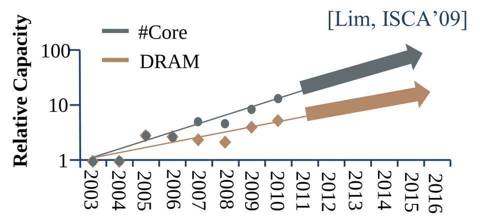
Runtime Solutions to Apply Non-volatile Memories in Future Computer Systems

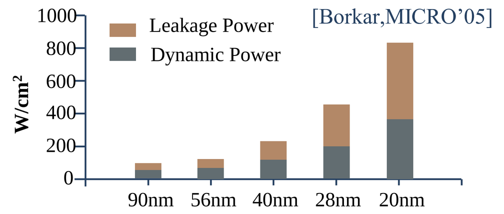
Hoda Aghaei Khouzani
Email: hoda@udel.edu

1. Introduction

I. Trends Affecting Main Memory

DRAM has been used as main memory from 1970. However, DRAM technology scaling is ending!

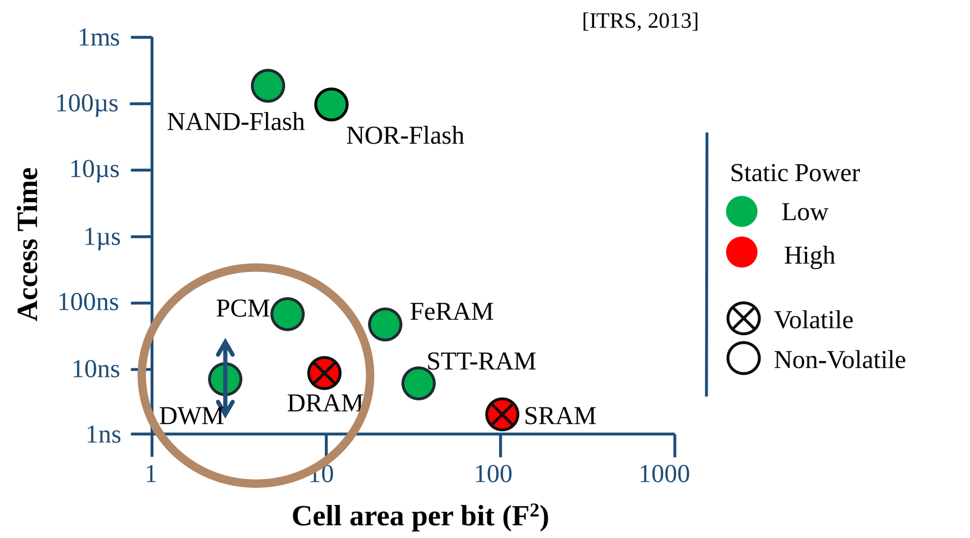




Memory capacity per core expected to drop by **30%** every two years [Mutlu, IMW'13]

DRAM consumes up to **40%** of system **energy** due to the need of consistent refresh cycles [Udipi, ISCA'10]

II. Emerging Memory Technologies



III. Potential Candidates for Future Main Memory

Phase Change Memory (PCM)	Domain Wall Memory (DWM)
Advantages:	Advantages:
– Higher scalability than DRAM	– Ultra dense
– Near zero static power	– Near zero static power
– Similar read performance and energy to DRAM	– Similar read/write performance and energy to DRAM
Challenges:	Challenge:
– Limited write endurance (10^6~10^9)	– Sequential access structure
– Long write latency (10x DRAM)	
– High write energy (4x DRAM)	

2. Prolonging PCM Limited Lifetime

I. Related Works on PCM Lifetime

Defect Tolerance	Write Reduction	Wear-leveling
• Applies when cells die • Increases access latency	• At bit level • Reduces average writes • Lifetime limits by the maximum	• At various levels of granularity • Balances writes • Extra writes due to remap

