FastCool: Leakage Aware Dynamic Thermal Management of 3D Memories

Lokesh Siddhu and Preeti Ranjan Panda
Department of Computer Science and Engineering
Indian Institute of Technology Delhi
{siddhulokesh, panda}@cse.iitd.ac.in

***Abstract*—3D memory systems offer several advantages in terms of area, bandwidth, and energy efficiency. However, thermal issues arising out of higher power densities have limited their widespread use. While prior works have looked at reducing dynamic power through reduced memory accesses, in these memories, both leakage and dynamic power consumption are comparable. Furthermore, as the temperature rises the leakage power increases, creating a thermal-leakage loop. We study the impact of leakage power on 3D memory temperature and propose turning OFF hot channels to meet thermal constraints. Data is migrated to a 2D memory before closing a 3D channel. We introduce an analytical model to assess the 2D memory delay and use the model to guide data migration decisions. Our experiments show that the proposed optimization improves performance by 27% on an average (up to 66%) over state-of-the-art strategies.**

I. Introduction

To ease performance bottlenecks caused by memory systems, researchers have proposed 3D (stacked 2D) memories which offer a smaller form factor and higher bandwidth; however, they have higher power density leading to frequent heating and cooling phases. Obtaining high performance under thermal constraints is a challenge in 3D memories [1].

Power consumption in memories comprises dynamic power and static/leakage power. Static power increases exponentially with temperature creating a positive feedback between temperature and leakage [2]. As a significant (∼52%) portion of the power consumed by memories is static power [3], it is essential to reduce static power to meet thermal constraints. In this work, we reduce static power by turning OFF hot channels.

We propose a novel data migration based strategy to maximize performance under thermal constraints for architectures with both 2D and 3D memories (Figure 1). Once the 3D memory heats up, data from hot channels is migrated to the 2D memory, and the affected 3D channels are turned OFF. Migrated data is accessed from the 2D memory. When the 3D memory cools down, data is brought back from the 2D memory. We also develop an analytical model to predict the 2D memory delay and guide migration decisions. We show that our strategy leads to small migration overheads, and results in an average runtime improvement of 27% (up to 66%) over state-of-the-art policies which reduce only dynamic power.

This paper makes the following contributions:

1) A study of the thermal-leakage loop resulting in a system-level performance optimization strategy respecting thermal constraints.

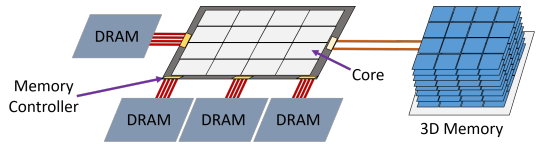

Fig. 1. 16-core Processor with off-chip 3D and 2D memories.

2) A thermal-aware data migration strategy for 3D memory.
3) A detailed analytical model to quantify the memory delay including queuing delays.

II. Related Work

Dynamic thermal management (DTM) policies periodically monitor the temperature and reduce the access rate of hot functional units using throttling or dynamic voltage/frequency scaling (DVFS). Limiting the performance overhead of DTM schemes is an active area of current research. Recent works have addressed the thermal management of 3D processors [4]–[6]. However, there has been limited work addressing thermal issues in 3D memories. Lo et al. discuss a thermal-aware page allocation policy where page allocation requests to a hot channel are reallocated to the coldest channel [1]. However, they do not consider heating due to accesses to existing pages which can eventually lead to throttling. For 2D memories, Lu et al. reduce static power by grouping pages with similar access locality into the same rank and placing idle ranks in low power modes [7]. Existing DTM strategies for 3D memories do not reduce the static/leakage power during DTM, which extends the stall time for cooling down. Further, there is a limited study of the effect of the thermal-leakage loop.

