Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case

Donghyuk Lee Yoongu Kim Gennady Pekhimenko
Samira Khan Vivek Seshadri Kevin Chang Onur Mutlu

Carnegie Mellon University

{*donghyu1, yoongukim, gpekhime, samirakhan, visesh, kevincha, onur*}*@cmu.edu*

Abstract

In current systems, memory accesses to a DRAM chip must obey a set of minimum latency restrictions specified in the DRAM standard. Such timing parameters exist to guarantee reliable operation. When deciding the timing parameters, DRAM manufacturers incorporate a very large margin as a provision against two worst-case scenarios. First, due to process variation, some outlier chips are much slower than others and cannot be operated as fast. Second, chips become slower at higher temperatures, and all chips need to operate reliably at the highest supported (i.e., worst-case) DRAM temperature (85°C). In this paper, we show that typical DRAM chips operating at typical temperatures (e.g., 55°C) are capable of providing a much smaller access latency, but are nevertheless forced to operate at the largest latency of the worst-case.

Our goal in this paper is to exploit the extra margin that is built into the DRAM timing parameters to improve performance. Using an FPGA-based testing platform, we first characterize the extra margin for 115 DRAM modules from three major manufacturers. Our results demonstrate that it is possible to reduce four of the most critical timing parameters by a minimum/maximum of 17.3%/54.8% at 55°C without sacrificing correctness. Based on this characterization, we propose Adaptive-Latency DRAM (AL-DRAM), a mechanism that adaptively reduces the timing parameters for DRAM modules based on the current operating condition. AL-DRAM does not require any changes to the DRAM chip or its interface.

We evaluate AL-DRAM on a real system that allows us to reconfigure the timing parameters at runtime. We show that AL-DRAM improves the performance of memory-intensive workloads by an average of 14% without introducing any errors. We discuss and show why AL-DRAM does not compromise reliability. We conclude that dynamically optimizing the DRAM timing parameters can reliably improve system performance.

1. Introduction

A DRAM chip is made of capacitor-based cells that represent data in the form of electrical charge. To store data in a cell, charge is injected, whereas to retrieve data from a cell, charge is extracted. Such *movement of charge* is not only the centerpiece of DRAM operation, but also the bottleneck of DRAM latency [36, 66]. This is due to two fundamental reasons. First, when injecting charge into a cell, a wire called the bitline — through which the charge is delivered — impedes the flow of charge [36, 66]. Owing to the large resistance and the large capacitance of the bitline, the cell experiences a *large RC-delay*, which increases the time it takes for the cell to become fully charged. Second, when extracting charge from a cell, the cell is incapable of mobilizing a strong flow of charge out of itself and into the bitline [36, 66]. Limited by the finite amount of charge stored in its small capacitor, the cell has an inherently *weak charge-drive*, which is further weakened as the cell loses more of its charge to the bitline. As a result, the cell cannot charge the bitline quickly (or even fully).

When a DRAM chip is accessed, it requires a certain amount of time before enough charge can move into the cell (or the bitline) for the data to be reliably stored (or retrieved). To guarantee this behavior, DRAM manufacturers impose a set of minimum latency restrictions on DRAM accesses, called *timing parameters* [25]. Ideally, timing parameters should provide just enough time for a DRAM chip to operate correctly. In practice, however, DRAM manufacturers *pessimistically incorporate a very large margin* into their timing parameters to ensure correct operation under *worst-case* conditions. This is because of two major concerns. First, due to *process variation*, some outlier cells suffer from a larger RC-delay than other cells, and require more time to be charged. For example, an outlier cell could have a very narrow connection (i.e., contact) to the bitline, which constricts the flow of charge and increases the RC-delay [37]. Second, due to *temperature dependence*, all cells suffer from a weaker charge-drive at high temperatures, and require more time to charge the bitline. DRAM cells are intrinsically leaky, and lose some of their charge even when they are not being accessed. At high temperatures, this leakage is accelerated exponentially [29, 41, 48, 57, 74], leaving a cell with less charge to drive the bitline when the cell is accessed — increasing the time it takes for the bitline to be charged.

Consequently, timing parameters prescribed by the DRAM manufacturers are dictated by the *worst-case cells* (the slowest cells) operating under the *worst-case conditions* (the highest temperature of 85°C [25]). Such pessimism on the part of the DRAM manufacturers is motivated by their desire to *(i)* increase chip yield and *(ii)* reduce chip testing time. The manufacturers, in turn, are driven by the extremely cost-sensitive nature of the DRAM market, which encourages them to adopt pessimistic timing parameters rather than to *(i)* discard chips with the slowest cells or *(ii)* test chips at lower temperatures. Ultimately, the burden of pessimism is passed on to the end-users, who are forced to endure much greater latencies than what is actually needed for reliable operation under common-case conditions.

In this paper, we first characterize 115 DRAM modules from three manufacturers to expose the excessive margin that is built into their timing parameters. Using an FPGA-based testing platform [29, 31, 41], we then demonstrate that DRAM timing parameters can be shortened to reduce DRAM latency without sacrificing any observed degree of DRAM reliability. We are able to reduce latency by taking advantage of the two large gaps between the worst-case and the "common-case." First, most DRAM chips are *not* exposed to the worst-case temperature of 85°C: according to previous studies [11, 12, 43] and our own measurements (Section 4.2), the ambient temperature around a DRAM chip is typically less than 55°C. Second, most DRAM chips do *not* contain the worst-case cell with the largest latency: the slowest cell for a typical chip is still faster than that of the worst-case chip (Section 7).

Based on our characterization, we propose Adaptive-Latency DRAM (AL-DRAM), a mechanism that dynamically optimizes the timing parameters for different modules at different temperatures. AL-DRAM exploits the *additional charge slack* present in the common-case compared to the worst-case, thereby preserving the level of reliability (at least as high as the worst-case) provided by DRAM manufacturers. We evaluate AL-DRAM on a real system [5, 6] that allows us to dynamically reconfigure the timing parameters at runtime. We show that AL-DRAM improves the performance of a wide variety of memory-intensive workloads by 14.0% (on average) without introducing errors. Therefore, we conclude that AL-DRAM improves system performance while maintaining memory correctness and without requiring changes to DRAM chips or the DRAM interface.

This paper makes the following contributions:

- We provide a detailed analysis of why we can reduce DRAM timing parameters without sacrificing reliability. We show that the latency of a DRAM access depends on how quickly charge moves into or out of a cell. Compared to the worst-case cell operating at the worst-case temperature (85°C), a typical cell at a typical temperature allows much faster movement of charge, leading to shorter access latency. This enables the opportunity to reduce timing parameters without introducing errors.
- Using an FPGA-based testing platform, we profile 115 DRAM modules (from three manufacturers) and expose the large margin built into their timing parameters. In particular, we identify four timing parameters that are the most critical during a DRAM access: `tRCD`, `tRAS`, `tWR`, and `tRP`. At 55°C, we demonstrate that the parameters can be reduced by an average of 17.3%, 37.7%, 54.8%, and 35.2% while still maintaining correctness. For some chips, the reductions are as high as 27.3%, 42.8%, 66.7%, and 45.4%.
- We propose a practical mechanism, *Adaptive-Latency DRAM (AL-DRAM)*, to take advantage of the above observation. The key idea is to dynamically adjust the DRAM timing parameters for each module based on its latency characteristics and temperature so that the timing parameters are dynamically optimized for the current operating condition. We show that the hardware cost of AL-DRAM is very modest, with no changes to DRAM.
- We evaluate AL-DRAM on a real system [5, 6] running real workloads by dynamically reconfiguring the timing parameters. For a wide variety of memory-intensive workloads, AL-DRAM improves system performance by an average of 14.0% and a maximum of 20.5% without incurring errors.

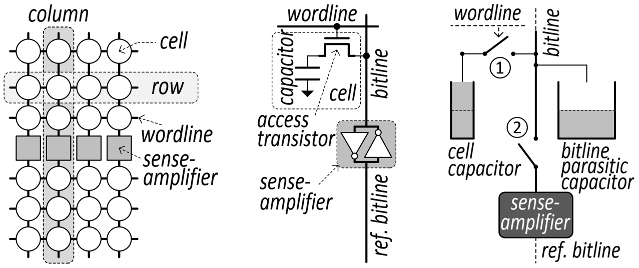
2. DRAM Background

To understand the dominant sources of DRAM latency, we first provide the necessary background on DRAM organization and operation.

2.1. DRAM Organization

Figure 1a shows the internal organization of a DRAM subarray [8, 34, 36, 62], which consists of a 2-D array of DRAM cells connected to a single row of *sense amplifiers* (a row of sense amplifiers is also referred to as a *row buffer*). The sense amplifier is a component that essentially acts as a latch — it detects the data stored in the DRAM cell and latches on to the corresponding data.

Figure 1b zooms in on the connection between a single DRAM cell and its corresponding sense amplifier. Each cell consists of *(i)* a capacitor that stores a bit of data in the form of electrical charge, and *(ii)* an access transistor that determines whether the cell is connected to the sense amplifier. The sense amplifier consists of two cross-coupled inverters. The wire that connects the cell to the sense amplifier is called the *bitline*, whereas the wire that controls the access transistor is called the *wordline*. Figure 1c depicts a simplified view of a cell as well as its bitline and sense amplifier, in which electrical charge is represented in gray. Switch ① represents the access transistor controlled by the wordline, and switch ② represents the on/off state of the sense amplifier.



(a) Subarray (b) Cell (c) Simplified View

Figure 1: DRAM Organization

2.2. DRAM Operation: Commands & Timing Constraints

As shown in Figure 2, a cell transitions through five different states during each access. In the first state ❶, which is called the *precharged* state, the cell is "fully" charged, while the bitline is only halfway charged (i.e., the bitline voltage is maintained at $\frac{1}{2}V_{DD}$). In practice, the cell is usually not completely charged because of a phenomenon called *leakage*, wherein the cell capacitor loses charge over time.

