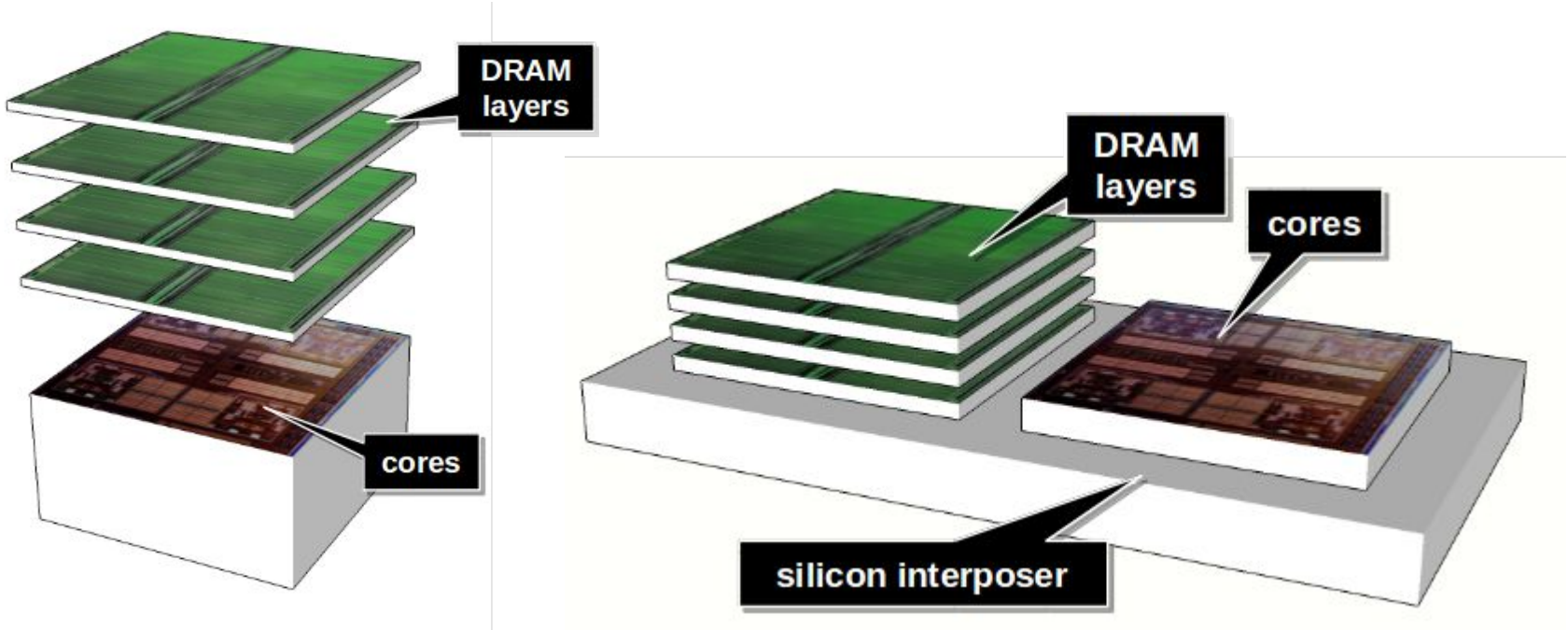
# DRAM Caches

Die-Stacked DRAM Caches

Rohith Mukku

# Die-stacked DRAM Caches

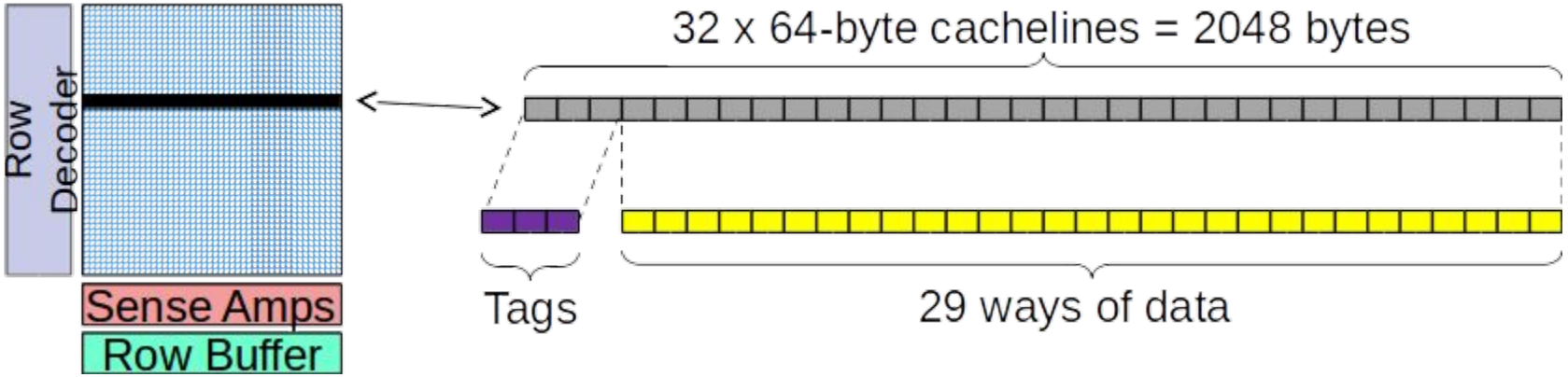


# Why Die-stacked DRAM as a cache

- As memory, difficulties?
