Reliability and Performance Trade-off Study of Heterogeneous Memories

Manish Gupta[1] David Roberts[2] Mitesh Meswani[2] Vilas Sridharan[3]
Dean Tullsen[1] Rajesh Gupta[1]
[1]Dept. of Computer Science and Engineering
University of California San Diego
{manishg, tullsen, rgupta}@cs.ucsd.edu
[2]AMD Research [3]RAS Architecture
Advanced Micro Devices, Inc.
{david.roberts, mitesh.meswani, vilas.sridharan}@amd.com

ABSTRACT

Heterogeneous memories, organized as die-stacked in-package and off-package memory, have been a focus of attention by the computer architects to improve memory bandwidth and capacity. Researchers have explored methods and organizations to optimize performance by increasing the access rate to faster die-stacked memory. Unfortunately, reliability of such arrangements has not been studied carefully thus making them less attractive for data centers and mission-critical systems. Field studies show memory reliability depends on device physics as well as on error correction codes (ECC). Due to the capacity, latency, and energy costs of ECC, the performance-critical in-package memories may favor weaker ECC solutions than off-chip. Moreover, these systems are optimized to run at peak performance by increasing access rate to high-performance in-package memory. In this paper, authors use the real-world DRAM failure data to conduct a trade-off study on reliability and performance of Heterogeneous Memory Architectures (HMA). This paper illustrates the problem that an HMA system which only optimizes for performance may suffer from impaired reliability over time. This work also proposes an age-aware access rate control algorithm to ensure reliable operation of long-running systems.

CCS Concepts

•Computer systems organization → Reliability; Availability; •Hardware → Aging of circuits and systems; Memory and dense storage;

Keywords

Reliability, Aging, Heterogeneous Memory, Die-stacking

1. INTRODUCTION

Die-stacking DRAM in a single package allows us to increase capacity and reach high bandwidth. Some proposals integrate off-package (high-capacity) memory and in-package (high-bandwidth) memory into one system. For the system to operate at its peak performance, as many accesses as possible should go to the highest-performing memory device. The goal of most of the research in this field is to track, predict and migrate data to the most appropriate location in order to increase performance [4, 18, 23, 28] or reduce energy [23, 32].

Despite of improvements in capacity and bandwidth, reliability and serviceability of the heterogeneous memories face greater challenges. Off-package DDR DRAM provides error protection using techniques such as ChipKill [6]. By contrast, while robust ECC schemes for in-package memory have been proposed [20,29], the constrained capacity for these memories implies that lower-overhead ECC is likely to be employed. For example, off-package memory uses additional DRAM chips to provide ChipKill, but adding additional DRAM dies is expensive for in-package die-stacked memories. For the purpose of this paper, we assume that off-package DRAM uses single-ChipKill ECC, and in-package die-stacked memory uses single error correct-double error detect (SEC-DED) ECC. For the rest of the paper, we will refer to single-ChipKill as ChipKill.

Errors can be due to transient faults or permanent faults. Accumulation of transient faults can be corrected, up to a certain extent, by a technique called scrubbing [2]. Memory scrubbing reduces the chance that transient faults result in error accumulation beyond the capability of error-correcting codes [19]. During scrubbing, the memory controller reads all memory locations and corrects errors. Errors resulting from permanent faults cannot be corrected by scrubbing, and may lead to fault accumulation over time. Accumulation of these faults makes the memory system more susceptible to an uncorrectable (and eventually undetectable) error as it ages. An uncorrectable error in memory can lead to a machine crash [24], application data corruption, and create security vulnerability [7, 16, 33]. Hence, reliability is important to avoid system downtime and ensure correct and secure operation. In order to maintain reliability, in the presence of aging, bad DIMMs in off-package memory can be replaced. However, replacing the die-stacked in-package memory requires replacing the entire chip.

In 2012, Sridharan et al. published a study of DRAM failures in the field [31]. Their study shows that ChipKill reduces the node failure rate attributed to uncorrected DRAM errors by 42x relative to SEC-DED. Hence, every access to a less resilient, die-stacked main memory device decreases the system reliability compared to accessing a more reliable off-package memory. In this paper, we show that making ju-

MEMSYS 2016 October 3–6, 2016, Washington, DC, USA

DOI: http://dx.doi.org/10.1145/2989081.2989113

dicious use of both in- and off-package memory can reduce the failure probability by 26x compared to a system that makes all of its memory accesses from the fastest in-package memory alone.

