A Study of DRAM Failures in the Field

Vilas Sridharan, Dean Liberty
RAS Architecture
AMD, Inc.
Boxborough, MA, USA
{vilas.sridharan, dean.liberty}@amd.com

***Abstract*—Most modern computer systems use dynamic random access memory (DRAM) as a main memory store. Recent publications have confirmed that DRAM errors are a common source of failures in the field. Therefore, further attention to the faults experienced by DRAM sub-systems is warranted. In this paper, we present a study of 11 months of DRAM errors in a large high-performance computing cluster. Our goal is to understand the failure modes, rates, and fault types experienced by DRAM in production settings.**

We identify several unique DRAM failure modes, including single-bit, multi-bit, and multi-chip failures. We also provide a deterministic bound on the rate of transient faults in the DRAM array, by exploiting the presence of a hardware scrubber on our nodes.

We draw several conclusions from our study. First, DRAM failures are dominated by permanent, rather than transient, faults, although not to the extent found by previous publications. Second, DRAMs are susceptible to large multi-bit failures, such as failures that affect an entire DRAM row or column, indicating faults in shared internal circuitry. Third, we identify a DRAM failure mode that disrupts access to other DRAM devices that share the same board-level circuitry. Finally, we find that chipkill error-correcting codes (ECC) are extremely effective, reducing the node failure rate from uncorrected DRAM errors by 42x compared to single-error correct/double-error detect (SEC-DED) ECC.

Keywords - soft error; hard error; single-event upset; DRAM; memory; reliability.

I. INTRODUCTION

Modern computer systems have long used dynamic random access memory (DRAM) as a main memory store, and architects have focused on the problems posed by DRAM failures for decades [1]. Furthermore, the amount of DRAM in computer systems continues to increase every year, and is predicted to increase 50x compared to 2009 levels by 2024 [2]. This implies that DRAM failures will be a growing concern for system architects and designers. Therefore, it is somewhat surprising that, until recently [3] [4] [5], relatively few studies existed of DRAM failures in the field. This relative lack of information about DRAM failures leaves architects to guess at the type, kind, and rate of DRAM failures to expect in modern systems.

A poor understanding of DRAM fault mechanisms can lead to the design of features that are poorly suited to the faults that actually occur. For example, memory scrubbing is a memory reliability technique that is often applied in conjunction with error-correcting codes (ECCs) [6]. A scrubber reads a memory location, fixes any errors that are correctable by the ECC, and writes the correct data back to the same location. This technique can correct a transient (non-permanent) fault and is thus most useful in a system that experiences many transient faults [7]. However, a scrubber cannot correct a permanent fault, and thus will not benefit memory that experiences a high rate of such faults.

In this paper, we present a study of DRAM failures from the "Jaguar" high-performance computing cluster at Oak Ridge National Laboratories (ORNL). Jaguar is a cluster of 18,688 two-socket nodes. During our experiment, each socket contained a 6-core AMD Opteron™ processor and four 2GB DDR-2 DIMMs. The ECC on Jaguar provides chipkill capability, which allows the memory sub-system to tolerate the failure of any single DRAM device [8]. Each node ran a version of the Linux operating system configured to log error events on the console.

Our experiment duration was approximately 11 months, from November 2009 to October 2010, comprising approximately fifty million DIMM-days of data. The Jaguar system was brought online in January 2009, so our experiment covered the second year of production lifetime for the majority of the DIMMs. The Jaguar system operators replaced DIMMs after the first uncorrected error from that DIMM. During our experiment, the system was typically 90-95% utilized except for planned maintenance windows.