- As a cache, the capacity can be increased many-folds
- The following slide explores the challenges faced DRAM cache

# Implementation of DRAM cache and its challenges

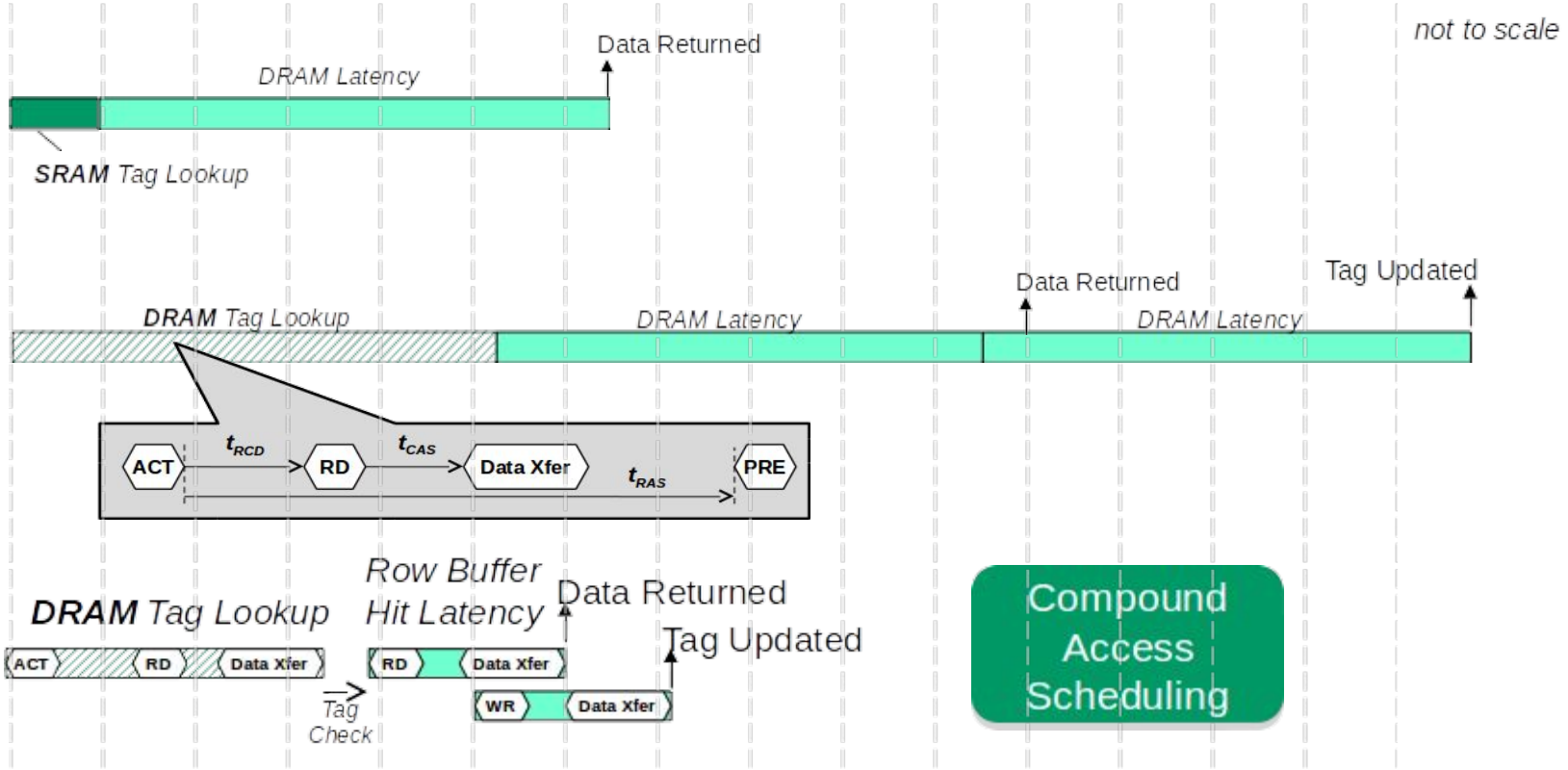
- Large Cacheline Sizes
- Sub-blocking
- Combining Tags and Data