In order to access data from a cell, the DRAM controller issues a command called `ACTIVATE`. Upon receiving this command, DRAM increases the wordline voltage, thereby connecting the cell to the bitline. Since the cell is at a higher voltage than the bitline, the cell then *drives* its charge into the bitline until their voltages are equalized at $\frac{1}{2}V_{DD} + \delta$. This is depicted in state ❷, which is called *charge-sharing*.

Subsequently, the sense amplifier is enabled, which then

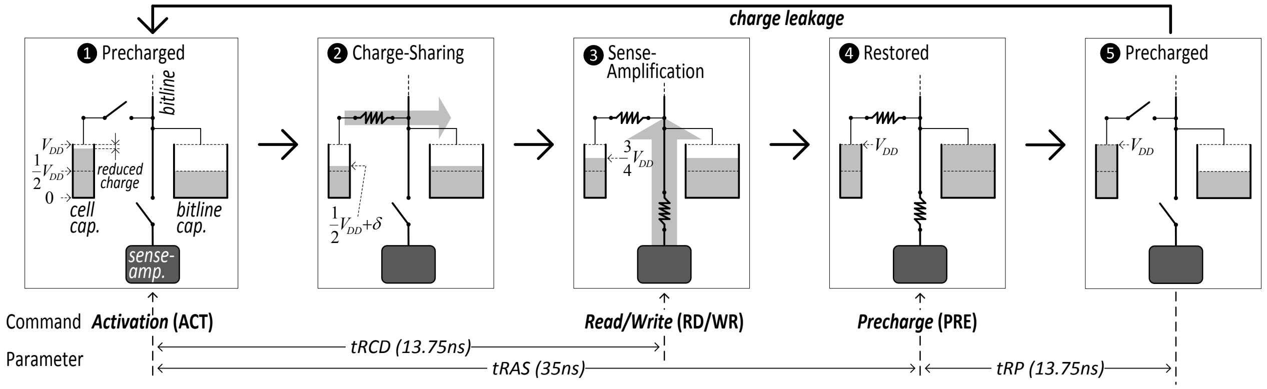


Figure 2: DRAM Operations, Commands and Parameters

senses and amplifies the difference in the bitline voltage and $\frac{1}{2}V_{DD}$. During this process, referred to as *sensing and amplification*, the sense amplifier drives the bitline voltage to V_{DD}. Since the cell is still connected to the bitline, this process also *injects* charge into the cell. Midway through the sense amplification process (state ❸), when the bitline reaches an intermediate voltage level (e.g., $\frac{3}{4}V_{DD}$), data can be read out or written into the bitline. This is done by issuing a `READ` or `WRITE` command to the corresponding cell. The time taken to reach this state (❸) after issuing the `ACTIVATE` is expressed as a timing parameter called `tRCD`.

After completing the sense amplification, the bitline voltage reaches V_{DD} and the cell charge is fully restored (state ❹). The time taken to reach this state after issuing the `ACTIVATE` is expressed as a timing parameter called `tRAS`. If there is a write operation, some additional time is required for the bitline and the cell to reach this state, which is expressed as a timing parameter called `tWR`.

Before we can access data from a different cell connected to the same bitline, the sense amplifier must be taken back to the precharged state. This is done by issuing a `PRECHARGE` command. Upon receiving this command, DRAM first decreases the wordline voltage, thereby disconnecting the cell from the bitline. Next, DRAM disables the sense amplifier and drives the bitline back to a voltage of $\frac{1}{2}V_{DD}$ (state ❺). The time taken for the precharge operation is expressed as a timing parameter called `tRP`.

At state ❺, note that the cell is completely filled with charge. Subsequently, however, the cell slowly loses some of its charge until the next access (cycling back to state ❶). The length of time for which the cell can reliably hold its charge is called the cell's *retention time*. If the cell is not accessed for a long time, it may lose enough charge to invert its stored data, resulting in an error. To avoid data corruption, DRAM refreshes the charge in all of its cells at a regular interval, called the *refresh interval*.

3. Charge & Latency Interdependence

As we explained, the operation of a DRAM cell is governed by two important concepts: *(i)* the quantity of charge and *(ii)* the latency it takes to move charge. These two concepts are closely related to each other — one cannot be adjusted without affecting the other. To establish a more quantitative relationship between charge and latency, Figure 3 presents the voltage of a cell and its bitline as they cycle through the precharged state, charge-sharing state, sense-amplification state, restored state, and back to the precharged state (Section 2).[1] This curve is typical in DRAM operation, as also shown in prior works [14, 27, 36, 66]. The timeline starts with an `ACTIVATE` at 0 ns and ends with the completion of `PRECHARGE` at 48.75 ns. From the figure, we identify three specific periods in time when the voltage changes slowly: *(i)* start of sense-amplification (part ①), *(ii)* end of sense-amplification (part ②), and *(iii)* end of precharging (part ③). Since charge is correlated with voltage, these three periods are when the charge also moves slowly. In the following, we provide three observations explaining why these three periods can be shortened for typical cells at typical temperatures — offering the best opportunity for shortening the timing parameters.

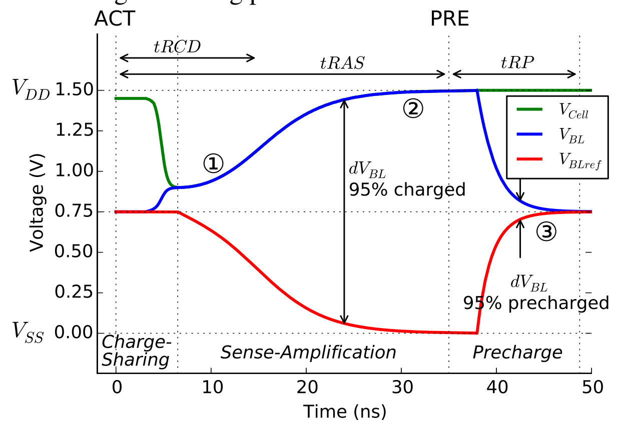


Figure 3: Phases of DRAM Voltage Levels

Observation 1. At the start of the sense-amplification phase, the higher the bitline voltage, the quicker the sense-amplifier is jump-started. Just as the amplification phase starts, the sense-amplifier detects the bitline voltage that was increased in the previous charge-sharing phase (by the cell donating its charge to the bitline). The sense-amplifier then begins to inject more charge into the bitline to increase the voltage even further — triggering a positive-feedback loop where the bitline voltage increases more quickly as the bitline voltage becomes higher. This is shown in Figure 3 where the bitline voltage ramps up faster and faster during the initial part of the amplification phase. Importantly, if the bitline has a higher voltage to begin with (at the start of sense-amplification), then the positive-

[1] Using 55nm DRAM parameters [55, 69], we simulate the voltage and current of the DRAM cells, sense-amplifiers, and bitline equalizers (for precharging the bitline). To be technology-independent, we model the DRAM circuitry using NMOS and PMOS transistors that obey the well-known MOSFET equation for current-voltage (SPICE) [56]. We do not model secondary effects.

feedback is able to set in more quickly. Such a high initial voltage is comfortably achieved by typical cells at typical temperatures because they donate a large amount of charge to the bitline during the charge-sharing phase (as they have a large amount of charge). As a result, they are able to reach states ❸ and ❹ (Figure 2) more quickly, creating the opportunity to shorten tRCD and tRAS.

Observation 2. At the end of the sense-amplification phase, nearly half the time (42%) is spent on injecting the last 5% of the charge into the cell. Thanks to the positive-feedback, the middle part of the amplification phase (part between ① and ② in Figure 3) is able to increase the bitline voltage quickly. However, during the later part of amplification (part ② in Figure 3), the RC-delay becomes much more dominant, which prevents the bitline voltage from increasing as quickly. In fact, it takes a significant amount of extra delay for the bitline voltage to reach V_{DD} (Figure 3) that is required to *fully* charge the cell. However, for typical cells at typical temperatures, such an extra delay may not be needed — the cells could already be injected with *enough* charge for them to comfortably share with the bitline when they are next accessed. This allows us to shorten the later part of the amplification phase, creating the opportunity to shorten tRAS and tWR.

Observation 3. At the end of the precharging phase, nearly half the time (45%) is spent on extracting the last 5% of the charge from the bitline. Similar to the amplification phase, the later part of the precharging phase is also dominated by the RC-delay, which causes the bitline voltage to decrease slowly to $\frac{1}{2}V_{DD}$ (part ③ in Figure 3). If we decide to incur less than the full delay required for the bitline voltage to reach exactly $\frac{1}{2}V_{DD}$, it could lead to two different outcomes depending on which cell we access next. First, if we access the *same* cell again, then the higher voltage left on the bitline works in our favor. This is because the cell — which is filled with charge — would have increased the bitline voltage anyway during the charge-sharing phase. Second, if we access a *different* cell connected to the same bitline, then the higher voltage left on the bitline may work as a handicap. Specifically, this happens only when the cell is devoid of any charge (e.g., storing a data of '0'). For such a cell, its charge-sharing phase operates in the opposite direction, where the cell steals some charge away from the bitline to decrease the bitline voltage. Subsequently, the voltage is "amplified" to 0 instead of V_{DD}. Nevertheless, typical cells at typical temperatures are capable of comfortably overcoming the handicap — thanks to their large capacitance, the cells are able to steal a large amount of charge from the bitline. As a result, this creates the opportunity to shorten tRP.

4. Charge Gap: Common-Case vs. Worst-Case

Based on the three observations, we understand that *timing parameters can be shortened if the cells have enough charge.* Importantly, we showed that such a criterion is easily satisfied for typical cells at typical temperatures. In this section, we explain what it means for a cell to be "typical" and why it has more charge at "typical" temperatures. Specifically, we examine two physical phenomena that critically impact a DRAM cell's ability to receive and retain charge: *(i)* process variation and *(ii)* temperature dependence.

4.1. Process Variation: Cells Are Not Created Equal

Process variation is a well-known phenomenon that introduces deviations between a chip's intended design and its actual implementation [13, 37, 60]. DRAM cells are affected by process variation in two major aspects: *(i)* cell capacitance and *(ii)* cell resistance. Although every cell is designed to have a large capacitance (to hold more charge) and a small resistance (to facilitate the flow of charge), some deviant cells may not be manufactured in such a manner [15, 26, 29, 30, 38, 41, 42]. In Figure 4a, we illustrate the impact of process variation using two different cells: one is a typical cell conforming to design (left column) and the other is the worst-case cell deviating the most from design (right column).