III. The FastCool Proposal

A. Motivation

We motivate our work by studying the thermal behavior of 3D memories and the effect of leakage power. Figure 2 shows the temperature profile of an 8-layer 3D memory system under uniform power dissipation with the heat sink placed on the top for cooling. The cooler upper layers have better heat dissipation capability than lower layers. Within a layer, the central ranks have the highest temperature as they get heated from all the sides [1].

DRAM manufacturers expand capacity by increasing memory cell density; this increases leakage causing significant static power dissipation. For a 1GB, 50 nm, 46 mm^2 3D memory [8], the static power is 39% of the total power at

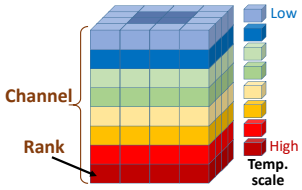

Fig. 2. Sixteen channel 3D Memory Hottest at bottom layer, central ranks

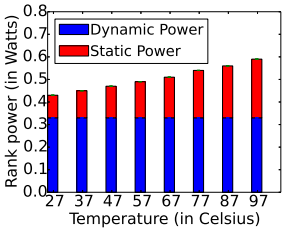

Fig. 3. Rank (8MB) power vs. temp. for a 3D memory using CACTI-3DD.

77°C. Further, leakage power increases with rising temperature (Figure 3). As an example, for the *lbm* workload (Figure 6) leakage increases temperature by ∼25°C (simulation details are presented in Section IV).

B. Thermal-Aware Channel Data Migration (TAM)

The temperature dependent leakage contributes significantly to 3D memory heating. To limit leakage power and meet thermal constraints, we propose using data migration at channel level granularity and turning OFF the hot channels. This has some advantages over approaches using a page-granularity that suffer from several challenges: data migration delays, profiling and counting memory accesses per page, and prediction of appropriate low power state and power down timeout [7]. These challenges can be conveniently addressed for meeting thermal constraints of 3D memories due to the presence of multiple channels which enables faster data migration. The slow varying characteristic of temperature (compared to the clock period) allows sufficient time to implement a coarse-grained cooling strategy such as data migration. Further, the coarse-grained channel-level profiling of accesses reduces the profiling overhead.

Our proposed thermal-aware channel data migration (which runs every epoch of duration T) is defined in terms of temperature thresholds $T1$, $T2$, $T3$, and $T4$ ($T4$>$T3$>$T2$>$T1$). We assume that the maximum temperature constraint $T4$ (such as 80°C) is specified for maintaining a safe temperature margin. Initially, each core is allocated a separate 3D memory channel to reduce memory interference [9]. Depending on the temperature, the thermal management scheme could be in any of the six states from *normal* to *throttle* (Figure 4). Below $T1$, the system works normally without any intervention. Above $T1$, data in the least accessed channels within the 3D memory is swapped with interior channels {5,6,9,10} (*migrate* state). If the memory cools down below $T1$, it transitions back to the *normal* state; else, data is migrated from the interior 3D channels to 2D memory and these channels are turned OFF. The migrated data is accessed from the 2D memory until the 3D memory cools down to temperature $T1\_C$ which is lower than $T1$ to prevent multiple transitions occurring around the threshold. The remaining channels are accessed normally without the processor being stalled.

On closing the interior channels, their temperature decreases, helping the adjacent channels to cool down. Now, the corner channels become the hottest as they are farthest from the interior channels. Hence, we close the diagonal corner