The major contributions of this work are:

- We demonstrate a reliability-performance trade-off study of HMA systems. We show that system failure probability can increase by more than a factor of 1000 in the seventh year compared to the failure probability in year 1.
- We show low and medium bandwidth benchmarks can ensure greater reliability (or lower failure probability) by only using off-package DDR memory at a cost of 37% and 45% performance degradation, respectively. However, a high bandwidth benchmark observes 51% drop in performance when all its memory access are restricted to more reliable off-package DDR memory.
- We show how a system with long-running shelf life can use the access rate to fast memory as a control knob to ensure sustained operation at a reduced failure rate over its lifetime.

The rest of this paper is organized as follows. Section 2 provides background information of die-stacked memory and existing research efforts to enhance their reliability. Section 3 elaborates our experimental methodology and tools used to study the performance-reliability trade-off. Section 4 presents our reliability, performance, and reliability vs. performance trade-off results. Section 5 demonstrates an aging-aware access rate control policy that uses access rate as a knob to ensure reliable operation of long-running systems. Section 6 discusses our methodology critically and the applicability of the aging-aware algorithm in the field. Section 7 summarizes and concludes this paper.

2. BACKGROUND

This section provides a quick overview of the organization of off-package and in-package die-stacked memory. We will also list challenges architects face in order to increase the reliability of in-package die-stacked memory.

Off-package memory is a 2D arrangement of DRAM chips. Each chip provides a fixed number of bits, x4 and x8 for 4-bit and 8-bit per cycle, respectively. Figure 1 (a) shows one side of a conventional off-package DDR x8 DIMM with eight DRAM chips for data and one chip for ECC. The additional 8-bit chip can provide single-bit correction and double-bit detection for 64-bit data. A memory with x4 DRAM chips can provide stronger error codes using ChipKill [6].

Figure 1 (b) shows a die-stacked DRAM with four layers. DRAM on die-stacked memory can provide as much as 128 bits per cycle, and the entire cache line is fetched from a single DRAM chip. ChipKill as implemented in off-package memory requires striping a cache line across multiple chips, and is infeasible in die-stacked DRAM. Striping a cache line across multiple chips in die-stacked memory reduces bank-level parallelism, which results in a performance overhead [17]. Moreover, activating and fetching data from different chips also has a power/energy overhead. Sim et al. presented an extension to the conventional ECC based design for die-stacked DRAM to provide ChipKill-level RAS, which is very expensive and unlikely to reach production in near future [29]. Nair et al. published Citadel [20] which could decrease the probability of failure by 700x by providing tri-dimensional parity. The research in the field has come up with various innovative ways to increase the reliability of die-stacked memories [12, 13, 20]. Micron's Hybrid Memory Cube (HMC) [22] provides a strong reliability for its 3D die-stacked memory. However, it requires special DRAM chips and changes to the memory controller. Hence, all of these solutions come at a cost of performance, energy, and complexity.

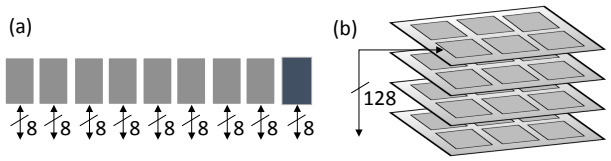

Figure 1: (a) Conventional off-package DDR memory with nine DRAM chips (Eight DRAMs for data bits and one DRAM for ECC bits). (b) Die-stacked high bandwidth memory.

3. EXPERIMENTAL METHODOLOGY

We used FaultSim [21] and modified Ramulator [15] to evaluate reliability and performance trade-offs.

3.1 Reliability simulations

FaultSim is an event-based DRAM fault simulator by Nair et al. [21]. The simulation framework uses DRAM failure rate data from Sridharan et al. [31]. FaultSim's event-based design and real-world failure statistics make it a fast and accurate tool for our reliability studies. A user-provided number of simulations are run. In each simulation, a fault is injected in a bit, word, column, row, or bank based on their FIT (Failure In Time) rates, a selected error-correction scheme is applied, and the outcome is recorded as detected, corrected, or uncorrected error. Injected faults can be transient or permanent, and an error occurs when the failed bit is used in the software. FaultSim assumes that every DRAM fault results in an error, which is a conservative assumption and provides an upper bound on the error rate for our studies. The faults are inserted to simulate a time duration of 348 weeks (approximately 7 years). We use the probability of uncorrected errors to measure the failure probability of a heterogeneous memory architecture.

FaultSim simulates SEC-DED and ChipKill error codes. SEC-DED and ChipKill have different starting reliability and aging curves [21]. We simulate a heterogeneous memory architecture with level one (M1) memory equipped with SEC-DED and level two (M2) with ChipKill. SEC-DED provides lower complexity and power, and ChipKill provides higher reliability.