# Main objectives in designing DRAM cache

1. Avoid fragmentation and minimize bandwidth wastage
2. Reduce SRAM overhead as low as possible
3. On cache hit, latency should be close to a DRAM access
4. On cache miss, proceed to main memory quickly without a stacked-DRAM access

# DRAM cache hits

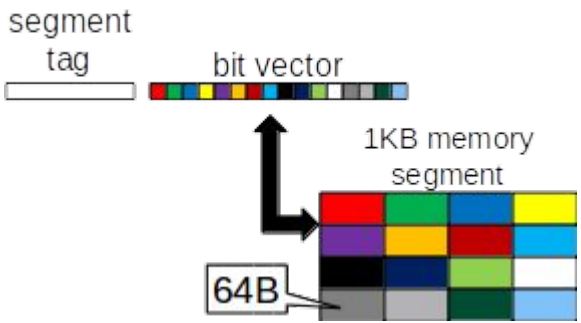


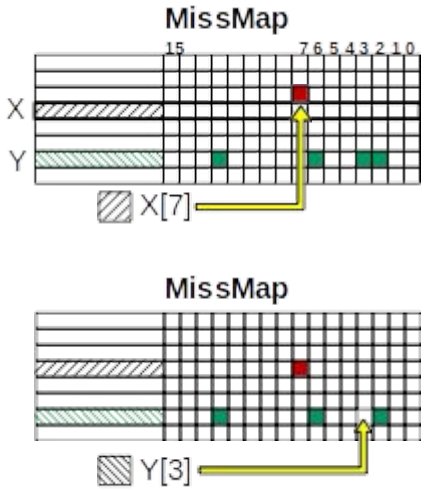
# Reducing Hit Latency

- Compound Access Scheduling
- Reserve the row buffer after tag read command
- Aim to maximize the hits
- For cache misses, MissMap technique

# MissMap

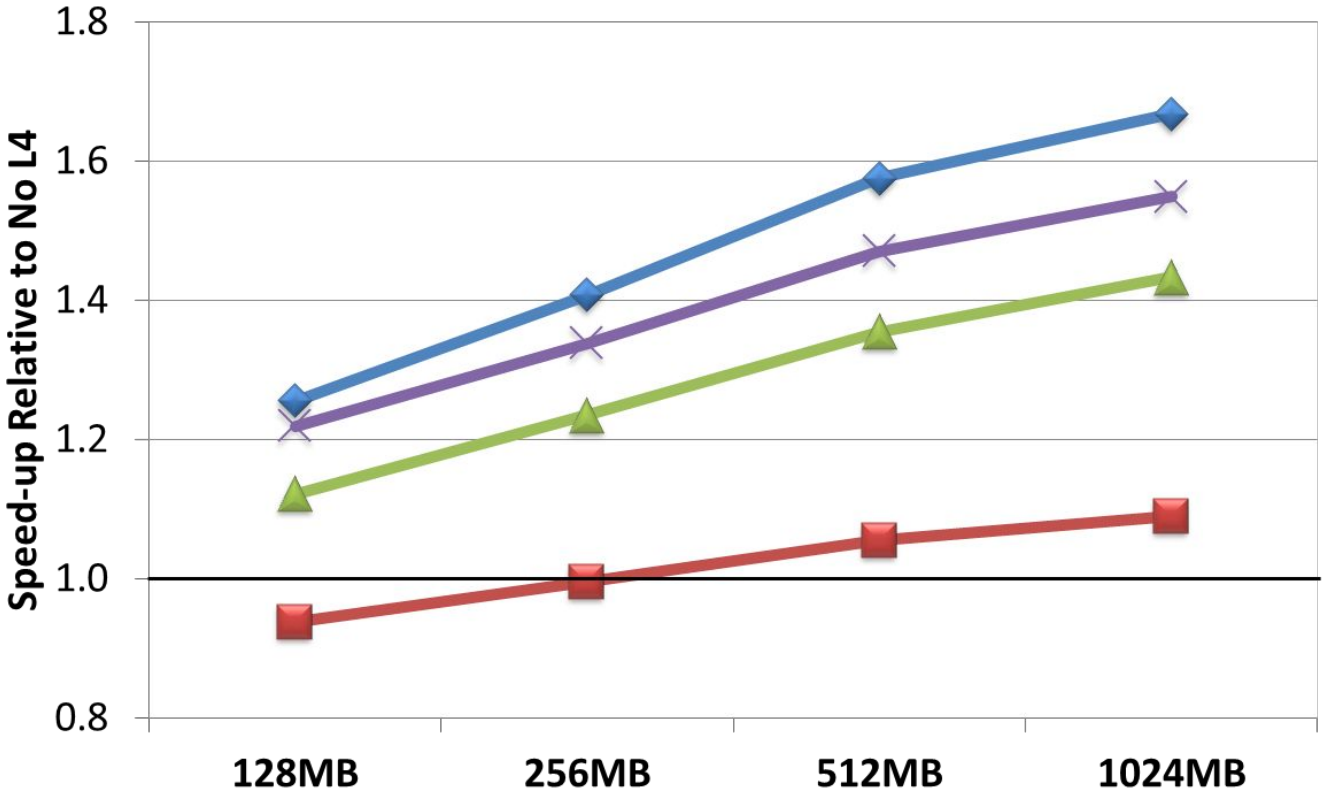
- Similar to conventional tag array
- Can hold more entries compared to tag array




