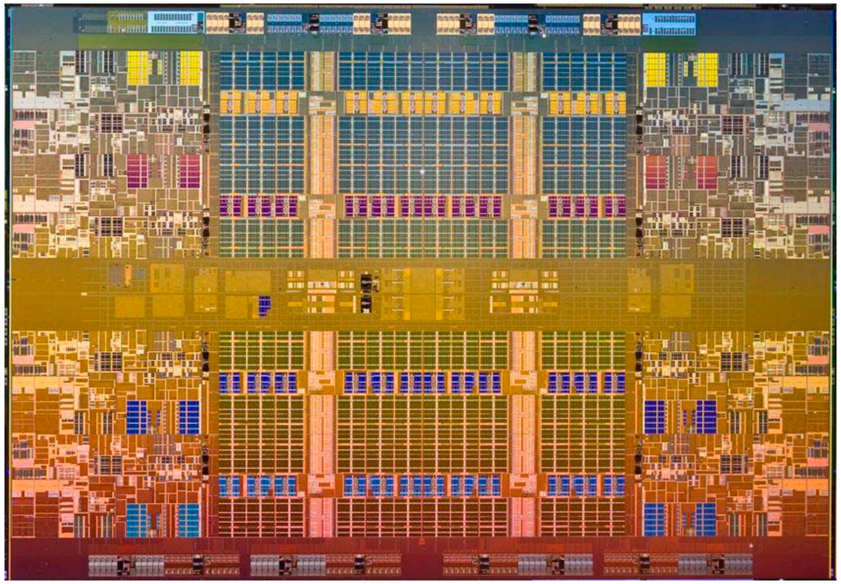
A Comprehensive Review of the Challenges and Opportunities Confronting Cache Memory System Performance


Photograph of Intel Xeon processor 7500 series die showing cache memories [1]

R. Kramer, M. Elmlinger, A. Ramamurthy, S. Timmireddy

Oregon State UNIVERSITY OSU

300% !

That's one estimate of how much a processor's core bandwidth requirements *exceed* the ability of the cache to supply data

[2,3]

Challenges Confronting Cache Memory System Performance

Some additional astonishing facts…

x30	That's the estimated required increase in cache memory bandwidth that is required for every x10 increase in processor transistor count.
50%	That's the estimated impact that cache memory has on the overall computer architecture's power requirements.
#1	… in cost. Cache memory is said to be the most expensive memory in the overall computer system.

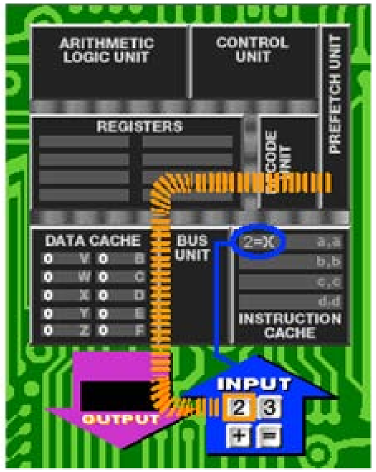
And many of those estimates were predicted over 30 years ago!

[2,3,4,5]

Today's Objectives and Contributions

To take you to the forefront of cache memory research opportunities to improve performance

- Advances in Cache Data Management: Prefetching, Bandwidth Management, Scheduling, and Data Placement (Abhishek Ramamurthy)
- Energy Efficiency Opportunities (Mathias Elmlinger)
- Advanced Topics in Cache Memory Research (Pranav Timmireddy)

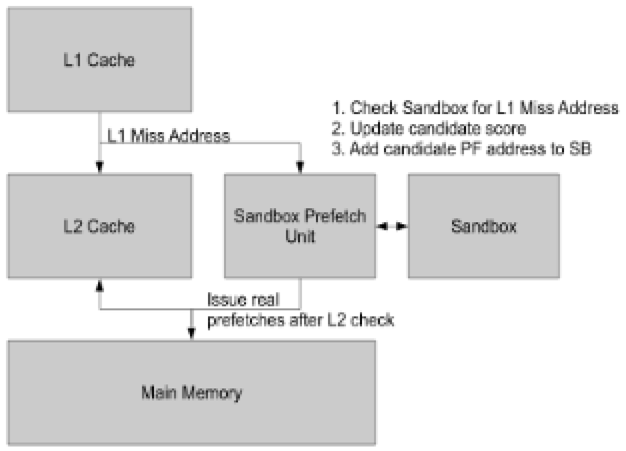


Advances in Cache Data Management: Prefetching, Bandwidth Management, and Data Placement

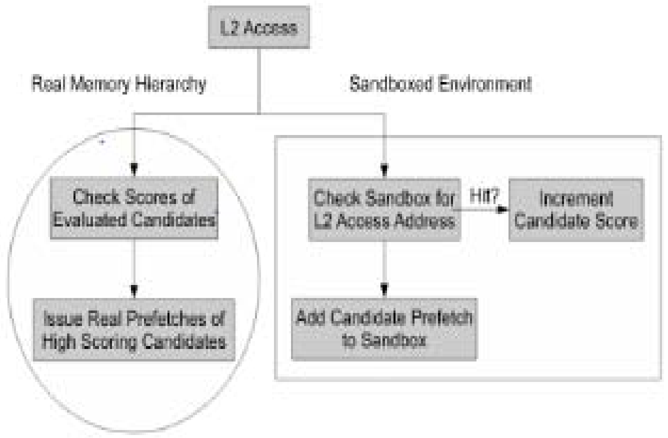
- Why is it necessary to have cache prefetching?
- What is the bottle neck involved in prefetching data from cache memory?
- How to improve the cache memory density?

Sandbox Prefetching Mechanism

- Technique is based on bloom filter (Howard Bloom, 1970).