In-package die-stacked memories may provide weaker reliability using schemes such as parity-based detection in order to reduce latency and capacity overhead [1]. Hence, assuming SEC-DED for in-package memory is a realistic assumption [5]. Moreover, in-package die-stacked memory's eight times higher density, and newer failure modes such as faulty through-silicon vias (TSV) also increase its failure probabilities. In the absence of field study data on failure rates of die-stacked memory, Citadel [20] scales FIT rates by eight times of the real-world DRAM field study [31], and performs

a sensitivity study by sweeping FIT rate for TSV failures. In this work, we illustrate the problem arising from different reliability levels of in-package and off-package memories. Scaling die-stacked memory's FIT rate by eight times, and including newer failure modes is only going to increase the gap in the reliability levels and make our argument stronger. For our study, we do not scale the FIT rates for die-stacked DRAM to be as conservative as possible. Hence, we simulate memory M1 and M2 with same densities and FIT rates. However, M1 and M2 are equipped with different ECC schemes, SEC-DED and ChipKill, respectively.

Failure probabilities of a heterogeneous memory system in the week w, $FP_{HMA}[w]$, can be computed using equations 1 and 2. Equation 1 defines the access rate as the ratio of number of accesses to the level 1 memory M1 over the total number of access to the memory. Given the failure probabilities of the individual memories in the week w, as $FP_{M1}[w]$ and $FP_{M2}[w]$, the failure probability of the overall system can be computed using equation 2, where **K1** and **K2** are the scaling factors to account for the difference in densities and geometries of M1 and M2 (For example, 8 Gb high-density in-package vs. 1 Gb DRAM memory capacity per die, and different number of channels, rank, bank etc.). **K1** and **K2** is equal to one for M1 and M2 with the same geometry and density.

$$Access\,Rate = \frac{(Number\,of\,M1\,Accesses)}{(Number\,of M1 + M2\,Accesses)} \quad (1)$$

$$\begin{aligned} FP_{HMA}[w] = \mathbf{K1} \times (Access\,Rate \times FP_{M1}[w]) \\ + \mathbf{K2} \times (1 - Access\,Rate) \times FP_{M2}[w] \end{aligned} \quad (2)$$

3.2 Performance simulations

Placing more heavily accessed pages in-package memory improves performance at cost of reliability. For performance evaluation, we use DDR3 [10] and HBM [11] memory standards as off-package and in-package die-stacked memory, respectively. HBM and DDR3 memories have comparable latencies of 40ns and 45ns, respectively. However, HBM provides 2X-8X higher bandwidth compared to DDR memory, depending on the organization. Hence, using one memory over another has performance implications that vary by workload. In this subsection, we discuss our performance evaluation methodology using Ramulator [15].

Ramulator is a DRAM simulator providing cycle-accurate performance models for different memory standards such as DDR3/4, LPDDR3/4, GDDR5, and HBM. In trace-driven mode, the simulator takes an input trace file. The trace file has the number of intervening non-memory instructions, memory address, and request type (read or write) for every memory request. The current version of Ramulator can simulate only one level of memory. We extended Ramulator to simulate two levels of heterogeneous memory. We simulate a 16-core system with HBM as in-package memory and DDR3 as off package memory. The complete CPU and memory configurations are in Table 1. Memory traces are generated using PinPlay [14] and cache filtering is done using Moola [27].

We use three workloads from the SPEC benchmark suite [8] as shown in Table 2. We arranged all the SPEC benchmarks in an increasing order of their MPKI (Misses Per Kilo Instructions) and chose mcf, astar, and cactus, a benchmark from high, medium, and low MPKI, respectively. We fixed randomly selected pages in HBM memory and the rest of the pages in off-package DDR3 memory to vary the access rate to HBM. HBM memory of size 4GB is simulated to fit the entire working set of astar and cactus workloads into in-package memory and 16GB (HBM) for mcf. We used 16GB DDR3 memory as level 2 off-package memory.

Processor	Values
Number of cores	16
Core Frequency	3.2GHz
Issue width	4-wide out-of-order
ROB size	128 entries
Caches	**Values**
L1 I-cache (private)	32KB, 2-way set-associative
L1 D-cache (private)	16KB, 4-way set-associative
L2 cache (shared)	16MB, 16-way set-associative
HBM Memory	**Values**
Capacity	4GB, 16GB
Bus frequency	500Mhz (DDR 1.0GHz)
Bus width	128 bits
Channels	8
Rank	1 Rank per Channel
Banks	8 Banks per Rank
ECC	SEC-DED
DDR3 (x4) Memory	**Values**
Capacity	16 GB
Bus frequency	800 MHz (DDR 1.6 GHz)
Bus width	64 bits
Channels	2
Ranks	1 Rank per channel
Banks	8 Banks per Rank
ECC	single-ChipKill

Table 1: Ramulator Simulation Configurations.