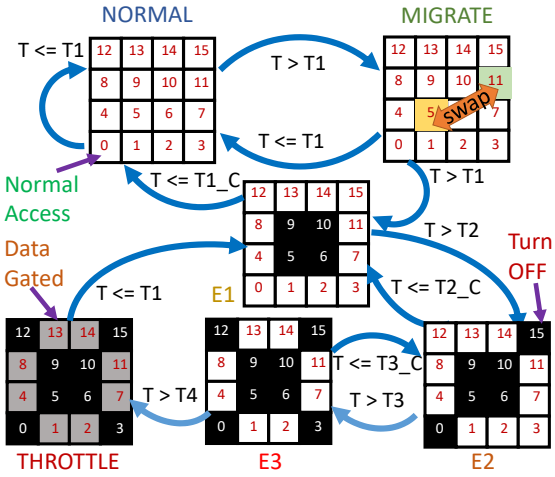

Fig. 4. Thermal-Aware data migration and channel shut down sequence

TABLE I
PARAMETER DEFINITIONS

Symbol	Description	Typical value
T	Time for an epoch	1 ms
C	Cache Line Size	64 Bytes
3D Memory Parameters		
b_{3D}	Per channel 3D memory bandwidth	8 GBps
L_{3D}	3D memory access latency	29 ns
s_{3D}	Size of data migrated, 3D → 2D	*Runtime*[1]
A_{3D}	Total accesses made to s_{3D} data (in the last T)	*Runtime*[1]
n_{3D}	Number of 3D channels (migrating 3D → 2D)	*Runtime*[1]
B_{3D}	Total 3D bandwidth of migrated channels	b_{3D} x n_{3D}
2D DRAM Parameters		
b_{2D}	Per channel bandwidth	12.8 GBps
L_{2D}	2D memory access Latency	45 ns
s_{2D}	Size of data migrated, 2D → 3D	*Runtime*[1]
A_{2D}	Accesses made to 2D memory (in the last T)	*Runtime*[1]
N_{2D}	Number of 2D memory channels	4
R_{2D}	Number of ranks per channel	2
BA_{2D}	Number of banks per rank	8
B_{2D}	Total 2D bandwidth of migrated channels	b_{2D} x N_{2D}
	Page Policy	Closed

[1] *Runtime* – Values are determined at runtime

channels {0,15} and {3,12} in sequence if the temperature rises to $T2$ and $T3$ respectively. Above $T4$, accesses to channels {1,2,4,7,8,11,13,14} are data gated (accesses prevented) and channels {0,3,5,6,9,10,12,15} remain turned OFF/power gated until the 3D memory cools down to $T1$ (*E1* state). We refer to the above algorithm as thermal-aware migration (TAM).

C. 2D Memory Delay Model

Migrating data from (faster) 3D to (slower) 2D memory consumes 2D memory bandwidth and increases the queuing delay which could increase the time to complete memory requests. To avoid such situations, we develop a delay model which estimates the time taken to complete 2D memory accesses. As the 3D memory heats up, we migrate data to (and subsequently access from) the 2D memory. The time to complete these requests consists of the data migration delay (*DMD*) and data access delay (*DAD*) which are impacted by both memory latency and bandwidth.

$$\textit{Time to complete 2D memory requests} = DMD + DAD \quad (1)$$

Data migration delay (DMD): Once the 3D memory is heated, we migrate s_{3D} (description in Table I) data to 2D memory. When the memory is cooled down, we copy back s_{2D} data to the 3D memory. During migration, both memories operate in parallel (because of multiple channels), and we can write the 2D memory while reading the 3D memory. The 2D and 3D memories operate at a bandwidth of B_{2D} and B_{3D}. Further, the slower of the two memories determine *DMD*.

$$DMD = (s_{3D} + s_{2D}) \times \max(\frac{1}{B_{3D}}, \frac{1}{B_{2D}}) \quad (2)$$

Data access delay (DAD): We evaluate the impact of bandwidth (DAD_B) and latency (DAD_L) separately and add them to obtain $DAD = DAD_B + DAD_L$. Once hot data is migrated from 3D to 2D memory, the total 2D memory accesses (A) increases by A_{3D} giving $A = A_{2D} + A_{3D}$ and the total data fetched for A accesses is $A \times C$. The delay due to bandwidth is given by $DAD_B = (A \times C)/B_{2D}$. DAD_L is the sum of the waiting time (when the memory is busy) in the queue (QD) and the memory latency delay (LD).

