Understanding and Alleviating Intra-Die and Intra-DIMM Parameter Variation in the Memory System

Meysam Taassori Ali Shafiee Rajeev Balasubramonian
University of Utah, UT, USA
{taassori,shafiee,rajeev}@cs.utah.edu

***Abstract*—Continued process scaling must overcome several manufacturing challenges. At the same time, industry is exploring many new memory technologies that require new manufacturing processes. In such challenging fabrication regimes, parameter variation (PV) and yield will be important problems. While many recent bodies of work have targeted PV in processors, few have targeted PV in the memory system. Mitigation techniques have either focused on refresh, or have focused on inter-die variation. In this work, with empirical measurements, we first show that PV and specifically intra-die PV is indeed a real phenomenon in modern DRAM chips. We show that this intra-die PV can impact timing parameters for different banks within a DRAM chip. In response to growing PV, memory timing parameters will likely be set very conservatively to accommodate the worst case. To overcome these worst-case limitations, we propose the design of a reconfigurable memory module that detects PV in the field and organizes the memory system into fast/slow regions. This requires changes to the memory controller and to buffer chips on DIMMs. Further, OS migration policies can move frequently accessed pages to the fast regions. This overall approach not only improves performance and energy, it also provides a configurable platform for systems that can tolerate errors or approximation. The proposed system yields an average performance improvement of 12.6% in DRAM systems, and 25.5% in NVM systems.**

I. Introduction

As logic and DRAM processes shrink to smaller dimensions, parameter variation (PV) emerges as a growing concern [27], [21]. This has already been well analyzed for processors and SRAM caches, e.g., [11], [35], [24], [3]. Only a few studies have examined PV in DRAM main memory [6], [40], [26], [26], [14], [17], [23], [12], [39]. Most of these studies typically focus on variations in DRAM cell retention times and refresh rates. However, PV can manifest itself in many other ways on a memory chip, ultimately impacting a variety of DRAM timing parameters. A Hynix study in 2011 shows that circuit delays on a DRAM wafer can vary by 30%, and the variation grows as voltages are reduced [21].

PV is also expected to be a significant problem in emerging memory technologies (PCM, memristors, STT-RAM, 3D stacks), especially in the early years when manufacturing processes are not mature [29].

The existence of PV in memory (and processor) chips is well known. The first step in dealing with PV is to bin dies into different classes based on their ability to pass various performance tests. Some of these classes offer higher performance and are sold at higher prices. Such binning takes care of a large degree of inter-die parameter variations.

This work was supported in parts by NSF grants 1302663 and 1423583.

However, binning does not address *intra-die parameter variation*. A given chip will inevitably have circuits with identical functionality that exhibit faster and slower speeds. The parameters for a given chip are ultimately determined by the worst-performing circuits. Modern DDR standards describe a number of timing parameters and it is assumed that those parameters will apply uniformly to every access from a given rank. With a growing interest in re-thinking the memory interface [30], and with an increase in PV, we believe that there is merit in identifying good/bad regions of memory and exposing their varying parameters to a controller on the memory package or on the processor. The term "good/bad" may describe speed, power, or error rates. Exploiting these regions will require smarter OS policies that place frequently accessed pages in faster or energy-efficient regions, or approximate memory systems that can place data in precise or error-prone regions based on criticality. Note that there is already a push to explicitly design parts of memory with different figures of merit for this exact purpose [20], [5], [32] – we show here that growing levels of PV will offer us non-uniform memory systems anyway and will require new solutions.

We first carry out an empirical evaluation of DRAM chips with a modified FPGA memory controller to analyze the nature of intra-die PV in modern DRAM chips. Our analysis shows that each bank within a DRAM chip (a *sub-bank*) indeed has varying error rates as DRAM timing parameters are made more aggressive. We therefore propose re-organizing how data is laid out in a rank or in a DIMM. Fast sub-banks in various chips are clubbed together to form a fast bank in a rank. The necessary logic for this reconfiguration is embedded in the buffer chip on a DIMM. The memory controller tracks different timing parameters for different banks. The OS monitors page activities to preferentially place hot pages in low-latency regions. In this work, we only focus on leveraging PV to reduce average memory latency; analyses of energy and approximate computing are left for future work.

II. Background

Parameter Variation

It is well-known that parameter variation grows as device dimensions shrink [27]. Parameter variations in logic and DRAM processes are typically attributed to changes in the effective gate length that are caused by systematic lithographic aberrations, and changes in threshold voltage that are caused by random doping fluctuations [27], [40], [6]. Most prior works on DRAM parameter variation have focused on how it impacts cell retention time [26], [14], [17], [23].

The work of Zhao et al. [40] develops a parameter variation model for 3D-stacked DRAM chips and quantifies the expected variation in leakage and latency in different banks. They show that parameter variation can cause most banks to have data read latencies that fall in the range of 12-26 cycles. The authors then propose a non-uniform latency 3D-stacked DRAM cache. With the variation profile described above, a commodity DRAM chip today would have to specify a conservative uniform latency of 30 cycles even though several requests can be serviced in half the time. Another study with an older DRAM process shows that latencies in different banks can vary by 18 ns because of parameter variation [22]. A more recent paper from Hynix shows that for 450 DRAM samples, the delay variation for circuits within a single wafer is almost 30% [21]. The delay variation grows as voltages shrink, implying that future technologies will likely see more variations. Emerging NVMs will also suffer from high PV, as shown for memristors by Niu et al. [29].

The above data points argue for more intelligent memory controllers that can automatically detect and exploit variations in DRAM timing parameters. In addition to the manufacturing variations described above, there are other sources of predictable and unpredictable variations at runtime because of the operating environment. Timing parameters vary predictably as temperature changes. The timing for some operations may vary unpredictably because of voltage supply noise. Some voltage fluctuations are predictable, either caused by static IR-drop [34] or dynamic LdI/dt [16]. The timing for every operation can also vary based on other simultaneous activities on the DRAM chip because some critical resources are shared (most notably, charge pumps and the power delivery network [34]).