4. EXPERIMENTAL RESULTS

4.1 Reliability results

The surface in Figure 2 shows the overall failure probability of an HMA system, FP_{HMA} as a function of access rate to M1 and time in weeks. Line 1 on the failure surface shows the variation of failure probability in a fixed week, and line 2 shows the variation of failure probability with a fixed access rate from week 0 to 348. The failure surface is constructed by moving line 1 and line 2 over week 0 to 348 and varying access rate from zero to one, respectively. The slope of line 1 increases as the system ages starting from week 0 to week 348, and the slope of line 2 increases with the increase in the access rate.

Failure probability increases with the increase in access rate because at the higher access rate the system makes more accesses to the less reliable M1 memory. Failure probability increases as the weeks progress because of accumulation of permanent faults. Faults accumulated over weeks 0 to n-1 make the system more vulnerable to encounter an uncorrectable error in week n. For access rate of one, FP_{HMA} increases at the fastest rate as the system ages and reaches the worst failure probability value of 0.0372 in week 348. The minimum failure probability is in week zero with an access rate of zero, 2.7×10^{-5}. Hence, when left uncontrolled,

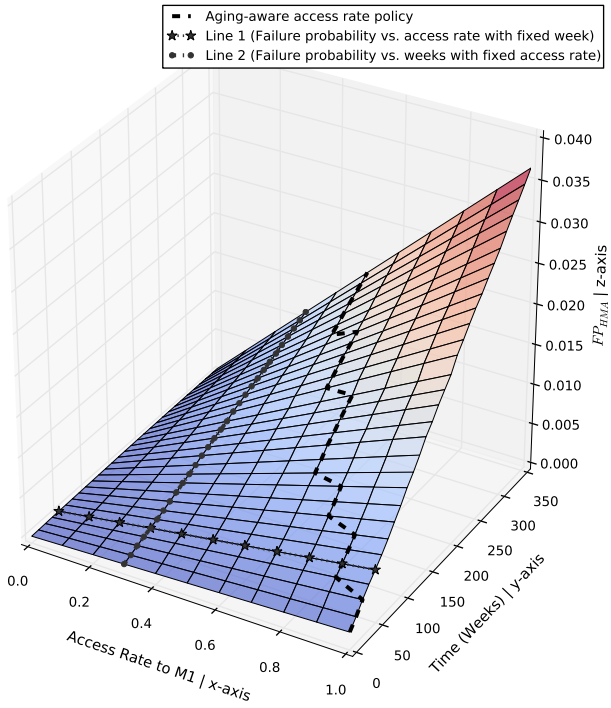

Figure 2: Failure probability of an HMA (FP_{HMA}) as function of access rate to M1 and time in weeks. FP_{HMA} varies from 2.7×10^{-5} to 0.0372, i.e. by 1377.77 times.

failure probability can increase by a factor of 1377.77x over the system's lifetime. However, using the result of our bound study we propose a system that reduces the access rate to M1 memory as it ages in order to keep the failure probability under acceptable limits, as shown by the aging-aware access rate policy. This policy is described in section 5.

4.2 Performance results

We observe an increase in performance as the access rate to HBM memory is increased from zero, as shown in Figure 3. The performance peaks at around 60% access rate and starts to drop as the access rate is increased further to 100%. Access rate of 100% is achieved by fixing the entire working set into HBM memory. With the entire working set in in-package (HBM) memory, the bandwidth provided by off-package (DDR3) memory is not utilized. The maximum performance is observed when the access to the heterogeneous memory system is distributed in a ratio of the bandwidth provided by off-package and in-package memory. Chou et al. showed similar results [3]. The access rate for the peak performance marginally drifts to the right as we move from mcf to astar, and from astar to cactus. The marginal drift of the peak performance point for lower bandwidth workloads can be explained by slightly lower latency HBM memory. A low bandwidth workload such as cactus can gain performance by slightly lower HBM latency (5ns difference) by moving more accesses to HBM over DDR3 memory than astar or mcf. We also observe that mcf's performance is the most sensitive to the access rate to HBM memory on either side of the peak performance point. Access rate of zero degrades mcf's performance by 51.3%, followed by 44.6%, and