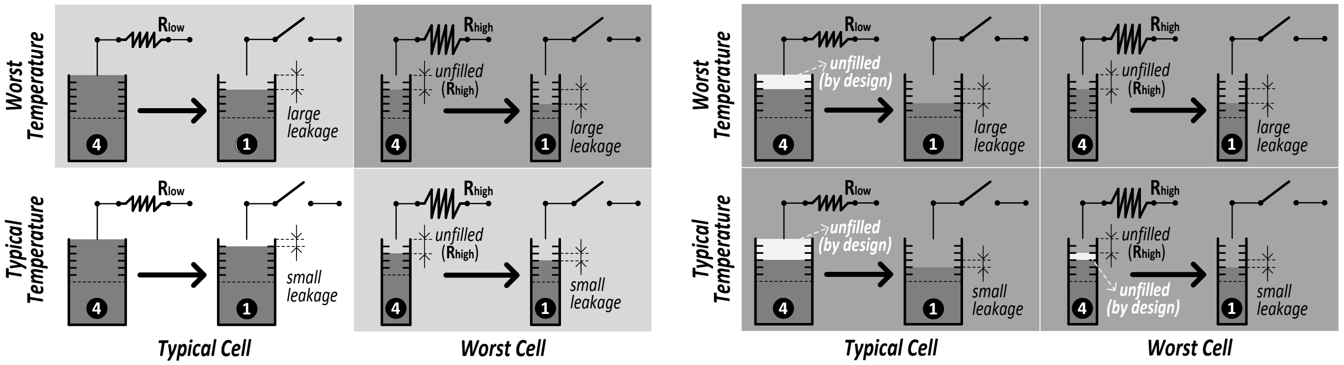
As we see from Figure 4a, the worst-case cell contains less charge than the typical cell in state ❹ (Restored state, as was shown in Figure 2). This is because of two reasons. First, due to its *large resistance*, the worst-case cell cannot allow charge to flow inside quickly. Second, due to its *small capacitance*, the worst-case cell cannot store much charge even when it is full. To accommodate such a worst-case cell, existing timing parameters are set to a large value. However, *worst-case cells are relatively rare.* When we analyzed 115 modules, the overwhelming majority of them had significantly more charge than what is necessary for correct operation (Section 7 will provide more details).

4.2. Temperature Dependence: Hot Cells Are Leakier

Temperature dependence is a well-known phenomenon in which cells leak charge at almost double the rate for every 10°C increase in temperature [29, 41, 48, 57, 74]. In Figure 4a, we illustrate the impact of temperature dependence using two cells at two different temperatures: *(i)* typical temperature (55°C, bottom row), and *(ii)* the worst-case temperature (85°C, top row) supported by DRAM standards.

As we see from the figure, both typical and worst-case cells leak charge at a faster rate at the worst-case temperature. Therefore, not only does the worst-case cell have less charge to begin with, but it is left with *even less* charge because it leaks charge at a faster rate (top-right in Figure 4a). To accommodate the combined effect of process variation *and* temperature dependence, existing timing parameters are set to a very large value. However, *most systems do not operate at 85°C* [11, 12, 43].[2] We measured the DRAM ambient temperature in a server cluster running a memory-intensive benchmark, and found that the temperature *never* exceeds 34°C — as well as never changing by more than 0.1°C per second. We show

[2] Figure 22 in [12] and Figure 3 in [43] show that the maximum temperature of DRAM chips at the highest CPU utilization is 60–65°C. While some prior works claim a maximum DRAM temperature over 80°C [76], each DIMM in their system dissipates 15W of power. This is very aggressive nowadays — modern DIMMs typically dissipate around 2–6W (see Figure 8 of [17], 2-rank configuration same as the DIMM configuration of [76]). We believe that continued voltage scaling and increased energy efficiency of DRAM have helped reduce the DIMM power consumption. While old DDR1/DDR2 use 1.8–3.0V power supplies, newer DDR3/DDR4 use only 1.2–1.5V. In addition, newer DRAMs adopt more power saving techniques (i.e., temperature compensated self refresh, power down modes [21, 46]) that were previously used only by Low-Power DRAM (LPDDR). Furthermore, many previous works [9, 39, 40, 43, 76] propose hardware/software mechanisms to maintain a low DRAM temperature and energy.



(a) Existing DRAM

(b) Our Proposal (Adaptive-Latency DRAM)

Figure 4: Effect of Reduced Latency: Typical vs. Worst (Darker Background means Less Reliable)

this in Figure 5, which plots the temperature for a 24-hour period (left) and also zooms in on a 2-hour period (right). In addition, we repeated the measurement on a desktop system that is not as well cooled as the server cluster. As Figure 6 shows, even when the CPU was utilized at 100% and the DRAM bandwidth was utilized at 40%, the DRAM ambient temperature never exceeded 50°C. Other works [11, 12, 43] report similar results, as explained in detail in Footnote 2. From this, we conclude that the majority of DRAM modules are likely to operate at temperatures that are much lower than 85°C, which slows down the charge leakage by an order of magnitude or more than at the worst-case temperature.

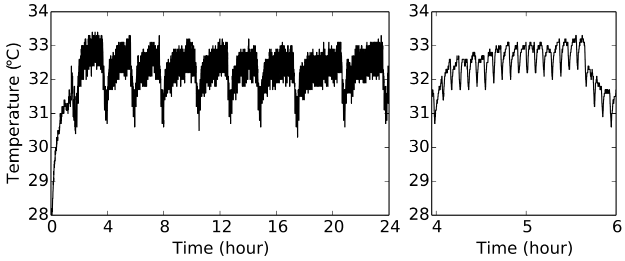


Figure 5: DRAM Temperature in a Server Cluster

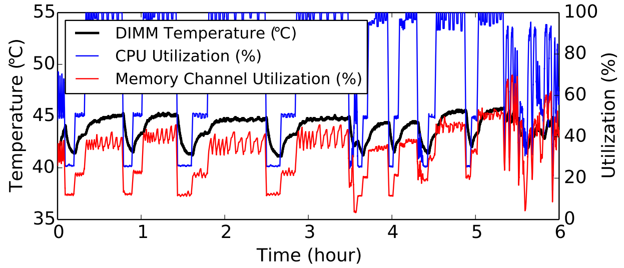


Figure 6: DRAM Temperature in a Desktop System

4.3. Reliable Operation with Shortened Timing

As explained in Section 3, the amount of charge in state ❶ (i.e., the precharged state in Figure 2) plays a critical role in whether the correct data is retrieved from a cell. That is why the worst-case condition for correctness is the top-right of Figure 4a, which shows the least amount of charge stored in the worst-case cell at the worst-case temperature in state ❶. However, DRAM manufacturers provide reliability guarantees even for such worst-case conditions. In other words, the amount of charge at the worst-case condition is still greater than what is required for correctness.

If we were to shorten the timing parameters, we would also be reducing the charge stored in the cells. It is important to note, however, that we are proposing to exploit *only* the *additional slack* (in terms of charge) compared to the worst-case. This allows us to provide as strong of a reliability guarantee as the worst-case.

In Figure 4b, we illustrate the impact of shortening the timing parameters in three of the four different cases: two different cells at two different temperatures. The lightened portions inside the cells represent the amount of charge that we are giving up by using the shortened timing parameters. Note that we are not giving up any charge for the worst-case cell at the worst-case temperature. Although the other three cells are not fully charged in state ❹, when they eventually reach state ❶, they are left with a similar amount of charge as the worst-case (top-right). This is because these cells are capable of either holding more charge (typical cell, left column) or holding their charge longer (typical temperature, bottom row). Therefore, optimizing the timing parameters (based on the amount of existing slack) provides the opportunity to reduce overall DRAM latency while still maintaining the reliability guarantees provided by the DRAM manufacturers.

In Section 7, we present the results from our characterization study where we quantify the slack in 115 DRAM modules. Before we do so, we first propose our mechanism for identifying and enforcing the shortened timing parameters.

5. Adaptive-Latency DRAM

Our mechanism, Adaptive-Latency DRAM (AL-DRAM), allows the memory controller to exploit the latency variation across DRAM modules (DIMMs) at different operating temperatures by using customized (aggressive) timing parameters for each DIMM/temperature combination. Our mechanism consists of two steps: *(i) identification* of the best timing parameters for each DIMM/temperature, and *(ii) enforcement*, wherein the memory controller dynamically extracts each DIMM's operating temperature and uses the best timing parameters for each DIMM/temperature combination.

5.1. Identifying the Best Timing Parameters

Identifying the best timing parameters for each DIMM at different temperatures is the more challenging of the two steps. We propose that DRAM manufacturers identify the best timing parameters at different temperatures for each DRAM chip during the testing phase and provide that information along with the DIMM in the form of a simple table. Since our proposal only involves changing four timing parameters (`tRCD`, `tRAS`, `tWR`, and `tRP`), the size of the table for, e.g., five, different temperature points is small and such a table can potentially be embedded in the Serial Presence Detect circuitry (a ROM present in each DIMM [24]). We expect this approach to have low cost as DRAM manufacturers already have an elaborate testing mechanism to identify faulty DIMMs. An alternative approach to perform this profiling step is to do it at the end-user system using online testing while the system is running. We leave the exploration of such an online testing mechanism to future work.

5.2. Enforcing Dynamic Timing Parameters

Dynamically enforcing the best timing parameters at the memory controller is fairly straightforward. The memory controller populates a hardware table with the best timing parameters for different temperatures for all the DIMMs connected to the controller. The memory controller divides time into regular intervals (e.g., 256 ms). At the beginning of each interval, it extracts the temperature of each DIMM. It then employs the best timing parameters corresponding to the temperature of each DIMM for all accesses during the remainder of the interval.