Over-Clocking

Gaming enthusiasts frequently resort to processor and memory "over-clocking" [28], [13], [36]. Processor and memory vendors expose a few parameters that can be set at boot-up time in the BIOS to allow a system to operate at frequencies higher than those in the specifications. For example, memory over-clocking today allows a change to the memory bus frequency, the DIMM voltage, and three DRAM timing parameters (tRP, tRCD, tCL) [36]. This is an effective coarse-grained approach to shrink timing margins and boost performance, while trading off some reliability. However, because the same timing parameters are applied to every circuit in the rank, over-clocking does not alleviate intra-die parameter variation, or even intra-rank parameter variation.

Recent Work on Timing Parameter Adjustments

The work of Liu et al. [26], [25] is the most comprehensive characterization of variable retention rates in DRAM cells. To alleviate this problem, Liu et al. propose adaptive policies that can apply non-uniform refresh rates to different memory regions. This is an example of a policy that alleviates intra-die PV, but is limited to the handling of refresh.

Two other recent works, by Chandrasekar et al. [8] and by Lee et al. [19], develop memory controllers that can modify timing parameters at run-time for entire ranks, while keeping errors in check. Chandrasekar et al. exploit the fact that DRAM chips have generous margins built into their timing parameters. Shaving these margins generally implies that the user can expect to see a higher error rate (that may or may not be tolerable). Lee et al. exploit the fact that timing parameters depend on temperature; so at lower temperatures, DRAMs can shave some of the timing margins without experiencing a higher error rate. Both of these bodies of work do not exploit intra-die PV. To a large extent, they also do not exploit inter-die PV since the same timing parameters are applied uniformly across the memory system. The motivational argument in these papers is that as a result of PV, timing parameters are dictated by worst-case circuits and worst-case environmental conditions, whereas in practice, most circuits can shave off their timing parameters and still work correctly.

We note that inter-die PV or even overly generous timing margins can be partially handled with binning. Once binning is performed, intra-die PV emerges as a primary problem. These prior approaches cannot address intra-die PV (apart from variable refresh [26]). Our work attempts to fill this gap by designing mechanisms that can exploit timing parameter variations within dies. We build on recent work by Zhang et al. [39] that attempts to address intra-die PV; a comparison to their work is discussed in Section VII.

III. Analyzing Parameter Variation

Intra-die variation includes systematic and random variations. The former is because of lithographic tools and exhibits spatial correlation. Random variation is caused by material fluctuations and manifests at finer granularity. Previous works [33], [9] have modeled both systematic and random variations as normal distributions.

Measurement Methodology

We analyze a number of DRAM chips to identify the extent of intra-die parameter variation. In order to vary DRAM timing parameters and measure error rates, we implement a customized memory controller on a Xilinx Virtex-7 FPGA VC707 evaluation kit [37] that has an external SO-DIMM. Because of FPGA system constraints, we were only able to take measurements on 1GB 1-rank SO-DIMMs. Measurements were taken on 30 SO-DIMMs, from three different memory vendors, and operating at 3 different speed grades (800, 667, and 533 MHz). Each SO-DIMM is comprised of eight DDR3 DRAM chips, with each chip being partitioned into eight sub-banks.

A single measurement involves the following. The memory controller on the FPGA is configured to use a certain set of timing parameters. We typically set all timing parameters to the default for that rank, except one parameter, say $tRCD$, which is made more aggressive. We then feed the memory controller with requests from a custom traffic generator. We first write a fixed pattern into the entire memory – this pattern includes all zeroes, all ones, and storing the cache line's address in the cache line itself (an easy way to generate a mix of 0s and 1s that can be easily verified without separate large metadata). We then sequentially read the contents of the entire memory, verify the result at the memory controller, and record any observed errors.

For space reasons, we only discuss our analysis of the $tRCD$ timing parameter. We observe similar trends for other timing parameters as well. The default $tRCD$ for the DRAM chips being studied is 13 ns. Because of the constraints of our

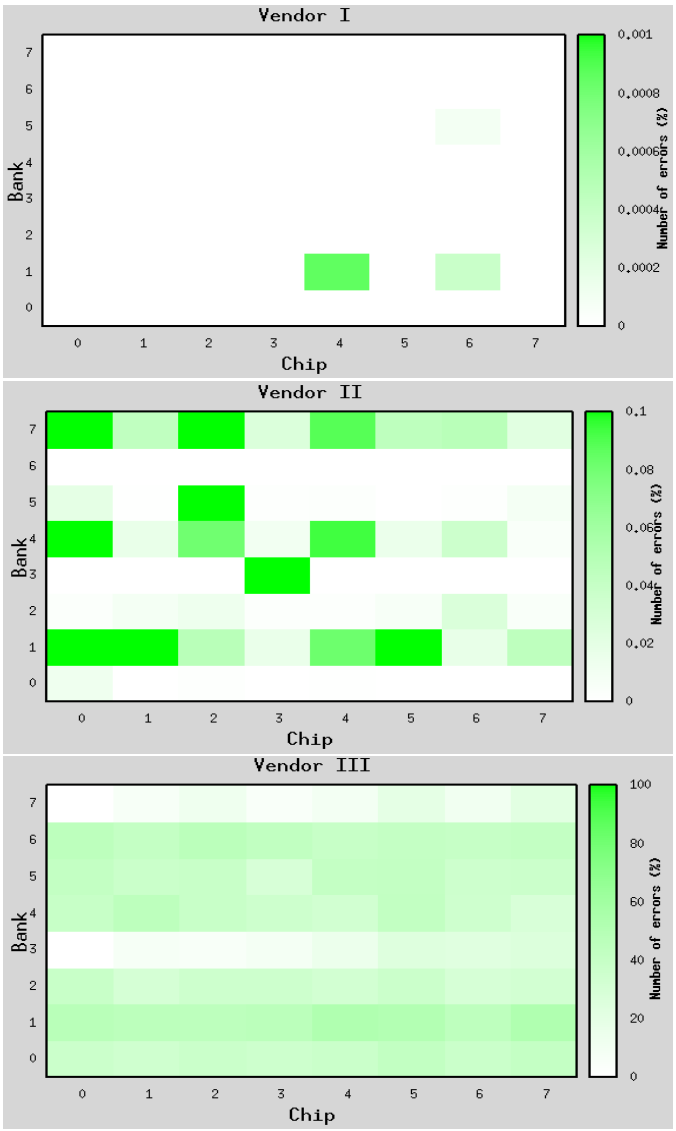

Fig. 1. Representative error rates for three SO-DIMMs from three different DRAM vendors, when operating with tRCD of 8.75 ns.

