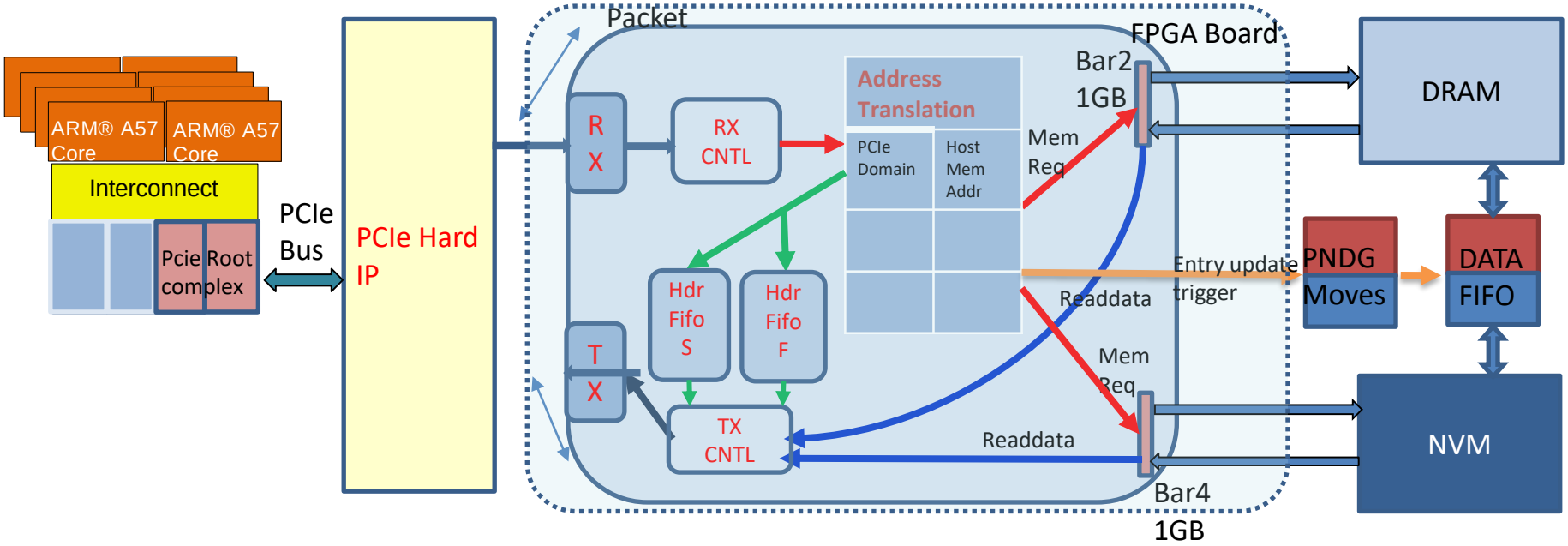


Experimental Methodology

ENGINEERING
TEXAS A&M UNIVERSITY

- Hardware setup: FPGA-based evaluation platform
  - ARM A57 processors
  - PCIe Gen. 3 8.0 Gbps
  - 128MB fast + 1GB slow memory attached via PCIe link
  - HMMU implemented in FPGA.
- Workload
  - Random address / random value memory test application
  - SPEC2017 CPU benchmarks