This approach should work well in practice as temperature does not change very frequently in real systems — our measurements on real server and desktop systems indicate that temperature changes at the rate of at most 0.1°C per second. In addition, existing DRAM designs such as LPDDR3 [23], and the recently announced DDR4 [22] already have in-DRAM temperature sensors to minimize self-refresh energy. By accessing the temperature value of in-DRAM temperature sensors during auto-refresh (usually performed every 7.8us), our mechanism monitors DRAM temperature without any performance penalty and frequently enough to detect even drastic temperature changes.

In Section 8, we evaluate the benefits of AL-DRAM on a real system equipped with a memory controller whose timing parameters can be dynamically changed. Our evaluation shows that AL-DRAM can significantly improve system performance for a wide variety of applications.

6. DRAM Latency Profiling Methodology

In this section, we describe our FPGA-based DRAM profiling (i.e., testing) infrastructure and the methodology we use to characterize DRAM modules for variation in access latency.

6.1. Profiling Infrastructure

To analyze the characteristics of latency variation in DRAM, we have built an FPGA-based testing platform [29, 31, 41] using Xilinx ML605 boards [72] connected to a host machine over a PCI-e bus [71]. The FPGA has a DDR3 DRAM memory controller [73]. We customize the memory controller so that we can change DRAM timing parameters from the host machine at runtime. To test DRAM modules at different operating temperatures, we built a heat chamber consisting of a heat encloser, a heater, and a temperature controller with a thermo-couple sensor. During the test, the temperature within the heat chamber is maintained within ±0.5°C of the target temperature. In all, we present results for 115 DRAM modules that are produced by three different DRAM manufacturers during the last 5 years (from 2010 to 2014). The majority of these are DDR3-1600 SO-DIMMs with standard timing parameters: `tRCD` = 13.75 ns, `tRAS` = 35 ns, `tWR` = 15 ns, and `tRP` = 13.75 ns.

6.2. Profiling Mechanism

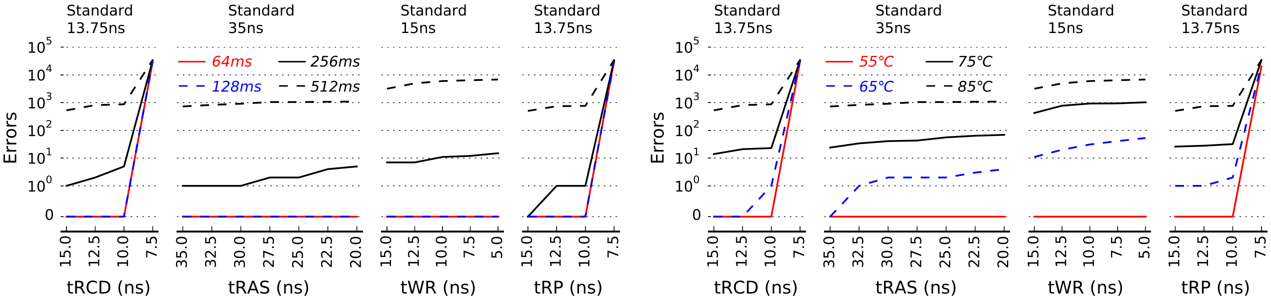
The goal of our profiling experiments is to determine the amount by which different timing parameters can be reduced without inducing any errors. For this purpose, we devise two types of tests: one for testing read operations and another for write operations. The read test aims to explore reductions in `tRCD`, `tRAS`, and `tRP`, whereas the write test aims to explore reductions in `tWR`. Both types of tests are carefully designed to induce a reasonable amount of coupling-noise between circuit components (i.e., cells, bitlines, and wordlines), as well as to generate power-noise. We now describe the steps involved in each of these tests in detail. Each test involves a test address, a test data pattern, and a target set of timing parameters.

Read Test: For a read test, we first write the test data pattern to the target address with the standard DRAM timing parameters. We follow it with a read to the target address with the *target*, smaller, timing parameters. This step could potentially corrupt the data if the cells involved in the operation are not restored with enough charge or if the corresponding bitlines are not precharged sufficiently. To test for data corruption, we wait for the refresh interval such that DRAM cells have the smallest charge due to charge leakage over time. Then, we perform another read operation to the target address. If the latter read data matches the originally-written test data, the test is considered successful — i.e., for the corresponding address, the timing parameters can be reduced to the target timing parameters. Otherwise, the test is considered a failure and the system logs the list of errors.

Write Test: For a write test, we first write the inverted test data pattern to the target address with the standard DRAM timing parameters. We then write the test data pattern with the target timing parameters. The original write (inverted test pattern) is to ensure that the test write actually flips the values on the bitlines, thereby making the `tWR` timing parameter relevant. After the test write, we wait for the refresh interval and perform a verification read. We check if the verification succeeded and log the list of errors, if any.

Coupling and Power-Noise Effects: We carefully design the tests to be close to the worst-case in coupling and power-noise (to stress the reliability of our proposal). From the standpoint of a single row, each test *(i)* writes/reads data to/from the row, and then *(ii)* reads data from the row for verification (access – wait – verify). However, from the standpoint of the entire DRAM chip, we test different rows in an overlapped manner, staggered by `tRC` (access, access, · · ·, access, verify, verify, · · ·, verify). We perform this test sequence twice with two different orders of row addresses (increasing/decreasing). Due to this overlapped manner of testing multiple rows, the tests exhibit the following effects.

- *Bitline-to-Cell Coupling:* Non-activated cells and bitlines are coupled with each other in as many as half of the cells per bitline (i.e., 256 times a 512-cell bitline).
- *Wordline-to-Wordline Coupling:* Each wordline is coupled



(a) Sweeping the Refresh Interval (Temperature: 85°C)

(b) Sweeping the Temperature (Refresh Interval: 512ms)

Figure 7: Effect of Varying Each Timing Parameter on Error Count

with adjacent wordlines in as many columns as in a row (i.e., 256 times for a 4k-cell row and 64-bit per access).

- *Power-Noise:* Power noise is close to the worst-case due to the maximum number of allowed row activations per time interval (i.e., activating a row every tRC).

Data Patterns: We repeat both tests for all DRAM addresses with eight different checkered data patterns [68] (0000, 0011, 0101, 1001, 0110, 1010, 1100, 1111). To test the bitline precharge effect, we use exclusive-OR data patterns (i.e., 0101 vs. 1010) for adjacent rows. Therefore, bitlines are amplified in the opposite direction from the previous row activation.

Iterating Tests: We repeat tests with a large number of reduced target timing parameters. We accumulate the erroneous cells from all the test patterns. Since cells of the same DRAM row are activated and precharged together, we run through all the DRAM addresses in the column-major order to reduce interference between tests of the cells in the same row. We perform 10 iterations of each test before deciding whether it is free from errors.

7. DRAM Latency Profiling Results and Analysis

In this section, we present and analyze the results of our profiling experiments. We first present the effect of reducing individual timing parameters on a single representative module (Section 7.1). We then present the results of reducing multiple timing parameters simultaneously (Section 7.2), analyze the timing slack present (Section 7.3), and quantify timing parameter reductions (Section 7.4) using this representative module. Finally, we summarize our profiling results for all 115 modules (Section 7.5).

7.1. Effect of Reducing Individual Timing Parameters

As discussed in Section 4, DRAM access latency is tightly coupled with the amount of charge in the accessed cell. To verify this on real DRAM chips, we need to adjust the amount of charge intentionally. Unfortunately, there is no way to quantitatively change the charge stored in DRAM cells. Instead, we indirectly adjust the charge amount by enabling the leakage of DRAM cells. Note that DRAM cells have less charge in two cases: *(i)* at longer refresh intervals and *(ii)* at higher temperatures. Using these characteristics, we design aggressive test environments for analyzing the effect of reducing each timing parameter by: *(i)* sweeping the refresh interval from the DRAM standard (64 ms) to very long refresh intervals (up to 512 ms) at the highest temperature in the DRAM standard (85°C), and *(ii)* sweeping the temperature at a very long refresh interval (512 ms).

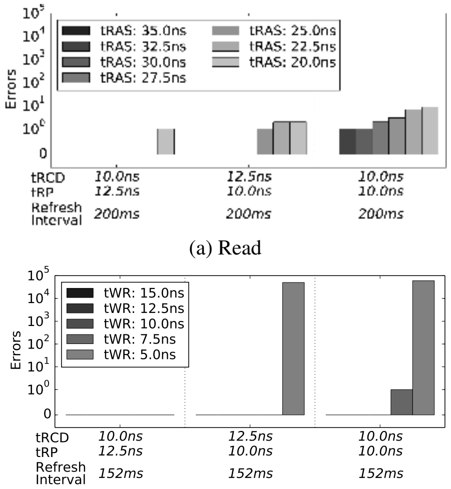
Sweeping the Refresh Interval: Figure 7a shows the number of errors when we reduce each of the four timing parameters (tRCD, tRAS, tWR, and tRP) at different refresh intervals while the temperature is kept at 85°C. We make three key observations from the figure. First, it is possible to reduce the timing parameters significantly without incurring errors. In particular, at the standard refresh interval of 64 ms, the four timing parameters can be reduced by (3.75 ns, 15 ns, 10 ns, 3.75 ns). Second, as the refresh interval increases, the number of errors starts to increase and the range of timing parameters for error-free operation decreases. In fact, increasing the refresh interval beyond 256 ms results in errors even with the standard timing parameters. Third, reducing tRCD or tRP below a certain point results in a drastic increase in the number of errors. This is because doing so causes a functional failure of DRAM — e.g., reducing tRCD below 10 ns does not give DRAM enough time to even activate the wordline completely, resulting in a malfunction of the entire row.

Sweeping the Temperature: Figure 7b shows the number of errors with reduced timing parameters at different temperatures (55°C to 85°C) while the refresh interval is kept at 512 ms. Note that the refresh interval is 8 times as long as its standard value of 64 ms, meaning that the number of errors shown in the figure are much larger than what they should be for commodity DRAM. From the figure, we find that the number of errors decreases at lower temperatures, similar to how it decreases at lower refresh intervals in the previous experiment. Therefore, we conclude that there is a significant opportunity to reduce timing parameters at lower temperatures.

