Exploiting NVRAM for Building Multi-Level Memory Systems

Dr. Krishna Kant
Intel Corporation

NVRAM Technologies

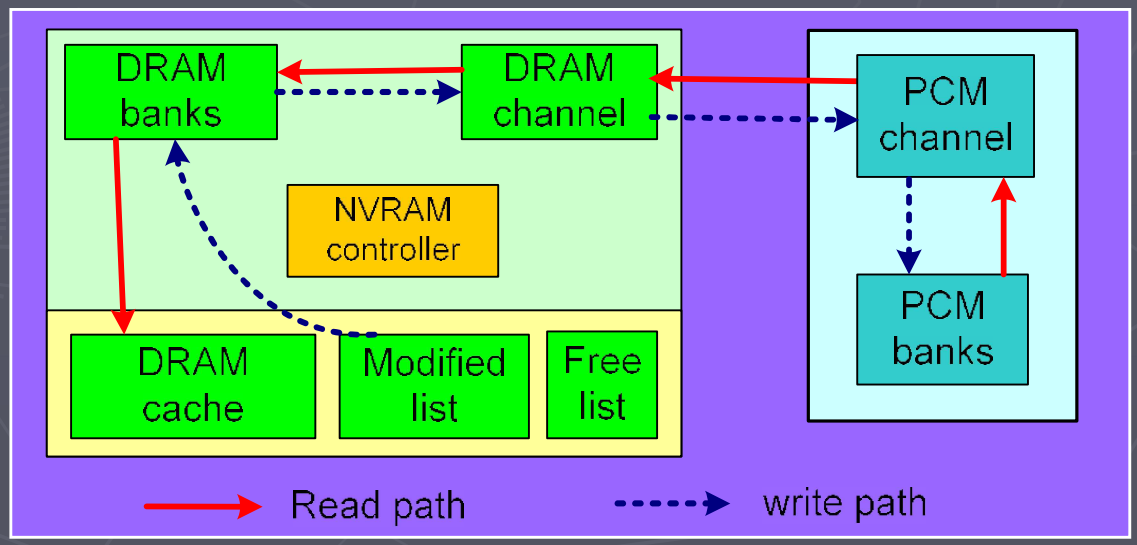
- Flash
  - Writes problematic: block only, need erasure
  - Degradation significant (100K write cycles)
- Phase Change Memory (PCM)
  - High density, fast reads, overwrite allowed, sectored R/W
  - Much longer life ~10M write cycles
- Other Technologies
  - Magneto-resistive RAM: Density big issue
  - Ferro-electric RAM: Very fast, density is an issue
  - Resistive RAM: Faster than PCM & lower power, not well developed.
  - Racetrack memory: Faster than PCM, not well developed.
  - SONOS (Si oxide nitride oxide Si), Nano RAM, millipede

NVRAM as Memory

- Most technologies unlikely to be fast enough to replace DRAM
- Second level memory?
  - Much more efficient than a file access model, but needs substantial FW/HW support.
  - Lower power than w/ DRAM only
  - Lower /GB cost than DRAM eventually
- Issues
  - Slow writes & limited lifetime an issue
  - Perhaps useful only for high locality workloads.
- Questions
  - Additional latency & its impact on throughput?
  - Power consumption?
  - To what extent can compression help mitigate extra latency?
  - Improve reliability by exploiting non-volatility?

NVRAM Memory Architecture

- DRAM (1st level memory): a fully associative cache
- NVRAM (2nd level): Accessed in units of pages & "sectors"
- Typical virtual memory mgmt
  - Implemented in FW/HW by NVRAM controller
  - Address translation, TLB, free/modified lists, etc.



PCM vs. DRAM

- Speed wrt DRAM
  - ~2x slower for reads, ~20 - 80X slower for writes
- Power consumption
  - Very little static power, about 10% of DRAM R/W power
- Page divided into sectors
  - Reads: Assume critical sector delivered first
  - Writes: Only the modified sector(s) written back.
- Small sector good for performance but expensive to maintain
  - Need dirty sector map in DRAM