L1 Cache
L1 Miss Address
1. Check Sandbox for L1 Miss Address
2. Update candidate score
3. Add candidate PF address to SB
L2 Cache
Sandbox Prefetch Unit
Sandbox
Issue real prefetches after L2 check
Main Memory

Sandbox Prefetch Architecture [14]

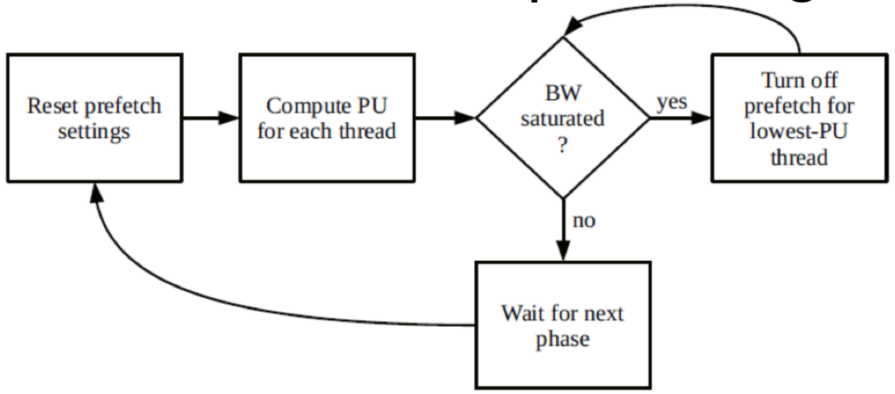

L2 Access
Real Memory Hierarchy
Sandboxed Environment
Check Scores of Evaluated Candidates
Issue Real Prefetches of High Scoring Candidates
Check Sandbox for L2 Access Address
Hit?
Increment Candidate Score
Add Candidate Prefetch to Sandbox

Sandbox Prefetch Action on L2 Access [14]

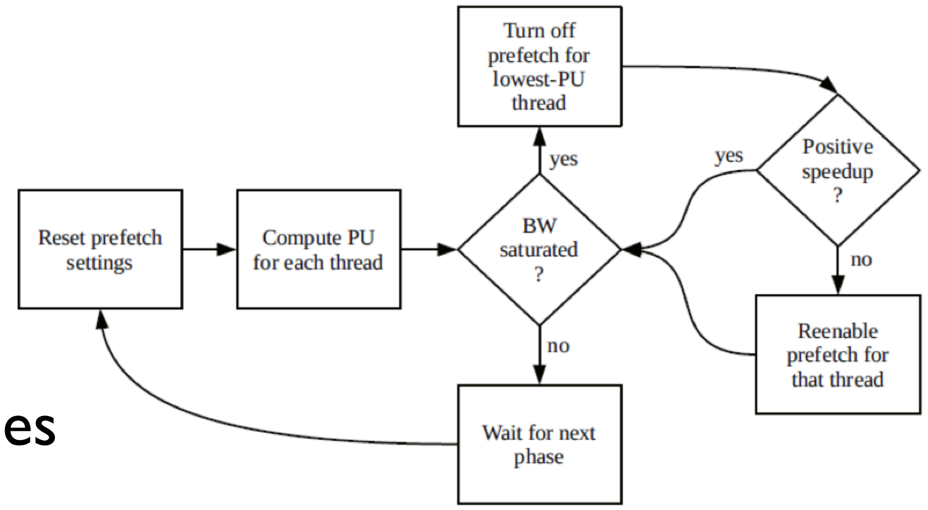
- SandBox Prefetching (SBP) improves Address Mapped Pattern Matching performance by 3.9% in a multicore environment.

Increasing Multicore Efficiency through Intelligent Bandwidth Shifting

- Technique provides better efficiency through assigning bandwidth for prefetching based on prefetch efficiency.


Reset prefetch settings
Compute PU for each thread
BW saturated ?
yes
Turn off prefetch for lowest-PU thread
no
Wait for next phase

Base Bandwidth shifting algorithm [16]

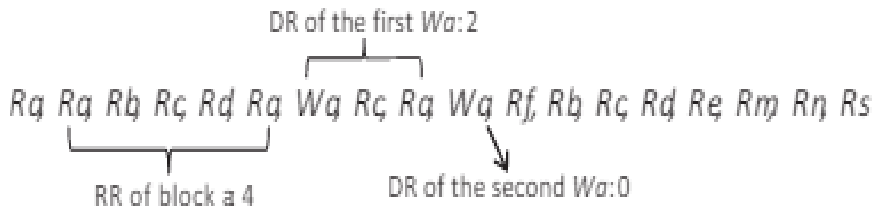

Turn off prefetch for lowest-PU thread
Positive speedup ?
yes
yes
Reset prefetch settings
Compute PU for each thread
BW saturated ?
no
no
Reenable prefetch for that thread
Wait for next phase

- Improved multicore efficiencies by 7% for random workloads.