- X[7] is being stored in DRAM cache
- Y[3] is being evicted from DRAM cache

# Performance of DRAM caches




Ideal SRAM Tags

CAS + MissMap

Comp. Acc. Sched.

Tags in DRAM

Average of Web-Index, SPECjbb05, TPC-C, & SPECweb05

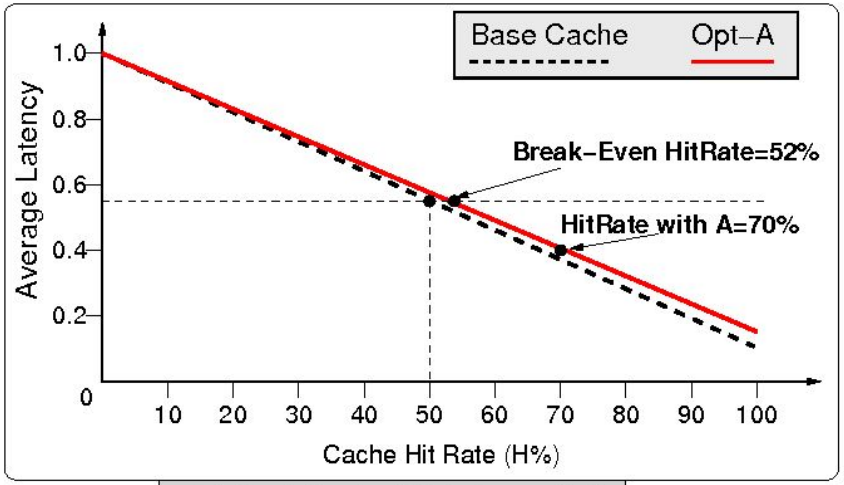
# DRAM Caches

Latency Trade-offs in Architecting DRAM Caches

# Latency vs Hit rate

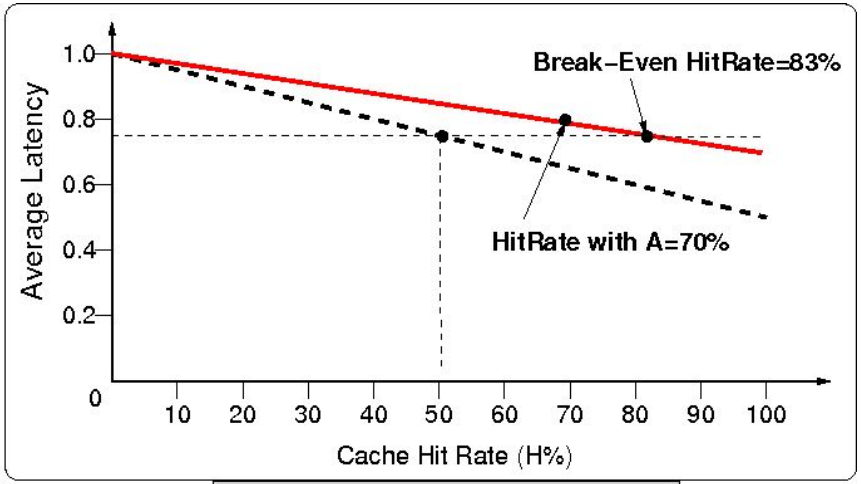
Memory access latency = 1, Hit rate = 50%, Optimized hit rate = 70%, hit latency with optimization = 0.14