Read Latency Issues

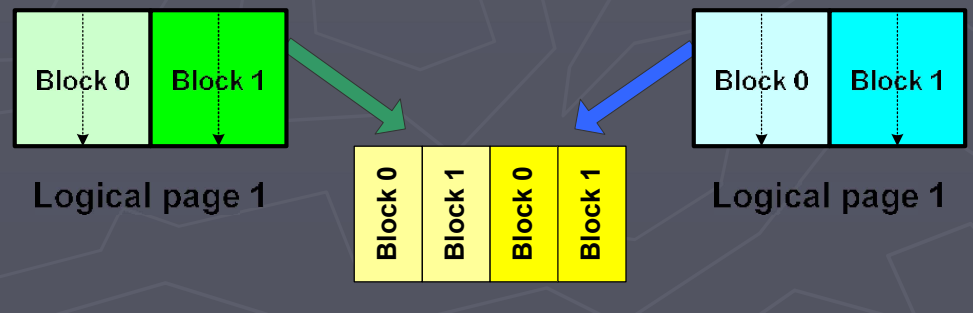
- Assumptions
  - Page size 4KB, sector size 1KB, CL size 64B
  - Read speed: 2x DRAM (e.g., 90 vs. 45 ns)
  - Write speed: 40x DRAM (e.g., 1.8 us)
- Read Latency Impact
  - Assuming critical sector first, need to read 8 CLs to obtain desired CL → 16x DRAM latency
  - 20% latency increase acceptable → 20/16 = 1.25% page miss ratio.
  - Need rather high locality of reference

Write Latency Impact

- Write latency hurts only when free list runs low.
- Page replenishment rate
  - Assuming 2 dirty sectors/page: 1/0.9 us = 1.1 M/sec
- Page demand rate
  - DRAM BW: DDR 1600 → 200 MT/sec → 12.8 GB/sec
  - Assume 50% channel utilization: 6.4 GB/sec → 1.6 M pages/sec
  - With 1.25% miss rate → 1.6 M x 0.0125 = 20 K pages/sec from NVRAM
- Demand ~1.8% of replenishment rate → Ok.
- Lifetime Issues
  - 20K page writes/sec → 1.75M page writes/day
  - 64 GB NVRAM → 16M pages → 0.11 writes/page/day

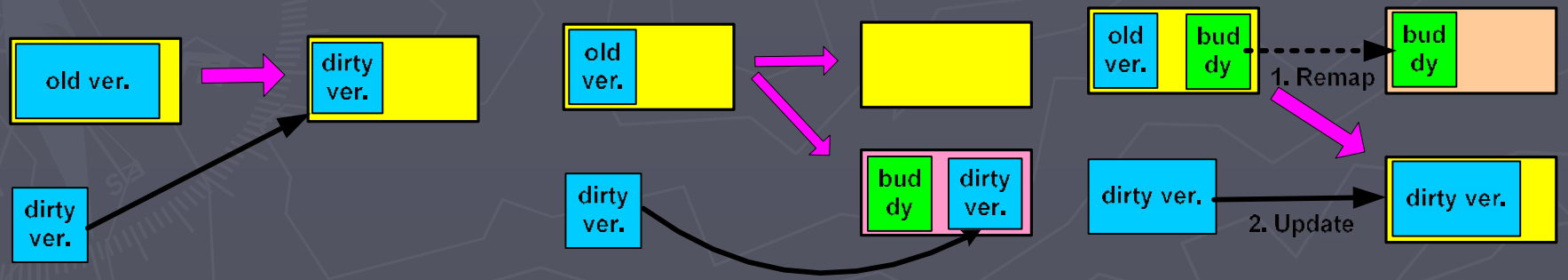
Exploiting Compression

- Scheme 1: No change to storage scheme (i.e., no NVRAM storage savings)
  - Still provides BW & power savings.
  - Can be exploited for better wear leveling
- Scheme 2: Compacted storage in NVRAM
  - Requires storage mgmt which could limit life