FPGA system, we were only able to test aggressive $tRCD$ values of 8.75 ns and 3.75 ns. Figure 1 shows a representative error profile for each of the three different memory vendors, when $tRCD$ is set to 8.75 ns. The figure is in the form of an error rate "heatmap" across the 64 sub-banks on each SO-DIMM. Errors are reported for every 8-bit unit read from a sub-bank in a memory half-cycle. The X-axis is organized by chips, and the Y-axis is organized by sub-banks.

Intra-Die Variation per Vendor

Figure 1 shows that for one of the vendors (Vendor III), very high error rates are observed for all sub-banks when operating at a $tRCD$ of 8.75 ns. This was also observed in other SO-DIMMs from that same vendor, indicating that those chips exhibit lower PV and lower margins than other chips in our analysis. It may be possible that these chips do exhibit PV, but that may only be evident at say $tRCD$ of 10 ns, which we were unable to test.

For the other two vendors, we observe that many of the sub-banks are able to operate with zero errors even though $tRCD$ has been lowered by 30%. A few sub-banks show erroneous behavior and can clearly not tolerate the aggressive $tRCD$. A single chip typically has both erroneous and error-free sub-banks, thus exhibiting intra-die variation.

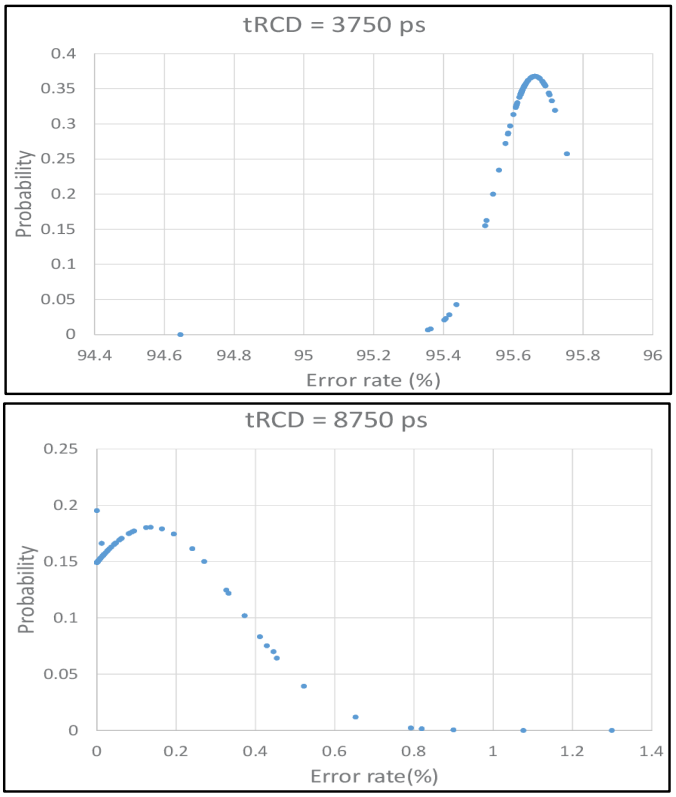

Fig. 2. A distribution of the number of sub-banks at each error rate for $tRCD$ of 3750 ps and 8750 ps.

When $tRCD$ is made 3.75 ns (not shown in the figures), all sub-banks have very high error rates. We also observed that the error profiles were not very sensitive to the speed grade, or the data pattern.

Distribution of Sub-Bank Latencies

Figure 2 takes all of our sub-bank measurements (640 in total) for Vendor I and plots the error distribution. The X-axis indicates the error rate for a sub-bank and the Y-axis shows the number of sub-banks with that error rate. This result shows that the goodness of a given sub-bank follows a Gaussian distribution and this Gaussian distribution shifts to the left as a timing parameter is made more conservative.

From this, it follows that the distribution of the lowest correct timing parameter for each sub-bank is also Gaussian (shown at the top of Figure 3). Chandrasekar et al. [8] also demonstrate this Gaussian distribution with SPICE simulations of DRAM circuits that suffer from PV.

Since we were unable to empirically measure error rates for a wide range of timing parameters, we have to estimate the values of μ and σ for the Gaussian distribution. As shown at the top of Figure 3, we assume that the $\mu + 3\sigma$ of this distribution is at the specified DRAM timing parameter of 13 ns for $tRCD$. Based on the reports [8], [21] that estimate typical variation of 30%, we assume that the distribution's $\mu - 3\sigma$ is at a $tRCD$ value that is 30% lower. Solving these

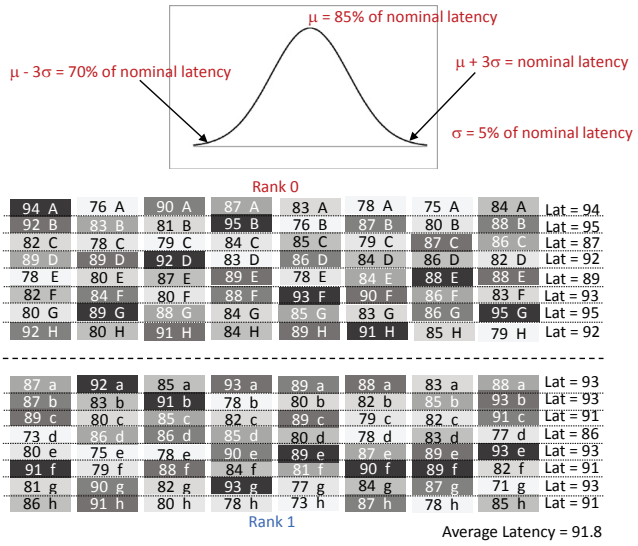

Fig. 3. Organizations A and B.

two equations yields the μ and σ values for the distribution we need, i.e., μ is $0.85 \times tRCD$ and σ is $0.05 \times tRCD$. For our subsequent analysis, we are going to assume that individual sub-banks on a DIMM have timing parameters that follow this Gaussian distribution. For example, Figure 3 shows an example DIMM with two ranks and 128 sub-banks, where the latency for each sub-bank has been drawn from the above normal distribution. Latency has been expressed as a number between 70% and 100%, so a sub-bank with latency 80% has timing parameters that are 20% lower than that of the worst-case timing parameters specified for that DRAM product (referred to as the *nominal latency*).