Cache access latency = 0.1




(a) Fast Cache [Hit Latency 0.1]

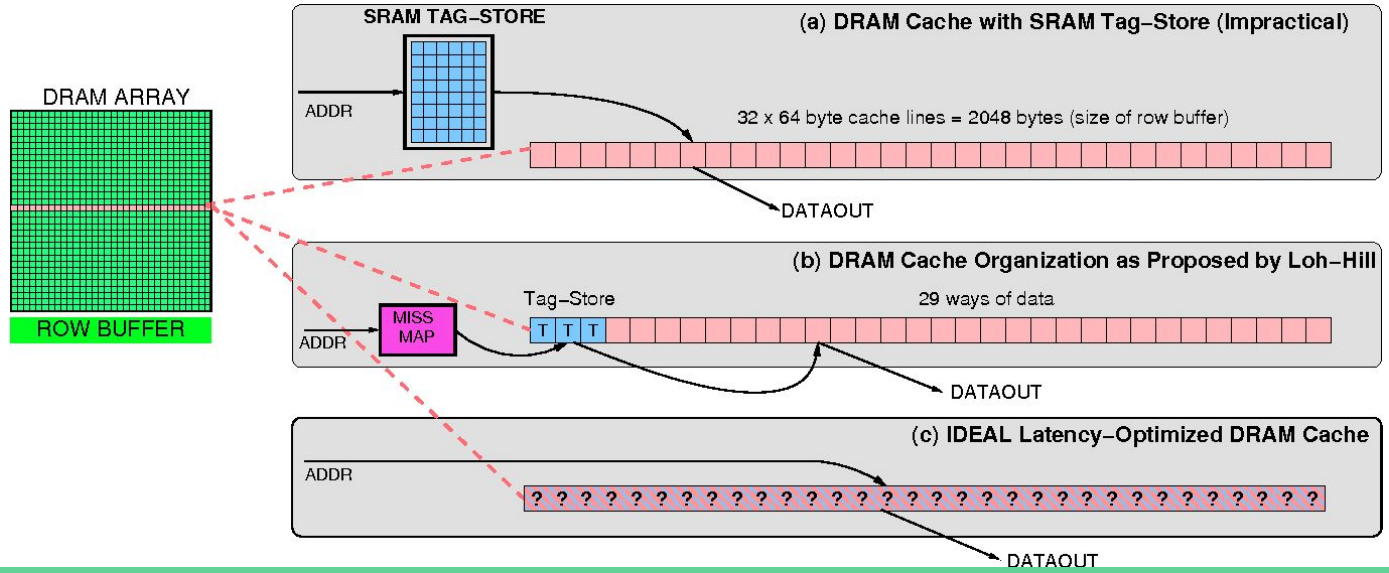
Cache access latency = 0.5



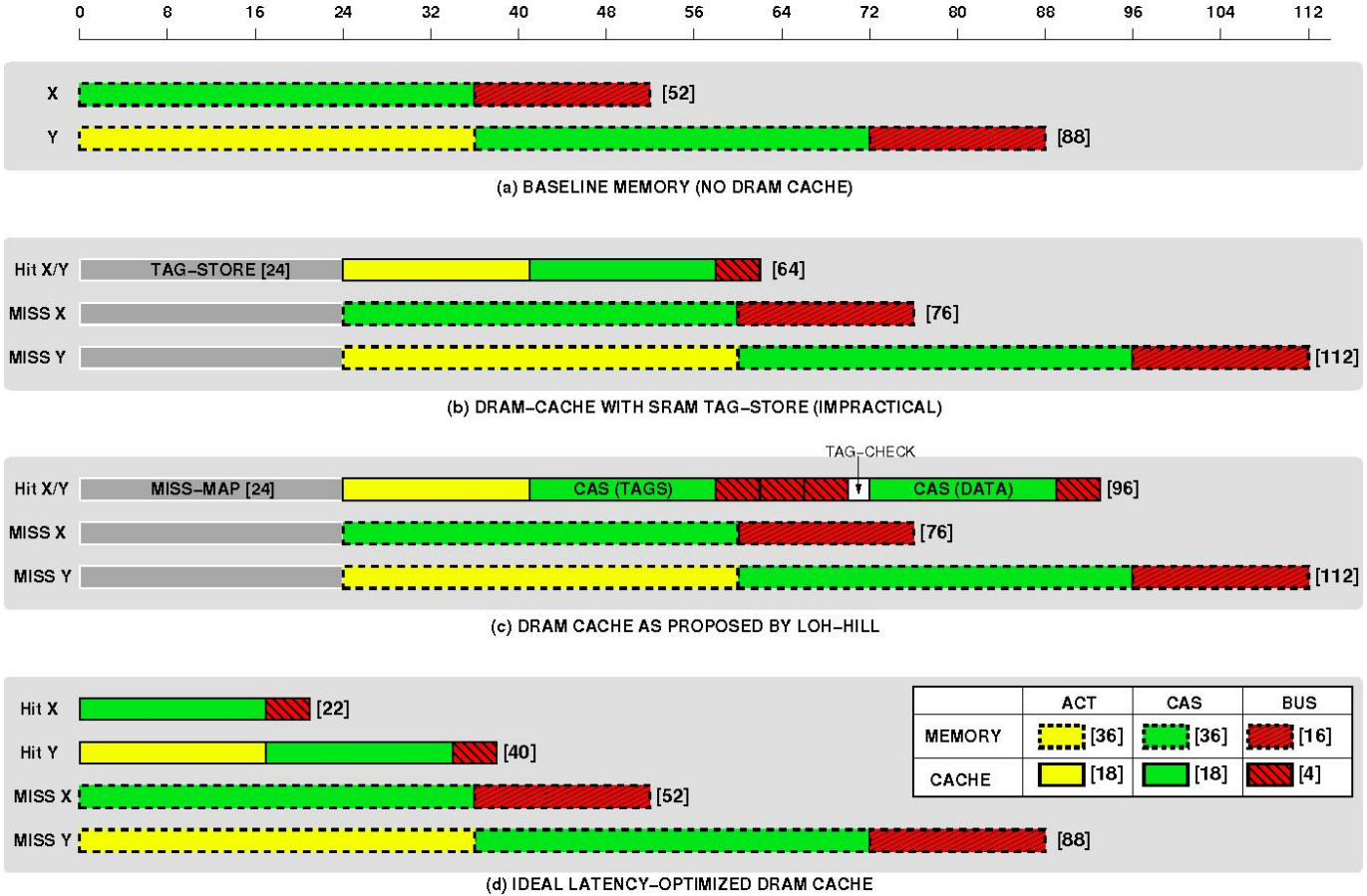

(b) Slow Cache [Hit Latency 0.5]

# Options for storing tag

- SRAM-Tag