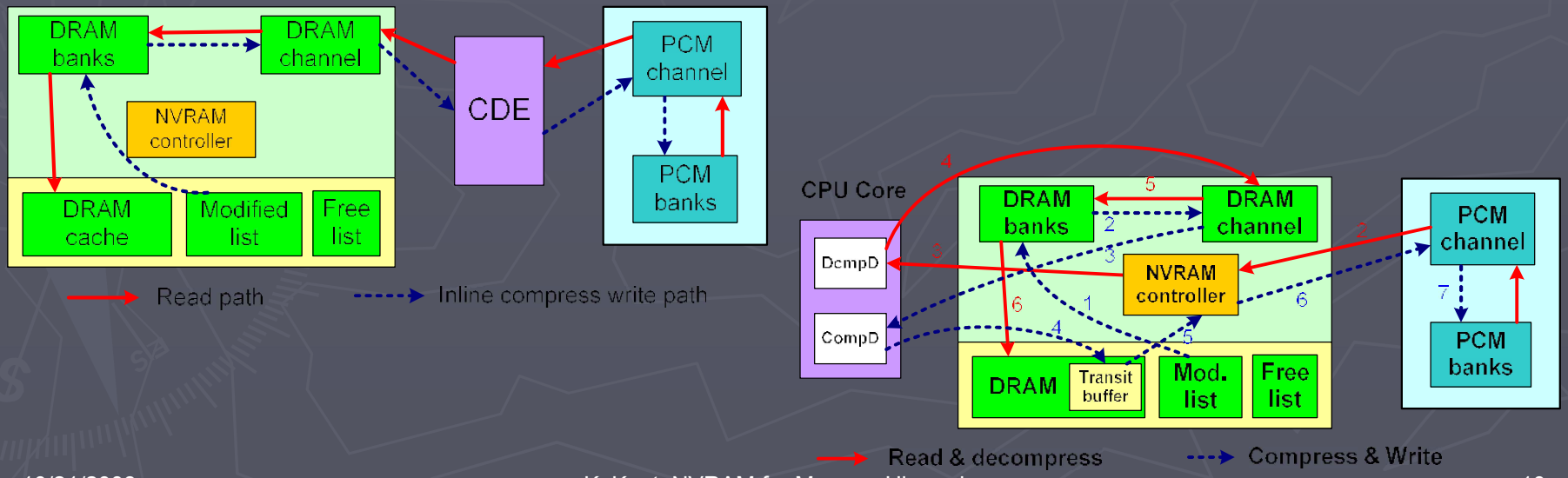
Storage Mgmt for Scheme 2

- Need to keep track of free & partially free pages.
- Occasional remapping required to consolidate partially free blocks
  - Not a desirable operation due to limited write cycles.
- Address translation mechanism for wear leveling can be exploited



Compression Architecture

- Multiple ways of doing compression
  - In NVRAM controller FW – Too slow?
  - HW based compression decompression engine (CDE)
  - Via a dedicated CPU core – need special features



Compression Rate

- Pipelined operations
  - Read: NVRAM data read || decompression || DRAM write
  - Write: DRAM read || compression || NVRAM write
- Compression/decompression at rates higher than other pipeline elements does not help
  - Optimal compression rate
  - Can we do optimal compression cheaply (e.g., in SW)?
- Tradeoff between compression rate & compressibility
  - Choice of compression algorithm
  - LZO – a version of LZ77, 10-20x faster than zlib, 20% worse compressibility.