Modified Base Bandwidth shifting algorithm [16]

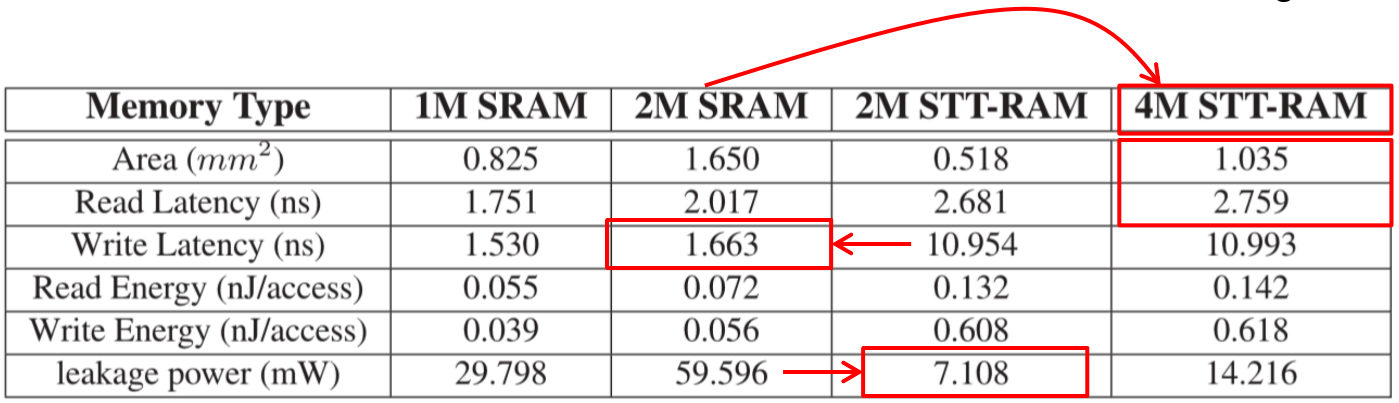
Adaptive Placement Policies for Data in Cache Memory Systems

- The technique provides a better way placing most frequently used data in cache and evicting the least used data block in cache.



Read range and Depth Range[17]

2x more storage AND ~60% less area



Memory Type	1M SRAM	2M SRAM	2M STT-RAM	4M STT-RAM
Area (mm^2)	0.825	1.650	0.518	1.035
Read Latency (ns)	1.751	2.017	2.681	2.759
Write Latency (ns)	1.530	1.663	10.954	10.993
Read Energy (nJ/access)	0.055	0.072	0.132	0.142
Write Energy (nJ/access)	0.039	0.056	0.608	0.618
leakage power (mW)	29.798	59.596	7.108	14.216

Area and read / write latency of SRAM and STT-RAM [17]

Energy Efficiency

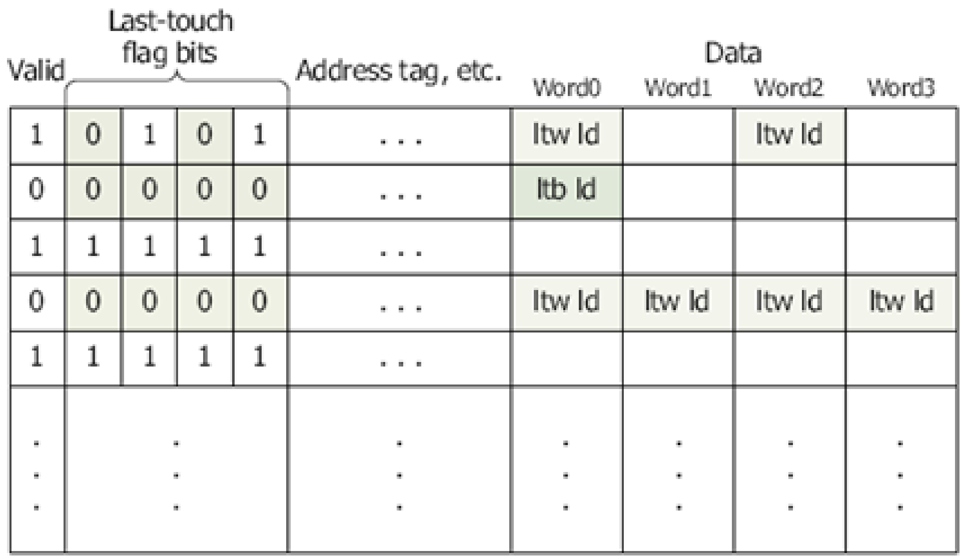
- Crucial factor: energy efficiency

- Why cache?
  - Large fraction of chip size
  - Estimated: 50% of energy dissipation by cache

- Approaches to improve energy efficiency
  - Software Self Invalidation (L1) and Data Compression (L2)
  - Exploiting row access locality (DRAM)
  - Improve Error Correcting Codes and Error Detection Codes (L1)
  - Isolation nodes and dynamic memory partitioning techniques (L1/L2)

Energy Efficiency
Software Self-Invalidation and Data Compression

- Invalidation
  - Through request
  - Last-touch load/store instructions

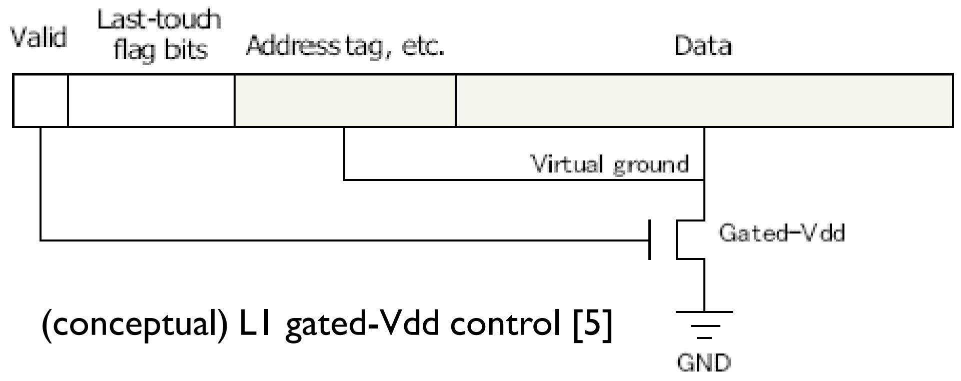


Valid	Last-touch flag bits				Address tag, etc.	Data			
						Word0	Word1	Word2	Word3
1	0	1	0	1	. . .	ltw ld		ltw ld	
0	0	0	0	0	. . .	ltb ld			
1	1	1	1	1	. . .				
0	0	0	0	0	. . .	ltw ld	ltw ld	ltw ld	ltw ld
1	1	1	1	1	. . .				
⋮	⋮				⋮	⋮	⋮	⋮	⋮