Design: Latency - Tag Serialization Latency (TSL)
- Tags-in-DRAM: The LH-Cache: Latency - TSL + Predictor Serialization (PSL)
- IDEAL Latency-Optimized Design



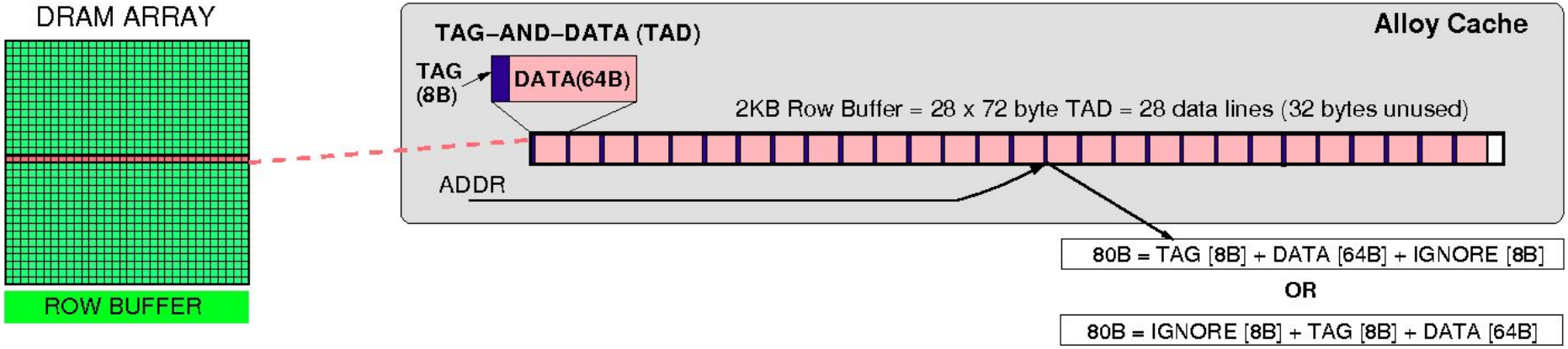
# Latencies in various cases of accesses X & Y