Latency delay (LD): Each memory request issued to a bank requires L_{2D} time to be serviced, assuming a closed page policy. However, multiple requests can be active simultaneously at the multiple banks in a rank. Similarly, multiple ranks and channels are active simultaneously, reducing LD. We have:

$$LD = \frac{A \times L_{2D}}{BA_{2D} \times R_{2D} \times N_{2D}} \quad (3)$$

Queuing delay (QD): To quantify memory waiting times we use queuing models that are characterized by the arrival process (access request) distribution, the number of servers (in our case, memory modules are servers) and service time distribution of the server. As our memory subsystem has a separate queue for each memory channel, we model each channel using an $M/M/1$ queue with multiple channels operating in parallel. An $M/M/1$ queue represents a system with a single queue per server, with the arrival (λ), service rates (μ) are assumed to be Poisson and exponential respectively. The expected waiting time is given by: $\lambda/(\mu \times (\mu - \lambda))$. Assuming accesses are uniformly distributed across channels, the per channel access rate $(A \times C)/(T \times N_{2D})$ and memory bandwidth (B_{2D}/N_{2D}) are the arrival (λ) and service rates (μ) respectively, giving:

$$QD = \frac{A \times C}{T \times B_{2D}} \times \frac{N_{2D}}{B_{2D} - (A \times C/T)} \quad (4)$$

The above analysis helps us estimate the time to complete the requests to 2D memory. If the total 2D memory delay is higher than a threshold, we can avoid migrating the data (Equation 7). Using Equations (1) to (4), the total 2D memory delay is: $\underbrace{DMD}_{\text{Migration Delay}} + \underbrace{DAD_B}_{\text{Bandwidth Delay}} + \underbrace{QD + LD}_{\text{Access Delay}}$.

D. The FastCool Algorithm

We outline the FastCool algorithm, which aims to reduce unnecessary transitions in the TAM algorithm and adds the following conditions on the state transitions.

Applications with higher memory access rates lead to higher temperatures. We observed experimentally (parameters in Section IV-A) that the temperature rises above 81°C (Figure 5) for access rates higher than 40.96 GBps (40K accesses per channel). Hence, as the temperature rises to $T1$ (Figure 4), we transition to *E1* state only if the total access count of channels {5,6,9,10} exceeds A_{MIN} (for our architecture A_{MIN} is 160K).

$$\text{Minimum Access Rate condition: } A_{3D} > A_{\text{MIN}} \quad (5)$$

This condition ensures that applications with predicted steady state temperatures less than 80°C do not migrate data.

As 3D memory heats up, TAM migrates data without checking whether the 2D memory is overloaded. We must ensure that $\lambda < \mu$ to avoid overloading. From Equation (4):

$$\text{Queuing stability condition: } A < B_{2D} \times T/C \quad (6)$$

TAM does not account for 2D memory speed before migrating hot data. Hence, a slow 2D memory could adversely affect the application runtime. To prevent such conditions, we set an upper bound on the estimated time to complete to 2D requests. The upper bound on the delay (D_{MAX}) is found experimentally: we varied D_{MAX} and measured the performance for various workloads. We selected the D_{MAX} value that gave the minimum average runtime. For our architecture, this value is 8.415 ms (experiments are omitted due to lack of space).

$$\text{Upper bound on 2D memory delay: } Delay < D_{MAX} \quad (7)$$

Using Equations (5) to (7), FastCool ensures that the transition to higher thermal emergency states are made only for high access rates such that the migrated data does not saturate the 2D memory bandwidth and the estimated 2D memory delay is lower than a threshold value.

IV. Experimental Results and Analysis

A. Methodology

For evaluating the proposed algorithm, we use a trace-based framework. We run the workload on Sniper 6.1 and collect a memory access trace assuming only 3D memory is present. Using the energy-per-access values (20.55 nJ per 64 bytes access) determined from CACTI-3DD [10], we convert the access trace to a power trace (with 1 ms epochs) which is fed to Hotspot 6.0 thermal simulator augmented to implement our strategy (including the high-level modeling for the 2D memory as described in section III-C). We add the temperature dependent leakage to the power trace to obtain the total power.