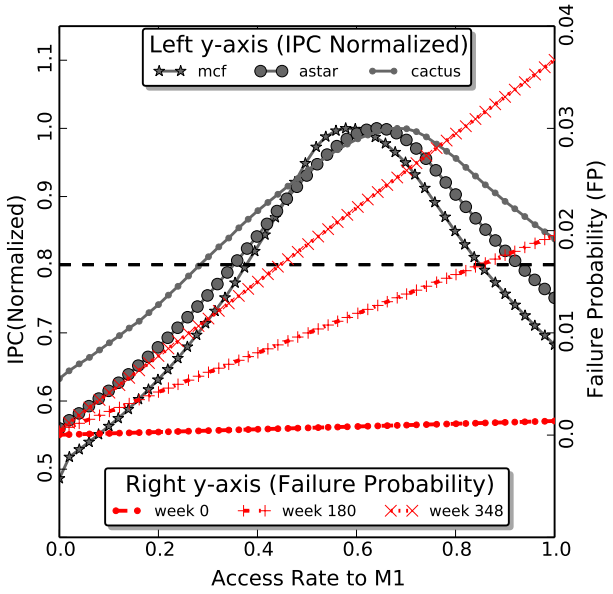

Figure 3: Left y-axis: Performance of mcf, astar, & cactus vs. access rate to M1. Right y-axis: FP = Failure Probability in week 0, 180 and 348 vs. access rate.

36.7% for high, medium and low bandwidth benchmarks, respectively.

4.3 Reliability vs. Performance Trade-off

The goal of this study is to evaluate the performance vs. reliability trade-off, as shown in Figure 3, where the left y-axis shows the normalized performance, and the right y-axis shows the failure probability in week 0, 180, and 348. In week 0, increasing access rate has very little effect on the failure probability. However, as the system ages and accumulates permanent memory faults, increasing access rate will adversely affect the failure probability. The performance cap of 80% IPC is shown by the dotted line in the figure. The average reduction in failure probabilities by running a workload at 80% of its peak IPC are 5.15×10^{-4}, 72.29×10^{-4}, and 137.98×10^{-4} in week 0, 180 and 348, respectively.

Our results show that a lower access rate will result in a greater reduction in failure probability in the mid and final week, than in the first week. In the first week memory faults accumulated by both memories (HBM and DDR3 memory) are corrected by SEC-DED for HBM and ChipKill for DDR3 memory. However, as weeks progress and memories age, SEC-DED loses its efficacy. The reduction in the failure probability (or gain in reliability) is 14x larger in the middle week and 26x larger in the final week compared to the first week.

5. AGING-AWARE ACCESS RATE CONTROL

In this section, we discuss the performance and reliability trade-off in more detail and present an aging-aware access rate control mechanism. We ran simulations with a cap on the failure probability and observed its effect on the performance of high, medium, and low bandwidth benchmarks over the system's lifetime. The results of the simulations are shown in Figure 4.

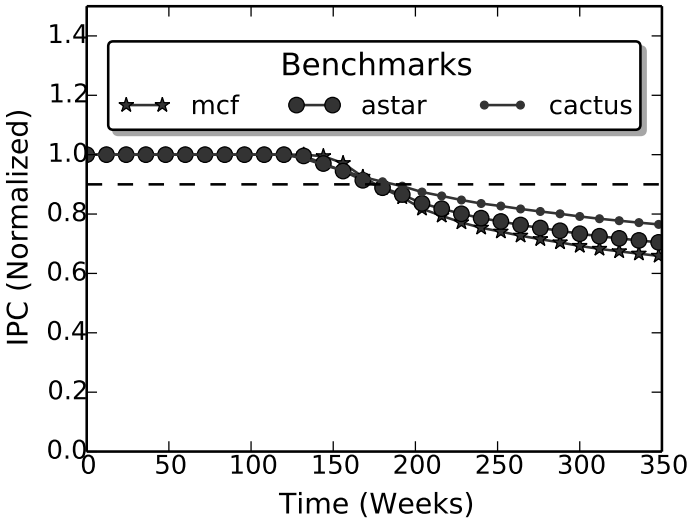

Figure 4: Aging-aware access rate control.