IV. Re-Organizing Memory

Having shown that intra-die PV manifests itself in commercial memory chips, and having established a Gaussian distribution for the individual speeds of sub-banks on a DIMM, we now develop solutions to alleviate the effects of intra-die PV.

Baseline DIMM – Organization A

First, consider how a modern DIMM is configured. As a running example in this section, we use the DIMM that was constructed in Figure 3 by taking random samples following our Gaussian distribution. Just as processors are placed in different frequency bins after testing, DRAM chips can also be placed in different speed bins. The selected speed bin for a DRAM chip is determined by the slowest sub-bank or region on the DRAM chip. While creating a DIMM, vendors use DRAM chips from the same speed bin. So all the chips on a DIMM have similar worst-case timing parameters, i.e., they all have at least one sub-bank that must operate near the worst-case timing parameter under worst-case conditions[1]. Indeed, this is what we observe in our example DIMM in Figure 3, where the worst sub-bank on every chip has a latency that is between 88-95% of the nominal latency. A modern memory controller uses the nominal latency uniformly across the entire rank. A memory controller like that proposed by Chandrasekar et al. [8] would apply the worst-case observed latency of 95% uniformly across the entire DIMM (assuming that binning has removed any generous timing margins and that we are operating in worst-case environments). We refer to such a memory controller as *Organization A*.

[1]The ability to operate at even lower latencies at lower temperatures, as proposed by Lee et al. [19] is an orthogonal technique that can be applied on top of our technique.

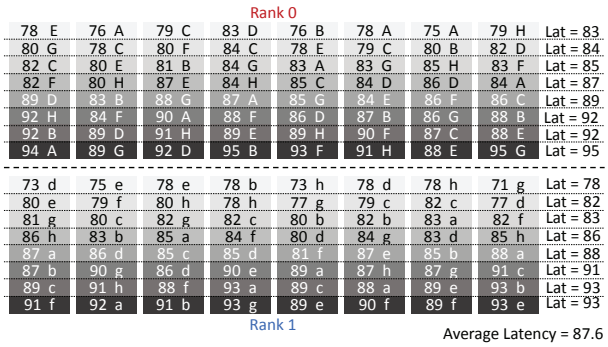

Fig. 4. Organization BR.

Variable Bank Latencies – Organization B

Now, we consider designs where the memory controller can use different and independent timing parameters for each bank. If each bank in the example DIMM in Figure 3 has its own latency, determined by the worst-case sub-bank in that bank, the bank latencies can again be a little lower than the nominal latencies. Figure 3 shows the per-bank latencies for such a DIMM and memory controller where each bank has independent timing parameters. The per-bank latencies in our example DIMM range from 86-95% of the nominal latency. We refer to such a memory controller as *Organization B*. If we assume that memory requests are uniformly distributed across all banks, the average memory access latency for requests to this example DIMM is 91.8% of the nominal latency. Because the worst-case sub-banks on the DIMM are only affecting their own bank, some of the other banks can execute with lower latencies. Therefore, this organization is able to yield lower latencies than Organization A.

Ganging Fast Sub-Banks – Organization BR

Next, we consider re-organization of ranks and banks so that low-latency sub-banks can be ganged together to form a low-latency bank. We first consider *bank re-organization*, shown in Figure 4. We refer to this as *Organization BR*. In BR, the sub-banks in each DRAM chip are re-numbered so that the sub-banks are sorted by latency – the lowest-latency sub-bank is numbered as sub-bank 0 and the highest-latency sub-bank is numbered as sub-bank 7. This is evident in Figure 4, where the slower banks with darker shading have moved to the bottom (logically speaking). With this sorting and re-numbering of sub-banks within a chip, we are able to gang similar sub-banks to form a bank. This yields low-latency bank 0's with latencies 83% and 78% of nominal latency for our example DIMM. Again, assuming a uniform distribution of memory traffic across banks, we see an average latency that is 87.6% of nominal latency.

Implementing BR – DRAM Chip Modification

In order to re-organize sub-banks in this manner, we need a mechanism that can re-number sub-banks within a chip.

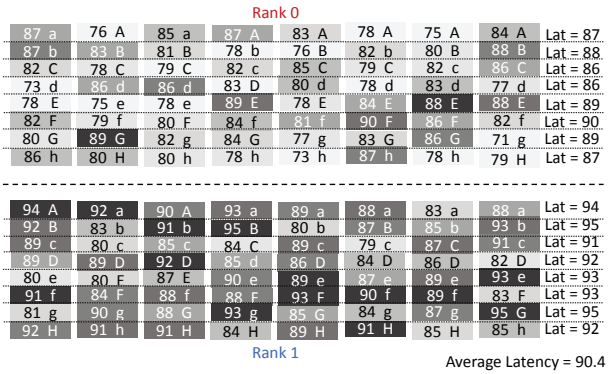

Fig. 5. Organization RR.

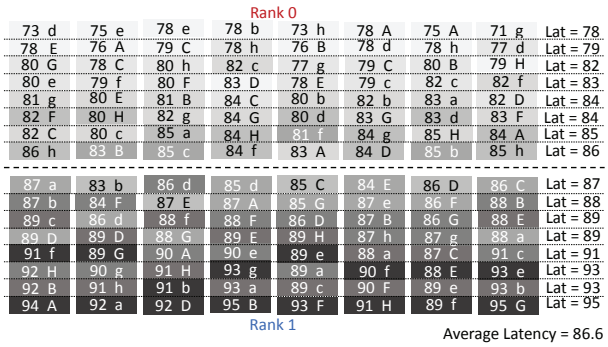

Fig. 6. Organization RBR.