7.2. Effect of Reducing Multiple Timing Parameters

The results of the previous experiment showed that there is a significant opportunity for reducing each timing parameter individually. However, we hypothesize that reducing one timing parameter may also decrease the opportunity to reduce another timing parameter simultaneously. For example, if we reduce the value of tRAS, the charge on a cell may not be fully restored by the end of the sense-amplification process. This may decrease the opportunity to reduce tRP, as the bitlines may have to be fully precharged to ensure that it can be reliably perturbed by the partially charged cell.

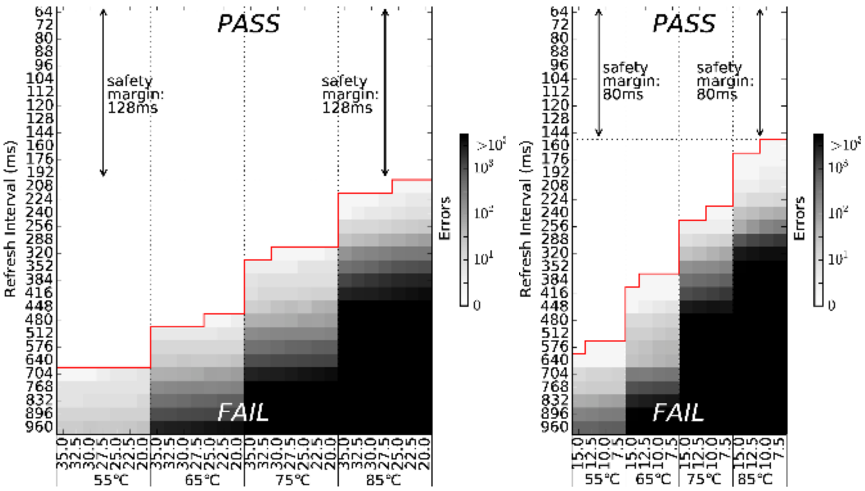
To study the potential for reducing multiple timing parameters simultaneously, we run our tests with all possible combinations of timing parameters in the following range: tRCD (12.5–10 ns), tRAS (35–20 ns), tWR (15–5 ns), tRP (12.5–10



(a) Read

(b) Write

Figure 8: Error Counts When Varying Multiple Timing Parameters Simultaneously



(a) Read Test (Varying tRAS)

(b) Write Test (Varying tWR)

Figure 9: Error Counts When Varying Temperature, Refresh Interval, and tRAS/tWR (tRCD/tRP: 12.5 ns)

ns). For each combination, we also vary the temperature from 55°C to 85°C and the refresh interval from 64 ms to 960 ms.

Figure 8 presents the number of errors for the read and write tests for multiple such combinations at 85°C. Our results validate our hypothesis. For example, although it was possible to *individually* reduce tRAS to 20 ns at 85°C and 200 ms refresh interval, tRAS can only be reduced to 32.5 ns if both tRCD and tRP are *also* reduced to 10 ns. In this section, we only present the results for specific combinations to show the effect of reducing multiple timing parameters clearly. In Section 7.4, we will present the test results with all timing parameter combinations and resulting potential reductions.

7.3. Effect of Temperature on Timing Slack

As discussed in Section 4, the length of the timing parameters is dictated by the weakest cell with the shortest retention time at 85°C. However, ensuring enough charge is stored in such a cell may not be enough to guarantee reliable operation — this is because the cell could be affected by other failure mechanisms that are difficult to foresee. Just to name one example, some DRAM cells are known to suffer from a phenomenon known as *variable retention time* (VRT) [29, 41], in which their retention time could change unpredictably between short and long values. As a counter-measure against such failures, DRAM manufacturers provide a built-in *safety-margin* in retention time, also referred to as *a guard-band* [4, 29, 70]. This way, DRAM manufacturers are able to guarantee that even the weakest cell is insured against various other modes of failure.

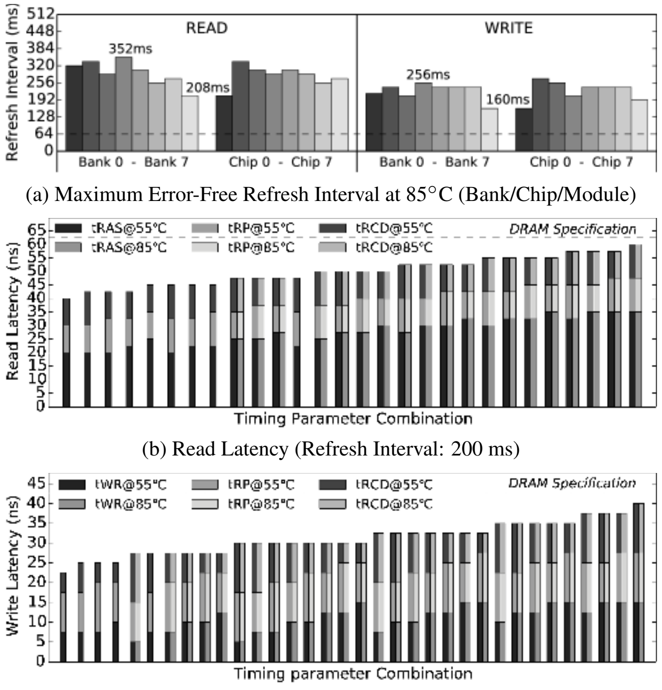
In order to quantify the safety-margin, we sweep the refresh interval from 64 ms to 960 ms. The safety-margin incorporated by the DRAM manufacturers is the difference between the highest refresh interval that exhibits no errors at 85°C and the standard refresh interval (64 ms). Figure 9 plots the number of errors (in gray scale) for varying refresh intervals for different temperatures (55°C to 85°C). For each temperature, we also vary tRAS (Figure 9a) and tWR (Figure 9b). A box at (x, y) represents the number of errors at a refresh interval y and tRAS/tWR of x. A white box indicates no errors, and a darker box indicates a large number of errors. The red line separates the error-free region from the region with at least one error.

We make several observations from the figure. First, as expected, for each temperature, increasing the refresh interval (i.e., going down on the y axis) leads to more errors. Second, for a given refresh interval and temperature, reducing the tRAS/tWR timing parameters (i.e., going right on the x axis for each temperature) also increases the number of errors. Third, at 85°C, the highest refresh interval for error-free operation is 192 ms for the read test and 144 ms for the write test — this implies a safety-margin of 128 ms for reads and 80 ms for writes. Fourth, the slack on the retention time increases with decreasing temperature, because retention time increases with decreasing temperature (Section 4). As shown in the figure, at 55°C, the margin on the retention time is 576 ms for reads and 448 ms for writes — these values are at least 4× higher than their safety-margins (at 85°C). In summary, there is significant room for reducing DRAM latency at lower temperatures while still ensuring reliable DRAM operation.

7.4. Potential Timing Parameter Reductions While Maintaining the Safety-Margin

So far, we have discussed the effect of reducing timing parameters both individually and simultaneously. We also have studied the safety-margin and the effect of the operating temperature on the slack in timing parameters. In this section, we study the possible timing parameter reductions of a DRAM module while maintaining the safety-margin.

We first measure the safety-margin of a DRAM module by sweeping the refresh interval at the worst operating temperature (85°C), using the standard timing parameters. Figure 10a plots the maximum refresh intervals of each bank and each chip in a DRAM module for both read and write operations. We make several observations. First, the maximum refresh intervals of both read and write operations are much larger than the DRAM standard (208 ms for the read and 160 ms for the write operations vs. the 64 ms standard). Second, for the smaller architectural units (banks and chips in the DRAM module), some of them operate without incurring errors even at much higher refresh intervals than others (as high as 352 ms for the read and 256 ms for the write test). This is because the error-free retention time is determined by the worst single cell in each



(a) Maximum Error-Free Refresh Interval at 85°C (Bank/Chip/Module)

(b) Read Latency (Refresh Interval: 200 ms)

(c) Write Latency (Refresh Interval: 152 ms)

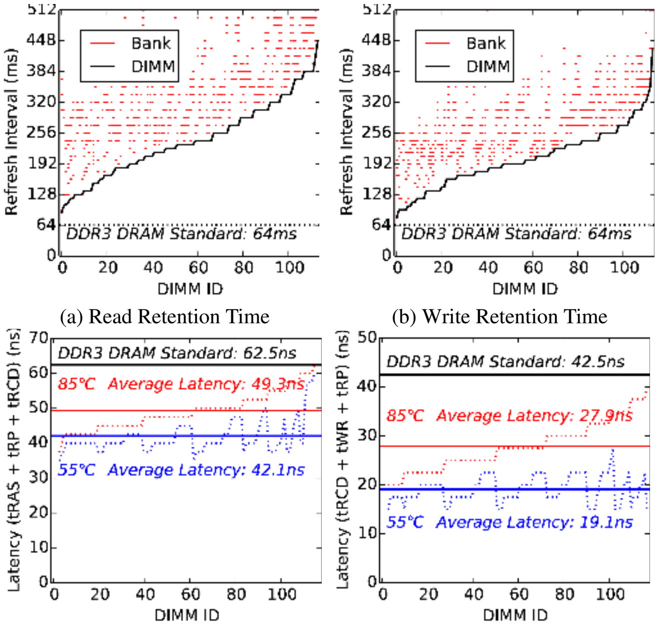
Figure 10: Latency Reductions While Maintaining the Safety-Margin

architectural component (i.e., bank/chip/module).

Based on this experiment, we define the *safe refresh interval* for a DRAM module as the maximum refresh interval that leads to no errors, minus an additional margin of 8 ms, which is the increment at which we sweep the refresh interval. The safe refresh interval for the read and write tests are 200 ms and 152 ms, respectively. We then use the safe refresh intervals to run the tests with all possible combinations of timing parameters. For each combination, we run our tests at two temperatures: 85°C and 55°C.

Figure 10b plots the error-free timing parameter combinations (tRCD, tRAS, and tRP) in the read test. For each combination, there are two stacked bars — the left bar for the test at 55°C and the right bar for the test at 85°C. Missing bars indicate that the test (with that timing parameter combination at that temperature) causes errors. Figure 10c plots same data for the write test (tRCD, tWR, and tRP).