We ran simulations with a cap of 300x on degradation of the failure probability, i.e. the access rate control mechanism engages once the failure probability is 300x higher than the initial failure probability. The system is allowed to run at its peak performance up until the cap on the failure probability is reached. The week, T_1, when the system crosses the cap on the failure probability is recorded, as shown in Table 2. Different workloads reach T_1 in different weeks depending upon the access rate for their peak performance. For example, cactus reaches the failure probability cap in week 108, while mcf could run until week 132 without crossing the cap on the failure probability. After the cap on failure probability is reached, access rate is reduced every week by a fixed amount, by choosing a statically profiled random page placement shown in Figure 3. The performance degradation is observed, and the week in which the performance falls below 90% is recorded in Table 2 as T_2. Different workloads have different values of T_2. The value of T_2 depends on the slope of the IPC degradation vs. access rate. Figure 3 shows that cactus performance has a lower sensitivity to access rate on either side of the peak performance point. Hence, lowering the access rate every week takes longer for cactus to fall below 90% of peak performance. In the final week, mcf, astar, and cactus operate at 65.9%, 70.4%, and 76.5% of their peak performance, respectively.

6. DISCUSSION

In this section, we discuss our usage of failure rates in more detail, ways to handle environmental variation in FIT rates, and cost of uncorrectable errors & recovery.

6.1 DRAM FIT rates

We use memory FIT rates from the work by Sridharan and Liberty [31]. In the field study work, experiments are conducted on 2.69 million DRAM devices, and average FIT Rate for various DRAM components (bit, row, column, word, bank, rank) are reported over a period of 11 months, for both transient and permanent faults. We use these average FIT rates to find the failure probability over a period of 7 years. Hence, in our simulation, we take into account FIT rate variation across different DRAM components. However,

WL(x16)	MT(GB)	IPC(Max, Min)	IPC_d(%)	T_1	T_2
mcf	16.02	(2.51, 1.22)	51.3	132	176
astar	2.63	(8.68, 4.86)	44.6	120	174
cactus	2.31	(21.36, 13.52)	36.7	108	187

Table 2: Workload Characteristics. WL:WorkLoads, MT:Memory Touched, IPC_d: % IPC degradation in the final week, T_1: Week when failure probability hits 300x of the initial failure probability value, and T_2: Week when the IPC falls below 10% of the peak IPC.

assuming a constant FIT rate over the entire 7 years is an approximation in our simulations. Longer field studies show a modestly decreasing FIT for permanent faults over the 2-3 year time horizon [30]. In the 5-7 year horizon; however, we will see a modestly increasing rate of DRAM faults; we won't be in the steep part of the bathtub curve. We add margin to the FIT rates reported from the field to approximate the FIT rate over all 7 years for our research. Similar approximations are employed in FaultSim [21], MEMRES [26], and Citadel [20].

To the best of our knowledge, there is no field study data on die-stacked memory error rates in the field. As heterogeneous memory penetrates the commercial data centers, it will open the doors for large-scale studies similar to the DRAM field study, and real-world fault and error rates can be incorporated in studies like ours. As the number of options to protect the in-package memories increases, it will also create an opportunity to evaluate their effect by enhancing fault simulators with new protection mechanisms.

6.2 Environmental variations in FIT rates

The research in the field provides evidence that geographical factors can have a significant impact on FIT rates. The longer field study work also shows that a system located at an altitude of 7,320 feet is more susceptible to transient errors than a system at an altitude of 43 feet [30]. Kim et al. [16] exploited the higher susceptibility of DRAM rows to voltage fluctuations to expose a security vulnerability. Hence, estimation of T_1 in our algorithm in the field, in the presence of variations discussed above, will require hardware and software modifications. x86 provides machine-check architecture (MCA) [9] registers which can be used to log corrected error events. The operating system should periodically record the information from MCA registers into a log file. The log files can include a variety of information such as memory addresses exhibiting the errors, the time of the error, type of error (corrected or uncorrected), and ECC syndrome bits. The log file can be periodically analyzed by the aging-aware access rate control module to decide when to start reducing the access rate to in-package die-stacked memory. Similar HW/SW facilities are used in the field study of DRAM errors by Sridharan et al. [31] to record the number of corrected and uncorrected errors in various DRAM components.

6.3 Cost of uncorrectable error and recovery

In this paper, we discuss the trade-offs between performance and probability of failure of an uncorrectable error. The cost of an uncorrectable memory error is system downtime, component replacements, data loss or security vulner-

abilities. Hence, memory systems such as in Google data centers employ error correcting codes to recover from such errors [25]. However, recovering from an error also has an associated cost. The cost can be broken down into four major components: performance, area, power and dollar cost.

In order to recover from an error in memory, error detection must be triggered at every memory access, e.g. ECC must read an additional 8 bytes of ECC-bits per 64 bytes. Error detection has a marginal performance overhead as it can happen in parallel. However, there is a 12.5% area, bandwidth, and power overhead. There is also an extra cost of using an ECC DIMM, which is roughly 10-20% compared to non-ECC DIMM for DDR memory. The dollar cost of enabling recovery for 3D die-stacked memory is higher than off-package DDR memory [22].