Emulation System



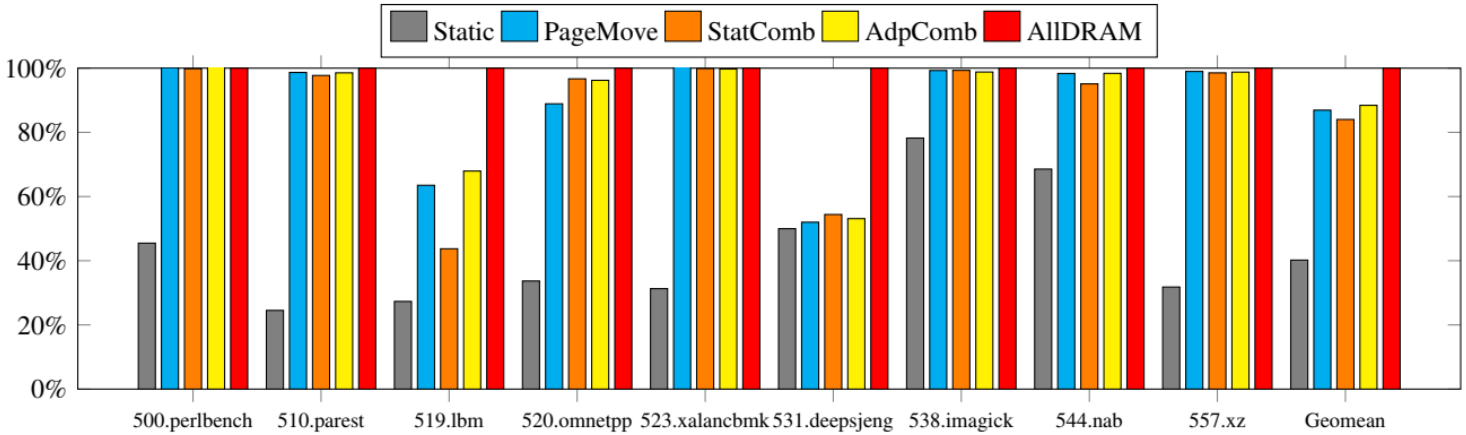
- Applications were run on real hardware, no software simulation involved.
- More accurate and much higher efficiency than software simulation.
- Generate arbitrary cycles of delays to mimic different NVM devices.
- Independent and parallel access to two memories.

Policies Under Test

ENGINEERING
TEXAS A&M UNIVERSITY

- Static: A baseline policy in which host requested pages are randomly assigned to fast and slow memory.
- PageMove: The whole 128MB DRAM is managed on the granularity of 4k pages.
- StatComb: 16MB out of the 128MB DRAM is reserved for sub-page block relocation, managed in the cache-like fashion. The remainder of the DRAM is managed on a full page. The threshold value is picked by heuristic.Prefetch is enabled.
- AdpComb: Same as StatComb, except that it uses an adaptive threshold to determine when the full page should be moved. Prefetch is enabled.
- AllDRAM: A baseline policy that presumes infinite DRAM resource, is used to estimate the upper performance bound.

Performance Speedup with SPEC 2017 Benchmark



- The page policy achieved 86.9% performance compared to all DRAM, AdpComb 88.4%, while static allocation yields only 40%.