# Latency-Optimized Cache Architecture

## Alloy Cache

- Direct-mapped structure with each block having both tag and data (TAD)



# Low-Latency Memory Access Predictor

- MissMap incurs PSL

Series Access Model (SAM) & Parallel Access Model (PAM)

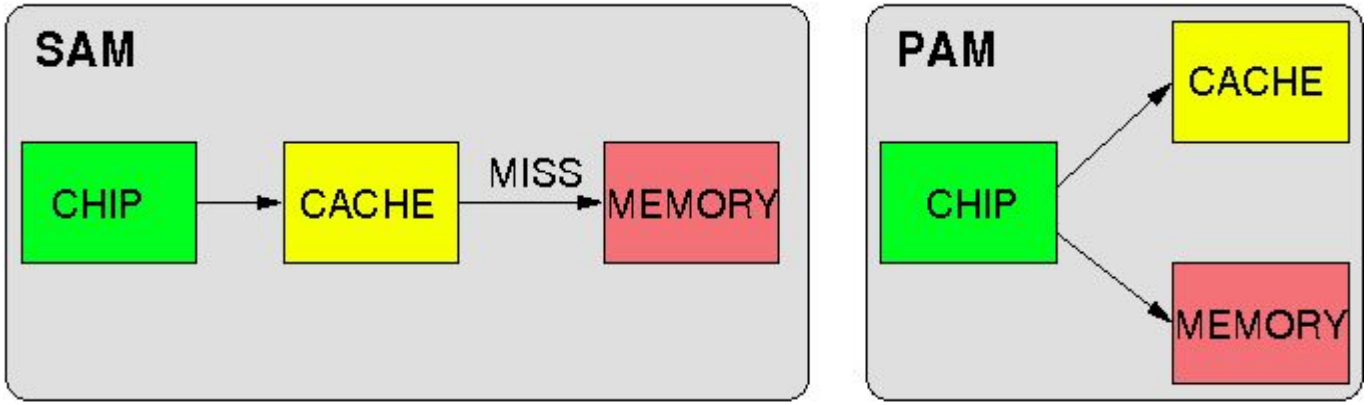



Figure 7: Cache Access Models: Serial vs Parallel

# Memory Access Predictor

- PAM - helps in saving latency (useful when cache hit rate is low)
- SAM - helps in bandwidth savings (useful when cache hit rate is high)
- Dynamic Access Model (DAM)
- History-Based Memory-Access Predictor (MAP)

# MAP

- Global-History Based MAP (MAP-G)
- Instruction-Based MAP (MAP-I)