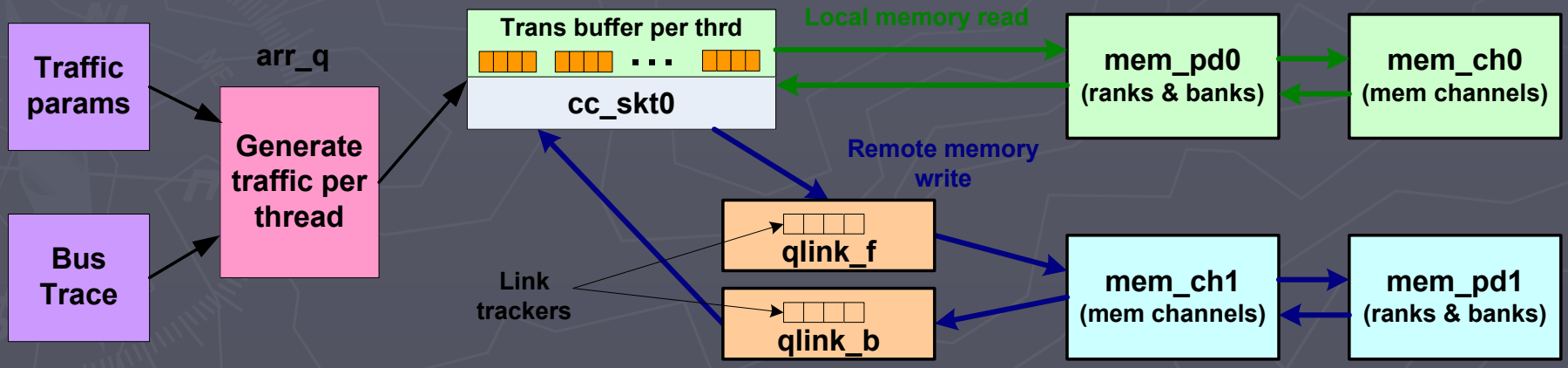
LZO Performance on Specweb2005

CBlock size (bytes)	ECOM			BANK			SUPP		
	comp ratio	comp rate	dcomp rate	comp ratio	comp rate	dcomp rate	comp ratio	comp rate	dcomp rate
256	3.322	424.7	906.9	2.348	333.9	932.1	1.416	240.5	1220.2
512	3.676	419.4	1001.6	2.530	325.8	1001.6	1.469	238.0	1220.2
1024	3.940	427.4	1032.4	2.664	315.1	1100.1	1.507	218.6	1290.6
2048	4.089	419.4	1525.2	2.747	327.4	1157.0	1.529	217.9	1342.2
4096	4.214	387.9	1525.2	2.804	299.6	1242.8	1.543	191.7	1398.1
8196	4.301	333.9	1560.7	2.845	247.6	1266.2	1.552	145.9	1398.1

- Only first 64MB of memory contents considered.
  - Better compressibility expected with whole thing
- Compressibility
  - Supp is poor (~1.5X), Ecom is great (~4X)
- Speed on 2.6Ghz Intel Core-2 duo
  - Compression 200-400 MB/sec, decomp 1.0-1.5 GB/sec.

Experimental Evaluation

- Evaluated w/ a very detailed simulator
  - 2-socket platform model – detailed DRAM model, Simpler CPU, interconnect & NVRAM models
  - Detailed implementation of NVRAM/DRAM mgmt.



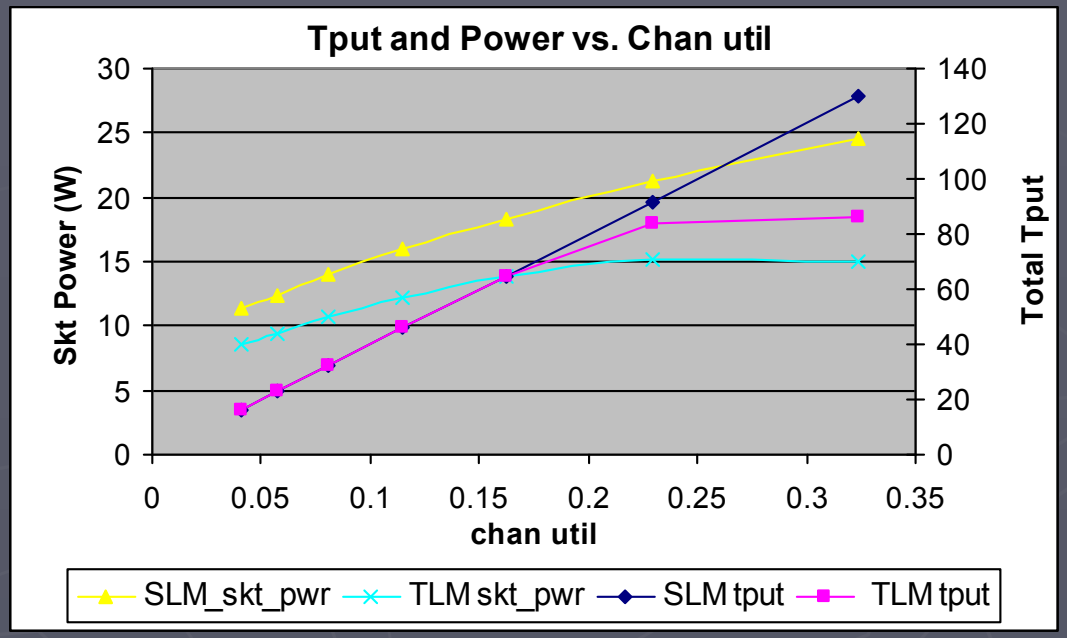
Power Control

- Realistic power control essential for power/performance evaluation.
- Power control techniques used
  - A combined proactive/reactive algorithm to use low power states for link & DRAM
    - Link: L0s & L1 states. L1 almost never entered.
    - DRAM: Fast & slow CKE states, slow CKE used a lot
  - Proactive selection of states + time based promotion.
  - Link width control used for lower link power
  - PCM: No explicit power control (static power negligible).