There are two possible ways to do this. The first approach implements a simple 8-entry permutation table on the DRAM chip itself. When the RAS signal is received on a DRAM chip, the three bits that indicate the sub-bank are changed to a different set of three bits based on the permutation table. The permutation table can be implemented with a total of 24 bits, and allows sub-banks to be arbitrarily permuted. The permutation table can be updated based on the memory controller's characterization of the speeds of individual sub-banks.

Buffered DIMM Basics

The second approach is to perform the permutation on a buffer chip on the DIMM. This can be implemented in buffered (LRDIMM [1]) or registered DIMMs (RDIMM [2]), which are both produced in high volume. When using RDIMMs, the address/command bus connects to a register chip on the RDIMM to reduce load on the bus. The register chip then produces new address/command signals on the RDIMM that drive every DRAM chip on the RDIMM. The buffered DIMM (LRDIMM) is the next step in this evolution. In an LRDIMM, a buffer chip on the DIMM serves as the interface for both address/command and data signals. It therefore also reduces the load on the data wires, enabling a large number of ranks on a single memory channel. Newer LRDIMMs implement multiple distributed buffer chips for data signals to reduce wiring overheads.

Implementing BR – Buffer Chip Modification

With RDIMMs or LRDIMMs, new logic can be introduced on the register or buffer chip to re-organize the DRAM chips into new bank configurations. When the memory controller makes a request for bank-0, the register or buffer chip looks up a permutation table and translates the bank-id bits into the appropriate set of three sub-bank-id bits for each chip in the rank. Typically, the bank-id bits are broadcast to all chips in the rank on a shared bus. But now, each chip would need a dedicated set of three wires on the DIMM that carry the sub-bank-id bits for that chip. The permutation table has a 3-bit entry for each of the 128 sub-banks in our example DIMM. The permutation table is a trivial overhead in either approach. The second approach introduces the overhead of three additional on-DIMM wires and register/buffer pins for every chip on the DIMM; but it has the advantage of using unmodified DRAM chips.

Sub-Banks in Fast/Slow Ranks – Organization RR

We next consider *rank re-organization*. This is depicted in Figure 5 as *Organization RR*. In RDIMMs, bit 0 of the data bus is connected directly to the first data pin in the first chip of both ranks of our example DIMM. In newer LR-DIMMs too, specific pins of the data bus are connected to specific pins of two DRAM chips, but through a small data buffer chip. When accessing Rank 0, the chip-selects for the chips in Rank 0 are activated. In essence, DRAM chips are paired and the correct chip from that pair is selected with an appropriate chip-select signal. With RR, we re-organize the chips that make up a rank. This re-organization can be different for different banks. Figure 5 shows how each sub-bank is mapped to either a fast "blue" rank or a slow "red" rank. When accessing bank 0 in fast rank 0, we are picking the faster sub-bank 0 from every pair of DRAM chips. When accessing bank 1 in fast rank 0, we are again picking the faster sub-bank 1 from every pair of DRAM chips. Depending on the bank being accessed, a different set of chips will comprise the rank. With this mechanism, we see an average latency that is 90.4% of nominal latency.

Implementing RR

To implement RR, each chip on the DIMM needs a dedicated chip-select signal. Modern RDIMMs and LRDIMMs have a dedicated chip select bus for each rank. The register or buffer chip will also need a 64-bit permutation table to indicate the chips that comprise a rank for each bank. If we have the option to modify the DRAM chip, this can be implemented with a simple 8-bit permutation table per chip in this example – depending on the bank being accessed, this table tells the chip if it should participate on chip-select high or low. One negative side-effect of this approach is that the memory controller has to enforce tFAW and tRRD constraints for every chip on the DIMM instead of for every rank on the DIMM.

Combining BR and RR – Organization RBR

Finally, we combine both of the above approaches to create *rank and bank re-organization* or *Organization RBR* in Figure 6. This approach first permutes the sub-banks in a chip so they are ordered from fast (sub-bank 0) to slow (sub-bank 7). When accessing bank 0 in the faster rank 0, it also then picks the faster sub-bank 0 from each pair of chips. With this approach, the average latency is 86.6% of nominal latency. The overheads are the sum of overheads of the previous two approaches, i.e., you either need a 32-bit permutation table

on the DRAM chip, or a 448-bit permutation table on the register/buffer chip along with 4 additional wires per chip.

Combining with Mini-Ranks

Finally, we observe that DIMMs that suffer from PV can also benefit from the concept of mini-ranks [41], [4]. In our designs so far, a single high-latency sub-bank has an impact on seven other sub-banks that must gang together to form a rank. However, if ranks are comprised of four chips, the impact of a high-latency sub-bank is limited to only three other sub-banks. Mini ranks have been shown to be beneficial in general because they increase parallelism in the memory system and reduce activation energy. Their favorable impact on PV management is another argument to move towards mini ranks in future systems.

OS Support for Page Migration

With the proposed DIMM re-organization techniques, we see a wide gap between the latencies of fast and slow banks. Instead of uniformly scattering working sets across all banks, OS policies can identify frequently accessed pages and move them to low-latency banks to further improve performance. Several recent studies [7], [34], [40], [20], [5] have introduced NUMA behavior within the memory system and relied on such OS policies to exploit low-latency memory regions.

Building on these prior works, we design a page migration policy tailored to our variable-latency memory system. Similar to prior policies, the OS must first track activities to individual pages over an epoch; this requires a table of counters at the memory controller. Once the epoch has finished, these statistics are analyzed. Up to N most highly accessed pages are moved from high-latency banks to low-latency banks. The latency of banks are determined not by their timing parameters, but the average observed latency in the last epoch. This ensures that too many pages are not moved into some banks, causing them to suffer from long queuing delays. The value of N and the epoch length are chosen to ensure that average migration penalties are much less than 1% of overall execution time (in our simulations, we model N, epoch length, and per-page migration latency of 64 pages, 15M cycles, and 10K cycles).

In addition to tracking hot pages and moving them to low-latency banks, the OS is also responsible for periodic diagnostics that can characterize the latency of each sub-bank. In systems with ECC protection, the error rates can be used to detect if the characterization has changed. In practice, one chip may yield errors at first, and these can typically be corrected with SECDED or chipkill.