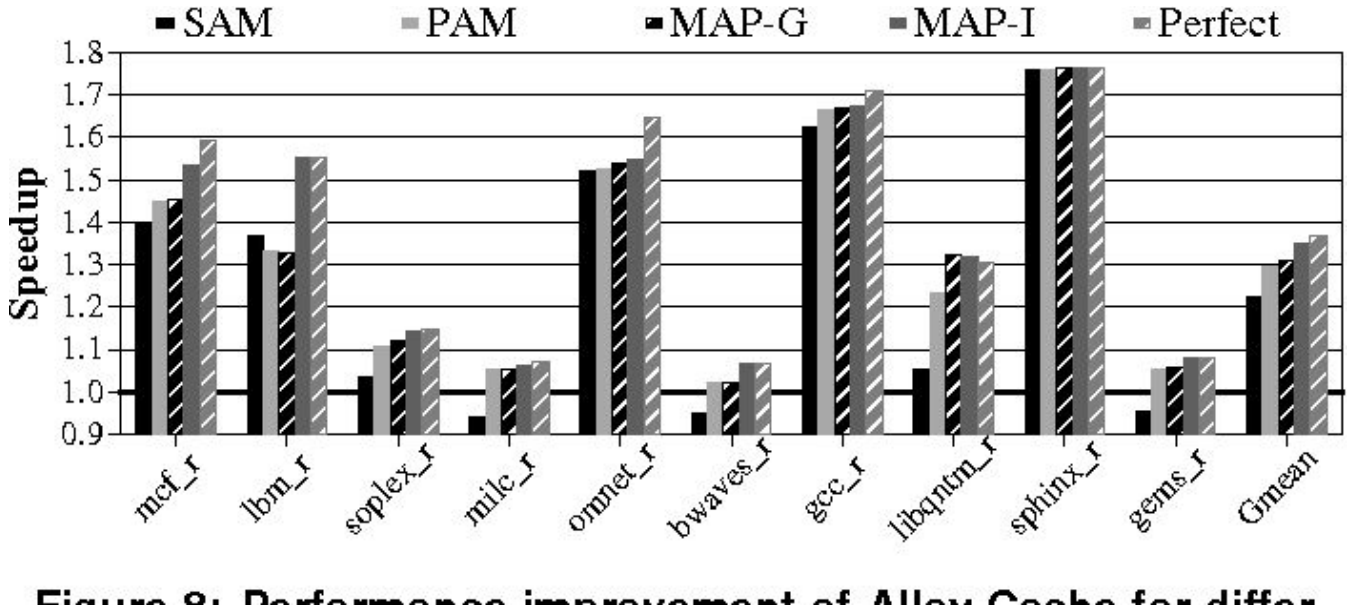



Figure 8: Performance improvement of Alloy Cache for differ

# Comparisons

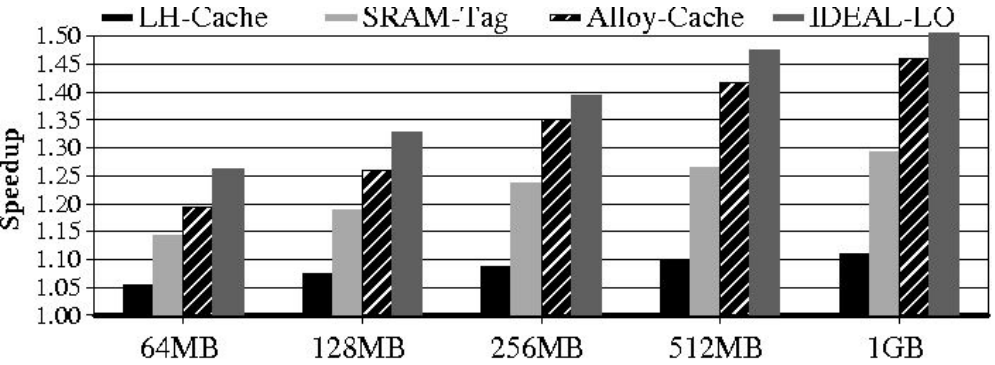



Figure 9: Performance impact across various cache size

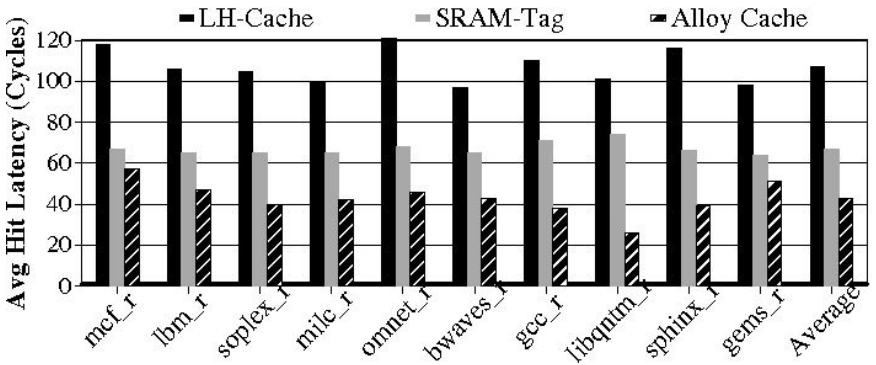



Figure 10: Average Hit-Latency: LH-Cache 107 cycles, SRAM-

# References


- Efficiently Enabling Conventional Block Sizes for Very Large Die-stacked DRAM Caches MICRO Porte Alegre, Brazil Gabriel H. Loh [1] and Mark D.

  (http://slideplayer.com/slide/4259051/)

- Fundamental Latency Trade-offs in Architecting DRAM Caches (https://www.semanticscholar.org/paper/Fundamental-Latency-Trade-offs-in-Architecting-DRA-Qureshi-Loh/e4aed18e1b965f90a86b57a787c52af44a8c20a4)

# THANK YOU