We make three observations. First, even at the highest temperature of 85°C, the DRAM module reliably operates with reduced timing parameters (24% for read, and 35% for write operations). Second, at the lower temperature of 55°C, the potential latency reduction is even higher (36% for read, and 47% for write operations). These latency reductions are possible *while* maintaining the safety-margin of the DRAM module. From these two observations, we conclude that there is significant opportunity to reduce DRAM timing parameters *without compromising reliability*. Third, multiple different combinations of the timing parameters can form the same overall value of the timing parameters. For example, three different combinations of (tRAS, tRP, and tRCD) show the same overall value of 42.5 ns. This might enable further optimization for the most critical timing parameter at runtime. We leave the exploration of such a fine-grained optimization to future work.



(a) Read Retention Time

(b) Write Retention Time

(c) Read Latency

(d) Write Latency

Figure 11: Analysis of Multiple DIMMs

7.5. Effect of Process Variation on Timing Slack

So far, we have discussed the effect of temperature and the potential to reduce various timing parameters at different temperatures for a single DRAM module. The same trends and observations also hold true for all of the other modules. In this section, we analyze the effect of process variation by studying the results of our profiling experiments on 115 DIMMs. We also present results for intra-chip process variation by studying the process variation across different banks within each DIMM.

Figure 11a (solid line) plots the highest refresh interval that leads to correct operation across all cells at 85°C within *each DIMM* for the read test. The red dots on top show the highest refresh interval that leads to correct operation across all cells within *each bank* for all 8 banks. Figure 11b plots the same data for the write test.

We draw two conclusions. First, although there exist a few modules which *just* meet the timing parameters (with a low safety-margin), a vast majority of the modules very comfortably meet the standard timing parameters (with a high safety-margin). This indicates that a majority of the DIMMs have significantly higher safety-margins than the worst-case module *even at the highest-acceptable operating temperature of 85°C*. Second, the effect of process variation is even higher for banks within the same DIMM, explained by the large spread in the red dots for each DIMM. Since banks within a DIMM can be accessed independently with different timing parameters, one can potentially imagine a mechanism that more aggressively reduces timing parameters at a bank granularity and not just the DIMM granularity. We leave this for future work.

To study the potential of reducing timing parameters for each DIMM, we sweep all possible combinations of timing parameters (tRCD/tRAS/tWR/tRP) for all the DIMMs at both the highest acceptable operating temperature (85°C) and a more typical operating temperature (55°C). We then determine the acceptable DRAM timing parameters for each DIMM for both temperatures while maintaining its safety-margin.

Figures 11c and 11d show the results of this experiment for the read test and write test respectively. The y-axis plots the sum of the relevant timing parameters (tRCD, tRAS, and tRP for the read test and tRCD, tWR, and tRP for the write test). The solid black line shows the latency sum of the standard timing parameters (DDR3 DRAM specification). The dotted red line and the dotted blue line show the most acceptable latency parameters for each DIMM at 85°C and 55°C, respectively. The solid red line and blue line show the average acceptable latency across all DIMMs.

We make two observations. First, even at the highest temperature of 85°C, DIMMs have a high potential of reducing their access latency: 21.1% on average for read, and 34.4% on average for write operations. This is a direct result of the possible reductions in timing parameters tRCD/tRAS/tWR/tRP — 15.6%/20.4%/20.6%/28.5% on average across all the DIMMs.[3] As a result, we conclude that process variation and lower temperatures enable a significant potential to reduce DRAM access latencies. Second, we observe that at lower temperatures (e.g., 55°C) the potential for latency reduction is even greater (32.7% on average for read, and 55.1% on average for write operations), where the corresponding reduction in timing parameters tRCD/tRAS/tWR/tRP are 17.3%/37.7%/54.8%/35.2% on average across all the DIMMs.

Given that DRAM latency is a significant bottleneck for system performance, this reduction in timing parameters will directly translate to improvement in overall system performance (as we will show in Section 8.2).

7.6. Analysis of the Repeatability of Cell Failures

In order to adopt a mechanism that exploits the DRAM latency variation in real systems, it is important to understand whether the latency-margin related errors remain consistent over time. For this purpose, we specifically measured how many cells consistently experience errors under different evaluation conditions. We perform this consistency test for five different scenarios listed in Table 1. For each scenario, we choose a refresh interval for which there are at least 1000 errors. We then repeat each test for 10 iterations and report the percentage of cells that appear in all 10 iterations.

As summarized in Table 1, the first four scenarios show that a very high fraction (more than 95%) of the erroneous cells consistently experience an error over multiple iterations of the same test. Even though the overlap ratio is high, it is not 100%, which means that the characteristic of some cells may be changing over time. We believe that these effects could be related to the VRT phenomenon (Section 7.3). For the fifth scenario (where parameters are set separately for reads and writes), the repeatability of errors is the lowest at 71%. We hypothesize that this is the result of different power-noise conditions (between activation and write) for these two different operations. This is why the read and write operations need to be profiled separately, since they are likely to sensitize errors in different sets of cells.

Scenario	Overlap (%)
10 iterations of the same test	96.94
10 iterations of eight different data patterns	96.01
10 iterations of eight timing-parameter combinations	98.99
10 iterations at two different temperatures (85 vs. 65°C)	96.18
10 iterations of two different test types (read vs. write)	71.59

Table 1: Repeatability and Consistency of Erroneous Cells

8. Real-System Evaluation

We evaluate AL-DRAM on a real system, whose detailed configuration is listed in Table 2. We chose this system for its AMD processor, which offers dynamic software-based control over DRAM timing parameters at runtime [5, 6]. We paired the processor with one or more DRAM modules that have ECC (error-correction code) support. For the purpose of minimizing performance variation, we disabled several optimization features of the system (e.g., dynamic core frequency scaling, adaptive DRAM row policy, and prefetching).

System	Dell PowerEdge R415 [10]
Processor	AMD Opteron 4386 (8 cores, 3.1GHz, 8MB LLC) [5, 6]
Main Memory	6 × 4GB ECC UDIMM (Single-/Dual-Rank) DDR3-1600 (800MHz clock rate, 1.25ns cycle time) Default (tRCD/tRAS/tWR/tRP): 13.75/35.0/15.0/13.75ns Reduced (tRCD/tRAS/tWR/tRP): **10.0/23.75/10.0/11.25ns**
Storage	128GB SSD (random read/write speed: 97K/90K IOPS)
OS	Linux 3.11.0-19-generic

Table 2: Evaluated System Configuration

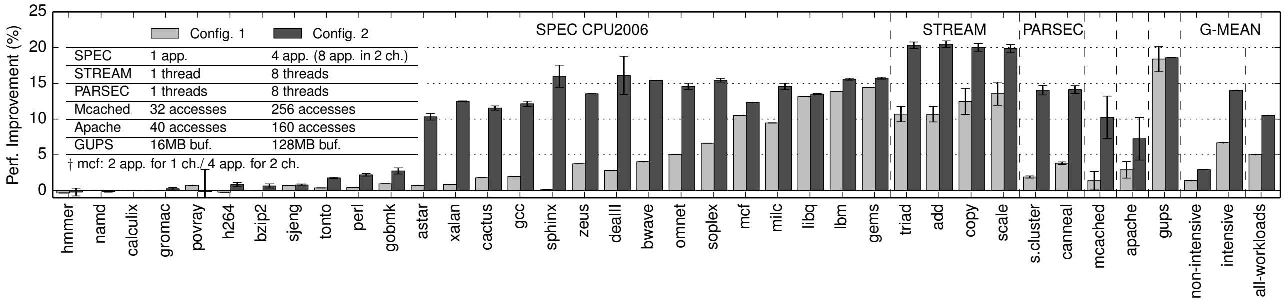
8.1. Tuning the Timing Parameters

First, we evaluate the possible latency reduction in DRAM modules without losing any data integrity. We stress the system with memory intensive workloads (99.1% CPU utilization with STREAM [44, 49] running in all eight cores) while reducing the timing parameters. The minimum values of the timing parameters that do not introduce any errors at 55°C or less define the maximum acceptable reduction in latency in the system. Table 2 shows that the potential reduction is 27%/32%/33%/18% for tRCD/tRAS/tWR/tRP, respectively. During the evaluation, the observed DRAM temperature range is 30°C–39°C (always less than 55°C, even at a very high CPU and memory utilization). Therefore, we need only one set of DRAM timing parameters for our real system evaluation.

8.2. Performance Improvement

Figure 12 shows the performance improvement of reducing the timing parameters in the memory system with one rank and one memory channel. We run a variety of different applications (SPEC, STREAM [44, 49], PARSEC [2], Memcached [1], Apache [3], and GUPS [18]) in two different configurations. The first one (Config. 1) runs only one thread and, the second one (Config. 2) runs multiple applications/threads (as described in the figure). We run each configuration 30 times (only SPEC is executed 3 times due to the large execution time), and present the average performance improvement across all the runs and their standard deviation as an error bar. Based on the last-level cache misses per kilo instructions (MPKI), we categorize our applications into memory-intensive

[3] Due to space constraints, we present only the *average* potential reduction for each timing parameter. However, detailed characterization of each DIMM can be found online at the SAFARI Research Group website [35].



	Config. 1	Config. 2
SPEC	1 app.	4 app. (8 app. in 2 ch.)
STREAM	1 thread	8 threads
PARSEC	1 threads	8 threads
Mcached	32 accesses	256 accesses
Apache	40 accesses	160 accesses
GUPS	16MB buf.	128MB buf.

† mcf: 2 app. for 1 ch./ 4 app. for 2 ch.