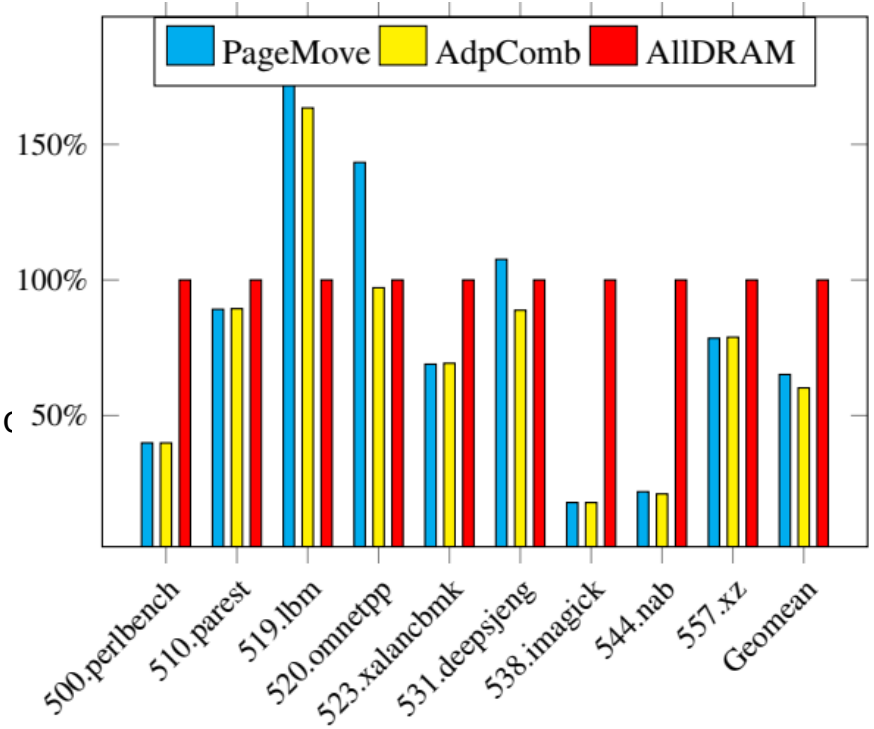
Energy Consumption

ENGINEERING
TEXAS A&M UNIVERSITY

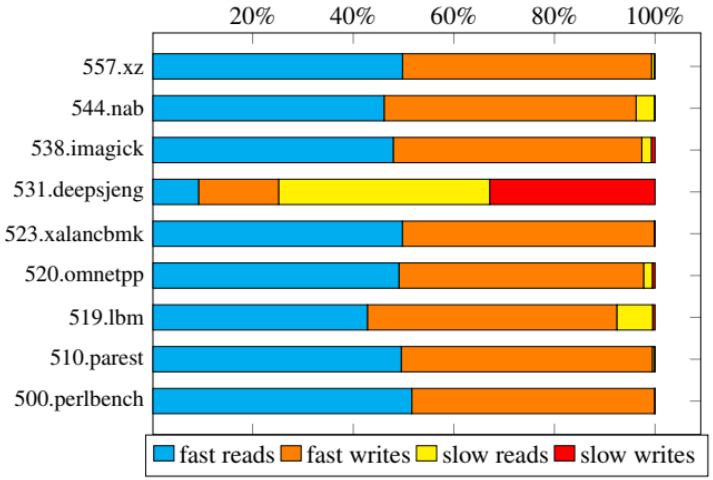
Table 4: Power Consumption of DDR4 and 3D-XPoint

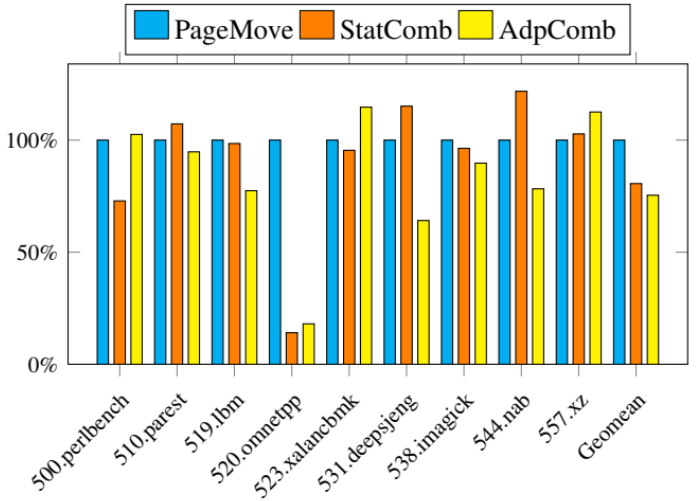
Technology	DDR4	3Dxpoint
Read Latency	50ns	100ns
Write Latency	50ns	300ns
Read Energy	4.2nJ	1.28nJ
Write Energy	3.5nJ	8.7nJ
Background Power	30mW/GB	~ 0

- $E_{total} = N_{rd} \times E_{rd} + N_{wr} \times E_{wr} + P_{static} \times T$
- Normalized against the energy consumption c
  AllDRAM.
- Energy Consumption(lower the better)
  - AdapComb:60.2% (Best)
  - PageMove:65.1%
  - StatComb:63.6%



NVM writes





- Combined mode has 20% less writes than the page move policy, which could save energy consumption and the NVM device lifetime.

Conclusion

ENGINEERING
TEXAS A&M UNIVERSITY

- Hybrid memory (DRAM and NVM) solves the dilemma of limited energy and increasing memory footprint of mobile applications.
- We used pure hardware memory management unit (HMMU) to avoid the prolonged kernel processes used by prior works.
- We carefully design the data placement and migration policies.
- Data could be managed in flexible granularities.
- Our optimal policy achieves 88.4% performance of AllDRAM, while saving ~40% energy.

ENGINEERING

TEXAS A&M UNIVERSITY

Thank you all for attendance!

Question?