There are several contributions of this research:

- A study of the impact of DRAM faults, failures, and errors in a production environment. A key finding of this study is that DRAM failure rate is a better predictor of DIMM health than the error rate.
- A method to provide a deterministic bound on the DRAM transient fault rate by using the memory scrubber on each node. We find that DRAMs experience both transient and permanent faults in the field, but that permanent faults constitute at least 70% of all DRAM failures.
- A detailed study of multi-bit failure modes, including row, column, bank, and multiple-bank faults. We find that multi-bit faults such as row, column and bank faults constitute almost 50% of the DRAM failures in Jaguar's memory system. We find that row faults tend to affect an entire row, while column faults tend to cluster around a few rows. We also identify a multiple-rank failure mode, where a fault in one DRAM affects multiple DRAMs sharing

SC12, November 10-16, 2012, Salt Lake City, Utah, USA
978-1-4673-0806-9/12/$31.00 ©2012 IEEE

the same board-level circuitry. These faults constitute almost 5% of the faults in our data set.
- An analysis of the effect of chipkill on the uncorrected error rate of a node. We find that chipkill reduces the node failure rate from uncorrected DRAM errors by 42x relative to SEC-DED ECC. We also find that a DIMM with a multiple-bank or multiple-rank fault has a significantly larger risk of a future uncorrected error than a DIMM with a single-bit or no fault.

The rest of this paper is organized as follows. Section II defines the terminology we use in this paper. Section III discusses related studies and describes the differences in our study and methodology. Section IV discusses the architecture and configuration of the Jaguar system and our data collection methodology. Section V presents aggregate system-level statistics, and Sections VI-X present our analysis of DRAM fault types and failure modes. Section XI discusses implications of our findings and presents our conclusions.

II. TERMINOLOGY

In this paper, we distinguish between a fault and an error using the following taxonomy [9]:

- A **fault** refers to an underlying cause of a failure, such as a particle-induced bit flip or a stuck-at bit.
- A **failure** is a deviation of a DRAM from its specified operation, meaning that it is not possible to read from or write to a location in that DRAM, or that a location in the DRAM returns data that differs from the data that was last written to the location. Note that ECC will often prevent a DRAM failure from propagating to higher architectural levels, but if the ECC cannot correct the failure, a node or system failure may result. A single fault normally causes a single DRAM failure, which can be incorrect or inaccessible data in one or more memory locations.
- An **error** is a symptom of a failure. A DRAM failure will return an error when it is read if the node provides higher-level detection such as parity or ECC. Note that each DRAM failure can cause many errors if the failed locations are accessed multiple times.

DRAM failures can be caused by many different types of faults [10]. These faults can be classified as:

- **Transient faults**, which cause incorrect data to be read from a memory location until the location is overwritten with correct data. These faults occur randomly and are not indicative of device damage [11]. Particle-induced upsets ("soft errors"), which have been extensively studied in the literature [1] [11], are one type of transient fault.
- **Hard faults**, which cause a memory location to consistently return an incorrect value (e.g., a stuck-at-0 fault). Hard faults are permanent and can be repaired only by replacing the faulty device [12].
- **Intermittent faults**, which cause a memory location to sometimes return incorrect values. Unlike hard faults, intermittent faults occur only under specific conditions (e.g., elevated temperature) [10]. Unlike transient faults, however, an intermittent fault is indicative of device damage or malfunction.

Distinguishing a hard fault from an intermittent fault in a running system requires knowing the exact memory access pattern to determine whether a memory location returns the wrong data on every access. In practice, this is impossible in a large-scale field study such as ours. Therefore, we group intermittent and hard faults together in a category of **permanent** faults.

Permanent faults have two distinguishing characteristics. First, a memory location that experiences an error due to a permanent fault has an increased probability of subsequent errors. Second, a permanent fault requires device replacement to repair. By contrast, a device that experiences a transient fault can be repaired by overwriting the location with correct data, and is at no greater risk of subsequent errors once the fault has been repaired.

In this paper, we present data on DRAM failures and DRAM failure rates. As noted, DRAM failures can be due to either permanent or transient faults. Unless otherwise specified, failure counts and rates in this paper include failures due to both permanent and transient faults.

A memory sub-system typically provides reliability features such as ECC or parity to detect and correct errors. Therefore, we classify DRAM errors as corrected, uncorrected, or undetected [13]:

- A **corrected error** is detected and corrected by the hardware and does not impact the correct operation of software. A corrected error may result in temporary or permanent performance degradation on the node.
- An **uncorrected error** is detected by the hardware but cannot be corrected. An uncorrected error can result in the termination of a process or a crash of the entire node, depending on the reliability features of the node.
- An **undetected error** is not detected by hardware. An undetected error may result in silent data corruption, or may be benign if the data in error does not affect program output [14].