L1 cache memory structure [5]

Energy Efficiency
Software Self-Invalidation and Data Compression

- Invalidation
  - Through request
  - Last-touch load/store instructions

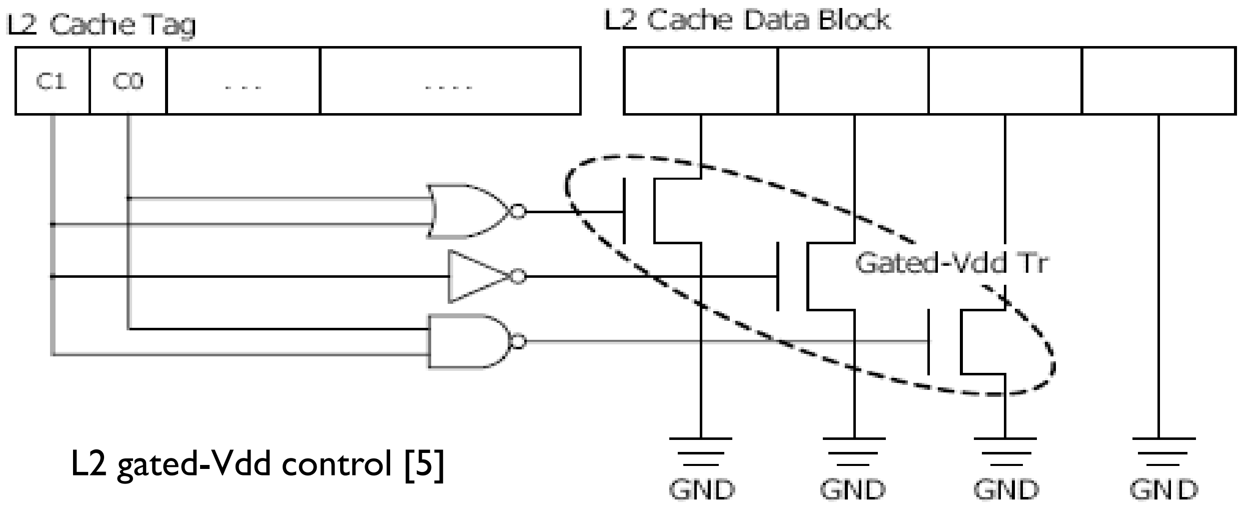


(conceptual) L1 gated-Vdd control [5]

- Reduction of up to 10% in terms of leakage energy

Energy Efficiency
Software Self-Invalidation and Data Compression

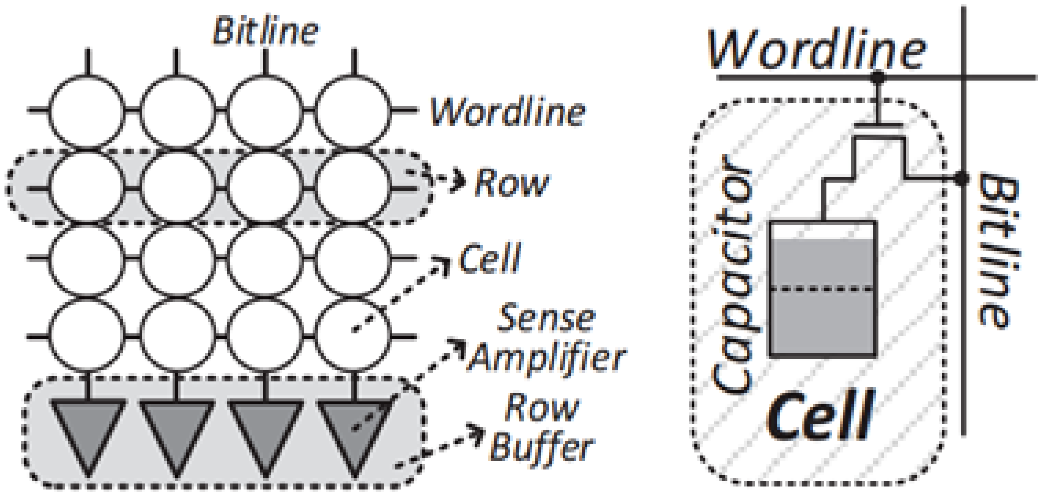
- Data compression
  - Less memory space used
  - More memory space can be turned off



L2 gated-Vdd control [5]

- Reduction of up to 25% in terms of leakage energy

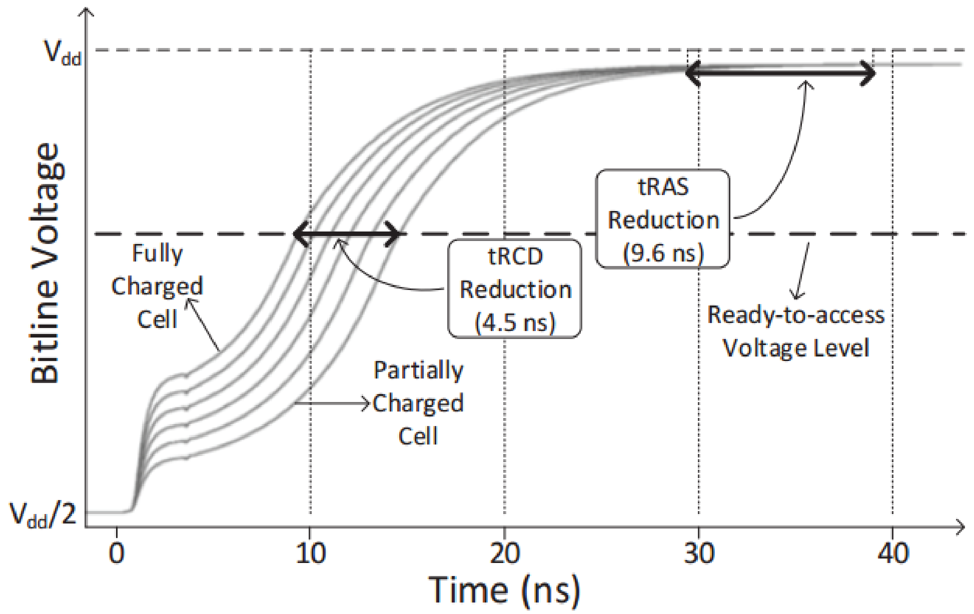
Energy Efficiency
Exploiting Row Access Locality