TABLE II
Workloads and Benchmark Details

Benchmark Details and Type	Workload
povray(x4), perlbench(x4), soplex(x4), lbm(x4)	*ppsl*
povray(x4), perlbench(x4), bwaves(x4), GemsFDTD(x4)	*ppbg*
bwaves(x4), GemsFDTD(x4), soplex(x4), lbm(x4)	*bgsl*
perlbench(x4), lbm(x8), GemsFDTD(x4)	*pllg*

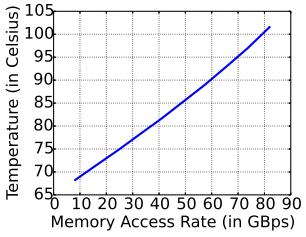

Fig. 5. Temperature exceeds 81°C with access rates >40.9 GBps

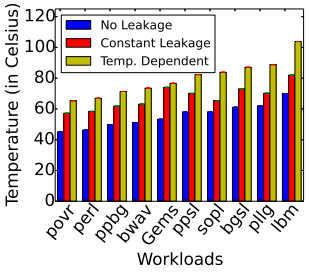

Fig. 6. Effect of leakage power on the 3D memory steady state temperature.

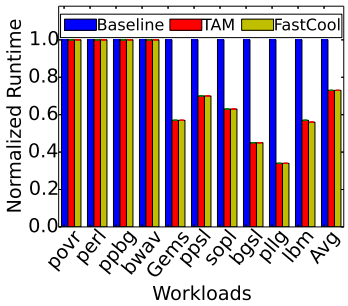

Fig. 7. Normalized runtime comparison

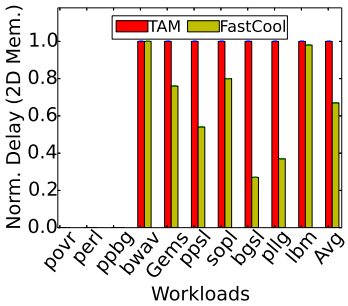

Fig. 8. 2D memory delay comparison

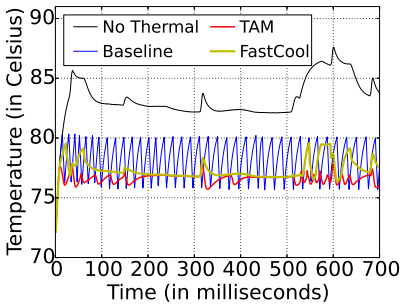

Fig. 9. Temperature trace

In our experiments, we use a processor with 16 cores, a 3D stacked memory (1GB, 15ns latency, 128 GBps, 16 channel, 8 ranks per channel, single bank per rank) and a 2D memory (see Table I). Core parameters are the same as in [5]. Temperature thresholds $\{T1, T2, T3\} = \{76.4, 76.9, 79.5\}$ and $\{T1\_C, T2\_C, T3\_C\} = \{71.4,76.5,77.4\}$ (in Celsius) are experimentally determined. Configuration parameters for Hotspot are the same as in [1]. We classify the SPEC 2006 benchmarks into three categories and present experimental results for two representative benchmarks from each category – compute-intensive (*povray*, *perlbench*), mixed (*bwaves*, *GemsFDTD*), and memory-intensive (*soplex*, *lbm*). 16 instances of each representative benchmark form the first six workloads. Mixes of the representative benchmarks are used to get the next four workloads (Table II), building 10 workloads. The workloads are simulated until each benchmark has completed at least 1B instructions, restarting the benchmark(s) completing early.

B. Results

Effect of Leakage Power: Figure 6 shows the effect of no leakage, constant leakage (25°C), and temperature dependent leakage power on steady-state memory temperature. Memory intensive workloads induce the highest temperatures. Thermal-leakage loop increases temperature, with its effect being dominant at high temperatures. A constant leakage model is unable to determine temperature accurately for all workloads.