V. Methodology

To evaluate the impact of our re-organized DIMMs, we perform cycle-accurate simulations with the USIMM memory timing model [10]. Memory access traces for 12 workloads are fed to USIMM's detailed memory timing model. Memory access traces are collected for 16 threads and filtered by a 4MB shared cache. The eventual trace for each workload has 250K memory accesses. We have modified USIMM to model per-bank timing parameters that are derived from a configuration file that describes each DIMM in our study. We model a single channel supporting 2 ranks and 16 cores to represent a future throughput-oriented system.

Table I describes our USIMM parameters. The timing parameters listed here are the nominal baseline timing parameters. In most of our analysis, we assume that per-sub-bank timing parameters follow the normal distribution described in Figure 3, with a 30% gap between the fastest and slowest sub-banks. We assume that timing parameters tRCD, tRP, tRAS, tWR, tFAW, and tRRD are varied in tandem, i.e., each is made F% faster than the nominal timing parameter, where F varies from 0 to 30 in most experiments. We also show a sensitivity study where F varies from 0 to 40, representing future memory systems that are even more plagued by PV. The systems are evaluated on memory-intensive workloads from SPEC2k6, PARSEC, and NAS parallel benchmark suites.

DRAM Parameters	
DRAM Frequency	1600 Mbps
Channels, ranks, banks	1 channel, 2 ranks/channel, 8 banks/rank
Write queue water marks	40 (high) and 20 (low), for each channel
Read Q Length	32 per channel
DRAM Timing Parameters (DRAM cycles)	t_{RC} = 39, t_{RCD} = 11 t_{RAS} = 28, t_{FAW} = 20 t_{WR} = 12, t_{RP} = 11 t_{RTRS} = 2, t_{CAS} = 11 t_{RTP} = 6, t_{DATA_TRANS} = 4 t_{CCD} = 4, t_{WTR} = 6, t_{RRD} = 5 t_{REFI} = 7.8μs, t_{RFC} = 640 ns

TABLE I
DRAM TIMING [15] PARAMETERS.

In addition to testing our approaches on a suite of different workloads, we also have to test our approaches on a suite of (simulated) DIMMs. We therefore construct a suite of 10 DIMMs, where the timing parameters for each sub-bank are drawn from a normal distribution. Each of these 10 DIMMs will exhibit different improvements with bank and rank re-organization depending on how the sub-bank latencies are scattered on the various memory chips. One of these 10 DIMMs is shown as an example in Figure 3.

VI. Results

A. DRAM

Monte-Carlo Simulations

We first carry out Monte-Carlo simulations to understand the impact of various rank/bank re-organization policies on bank latencies. Each data point in our simulation constructs a DIMM by drawing sub-bank timing parameters from a normal distribution. We draw 1350 random DIMM samples in these experiments and compute a distribution of bank latencies with each approach.

Figure 7 shows the cumulative distribution of bank speeds with different re-organization schemes. The distribution shifts further to the left as the re-organization scheme is provided more flexibility. In the baseline, most banks have a latency of 97% of nominal or higher. With the RBR schemes, more than 40% of banks are at least 10% faster than the nominal latency.

Room for Improvement

Figure 8 shows the performance improvement as various timing parameters are improved by 30% for all banks. We consider each timing parameter in isolation, as well as the combination of all timing parameters. This graph shows the relative importance of each timing parameter (tRAS and tRP are most important), as well as an upper bound on the performance improvement (17.5%) with PV-aware bank latencies.

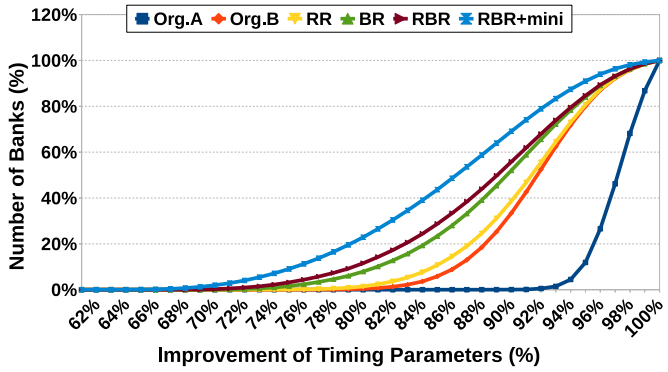

Fig. 7. Cumulative distribution of bank speeds for different bank reorganization methods. The X axis shows the latency of a sub-bank as a percentage of the nominal latency; the Y axis represents the percentage of sub-banks that are faster than that speed.

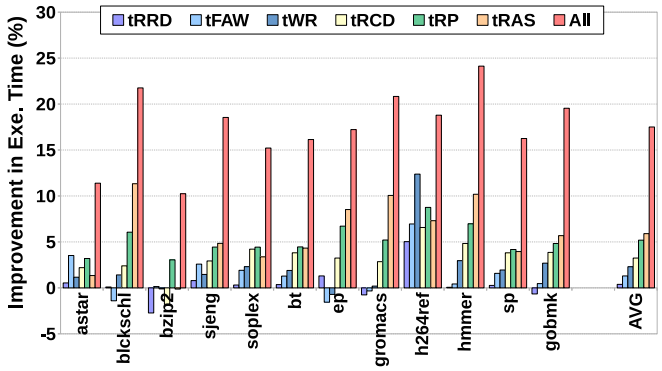

Fig. 8. Effect of reduced tRAS, tWR, tRCD, tRP, tFAW, tRRD, and All on execution time. Each parameter is reduced by 30%.

Improvements for the Suite of DIMMs

Figure 9 shows the performance improvement (averaged across the suite of 10 sample DIMMs) for each of our benchmark programs for each re-organization approach. We see that the RBR policy yields an average performance improvement of 5.8% across all benchmarks. The contribution of bank re-organization is higher than that of rank re-organization (consistent with the examples we worked out in Figures 4 and 5). The behavior across the 10 sample DIMMs is relatively uniform.