Figure 12: Real-System Performance Improvement with AL-DRAM (Each Error Bar Shows the Standard Deviation across Multiple Runs)

or non-intensive groups, and report the geometric mean performance improvement across all applications from each group. We draw three key conclusions from Figure 12. First, AL-DRAM provides significant performance improvement over the baseline (as high as 20.5% for the very memory-bandwidth-intensive STREAM applications [44, 49]). Second, when the memory system is under higher pressure with multi-core/multi-threaded applications, we observe significantly higher performance (than in the single-core case) across all applications from our workload pool. Third, as expected, memory-intensive applications benefit more in performance than non-memory-intensive workloads (14.0% vs. 2.9% on average). We conclude that by reducing the DRAM timing parameters using our approach, we can speed up a system by 10.5% (on average across all 35 workloads on the multi-core/multi-thread configuration).

8.3. Reliability of Reduced Timing Parameters

By reducing the timing parameters, we are also stripping away the excessive margin in DRAM's electrical charge. The remaining margin should be enough for DRAM to achieve correctness by overcoming process variation and temperature dependence (as we discussed in Section 4.3). To verify the correctness of our experiments, we ran our workloads for 33 days non-stop, and examined their and the system's correctness with reduced timing parameters. Using the reduced timing parameters over the course of 33 days, our real system was able to execute 35 different workloads in both single-core and multi-core configurations (see Figure 12) while preserving correctness and being error-free.

Note that these results do *not* absolutely guarantee that no errors can be introduced by reducing the timing parameters. Our real-system experiments are limited in their statistical significance, since they involve a small sample population (six ECC DRAM modules) over a relatively short test duration (33 days). However, DRAM manufacturers *already have the necessary testing methodology to guarantee reliable operation with reduced timing parameters that are appropriately chosen.* Existing industrial-grade methodology for measuring and ensuring reliability (at different timing parameters) is typically based on millions of hours of aggregate test time, which is clearly beyond the scope of this paper but is also clearly doable by DRAM manufacturers. Thus, we believe that we have demonstrated a proof-of-concept which shows that DRAM latency can be reduced at little-to-no apparent impact on DRAM reliability.

8.4. Sensitivity Analysis

Number of Channels, and Ranks: We analyze the impact of increasing the number of ranks and channels on the performance achieved with AL-DRAM in Figure 13. Note that adding ranks enables more memory parallelism while keeping the total memory bandwidth constant, but adding channels also increases the total memory bandwidth. The 2-channel systems we evaluate are overprovisioned for the workloads we evaluate as our workloads exert little pressure on memory in such systems. We make two major observations. First, AL-DRAM significantly improves system performance even on highly-provisioned systems with large numbers of channels, and ranks: 10.6%/5.2%/2.9% on average across our memory-intensive/multi-core workloads in a 2-rank 1-channel/1-rank 2-channel/2-rank 2-channel system, respectively. Second, the benefits AL-DRAM are higher when the system is more memory parallelism- and bandwidth-constrained, as expected, because memory latency becomes a bigger bottleneck when limited parallelism/bandwidth causes contention for memory. As on-chip computational power continues to increase at a much faster rate than the amount of off-chip memory bandwidth [20, 58] due to the limited pin count, and as future applications become increasingly data-intensive, it is very likely that future systems will be increasingly memory bandwidth constrained [51]. We conclude that AL-DRAM will likely become more effective on such future systems.

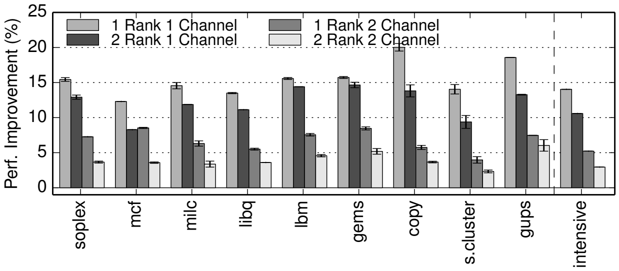


Figure 13: AL-DRAM Performance Improvement on a Real System with Different Rank/Channel Configurations

Heterogeneous Workloads: We evaluate the performance impact of AL-DRAM for heterogeneous workloads generated by combining two different applications from those listed in Figure 13. We observe that AL-DRAM provides significant weighted speedup improvement over the baseline system: 7.1% on average in a 1-rank 2-channel system.

Row Policy: Performance improvement can depend on the row buffer management policy used in the memory system. We use an in-house out-of-order multi-core simulator with a detailed DRAM timing model to analyze the sensitivity of our results to two policies (as our tested system does not have any flexibility to change row policy): open-row & closed-row [32]. We evaluate 51 multi-core workloads (randomlyselected from SPEC CPU2006, STREAM, TPC, and GUPS) in a 1-rank 2-channel system. Our evaluation shows that AL-DRAM provides similar performance improvements for both policies (11.4%/11.0% improvement for open/closed row policies over the baseline system).

Energy Efficiency: By reducing the timing parameters and overall execution time, AL-DRAM improves energy efficiency. We estimate the DRAM energy consumption of AL-DRAM using a DRAM energy calculator [47]. When using 4Gbyte modules with DDR3-1600, AL-DRAM reduces DRAM power consumption by 5.8% (in a current specification, I_{DD1} [59]). The major energy reduction is attributed to the reduction in row activation time.

9. Related Work

To our knowledge, this is the first work to *(i)* provide a detailed qualitative and empirical analysis of the relationship between process variation and temperature dependence of modern DRAM devices on the one side, and DRAM access latency on the other side (we directly attribute the relationship between the two to the amount of charge in cells), *(ii)* experimentally characterize a large number of existing DIMMs to understand the potential of reducing DRAM timing constraints, *(iii)* provide a practical mechanism that can take advantage of this potential, and *(iv)* evaluate the performance benefits of this mechanism by dynamically optimizing DRAM timing parameters on a real system using a variety of real workloads.

Several works investigated the possibility of reducing DRAM latency by either exploiting DRAM latency variation [7, 65] or changing the DRAM architecture [34, 36, 45, 61, 62, 66]. We discuss these below.

DRAM Latency Variation: Recently, Chandrasekar et al. [7] evaluate the potential of relaxing some DRAM timing parameters to reduce DRAM latency. This work observes latency variations across DIMMs as well as for a DIMM at different operating temperatures. However, there is no explanation as to why this phenomenon exists. In contrast, our work *(i)* identifies and analyzes the root cause of latency variation in detail, *(ii)* provides a practical mechanism that can relax timing parameters, and *(iii)* provides a real system evaluation of this new mechanism, using real workloads, showing improved performance and preserved reliability.

Shin et al. [65] show that recently-refreshed rows have more charge, and propose a mechanism (NUAT) to access the recently-refreshed rows with reduced latency (especially, `tRCD`/`tRAS`). Even though some of the observations in [65] are similar to ours, the approaches to leverage them are different. AL-DRAM exploits temperature dependence in a DIMM and process variations across DIMMs. NUAT uses the time difference between a row refresh and an access to the row (hence its benefits are dependent on when the row is accessed after it is refreshed). AL-DRAM provides latency benefits regardless of access pattern and NUAT provides latency benefits regardless of DRAM temperature. Therefore, the two techniques are complementary and can potentially be combined for better performance. We note that *(i)* AL-DRAM has very little overhead compared to NUAT, and *(ii)* compared to [65], our work *experimentally* analyzes the dependence of DRAM latency on temperature and process variation in *real* DRAM chips.

Low Latency DRAM Architectures: Previous works [8, 16, 34, 36, 45, 51, 61, 62, 66, 75] propose new DRAM architectures that provide lower latency. These works improve DRAM latency at the cost of either significant additional DRAM chip area (i.e., extra sense amplifiers [45, 61, 66] or additional SRAM cache [16, 75]), specialized protocols [8, 34, 36, 62] or both. Our proposed mechanism requires *no changes* to the DRAM chip and DRAM interface, and hence there is (almost) no overhead. Furthermore, AL-DRAM is largely orthogonal to these proposed designs, and can be applied in conjunction with them, providing greater cumulative reduction in latency.

Binning or Over-Clocking of DRAM: AL-DRAM has multiple sets of DRAM timing parameters for different temperatures and dynamically optimizes the timing parameters at runtime. Therefore, AL-DRAM is different from simple binning (performed by manufacturers) or over-clocking (performed by end-users [19, 52]) that are used to figure out the highest *static* frequency or lowest *static* timing parameters for DIMMs.

Other Methods for Lower Memory Latency: Other methods of reducing/hiding memory latency, e.g., by employing prefetching (e.g., [67]), memory/cache compression (e.g., [53, 54, 64]), better memory scheduling (e.g., [32, 33, 50]), or better caching (e.g., [28, 63]), are orthogonal to our approach.

10. Conclusion

This paper introduces Adaptive-Latency DRAM (AL-DRAM), a simple and effective mechanism for dynamically tailoring the DRAM timing parameters for the current operating condition without introducing any errors. The standard DRAM timing constraints are grossly overprovisioned to ensure correct operation for the cell with the lowest retention time at the highest acceptable operating temperature. We make the observation that a significant majority of DRAM modules do *not* exhibit the worst case behavior and that most systems operate at a temperature much lower than the highest acceptable operating temperature, enabling the opportunity to significantly reduce the timing constraints. Based on these observations, AL-DRAM dynamically measures the operating temperature of each DIMM and employs timing constraints optimized for *that DIMM at that temperature*. Results of our latency profiling experiments on 115 modern DRAM modules show that our approach can significantly reduce four major DRAM timing constraints by 17.3%/37.7%/54.8%/35.2 averaged across all 115 DIMMs tested. This reduction in latency translates to an average 14% improvement in overall system performance across a wide variety of memory-intensive applications run on a real multi-core system.

We conclude that AL-DRAM is a simple and effective mechanism to reduce DRAM latency. We hope that our experimental exposure of the large margin present in the standard DRAM timing constraints will inspire other approaches to optimize DRAM chips, latencies, and parameters at low cost.

Acknowledgments

We thank the anonymous reviewers for their valuable suggestions. We thank Uksong Kang, Jung-Bae Lee, and Joo Sun Choi from Samsung, and Michael Kozuch from Intel for their helpful comments. We acknowledge the support of our industrial partners: Facebook, IBM, Intel, Microsoft, Qualcomm, VMware, and Samsung. This research was partially supported by NSF (grants 0953246, 1212962, 1065112), Semiconductor Research Corporation, and the Intel Science and Technology Center for Cloud Computing. Donghyuk Lee is supported in part by a Ph.D. scholarship from Samsung and the John and Claire Bertucci Graduate Fellowship.