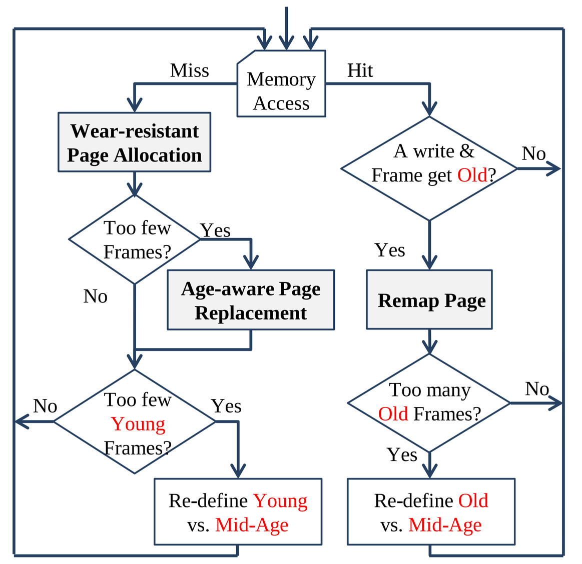
II. Basic Elements of Our Proposed Wear-leveling

✓ **Inter-process**: as endurance is a lifetime factor, wear leveling is done across different processes.

✓ **Segment-based**: segment information is used to define *hot/cold* pages.

✓ **Age-aware**: page allocation uses *young* and *old* page frames to hold *hot* and *cold* pages, respectively.

III. System Overview

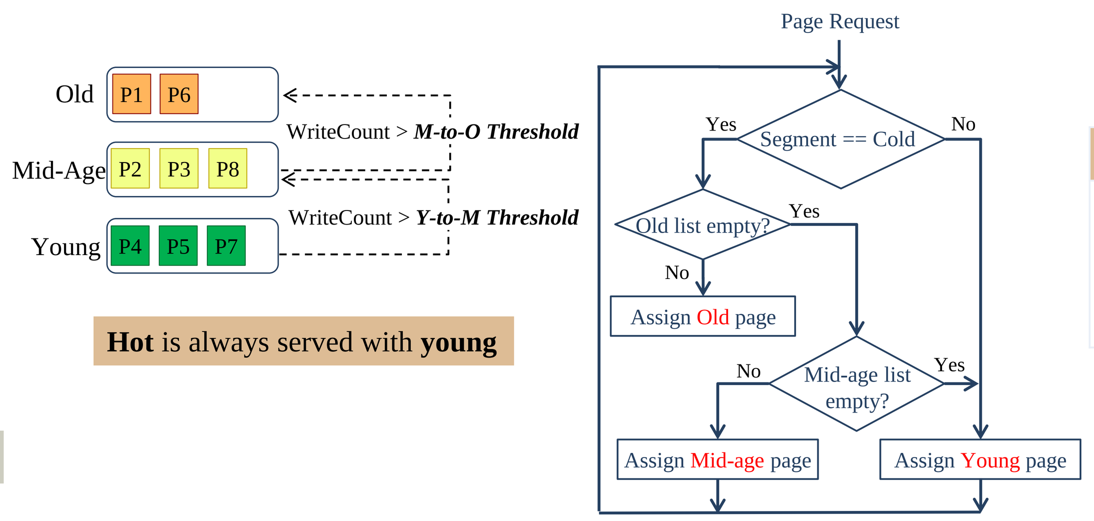


Following are added to the system:

- **Three procedures**
  - Wear-resistant page allocation
  - Age-aware page replacement
  - Remap hot page
- **Two thresholds**
  - Young-to-Midage threshold
  - Midage-to-Old threshold

IV. Wear-resistant Page Allocation

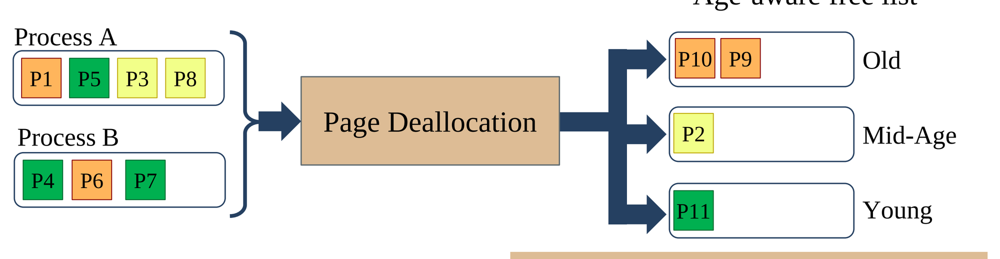
We maintain Age-aware free list to quickly find the appropriate physical page.