III. RELATED WORK

During the past few years, several studies have been published studying DRAM failures in the field. In 2006, Schroeder and Gibson published a study on failure data from high-performance computer systems at Los Alamos National Labs [3]. In 2007, Li et al. published a study of memory errors on three different data sets, including a server farm of an Internet service [5]. In 2009, Schroeder et al. published a large-scale field study using Google's server fleet [4]. In 2010, Li et al. published an expanded study of memory errors on an Internet server farm and other sources [15]. In 2012, Hwang et al. published an expanded study on Google's server fleet as well as two IBM Blue Gene clusters [16], and

Sridharan and Liberty presented a summary of DRAM errors [17].

Due to the sheer size of our data set, our study has significantly more statistical power than many of these previous studies [3] [5] [15]. As a result, the data in our study is more likely to be representative of DRAM failures in general than smaller-scale studies. Furthermore, the scale of our study allows us to evaluate techniques such as chipkill that operate on relatively rare events such as DRAM device failures. Our study also has significantly more detail and analysis on failure patterns and modes than Sridharan [17].

Two of the cited studies have similar or larger scale than ours. First, Schroeder et al. analyzed error rates with respect to several factors such as DIMM capacity and organization, system utilization, temperature, and age [4]. Several of the study's findings (e.g., failure rates, incidence of transient faults) differ substantially from our findings. This is due to a methodological difference. Schroeder et al. calculated rates using the number of observed errors over time [18]. We calculate rates using the number of observed DRAM failures over time.

This has a significant impact on our results and makes it difficult to compare the two studies. For example, a failure due to a permanent fault will persist until the failing DRAM is replaced, potentially days or weeks after the first detected error. During this time, every access to the failing location can result in an error. In contrast, a transient fault persists only until the memory element is written, which may be just a few DRAM cycles after the first error [11]. As a result, a typical transient fault will generate many fewer errors than a typical permanent fault.

More recently, Hwang et al. analyzed error logs for patterns of failing addresses, the impact of chipkill on the rate of corrected errors, and the effectiveness of page retirement techniques [16]. However, the authors are unable to conclusively differentiate errors caused by transient faults from those caused by permanent faults. For example, the authors report a significant incidence of repeat errors from the same address, but it is impossible to determine whether this is due to a permanent fault or to a transient fault that was read multiple times before being overwritten. The authors also perform their analyses on error rates, making it difficult to differentiate patterns due to failure modes from artifacts of system activity. The study also does not examine the impact of chipkill on a node's uncorrected error rate.

There have also been many laboratory studies of DRAM failures dating back several decades [19] [20] [21] [22]. Most of these studies focus solely on particle-induced transient faults. Of particular interest are the studies by Borucki et al. and Quinn et al., which identified large multi-bit failures as a possible consequence of particle strikes [21] [22]. However, while all these studies examine failures caused by neutron or alpha particle strikes, they do not necessarily represent the total range of failures experienced by DRAM in the field.

IV. System Configuration and Data Collection

The Jaguar system at Oak Ridge National Laboratory is a cluster of 18,688 two-socket nodes. During our experiment, each two-socket node contained two 6-core 2.6GHz AMD Opteron™ processors and four memory channels. A memory channel consists of 72 DQ signals, 64 for data bits and eight for check bits. Each memory channel is populated with two 2GB DDR-2 DIMMs for a total of 4GB of DRAM per memory channel, or 16GB per node. Each DIMM consists of 18 DRAM devices. Therefore, the Jaguar system has approximately 2.69 million DRAM devices in total.