7. CONCLUSION

In this paper, we have evaluated performance vs. reliability trade-off of the current heterogeneous memory architectures. We discuss the difference in reliability characteristics of die-stacked high bandwidth memory and off-package high capacity DDR memory. We also discuss the architectural challenges in enhancing the reliability of die-stacked memories. We argue that in the presence of architectural challenges to increase the reliability of the current die-stacked memories, system designers should consider aging-aware data placement policies. A data placement policy that only optimizes for performance will degrade the system's reliability over its lifetime. We propose and show the benefits of a policy that uses access rate as a knob to reduce the probability of the system failure to 137.98×10^{-4} in the final week i.e. 26x reduction when compared with the first week. We also present an aging-aware access rate control policy that decreases access rate to the lower reliability die-stacked memory as it age to ensure lower failure probability. Hence, moving from a performance focused to an aging-aware usage of HMA will result in a sustained and reliable operation in the system's final years.

We have described and discussed some of the pertinent challenges in using current fault simulation tools. Future work should fill some of the gaps required to study the reliability of heterogeneous memory architectures. In the absence of adaptive data placement policies and a die-stacked memory with uncorrectable DRAM errors will leave us with no option but to discard the die, even if it has a functional APU.

8. REFERENCES

[1] NVIDIA Pascal architecture. NVIDIA, 2016.

[2] Juan J. Serrano Abdallah M. Saleh and Janak H. Patel. Reliability of Scrubbing Recovery Techniques for Memory Systems. IEEE Transactions on Reliability, pages pp. 144–122, April, 1990.

[3] Moinuddin K. Qureshi Chiachen Chou, Aamer Jaleel. BATMAN: Maximizing Bandwidth Utilization of Hybrid Memory Systems. *Tech Report*, TR-CARET-2015-01(1):297–310, March 2015.

[4] Chiachen Chou, Aamer Jaleel, and Moinuddin K. Qureshi. CAMEO: A Two-Level Memory Organization with Capacity of Main Memory and Flexibility of Hardware-Managed Cache. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-47, pages 1–12, Washington, DC, USA, 2014. IEEE Computer Society.

[5] Mike O' Connor. HBMv1. NVIDIA. http://www.cs.utah.edu/thememoryforum/mike.pdf.

[6] T. J. Dell. A White Paper on the Benefits of Chipkill-correct ECC for PC server Main Memory. IBM, Technical Report, November, 1997.

[7] Sudhakar Govindavajhala and Andrew W. Appel. Using Memory Errors to Attack a Virtual Machine. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP '03, pages 154–, Washington, DC, USA, 2003. IEEE Computer Society.

[8] John L. Henning. SPEC CPU2006 Benchmark Descriptions. *SIGARCH Comput. Archit. News*, 34(4):1–17, September 2006.

[9] AMD Inc. AMD64 Architecture Programmer's Manual Revision 3.17. 2011.

[10] JEDEC. JESD79-3 DDR3 SDRAM Standard. June, 2007.

[11] JEDEC. JESD235 High Bandwidth Memory (HBM) DRAM. Oct, 2013.

[12] Hyeran Jeon, Gabriel H. Loh, and Murali Annavaram. Efficient RAS Support for Die-stacked DRAM. In *2014 International Test Conference, ITC 2014, Seattle, WA, USA, October 20-23, 2014*, pages 1–10, 2014.

[13] Xun Jian, Vilas Sridharan, and Rakesh Kumar. Parity Helix: Efficient Protection for Single-Dimensional Faults in Multi-dimensional Memory Systems. In *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*, HPCA '16, 2016.

[14] Chi keung Luk, Robert Cohn, Robert Muth, Harish Patil, Artur Klauser, Geoff Lowney, Steven Wallace, Vijay Janapa, and Reddi Kim Hazelwood. Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation. In *In PLDI '05: Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation*, pages 190–200. ACM Press, 2005.

[15] Y. Kim, W. Yang, and O. Mutlu. Ramulator: A Fast and Extensible DRAM Simulator. *IEEE Computer Architecture Letters*, PP(99):1–1, 2015.

[16] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In *Proceeding of the 41st Annual International Symposium on Computer Architecuture*, ISCA '14, pages 361–372, Piscataway, NJ, USA, 2014. IEEE Press.