Hot is always served with **young**

V. Age-aware Page Deallocation

In OS there is an in-use list per process



We applied a constrained clock algorithm, the constrain is the upper bound to the write count of page.

If the young list empty
upper bound = Y-to-M threshold
Else
upper bound = wear out limit

VI. Evaluation

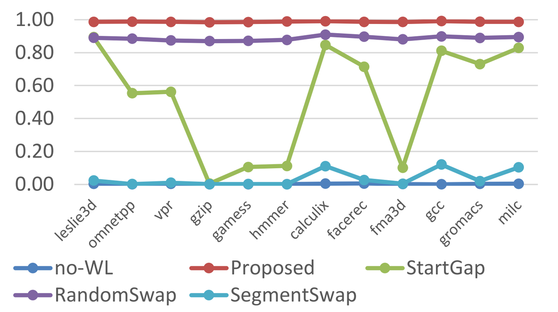
- **Benchmarks:** SPEC 2000/2006. Memory traces collected with *Pin*.
- **PCM main memory:** 128MB with 10^6 wear out limit per cell.
- **Y-to-M threshold:** 10^4 initially. **M-to-O threshold:** 2×10^5 initially.

Compare with

no-WL	Start Gap	Random Swap	Segment Swap
No wear leveling	• Maintain an extra empty line • Every 50 writes, swap the empty line with the next line	• Every 512 writes, swap the last written page with a randomly selected page	• Group pages into 1MB segment. • Swap the hottest and coldest segments every 2×10^5 writes

Normalized Lifetime:

$$\frac{\text{Total LLC Writes}}{\text{Wear out limit} \times \text{Total pages}}$$



Proposed scheme explores more than 98% of PCM writes.

Overhead:

$$\frac{(\text{Total Page Faults} + \text{Total Remaps}) \times 128}{\text{Total LLC Writes}} \qquad \frac{\text{Page Size}}{\text{LLC Block Size}} = \frac{2^{12}}{2^5}$$

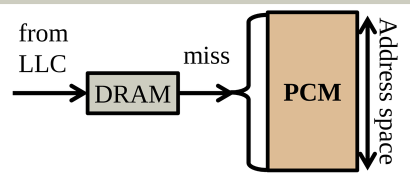
On Average

no-WL	Proposed	StartGap	Random Swap	Segment Swap
6.75%	7.25%	146%	67.25%	36.5%

3. Addressing PCM Write Latency and Energy

I. DRAM-PCM hybrid architecture

- Use a **small** DRAM (3% of PCM size) as a cache for PCM, to absorb CPU writes.
- Two step access policy, first access DRAM, if **miss**, access PCM.



II. Criticality of DRAM misses

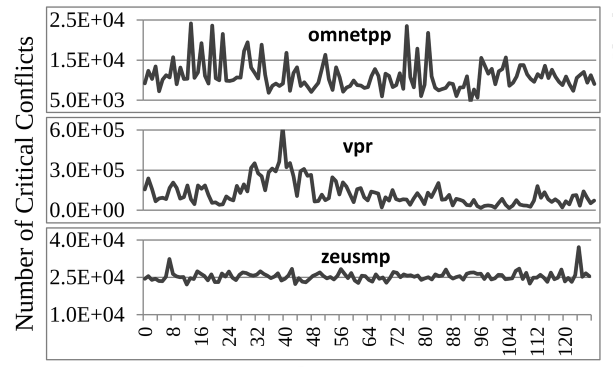
1 Memory performance measured by Average Memory Access Time (AMAT)

AMAT = DRAM Access Time + **DRAM Miss Rate** × PCM Access Time

2 Memory endurance measured by maximum number of writes in PCM

A PCM write happens upon a **DRAM miss** that replaces a dirty block

- DRAM Conflict Misses:
  - Hit in PCM
- Critical Conflicts:
  - Generates writeback to PCM



High variation across DRAM sets!!