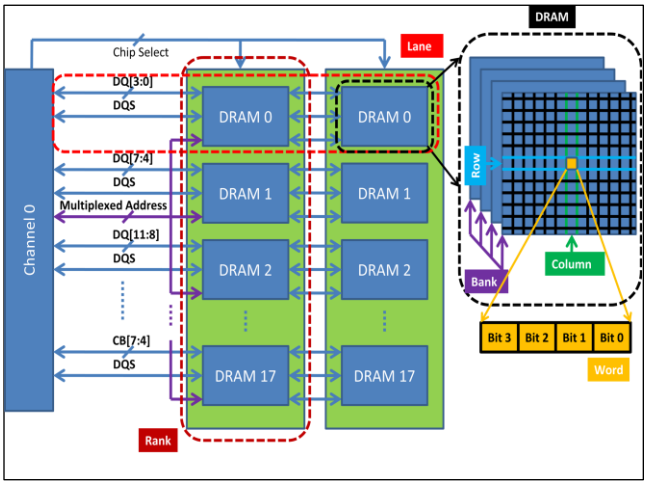


Figure 1. A simplified logical view of a single channel of the DRAM memory subsystem on each node in the Jaguar system.

A. DRAM Configuration

Figure 1 shows a simplified logical view of a single memory channel. Each DIMM contains one **rank** of 18 DRAM devices, each with four data (DQ) signals (known as an x4 DRAM device). Sixteen of the DRAM devices are used to store data bits and two are used to store check (ECC) bits. A memory request accesses all devices on a rank in parallel by activating a chip-select signal dedicated to that rank.

A **lane** is a group of DRAM devices that share data (DQ) signals. In Jaguar, each memory channel has 18 lanes, each with two DRAM devices. DRAMs in the same lane also share a strobe (DQS) signal, which is used as a source-synchronous clock signal for the data signals.

Each DRAM device contains eight internal **banks** that can be accessed in parallel. Logically, each bank is organized into **rows** and **columns**. Row and column addresses are delivered on a shared command/address bus. Each row/column address pair uniquely identifies a 4-bit **word** in the DRAM device. Internally, a DRAM bank is constructed as multiple **sub-arrays** that are accessed in parallel and that each contribute exactly one bit to every 4-bit word [23].

B. Error Detection and Correction

DRAM sub-systems are typically protected by an error detection and correction code, referred to as an ECC. A conventional ECC stores several additional check bits along with each data word [24]. These check bits are encoded to allow detection of some errors in both the data and check bits. For certain errors, ECCs can also identify the specific data bits in error, allowing the hardware to correct the data.

There are many variants of ECC in use. Single Error Correct-Double Error Detect (SEC-DED) ECC allows correction of a single-bit error and detection of a double-bit

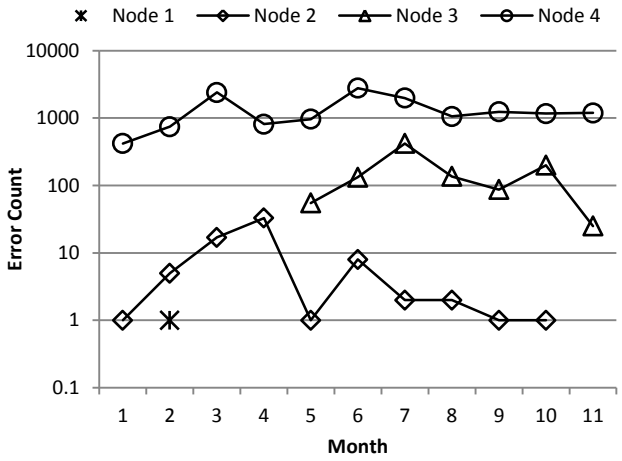


Figure 2. Corrected errors per month for several representative nodes. The number of errors per node varies substantially, but each node experienced exactly one DRAM failure. Node 1 experienced a transient fault, while nodes 2-4 each experienced a permanent fault.

error. SEC-DED was frequently used on DRAM sub-systems for many years, and is still in common use on CPU cache memories [7] [8].

The ECC used on the Jaguar system, by contrast, is a single-symbol correct (SSC) ECC. A symbol is a group of eight adjacent data bits (bits 0-7, 8-15, etc.). This code can correct any number of bit errors within one symbol. The memory system is laid out such that the bits from a single DRAM device contribute to only one symbol in the ECC word. This allows the ECC to correct any number of failing bits from a single DRAM device, a capability referred to as chipkill [8].