DRAM Sub-Array (left) and DRAM cell (right) [6]

- Timing to access rows based on amount of charge
- Keep track of charge of recently accessed rows
  - Table in main memory controller
  - Hit: lower timing parameters

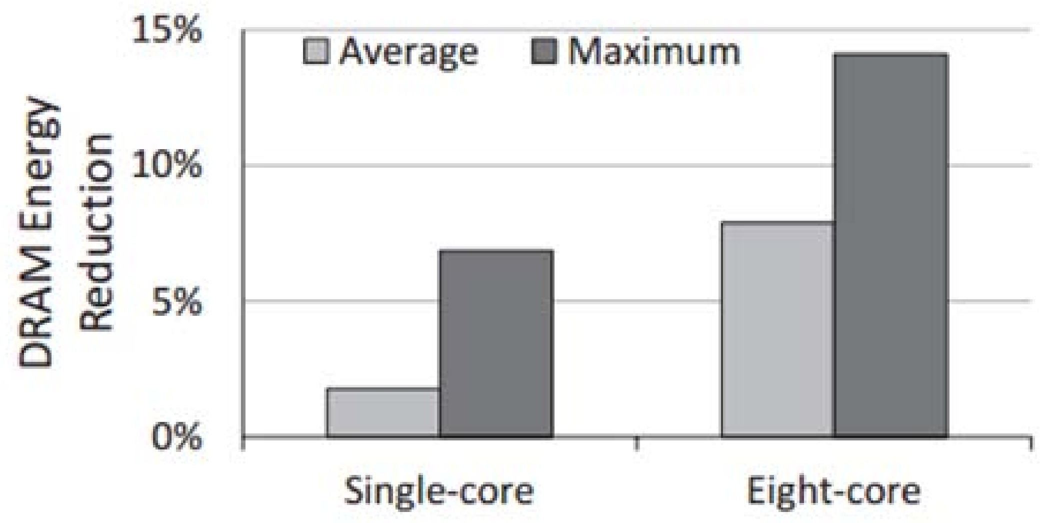
Energy Efficiency
Exploiting Row Access Locality



Effect of initial cell charge on bit line voltage [6]

- Timing to access rows based on amount of charge
- Keep track of charge of recently accessed rows
  - Table in main memory controller
  - Hit: lower timing parameters

Energy Efficiency
Exploiting Row Access Locality



DRAM energy reduction of ChargeCache [6]

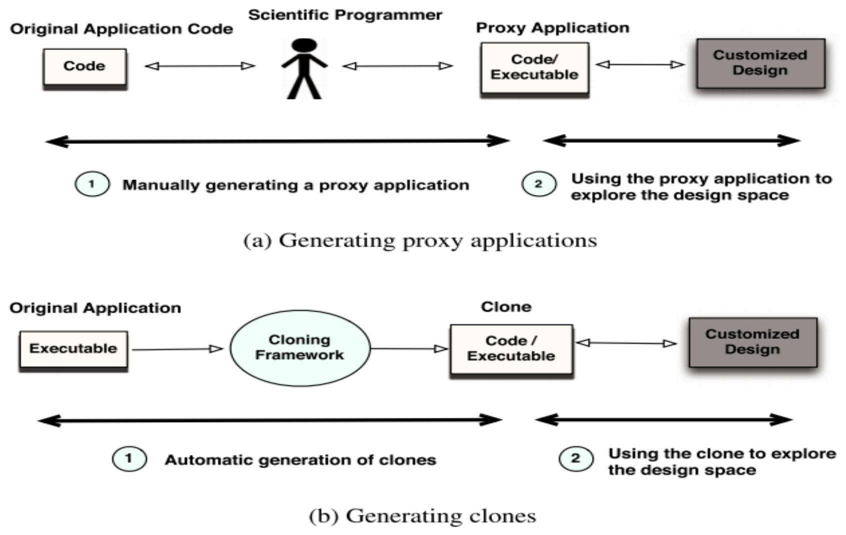
- Single-core: 1.8% average (max. 6.9%)
- Eight-core: 7.9% average (max. 14.1%)