[17] Chang Joo Lee, Veynu Narasiman, Onur Mutlu, and Yale N. Patt. Improving Memory Bank-level Parallelism in the Presence of Prefetching. In *Proceedings of the 42Nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 42, pages 327–336, New York, NY, USA, 2009. ACM.

[18] M. R. Meswani, S. Blagodurov, D. Roberts, J. Slice, M. Ignatowski, and G. H. Loh. Heterogeneous Memory Architectures: A HW/SW Approach for Mixing Die-stacked and Off-package Memories. In

High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on, pages 126–136, Feb 2015.

[19] Shubhendu S. Mukherjee, Joel Emer, Tryggve Fossum, and Steven K. Reinhardt. Cache Scrubbing in Microprocessors: Myth or Necessity? In *Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'04)*, PRDC '04, pages 37–42, Washington, DC, USA, 2004. IEEE Computer Society.

[20] Prashant J. Nair, David A. Roberts, and Moinuddin K. Qureshi. Citadel: Efficiently Protecting Stacked Memory from Large Granularity Failures. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-47, pages 51–62, Washington, DC, USA, 2014. IEEE Computer Society.

[21] Prashant J. Nair, David A. Roberts, and Moinuddin K. Qureshi. FaultSim: A Fast, Configurable Memory-Reliability Simulator for Conventional and 3D-Stacked Systems. *ACM Trans. Archit. Code Optim.*, 12(4):44:1–44:24, December 2015.

[22] J. T. Pawlowski. Hybrid Memory Cube: Breakthrough DRAM Performance with a Fundamentally Re-Architected DRAM Subsystem. Hot Chips, 2011.

[23] Jee Ho Ryoo, Karthik Ganesan, Yao-Min Chen, and Lizy Kurian John. i-MIRROR: A Software Managed Die-stacked DRAM-Based Memory Subsystem. In *27th International Symposium on Computer Architecture and High Performance Computing, SBAC-PAD 2015, Florianópolis, Brazil, October 17-21, 2015*, pages 82–89, 2015.

[24] Bianca Schroeder and Garth A. Gibson. A Large-scale Study of Failures in High-performance Computing Systems. In *Proceedings of the International Conference on Dependable Systems and Networks*, DSN '06, pages 249–258, Washington, DC, USA, 2006. IEEE Computer Society.

[25] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. DRAM Errors in the Wild: A Large-scale Field Study. In *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '09, pages 193–204, New York, NY, USA, 2009. ACM.

[26] Hongzhong Zheng Shaodi Wang, Henry Hu and Puneet Gupta. MEMRES: A Fast Memory System Reliability Simulator. In *SELSE-15: The 15th Workshop on Silicon Errors in Logic - System Effects*, 2015.

[27] Charles F. Shelor and Krishna M. Kavi. Moola: Multicore Cache Simulator. In *30th International Conference on Computers and Their Applications CATA-2015*, 2015.

[28] Jaewoong Sim, Alaa R. Alameldeen, Zeshan Chishti, Chris Wilkerson, and Hyesoon Kim. Transparent Hardware Management of Stacked DRAM As Part of Memory. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-47, pages 13–24, Washington, DC, USA, 2014. IEEE Computer Society.

[29] Jaewoong Sim, Gabriel H. Loh, Vilas Sridharan, and Mike O'Connor. Resilient Die-stacked DRAM Caches. In *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ISCA '13, pages 416–427, New York, NY, USA, 2013. ACM.

[30] Vilas Sridharan, Nathan DeBardeleben, Sean Blanchard, Kurt B. Ferreira, Jon Stearley, John Shalf, and Sudhanva Gurumurthi. Memory Errors in Modern Systems: The Good, The Bad, and The Ugly. *SIGARCH Comput. Archit. News*, 43(1):297–310, March 2015.

[31] Vilas Sridharan and Dean Liberty. A Study of DRAM Failures in the Field. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 76:1–76:11, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[32] ChunYi Su, David Roberts, Edgar A. León, Kirk W. Cameron, Bronis R. de Supinski, Gabriel H. Loh, and Dimitrios S. Nikolopoulos. HpMC: An Energy-aware Management System of Multi-level Memory Architectures. In *Proceedings of the 2015 International Symposium on Memory Systems*, MEMSYS '15, pages 167–178, New York, NY, USA, 2015. ACM.

[33] Jun Xu, Shuo Chen, Zbigniew Kalbarczyk, and Ravishankar K. Iyer. An Experimental Study of Security Vulnerabilities Caused by Errors. In *2001 International Conference on Dependable Systems and Networks (DSN 2001) (formerly: FTCS), 1-4 July 2001, Göteborg, Sweden, Proceedings*, pages 421–432, 2001.