Model Parameters

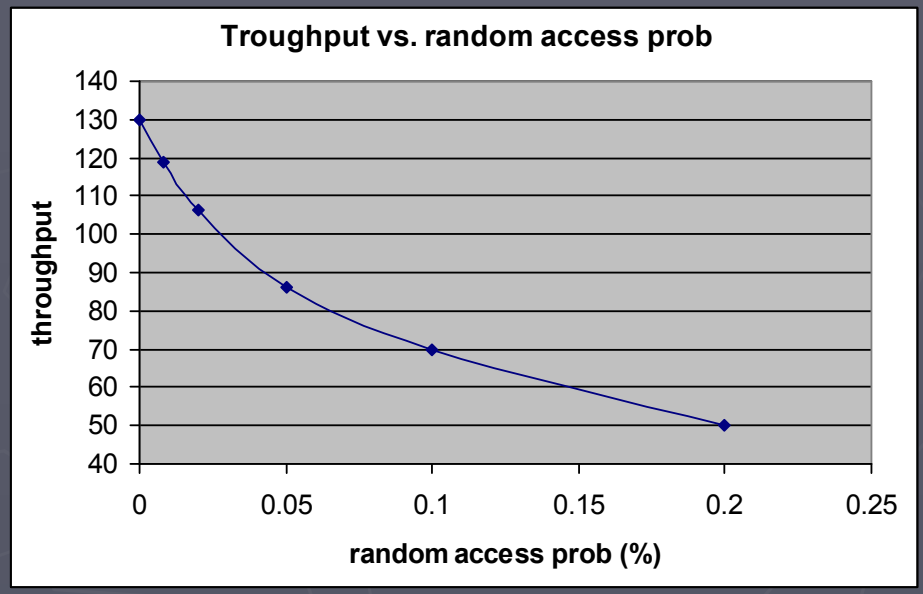
- Basic configuration
  - DRAM: DDR 1600, 2 chan/skt, 1 or 2 dimms/chan, 2 ranks/dimm, 8 banks/rank
  - NVRAM: DDR 800, 4 chan/skt, 1 dimm/chan, …
  - Remote access: 50% traffic/skt, 15% remote, 9.6GT/s link
- Compare 8 GB DRAM only w/ 4 GB DRAM + 8 GB NVRAM
- Access locality: 3 choices
  - Random jump in entire address space w/ prob 0.008-0.2%
  - Nearby page jump w/ 0.1, 0.5 & 1.0% prob. 0.04-1.0%
  - Zipf distribution for page distance
- Latency sensitivity can be controlled by CPU parms
  - Primarily consider latency insensitive workloads

Experimental Results



- No compression, Medium locality
  - Random page after 2000 refs, page change every 400 refs
- Power savings with TLM: 2 – 9 watts, but significant drop in tput at high utilizations

Experimental results



- No compression
- Confirms rapid drops in throughput as the locality decreases

Compression Results

sector size	block size	Comp ratio	decmp_rate	comp_rate	frac_read_lat	frac_write_lat	rel_perf_uc	rel_perf_c
512	1	1.694	1.398	0.503	0.717	0.634	0.914	0.937
512	2	1.826	1.341	0.483	1.066	0.807	0.914	0.909
512	4	1.948	1.312	0.472	1.726	0.885	0.914	0.861
512	8	2.051	1.297	0.467	3.000	0.871	0.914	0.781
1024	1	1.826	1.341	0.483	**0.631**	**0.583**	**0.864**	**0.909**
1024	2	1.948	1.312	0.472	1.022	0.639	0.864	0.861
1024	4	2.051	1.297	0.467	1.776	0.629	0.864	0.781
2048	1	1.948	1.312	0.472	0.563	0.546	0.777	0.861
2048	2	2.051	1.297	0.467	0.978	0.537	0.777	0.781
4096	1	2.051	1.297	0.467	0.515	0.520	0.647	0.781

- Optimal sector & compression block size: 1KB
- Optimal rates: Compr: 483 MB/s, Decompr: 1.34 GB/s
- Benefits
  - Reduces read latency to 63% & write latency to 58%
  - Performance benefit (sample): 86.4% to 91%
  - NVRAM power reduction by compression ratio ~2X for NVRAM

Conclusions

- Use of NVRAM to build memory hierarchy.
- Useful for low locality workloads
  - Small tput degradation but big power savings.
  - At low utilization levels, degradation may be insignificant (but power savings also small).
- Compression can be useful
  - Optimal compression can reduce read latencies by a few 10's of percentage.
  - Additional power impacts: perhaps <100mW.
- Need more detailed studies w/ more realistic CPU models.