Impact of Migration

Figure 9 also shows the average performance improvement for the RBR re-organization method augmented with smart

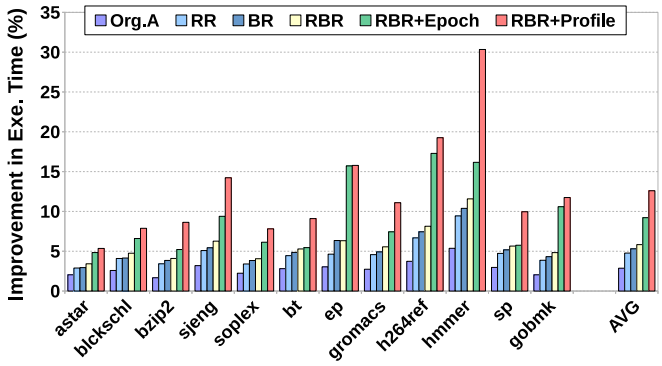

Fig. 9. Effect of different schemes of reorganization on execution time for different benchmarks.

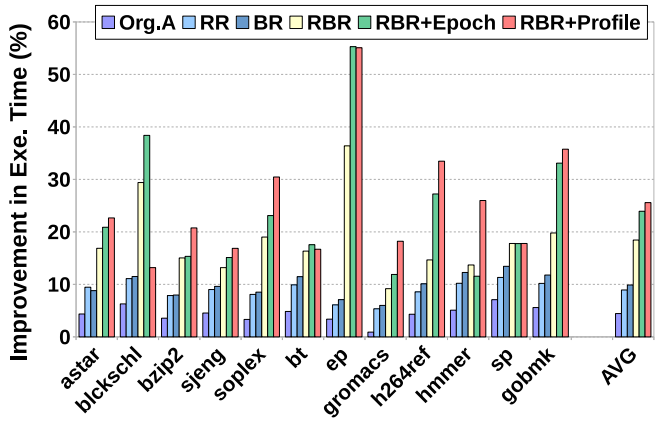

Fig. 10. Impact of reorganization schemes on PCM.

page placement. We show the impact of a profile-guided page placement that places pages in optimal locations at the start of the simulation (based on knowledge about the future), and a more practical page migration scheme that moves pages after every epoch, based on statistics gathered in the previous epoch. The profile-based approach is able to boost the performance improvement to 12.6%, and the practical migration scheme is able to come within a few percentage points of this approach, yielding a performance improvement of 9.2%.

Sensitivity Analysis

In the results so far, with a standard deviation σ of 5%, we observed that average bank latency was 10% lower than the nominal latency. When σ is increased to 10% and 15%, we observed that average bank latencies were 19% and 28% lower than nominal. Therefore, the benefits of re-organization grow somewhat linearly with the extent of PV.

B. Phase Change Memory

In this section, we apply different reorganization schemes to a PCM-based main memory. The most critical characteristic of PCM cells is the minimum current to set and reset a cell [38]. Zhang et al. [38] study different design parameters that affect PCM write current and introduce a model to characterize the impact of process variation on write current in PCM, which in turn impacts performance and endurance. Similar to DRAM, reorganization can mitigate the impact of process variation in PCM by clubbing sub-banks with similar tWR.

To evaluate the effect of reorganization on PCM performance, we use timing parameters and the model of process variation suggested in [38] ($\sigma = 8\%$ and $\mu = 75\%$ of nominal value for tWR). PCM timing parameters are based on the evaluation by Lee et al. [18]. Figure 10 shows the impact of re-organization and page migration on the performance of a PCM-based system. Because of the large impact of tWR on PCM performance, we see that our proposals can yield a significant average improvement of 25.5%.

VII. Related Work

The notion of identifying/defining "good/bad" regions of memory and exposing them to other parts of the system stack has been considered by others. For example, Sampson et al. [32] leverage this concept for approximate storage, while Lee et al. [20] leverage this for high performance. These proposals explicitly make regions of memory non-uniform. In

our work, we are developing techniques to manage inherent PV-induced non-uniformity.

In Section II, we have already discussed recent works that adjust memory timing parameters to address inter-die parameter variation and generous timing margins [9], [19], or variable retention times [26], [25], [31]. Emerging non-volatile memories are also affected by PV, e.g., PCM [38] and memristors [29]. Architectural solutions to this problem include adaptive programming currents in PCM [38].

A recent project by Zhang et al. [39] attempts bank re-organization to address the problem of high write recovery time (tWR) in future DRAMs. In our work, we consider more timing parameters, support our hypothesis with empirical measurements, evaluate a variety of re-organization approaches, and augment the system with migration policies.

VIII. Conclusions

In this work, we empirically show the existence of intra-die PV in modern DRAM chips. We make the case that bank/rank re-organization is easy to implement by adding a few on-DIMM wires and/or adding small permutation tables to buffer chips or DRAM chips. Our results show that the best re-organization scheme yields an average performance improvement of nearly 6%. This improvement grows to nearly 13% if the OS can place frequently accessed pages in low-latency banks. The improvement is much higher (26%) when applied to future PCM systems with high and varied write latencies.

References

[1] "Load-Reduced DIMMs," http://www.micron.com/products/dram-modules/lrdimm.

[2] "Registered DIMMs," http://www.micron.com/products/dram-modules/rdimm.

[3] A. Agarwal, B. Paul, H. Mahmoodi, A. Datta, and K. Roy, "A Process-Tolerant Cache Architecture for Improved Yield in Nanoscale Technologies," *IEEE Transactions on VLSI*, 2005.

[4] J. Ahn, N. Jouppi, and R. S. Schreiber, "Future Scaling of Processor-Memory Interfaces," in *Proceedings of SC*, 2009.

[5] J. Ahn, J. Lee, S. O, Y. Ro, and Y. Son, "Reducing Memory Access Latency with Asymmetric DRAM Bank Organizations," in *Proceedings of ISCA*, 2013.

[6] K. Argawal and J. Hayes, "Method and Apparatus for Measuring Statistics of DRAM Parameters with Minimum Perturbation to Cell Layout and Environment," 2010, United States Patent, Number US 7,768,814 B2.

[7] M. Awasthi, D. Nellans, K. Sudan, R. Balasubramonian, and A. Davis, "Handling the Problems and Opportunities Posed by Multiple On-Chip Memory Controllers," in *Proceedings of PACT*, 2010.

[8] K. Chandrasekar, S. Goossens, C. Weis, M. Koedam, B. Akesson, N. Wehn, and K. Goossens, "Exploiting Expendable Process-Margins in DRAMs for Run-Time Performance Optimization," in *Proceedings of DATE*, 2014.