Idea: when bring a page to DRAM, map it to a less conflicting set

III. How to reduce DRAM misses?

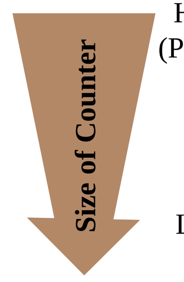
We exploit the flexibility of Logical to Physical page mapping

DRAM set of a page → Physical Page Frame Number (PFN) → Logical to Physical page mapping

determined by — determined by

IV. Count and store conflict misses

2-bit saturating counters per DRAM sets are used.



Higher Overhead (Power and Storage)

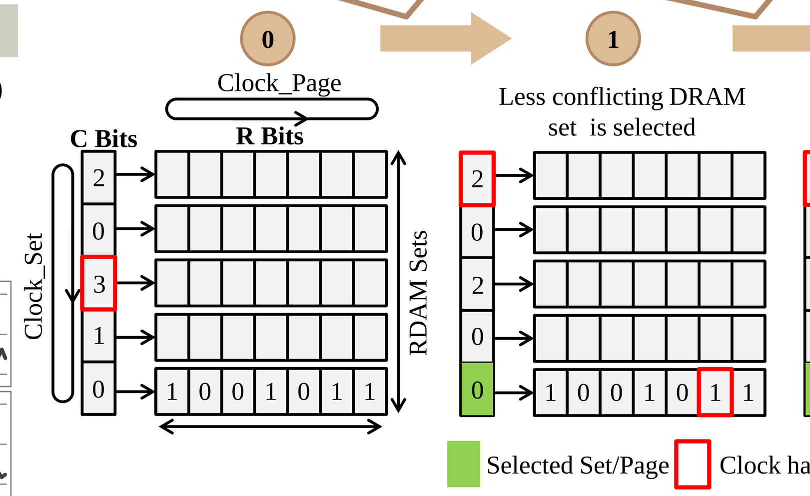
Lower Accuracy

For a 2MB, 4-way associative DRAM, storage overhead is 256 bits out of 2MB (**< 0.01%**)

V. Conflict aware page allocation

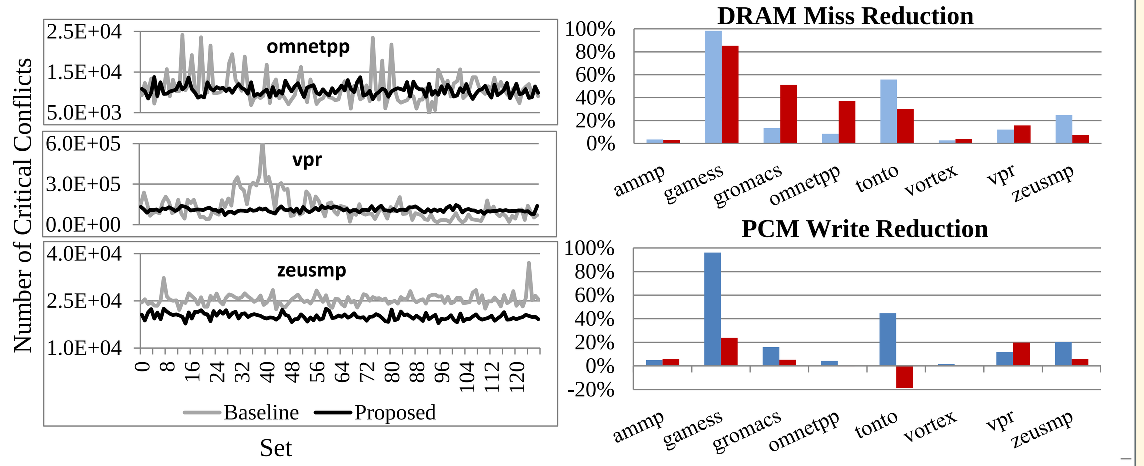
1. Find a less conflicting set: Instead of Comparing all sets which is not efficient, we use clock algorithm (**Clock_Set**).