Runtime Comparison: Figure 7 shows the runtimes of the proposed strategies TAM and FastCool with respect to the baseline. We use 3D memory page allocation as the baseline [1], which uses page allocation to reduce temperature; this reduces neither dynamic nor leakage power, causing frequent memory stalls. At high temperatures, leakage power is comparable to dynamic power, and it is hard to ensure a fast cool down with throttling alone. TAM and FastCool turn OFF the hot channels (following data migration to 2D memory) – reducing the leakage power and system throttling events. The migrated data is accessed from the 2D memory ensuring all the applications make progress. We observe runtime reductions of 27% on an average and up to 66%.

Impact of considering the delay model: FastCool and TAM achieve comparable runtimes, but FastCool places lower burden on the 2D memory and reduces the time to complete 2D memory requests by 33% as compared to TAM (Figure 8). We illustrate this using a temperature-time trace for the *bgsl* workload. As we observe in Figure 9, without thermal constraints, the temperature exceeds 80°C. However, the baseline strategy throttles at 80°C ($T4$) until it cools down to 76.4°C. Since TAM and FastCool reduce leakage and dynamic power, they need not throttle and can maintain the temperature between 76–79.4°C. From 500-700 ms, TAM closes hot channels and maintains a temperature of about $T2$ (76.9°C), migrating between $E1$ and $E2$. However, FastCool does not close channels at 76.9°C (because of the minimum access rate condition) and the temperature rises to 79.4°C, which effectively utilizes the temperature headroom, avoiding unnecessary migrations.

V. Conclusion

We modeled the effect of leakage on 3D memory temperature and proposed a novel algorithm which turns hot channels OFF to maximize performance under thermal constraints. Data is migrated to 2D memory before closing a 3D memory channel. We developed an analytical model to quantify the 2D memory delay and used it to guide decisions. We evaluated the strategy using SPEC2006 workloads running on a 3D+2D memory platform. Our simulation results show an average runtime improvement of 27% (up to 66%) over state-of-the-art policies which reduce dynamic power alone.

Acknowledgments

We express our gratitude to Rajesh Kedia for several useful suggestions on this work. We thank the Visvesvaraya PhD fellowship for the financial support to Lokesh Siddhu.

References

[1] W. Lo *et al.*, "Thermal-aware dynamic page allocation policy by future access patterns for Hybrid Memory Cube (HMC)," in *DATE*, 2016.
[2] M. Zapater *et al.*, "Leakage and temperature aware server control for improving energy efficiency in data centers," in *DATE*, 2013.
[3] H. Huang, K. G. Shin, C. Lefurgy, and T. Keller, "Improving energy efficiency by making DRAM less randomly accessed," in *ISLPED*, 2005.
[4] C. H. Liao, C. H. P. Wen, and K. Chakrabarty, "An online thermal-constrained task scheduler for 3D multi-core processors," in *DATE'15*.
[5] J. Meng *et al.*, "Optimizing energy efficiency of 3-D multicore systems with stacked DRAM under power and thermal constraints," in *DAC'12*.
[6] G. L. Loi *et al.*, "A thermally-aware performance analysis of vertically integrated (3D) processor-memory hierarchy," in *DAC*, 2006.
[7] Y. Lu *et al.*, "Rank-aware dynamic migrations and adaptive demotions for DRAM power management," *TC*, 2016.
[8] J. Jeddeloh and B. Keeth, "Hybrid memory cube new DRAM architecture increases density and performance," in *VLSIT*, 2012.
[9] Muralidhara *et al.*, "Reducing memory interference in multicore systems via application-aware memory channel partitioning," in *MICRO*, 2011.
[10] K. Chen *et al.*, "CACTI-3DD: Architecture-level modeling for 3D die-stacked DRAM main memory," in *DATE*, 2012.