Advanced Topics in Cache Memory Research

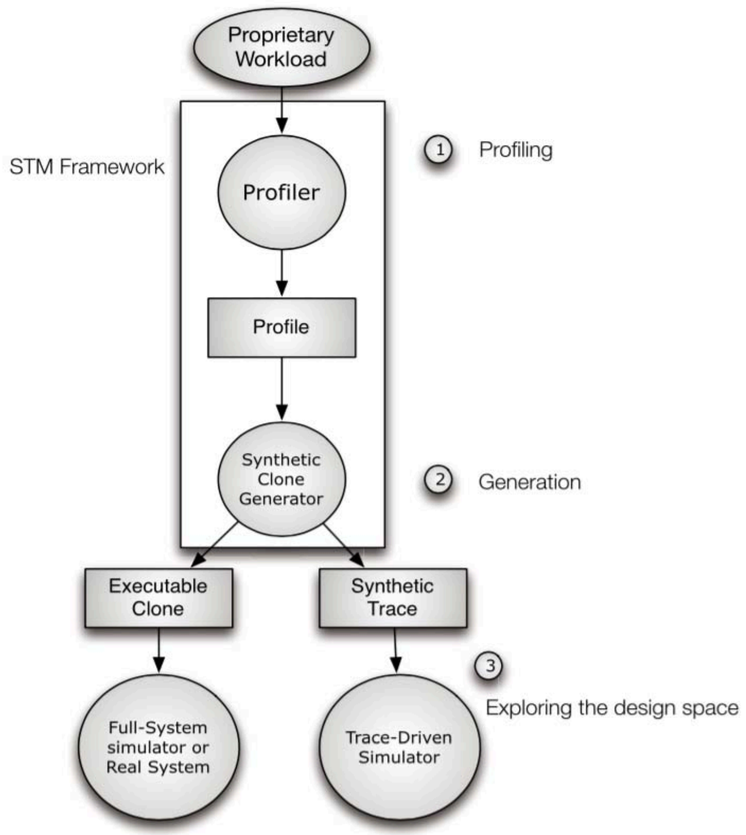
- STM : Cloning the Spatial and Temporal Memory Access Behavior
- RADAR (Runtime- Assisted Dead Region) Management for Last Level Caches

STM: Spatial and Temporal Cloning

- Transition probability table indexed by stride history pattern is used to capture the spatial locality
- A combination of stack distance profile and stride pattern table

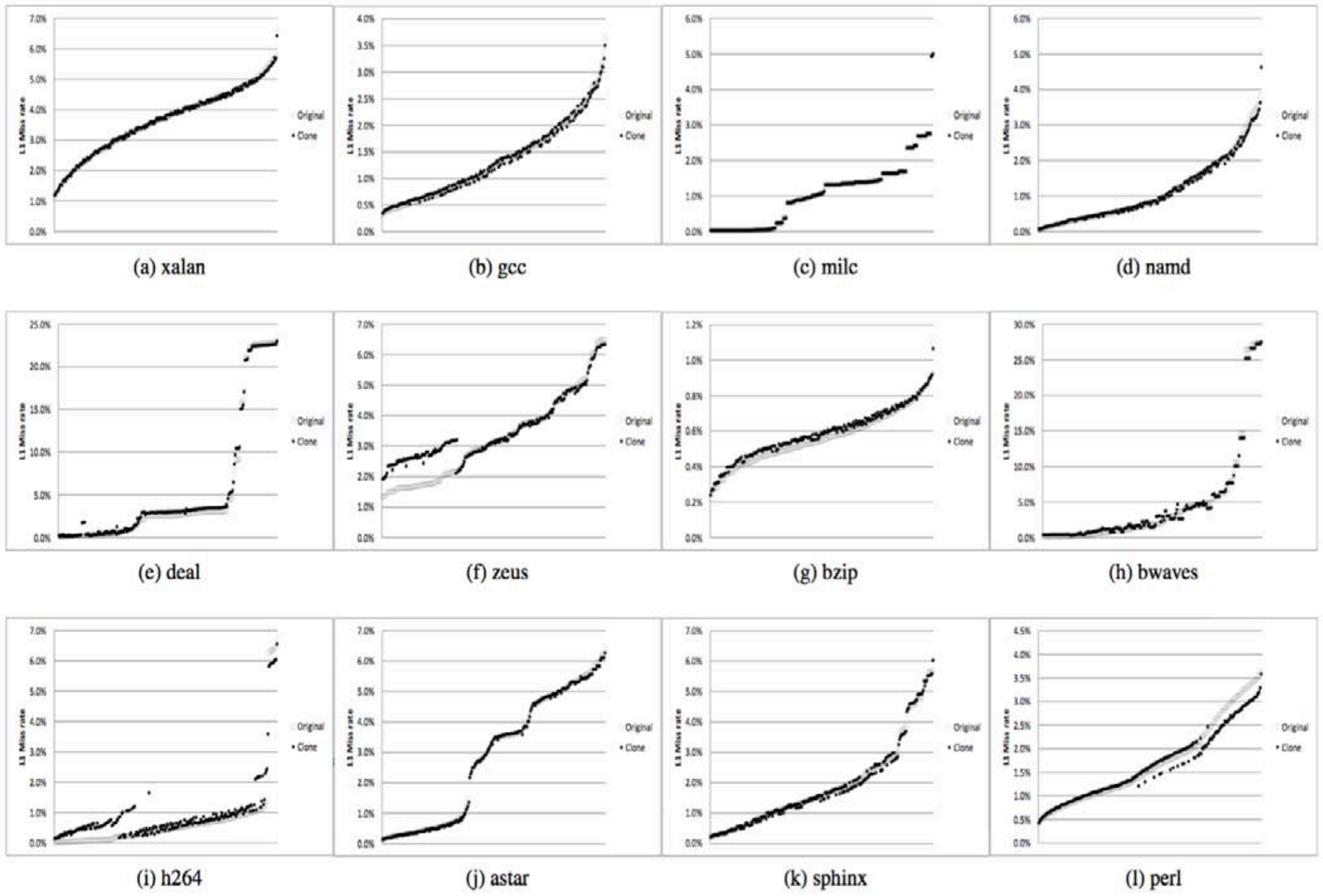

Original Application Code
Scientific Programmer
Proxy Application
Code
Code/ Executable
Customized Design
1 Manually generating a proxy application
2 Using the proxy application to explore the design space
(a) Generating proxy applications
Original Application
Clone
Executable
Cloning Framework
Code / Executable
Customized Design
1 Automatic generation of clones
2 Using the clone to explore the design space
(b) Generating clones