2. Finding a page in selected set: Organized pages in groups based on their DRAM set, and only search the reference bit of pages mapped to the selected set (**Clock_Page**)



VI. Evaluation

- PCM main memory: 64 MB, page size of 4KB
- DRAM Cache: 2MB, 4-way associative
- We use two DRAM replacement algorithms: LRU (least-recently-used) and CF (clean-first)



On average, our technique reduces:

✓ **Standard deviation** of conflict misses across DRAM sets by **63%**.

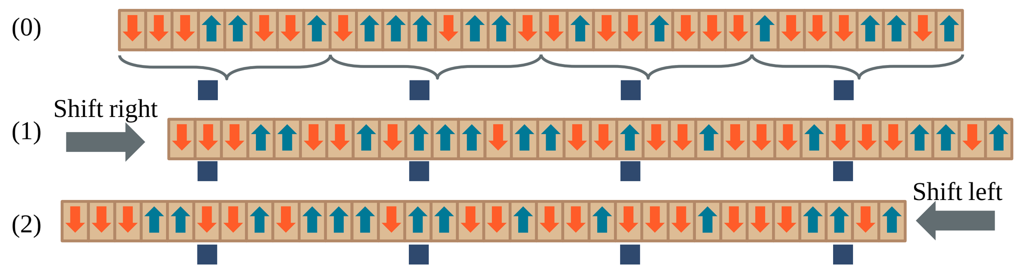
✓ **DRAM misses** by **27%** for LRU and **29%** for CF

✓ **PCM writes** by **25%** for LRU and **9%** for CF

4. Reducing DWM Access Latency

I. Domain Wall Memory Structure

- Made of hundreds of millions of ferromagnetic nanowires, called tracks.
- Each track stores many bits in the form of magnetized regions called domain.
- To read/write data several access ports are placed at fixed position.
- Shift operation required to access each bit.



Performance and Energy consumption is proportional to the number of shifts

II. Related Works DWM Main memory

Previous designed aim at reducing data accesses through storing data vertically.

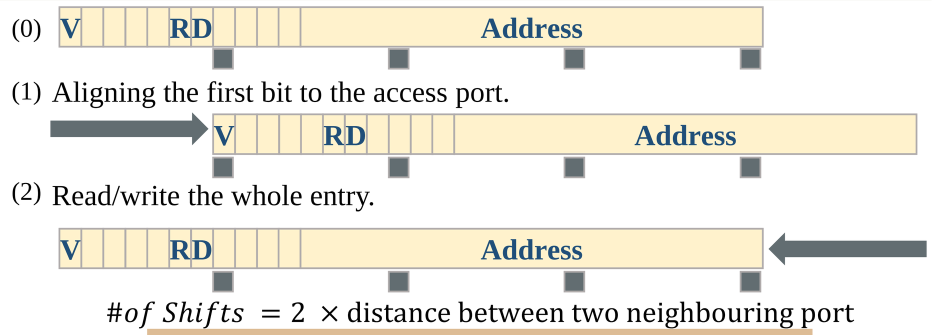
Our work differs from them by
Considering **metadata** accesses, specifically page table.
(1) Reducing shifts
(2) Leveraging access port position for metadata interpretation

III. Page Table

Page table contains the virtual-to-physical page address mapping. It adopt a hierarchical structure, which requires a 4-step process to Read and write a page table entry (PTE).