ECC detection and correction is performed on every read access to DRAM. In addition, each node in the Jaguar system also has a hardware memory scrubber [6]. The hardware scrubber periodically reads every location in memory. Its goal is to correct any latent (unaccessed) correctable errors before a second failure creates an uncorrectable error in the ECC word. The time that the scrubber takes to cycle through every location in a node's DRAM memory is a **scrub interval**. A scrub interval on nodes in the Jaguar system is on the order of a few hours.

C. Data Collection Methodology

The data collection infrastructure on Jaguar contains both hardware and software components. These components work together to record detected errors on each node, including both corrected and uncorrected errors. Note that this infrastructure does not attempt to record any undetected errors.

The hardware memory controller in each node logs corrected error events in registers provided by the x86 machine-check architecture (MCA) [25]. Each node's operating system is configured to poll the MCA registers once every few seconds and record any events it finds to the node's console log. These console logs are then collected by the administrators and saved for later analysis. The console logs contain a variety of information, including the physical address associated with the error, the time the error was recorded, the type of error (corrected or uncorrected), and the ECC syndrome associated with the error.

Hardware can log many corrected errors during a single software polling interval. Because there is only one set of MCA registers per core, hardware cannot guarantee that all errors are communicated to software and recorded in the console logs. To track this case, the x86 MCA registers provide an overflow bit to indicate that at least one error was not logged [25]. However, this bit gives no information on how many errors were missed. Therefore, a node's console log can be viewed as a statistical sample of all corrected errors on the node.

The hardware also logs uncorrected error events to the MCA registers. These errors may result in an immediate reset of the node and thus cannot reliably be captured by the software polling mechanism. The MCA registers preserve their values across a warm reset, however, and uncorrected errors are logged after the node resets.

D. Limitations

Information not tracked by this research limits the analyses that we can perform. We do not track DIMM vendor information and thus cannot distinguish differences in vendor failure rates. We also do not track temperature or other environmental conditions, which have been shown to have an effect on certain DRAM failures [10]. Finally, because all DIMMs in the system are of the same capacity and organization, we cannot analyze differences due to these factors.

V. Aggregate Statistics

In this section, we present aggregate data on failures and errors across the Jaguar system. The failure rates observed in our data set indicate that two DRAM devices (0.00008% of all DRAM devices in Jaguar) will experience more than one fault in a single year. By contrast, more than 2,000 DRAM devices (0.09% of all DRAMs) experience a single fault, a difference of approximately three orders of magnitude. Therefore, we make the simplifying assumption that each DRAM in our system experiences exactly one fault over our experiment duration.

A. Node Behavior Over Time

It is first instructive to look at the behaviors of individual nodes in our data set. Figure 2 shows a plot of corrected errors over time for several representative nodes. Each node had exactly one failing DRAM, and the errors in each node were confined to a single memory location.

Node 1 had only one error reported during the entire 11-month period. Node 2 had between zero and fifty errors per month. As discussed, we can say with high probability that these errors are likely due to a single fault. In contrast, nodes 3 and 4 each recorded hundreds or thousands of errors per month. Due to the sampling behavior noted in Section IV.C, each of these nodes may have actually experienced millions of corrected errors per month. Node 3 did not experience any errors until a few months into the measurement window, indicating that this node developed a fault during the measurement interval.

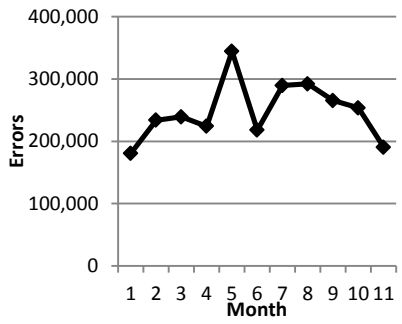


Figure 3. Corrected DRAM errors per month.

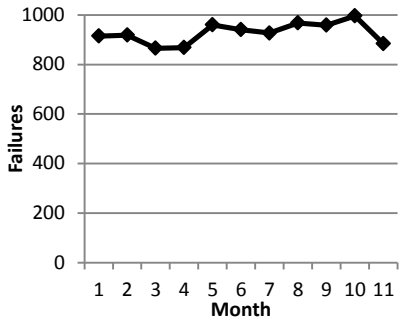


Figure 4. Activated DRAM failures per month.