Proxy application versus cloning [19]



STM Framework [19]

STM – Cont'd

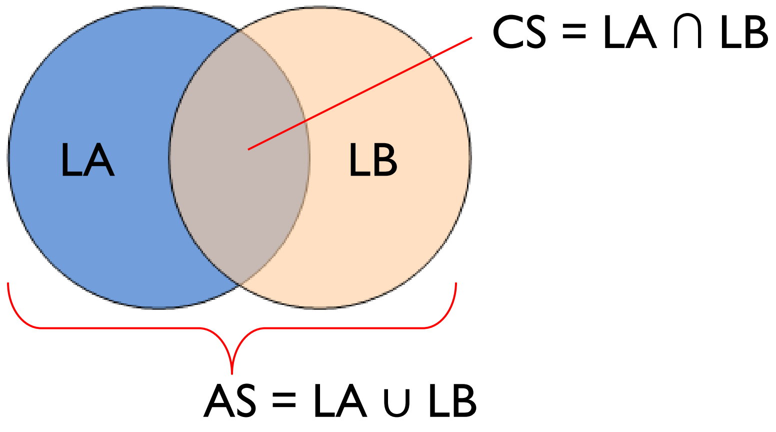


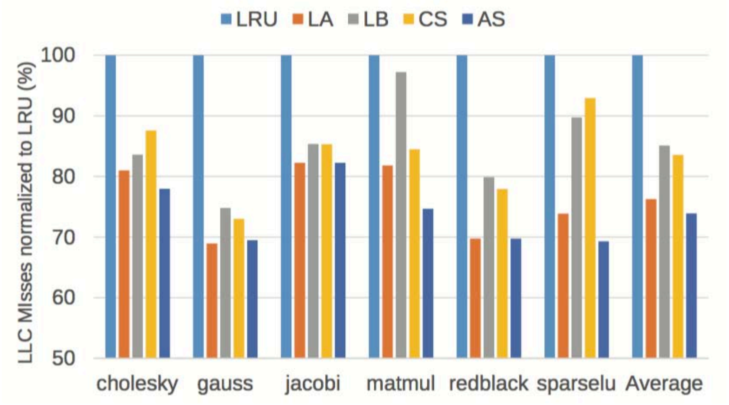
STM on different benchmarks [19]

Original vs clone L1 miss rate across different cache prefetchers and configurations

RADAR: Runtime- Assisted Dead Region Management

- Efficient management of LLCs is essential
- Existing protocols use either dynamic or static techniques.
- RADAR is a hybrid static/dynamic technique which improves LLC efficiency.
- Look Ahead (LA), Look Back (LB), Conservative combined Scheme (CS = LA ∩ LB), Aggressive combined Scheme (AS = LA ∪ LB).





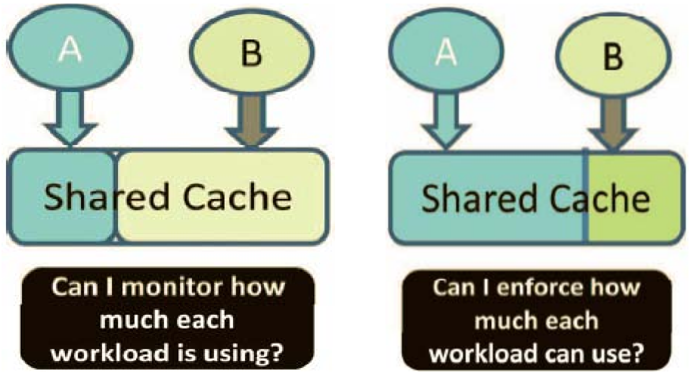
LLC miss rate for different RADAR [21]

- Aggressive Combined scheme performs best and more than 26% reduction in LLC misses over the baseline LRU.

Taking Research into Reality

Such opportunities do translate into results.

An example: two cache bandwidth Quality of Service concepts called CMT (Cache Monitoring Technology) and CAT (Cache Allocation Technology) took over 10 years to go from research to silicon.


A
B
Shared Cache
Can I monitor how much each workload is using?
A
B
Shared Cache
Can I enforce how much each workload can use?

Overview of CMT (Left) and CAT (Right) [24]

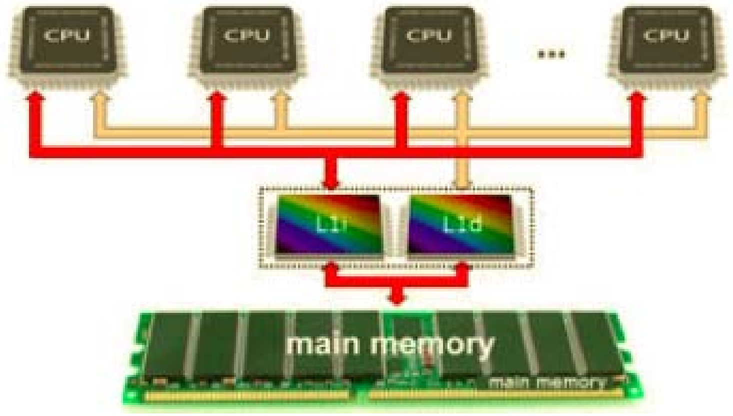
On June 4, 2013, Intel introduced the Xeon "Haswell" 4th generation processor employing both CMT and CAT technologies [29].

…. providing as high as a <u>450%</u> improvement [24].

Conclusion and Future Work

While cache memory system advances continue to be made… these advances are consistently offset by the ever increasing requirements for multicore processors.

One estimate is that by 2020, multicore processors will reach *zetta-flop* (10^{21}) speeds [25].