Most frequently accessed control bits are V, R, D

IV. Intuitive Read/Write a PTE

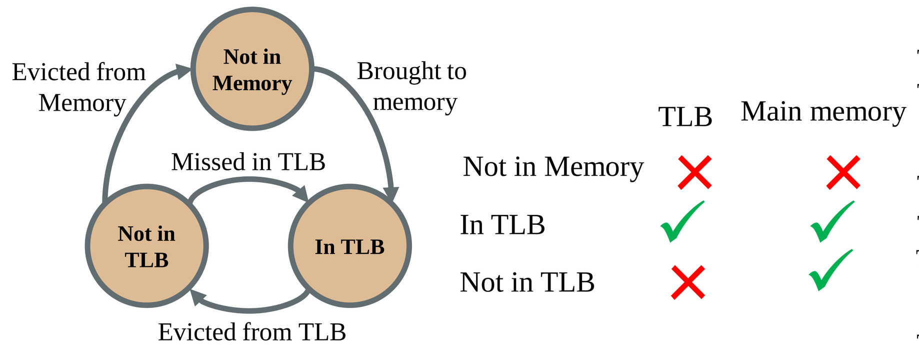


$\#of\ Shifts = 2 \times$ distance between two neighbouring port

Observation: There is no need to read the entire PTE

V. Observation on PTE States

In system with TLB, PTE have three states



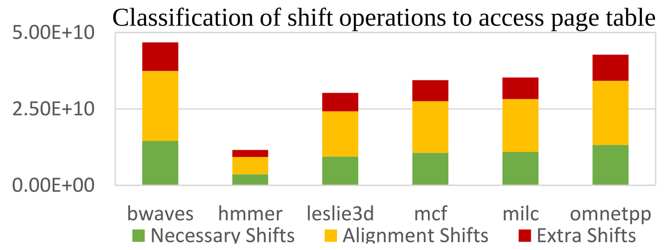
Transition	Action	Accessed Field in PTE
Not in Memory → In TLB	Write	The entire entry
In TLB → Not in TLB	Update	Referenced (R) bit and/or Dirty (D) bit
Not in TLB → In TLB	Read	The entire entry
Not in TLB → Not in Memory	Update	Validity (V) bit

VI. Type of Shifts

Necessary Shift: Shifting through the bits that are listed in table.

Extra Shift: Shifting through the bits that are **not** listed in table

Alignment Shift: Align first bit with access port.



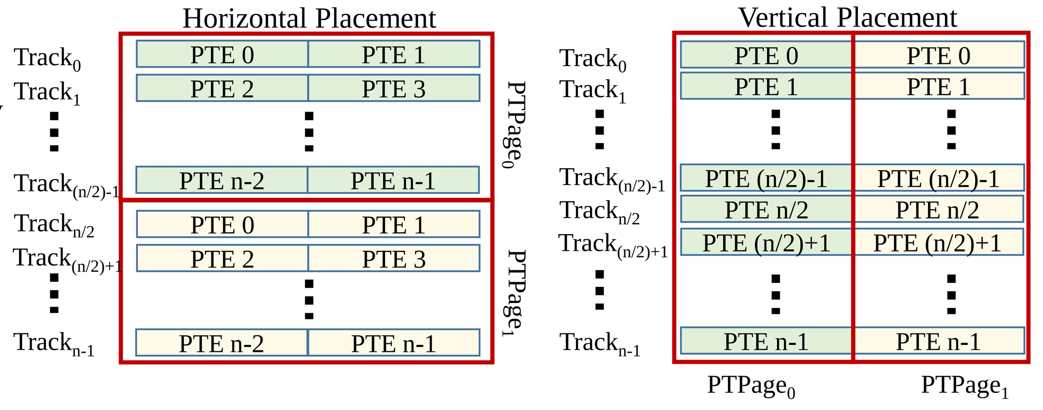
VII. Pre-alignment Technique

Pre-align based on the state:
In TLB: Pre-align to R bit
Not in TLB: Pre-align to V bit
Not in Memory: Pre-align to the bit adjacent to the V bit

Two advantages:
(1) Remove the alignment shifts from the PTE access path
(2) The state of PTE is known prior to reading PTE

VIII. PTpage Placement

PTEs are stored in granularity of pages. (**PTpage**)