References

[1] Memcached. http://memcached.org/.

[2] PARSEC Benchmarks. http://parsec.cs.princeton.edu/.

[3] The Apache HTTP Server Project. http://httpd.apache.org/.

[4] J.-H. Ahn et al. Adaptive Self Refresh Scheme for Battery Operated High-Density Mobile DRAM Applications. In *ASSCC*, 2006.

[5] AMD. *AMD Opteron 4300 Series processors*. http://www.amd.com/en-us/products/server/4000/4300.

[6] AMD. BKDG for AMD Family 16h Models 00h-0Fh Processors, 2013.

[7] K. Chandrasekar et al. Exploiting Expendable Process-margins in DRAMs for Run-time Performance Optimization. In *DATE*, 2014.

[8] K.-W. Chang et al. Improving DRAM performance by parallelizing refreshes with accesses. In *HPCA*, 2014.

[9] H. David et al. Memory power management via dynamic voltage/frequency scaling. In *ICAC*, 2011.

[10] Dell. Dell PowerEdge R415 Spec Sheet. http://www.dell.com/learn/us/en/04/shared-content~data-sheets~en/documents~r415-specsheet.pdf, 2009.

[11] N. El-Sayed et al. Temperature Management in Data Centers: Why Some (Might) Like It Hot. In *SIGMETRICS*, 2012.

[12] N. El-Sayed et al. Temperature Management in Data Centers: Why Some (Might) Like It Hot. In *Technical Report, CSRG-615, University of Toronto*, 2012.

[13] P. Friedberg et al. Modeling within-die spatial correlation effects for process-design co-optimization. In *ISQED*, 2005.

[14] P. Gillingham et al. High-speed, high-reliability circuit design for megabit DRAM. In *JSSC*, 1991.

[15] T. Hamamoto et al. On the Retention Time Distribution of Dynamic Random Access Memory (DRAM). In *IEEE TED*, 1998.

[16] H. Hidaka et al. The cache DRAM architecture: A DRAM with an on-chip cache memory. *IEEE Micro*, March 1990.

[17] HP. Configuring and using DDR3 memory with HP ProLiant Gen8 Servers, 2012. http://h20000.www2.hp.com/bc/docs/support/ SupportManual/c03293145/c03293145.pdf.

[18] HPC Challenge. GUPS. http://icl.cs.utk.edu/projectsfiles/hpcc/RandomAccess/.

[19] Intel. Intel Extreme Memory Profile (Intel XMP) DDR3 Technology. http://www.intel.com/content/www/us/en/chipsets/extreme-memory-profile-ddr3-technology-paper.html, 2009.

[20] E. Ipek et al. Self-optimizing memory controllers: A reinforcement learning approach. In *ISCA*, 2008.

[21] JEDEC. Application Specific Memory for a Server Centric World. http://www.jedec.org/sites/default/files/EugeneKim_dcrp.pdf, 2012.

[22] JEDEC. DDR4 SDRAM Standard, 2012.

[23] JEDEC. Low Power Double Data Rate 3 (LPDDR3), 2012.

[24] JEDEC. *Standard No. 21C. Annex K: Serial Presence Detect (SPD) for DDR3 SDRAM Modules*, Aug. 2012.

[25] JEDEC. *Standard No. 79-3F. DDR3 SDRAM Specification*, July 2012.

[26] U. Kang et al. Co-architecting controllers and DRAM to enhance DRAM process scaling. In *The Memory Forum*, 2014.

[27] B. Keeth and R. J. Baker. *DRAM Circuit Design: A Tutorial*. Wiley-IEEE Press, 2000.

[28] S. Khan et al. Improving Cache Performance by Exploiting Read-Write Disparity. In *HPCA*, 2014.

[29] S. Khan et al. The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study. In *SIGMETRICS*, 2014.

[30] K. Kim and J. Lee. A New Investigation of Data Retention Time in Truly Nanoscaled DRAMs. In *EDL*, 2009.

[31] Y. Kim et al. Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In *ISCA*, 2014.

[32] Y. Kim et al. ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers. In *HPCA*, 2010.

[33] Y. Kim et al. Thread Cluster Memory Scheduling: Exploiting Differences in Memory Access Behavior. In *MICRO*, 2010.

[34] Y. Kim et al. A case for exploiting subarray-level parallelism (SALP) in DRAM. In *ISCA*, 2012.

[35] D. Lee et al. Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case. http://www.ece.cmu.edu/~safari/tools/aldram-hpca2015-fulldata.html.

[36] D. Lee et al. Tiered-latency DRAM: A low latency and low cost DRAM architecture. In *HPCA*, 2013.

[37] J. Lee et al. Simultaneously Formed Storage Node Contact and Metal Contact Cell (SSMC) for 1Gb DRAM and Beyond. In *IEDM*, 1996.

[38] Y. Li et al. DRAM Yield Analysis and Optimization by a Statistical Design Approach. In *IEEE TCSI*, 2011.

[39] J. Lin et al. Software Thermal Management of DRAM Memory for Multicore Systems. In *SIGMETRICS*, 2008.

[40] J. Lin et al. Thermal Modeling and Management of DRAM Systems. In *ISCA*, 2007.

[41] J. Liu et al. An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms. In *ISCA*, 2013.

[42] J. Liu et al. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *ISCA*, 2012.

[43] S. Liu et al. Hardware/software techniques for DRAM thermal management. In *HPCA*, 2011.

[44] J. D. McCalpin. STREAM: Sustainable Memory Bandwidth in High Performance Computers. Technical report, University of Virginia, 1991-2007.

[45] Micron. RLDRAM 2 and 3 Specifications. http://www.micron.com/products/dram/rldram-memory.

[46] Micron. Mobile DRAM Power-Saving Features and Power Calculations. http://www.micron.com/~/media/documents/products/technical-note/dram/tn4612.pdf, 2005.

[47] Micron. DDR3 SDRAM System-Power Calculator, 2010.

[48] Y. Mori et al. The origin of variable retention time in DRAM. In *IEDM*, 2005.

[49] T. Moscibroda and O. Mutlu. Memory Performance Attacks: Denial of Memory Service in Multi-core Systems. USENIX Security, 2007.

[50] S. P. Muralidhara et al. Reducing memory interference in multicore systems via application-aware memory channel partitioning. In *MICRO*, 2011.

[51] O. Mutlu. Memory Scaling: A Systems Architecture Perspective. In *MemCon*, 2013.

[52] Nvidia. Extreme DDR3 Performance with SLI-Ready Memory. http://www.nvidia.com/docs/IO/52280/NVIDIA_EPP2_TB.pdf, 2008.

[53] G. Pekhimenko et al. Linearly Compressed Pages: A Main Memory Compression Framework with Low Complexity and Low Latency. In *PACT*, 2012.

[54] G. Pekhimenko et al. Base-Delta-Immediate Compression: A Practical Data Compression Mechanism for On-Chip Caches. In *PACT*, 2012.

[55] Rambus. DRAM Power Model. http://www.rambus.com/energy, 2010.

[56] B. Razavi. *Design of Analog CMOS Integrated Circuits*. McGraw-Hill, 2000.

[57] P. J. Restle et al. DRAM variable retention time. In *IEDM*, 1992.

[58] B. M. Rogers et al. Scaling the Bandwidth Wall: Challenges in and Avenues for CMP Scaling. In *ISCA*, 2009.

[59] Samsung. DRAM Data Sheet. http://www.samsung.com/global/business/semiconductor.

[60] S. R. Sarangi et al. Varius: A model of process variation and resulting timing errors for microarchitects. In *IEEE Transactions on Semiconductor Manufacturing*, 2008.

[61] Y. Sato et al. Fast Cycle RAM (FCRAM); a 20-ns random row access, pipe-lined operating DRAM. In *Symposium on VLSI Circuits*, 1998.

[62] V. Seshadri et al. RowClone: Fast and Energy-efficient in-DRAM Bulk Data Copy and Initialization. In *MICRO*, 2013.

[63] V. Seshadri et al. The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing. In *PACT*, 2012.

[64] A. Shafiee et al. MemZip: Exploring Unconventional Benefits from Memory Compression. In *HPCA*, 2014.

[65] W. Shin et al. NUAT: A Non-Uniform Access Time Memory Controller. In *HPCA*, 2014.

[66] Y. H. Son et al. Reducing Memory Access Latency with Asymmetric DRAM Bank Organizations. In *ISCA*, 2013.

[67] S. Srinath et al. Feedback Directed Prefetching: Improving the Performance and Bandwidth-Efficiency of Hardware Prefetchers. In *HPCA*, 2007.

[68] A. J. van de Goor and I. Schanstra. Address and Data Scrambling: Causes and Impact on Memory Tests. In *DELTA*, 2002.

[69] T. Vogelsang. Understanding the Energy Consumption of Dynamic Random Access Memories. In *MICRO*, 2010.

[70] M.-J. Wang et al. Guardband determination for the detection of off-state and junction leakages in DRAM testing. In *Asian Test Symposium*, 2001.

[71] Xilinx. *Virtex-6 FPGA Integrated Block for PCI Express*, 2011. http://www.xilinx.com/support/documentation/ip_documentation/mig/v3_92/ug406.pdf.

[72] Xilinx. *ML605 Hardware User Guide*, Oct. 2012. http://www.xilinx.com/support/documentation/boards_and_kits/ug534.pdf.

[73] Xilinx. *Virtex-6 FPGA Memory Interface Solutions*, 2013. http://www.xilinx.com/support/documentation/ip_documentation/mig/v3_92/ug406.pdf.

[74] D. Yaney et al. A meta-stable leakage phenomenon in DRAM charge storage - Variable hold time. In *IEDM*, 1987.

[75] Z. Zhang et al. Cached DRAM for ILP processor memory access latency reduction. *IEEE Micro*, July 2001.

[76] Q. Zhu et al. Thermal management of high power memory module for server platforms. In *ITHERM*, 2008.