Envisioned optical RAM cache architecture [25]

With such demands, the need for additional breakthroughs in the area of cache memory architectures remains critical.

References

[1] Intel Corporation, "Photograph of Intel Xeon processor 7500 series die showing cache memories," 31 March 2010. [Online]. Available: https://phys.org/news/2010-03-intel-xeon-processor-series.html. [Accessed 20 February 2017].

[2] IBM Products Division, "System/370 model 168 theory of operation/diagrams manual (volume 1)," Poughkeepsie, NY, 1976.

[3] J. R. Goodman, "USING CACHE MEMORY TO REDUCE PROCESSOR-MEMORY TRAFFIC".

[4] H. Bajwa and X. Chen, "Low-Power High-Performance and Dynamically Configured Multi-Port Cache Memory Architecture".

[5] K. Tanaka, "Cache Memory Architecture for Leakage Energy Reduction," in *International Workshop on Innovative Architecture for Future generation Processors and Systems 2007*, 2007.

[6] H. Hassan, G. Pekhimenko, N. Vijaykumar, V. Seshadri, D. Lee, O. Ergin and O. Mutlu, "ChargeCache: Reducing DRAM Latency by Exploiting Row Access Locality," *IEEE,* 2016.

[7] C. K. Tang, "Cache system design in the tightly coupled multIproeessor system," *AFIPS Proceedings,* vol. 45, pp. 749-753, 1976.

[8] L. M. Censier and P. Feautrier, "A new solution to coherence problems in multicache systems," *IEEE Transactions on Computers,* Vols. C-27. No. 10, pp. 1112-1118, 1978.

[9] G. S. Rao, "Performance Analysis of Cache Memories," *Journal of the ACM,* vol. 25, pp. 378-395, 1978.

[10] R. L. Norton and J. L. Abraham, "Using write back cache to improve performance of multiuser multiprocessor," in *Int. Conf. on Par. Proc., IEEE cat. no. 82cH1794-7*, 1982.

[11] M. C. Easton and R. Fagin, "Cold-start versus warm-start miss ratios," *CACM,* vol. 21. No. 10, pp. 866-872, 1978.

[12] C. Bell, J. Judge and J. McNamara, "Computer engineering: a DEC view of hardware system design," *Digital Press,* 1978.

[13] E. Akanksha, H. Shanvas and V. Nallusamy, "Modern CPU's Memory Architecture - A Programmer's Outlook," *Modern Education and Computer Science Press,* no. Published Online April 2012 in MECS, 2012.

[14] S. H. Pugsley, Z. Chishti, C. Wilkerson, P.-f. Chuang, R. L. Scott, A. Jaleel, S.-L. Lu, K. Chow and R. Balasubramonian, "Sandbox Prefetching: Safe Run-Time Evaluation of Aggressive Prefetchers," *IEEE,* pp. 1-12, 2014.

References

[15] N. Beckmann, P.-A. Tsai and D. Sanchez, "Scaling Distributed Cache Hierarchies through Computation and Data Co-Scheduling," p. IEEE, 2015.

[16] V. Jimenez, F. P.O'Conell and F. Cazorla, "Increasing Multicore System Efficiency through Intelligent Bandwidth Shifting," *IEEE,* pp. 39-50, 2015.

[17] Z. Wang, D. A. Jimenez, C. Xu, G. Sun and Y. Xie, "Adaptive Placement and Migration Policy for an STT-RAM-Based Hybrid Cache," *IEEE,* 2014.

[18] H. Farbeh and S. G. Miremadi, "PSP-Cache: A Low-Cost Fault-Tolerant Cache Memory Architecture," *EDAA,* 2014.

[19] A. Awad and Y. Solihin, "STM : Cloning the Spatial and Temporal Memory Access Behavior," *IEEE,* 2014 .

[20] N. Beckmann and D. Sanchez, "Talus: A Simple Way to Remove Cliffs in Cache Performance," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA),* San Francisco, 2015.

[21] M. Manivannan, V. Papaefstathiou, M. Pericàs and P. Stenström, "RADAR: Runtime-Assisted Dead Region Management for Last-Level Caches," *IEEE,* 2016.

[22] G. Kurian, S. Devadas and O. Khan, "Locality-Aware Data Replication in the Last-Level Cache," *IEEE,* 2014.

[23] S. Khan, A. R. Alameldeen, C. Wilkerson, O. Mutlu and D. A. Jimenez, "Improving Cache Performance Using Read-Write Partitioning," *IEEE,* 2014.

[24] A. Herdrich, E. Verplanke, P. Autee, R. Illikkal, C. Gianos, R. Singhal and R. Iyer, "Cache QoS: From Concept to Reality in the Intel Xeon Processor E5-2600 v3 Product Family," *IEEE,* 2016.

[25] P. Maniotis, D. Fitsios, G. Kanellos and N. Pleros, "Optical Buffering for Chip Multiprocessors: A 16GHz Optical Cache Memory Architecture," *Journal of lightware technology,* 2013.

[26] N. P. Jouppi, "Improving Direct Mapped Cache Performance by the Addition of a Small Fully Associative Cache and Prefetch Buffers," *Digital Corporation - Western Research Laboratory,* Vols. WRL Technical Note TN-14, pp. 1-36, 1990.

[27] "Cache Prefetching citing N. Jouppi," [Online]. Available: https://en.wikipedia.org/wiki/Cache_prefetching. [Accessed 16 Feb. 2017].

[28] K. M. Qureshi and Y. N. Patt, "Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches," in *The 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06),* Orlando, 2006.

[29] P. Moorhead, "Intel's Newest Core Processors: All About Graphics And Low Power," *Forbes ,* 4 June 2013.

Questions?

THANK YOU!