PTEs in the same page have high spatial locality ⇒ highly conflicting

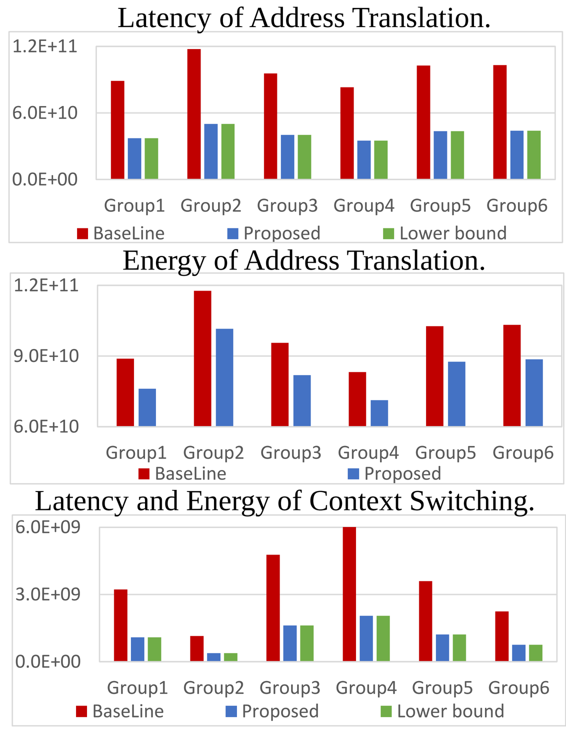
Less spatial locality between PTEs of different pages ⇒ less conflicting

IX. Evaluation

- **Benchmarks:** SPEC 2000/2006. TLB traces collected with *Pin*.
- **Instruction and Data TLB:** 4-way associative 128-entry
- **PCM main memory:** 4 GB, page size of 4KB
- **Page Table Reserved Size:** 2MB
- **Compared with:** DWM main memory, no- pre-alignment, horizontal

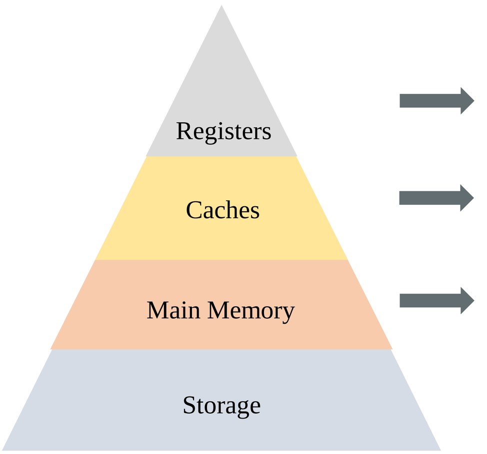
	Benchmarks
Group1	gemsFDTD, gobmk, gromacs, h264ref
Group2	astar, bwaves, bzip, cactus
Group3	calculix, deal, gamess, gcc
Group4	hmmer, zeusmp, leslie3d, omnetpp
Group5	mcf, milc, namd, libquantum
Group6	perlbench, sjeng, soplex, tonto

Lower bound = necessary shifts only



5. Under going and Future Works

I. Overview



Memory Hierarchy

- DWM-based Stack Architecture
- STT-RAM-based Cache
- PCM-based Main memory; DWM-based Main memory

II. DWM-based Stack Architecture

Energy harvesting devices become popular

Concerns:	Solutions:
(1) Intermittent power supply	Non-volatile memory register files [Ma, Micro'15]
(2) Tend to be smaller everyday	Alternative architecture such as Stack architecture

DWM is a good choice for stack.

Goal: Improving the reliability, energy, and execution time of energy harvesting devices

III. STT-RAM-based Cache

- Over 50% of chip area devoted to cache.
- STT-RAM is more scalable than SRAM.
- STT-RAM has long write latency and high write energy,
  - however they have a trade of with non-volatility property

[Jog, DAC'12]

Retention Time	10 years	1 sec	10 ms
Write Latency @2GHz	22 cycles	12 cycles	6 cycles

Goal: Explore the flexibility of cache replacement algorithm to reduce retention time and improve access latency and energy.

6. Conclusion

- Showed the necessity to change main memory technology.
- Introduced two candidates:
  - Phase Change Memory (PCM)
  - Domain Wall Memory (DWM)
- Prolonged PCM main memory lifetime
  - Proposed a segment-aware page allocation.
- Overcame write limitation of PCM
  - Proposed a conflict-aware page allocation for PCM-DRAM hybrid main memory.
- Improved the performance and energy page table accesses
  - Proposed a pre-alignment technique as well as a new placement in DWM main memory.