[9] K. Chandrasekar, C. Weis, B. Akesson, N. Wehn, and K. Goossens, "Towards Variation-Aware System-Level Power Estimation of DRAMs: An Empirical Approach," in *Proceedings of DAC*, 2013.

[10] N. Chatterjee, R. Balasubramonian, M. Shevgoor, S. Pugsley, A. Udipi, A. Shafiee, K. Sudan, M. Awasthi, and Z. Chishti, "USIMM: the Utah SImulated Memory Module," University of Utah, Tech. Rep., 2012, UUCS-12-002.

[11] E. Chun, Z. Chishti, and T. Vijaykumar, "Shapeshifter: Dynamically Changing Pipeline Width and Speed to Address Process Variations," in *Proceedings of MICRO*, 2008.

[12] S. Desai, "Process Variation Aware DRAM Design Using Block-Based Adaptive Body Biasing Algorithm," Master's thesis, Utah State University, 2012.

[13] C. Garbella, "A Basic Guide to Overclocking and System Building," http://www.overclockers.com/a-basic-guide-to-overclocking-and-system-building/.

[14] T. Hamamoto, S. Sugiura, and S. Sawada, "On the Retention Time Distribution of Dynamic Random Access Memory (DRAM)," *IEEE Trans. on Electron Devices*, 1998.

[15] JEDEC, *JESD79-4: JEDEC Standard DDR4 SDRAM*, 2012.

[16] R. Joseph, D. Brooks, and M. Martonosi, "Control Techniques to Eliminate Voltage Emergencies in High Performance Processors," in *Proceedings of HPCA*, 2003.

[17] K. Kim and J. Lee, "A New Investigation of Data Retention Time in Truly Nanoscaled DRAMs," *IEEE Electron Device Letters*, 2009.

[18] B. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting Phase Change Memory as a Scalable DRAM Alternative," in *Proceedings of ISCA*, 2009.

[19] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang, and O. Mutlu, "Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common Case," in *Proceedings of HPCA*, 2015.

[20] D. Lee, Y. Kim, V. Seshadri, J. Liu, L. Subramanian, and O. Mutlu, "Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture," in *Proceedings of HPCA-19*, 2013.

[21] H.-W. Lee, K.-H. Kim, Y.-K. Choi, J.-H. Shon, N.-K. Park, K.-W. Kim, C. Kim, Y.-J. Choi, and B.-T. Chung, "A 1.6V 1.4Gb/s/pin Consumer DRAM with Self-Dynamic Voltage-Scaling Technique in 44nm CMOS Technology," in *Proceedings of ISSCC*, 2011.

[22] S. Lee, C. Choi, J. Kong, W. Lee, and J. Yoo, "An Efficient Statistical Analysis Methodology and Its Application to High-Density DRAMs," in *Proceedings of ICCAD*, 1997.

[23] Y. Li, H. Schneider, F. Schnabel, and R. Thewes, "DRAM Yield Analysis and Optimization by a Statistical Design Approach," *IEEE Trans. on Circuits and Systems*, 2011.

[24] X. Liang, R. Canal, G. Wei, and D. Brooks, "Process Variation Tolerant 3T1D-Based Cache Architectures," in *Proceedings of MICRO*, 2007.

[25] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu, "An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms," in *Proceedings of ISCA*, 2013.

[26] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu, "RAIDR: Retention-aware intelligent DRAM refresh," in *Proceedings of ISCA*, 2012.

[27] S. R. Nassif, "Modeling and analysis of manufacturing variations," in *Proceedings of IEEE Conf. Custom Integr. Circuits*, 2001.

[28] R. Nelson, "A Newbie's Guide to Overclocking Memory," http://www.overclockers.com/a-newbies-guide-to-overclocking-memory/.

[29] D. Niu, Y. Chen, C. Xu, and Y. Xie, "Impact of Process Variations on Emerging Memristor," in *Proceedings of DAC*, 2010.

[30] T. Pawlowski, "The Future of Memory Technology," 2014, keynote at The Memory Forum.

[31] M. K. Qureshi, D.-H. Kim, S. Khan, P. Nair J., and O. Mutlu, "AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems," in *Proceedings of 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015.

[32] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, "Approximate Storage in Solid-State Memories," in *Proceedings of MICRO*, 2013.

[33] S. R. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "VARIUS: A Model of Parameter Variation and Resulting Timing Errors for Microarchitects," *Semiconductor Manufacturing, IEEE Transactions on*, vol. 21, no. 1, 2008.

[34] M. Shevgoor, J.-S. Kim, N. Chatterjee, R. Balasubramonian, A. Davis, and A. Udipi, "Quantifying the Relationship between the Power Delivery Network and Architectural Policies in a 3D-Stacked Memory Device," in *Proceedings of MICRO*, 2013.

[35] R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "Mitigating Parameter Variation with Dynamic Fine-Grain Body Biasing," in *Proceedings of MICRO*, 2007.

[36] G. Torres, "Memory Overclocking," http://www.hardwaresecrets.com/article/Memory-Overclocking/152/.

[37] Xilinx, "Virtex-7 FPGA VC-707 Evaluation Kit," http://http://www.xilinx.com/products/boards-and-kits/ek-v7-vc707-g.html.

[38] W. Zhang and T. Li, "Characterizing and mitigating the impact of process variations on phase change based memory systems," in *Proceedings of MICRO*, 2009.

[39] X. Zhang, Y. Zhang, B. R. Childers, Yang, and Jun, "Exploiting DRAM Restore Time Variations in Deep Sub-micron Scaling," in *Proceedings of DATE-15*, 2015.

[40] B. Zhao, Y. Du, Y. Zhang, and J. Yang, "Variation-Tolerant Non-Uniform 3D Cache Management in Die Stacked Multicore Processor," in *Proceedings of MICRO*, 2009.

[41] H. Zheng, J. Lin, Z. Zhang, E. Gorbatov, H. David, and Z. Zhu, "Mini-Rank: Adaptive DRAM Architecture For Improving Memory Power Efficiency," in *Proceedings of MICRO*, 2008.