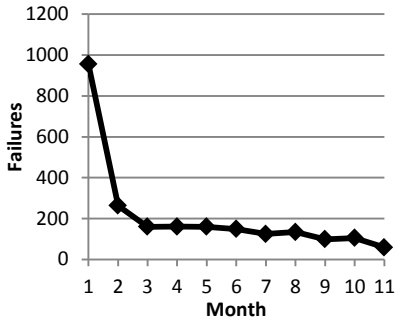


Figure 5. New DRAM failures per month.

% Failing DRAM devices	0.09%
% of DIMMs with a failing DRAM	1.6%
Failure rate (FIT/Mbit)	0.066
Failure rate (FIT/DRAM device)	66.1

Table 1. Failure rates and fraction of failing devices in the Jaguar system. Failure rate is given in FIT, or failures per billion device-hours of operation.

B. Total Errors and Failures

In this section, we examine the total number of logged errors and the total number of failures per month in our data set in order to determine the overall health of the DRAM population in the system.

Figure 3 shows the total number of corrected errors logged per month in our data set. The average number of errors is just under 250,000 per month, but the number of errors varies by more than 90%, from a low of 180,559 in the first month to a high of 344,365 in the fifth month of the study. This translates to an average of 6.6 errors per node per month across the entire system.

Figure 4 and Figure 5 show the total number of failures in our data set. Figure 4 shows the total number of **activated** DRAM failures per month in our data set. We say a DRAM failure is activated in a given month if the DRAM produced at least one error in that month. For example, a DRAM failure that caused errors in months 3, 4, 5, and 7, would be counted exactly once in each of those months. On average, there are 927.5 activated DRAM failures per month, and the number of activated failures varies by 15%, from a low of 866 in month three to a high of 996 in month ten.

Figure 5 shows the number of **new** failures per month in our data set. Each DRAM failure is counted as new in the first month that it produced any errors. The spike in months one and two are due to failures accumulated in the DRAMs before the start of our experiment. Note that the rate of new failures in months 3-11 appears to decline slightly over time. The majority of the DIMMs in Jaguar were installed in January 2009, so our experiment is in the second year of the DRAM devices' lifetimes. Therefore we hypothesize that the DRAMs are still in the "early life" phase of the classic hardware reliability bathtub curve and thus are expected to show a declining failure rate [26].

One major observation is that the number of failures per month shows substantially lower variability than the number of errors per month. This can be attributed in part to changes over time in access patterns, utilization, and workloads, which all affect the frequency of error detection. The number of errors may also depend on external factors such as temperature that cause intermittent faults to manifest more frequently [10]. For example, the number of errors increased by 53.8% between months 4 and 5, but the number of activated failures increased by only 10.6%. Our conclusion from this data is that the failure rate is a better assessment of the health of a DIMM population than the error rate, because the error rate can vary widely due to factors unrelated to DIMM health.

C. DRAM Failure Rates

Table 1 shows aggregate failure rates for DRAM in the system, including the failure rate per megabit and fraction of DRAMs and DIMMs experiencing a failure. The failure rate is the average rate of new failures per month, excluding months one and two to eliminate the spike at the start of the experiment.

The table shows that 1.6% of DIMMs, or 0.09% of DRAM devices, experienced a failure during the experiment. The calculated failure rate of 0.066 FIT/Mbit translates to one failure approximately every six hours across the Jaguar system. Obviously, DRAM failures are not rare events in large systems such as Jaguar. Note that this failure rate is not directly comparable to the failure rates calculated by Schroeder et al. due to the methodology differences discussed in Section III, although the number of failures per DIMM is in line with their corrected-error incidence per DIMM for DDR-2 DRAM [4]. Our calculated failure rate is also in line with other published studies [5].

VI. TRANSIENT VERSUS PERMANENT FAULTS

To estimate the rate of different fault types in the DRAM array, we must ascertain the type of fault based on the pattern of errors logged on each node. Specifically, we would like to differentiate transient faults from permanent faults, which is difficult because we do not know workload memory access patterns. Thus, guarantees on fault classes are difficult to achieve without laboratory testing of each faulty DRAM.

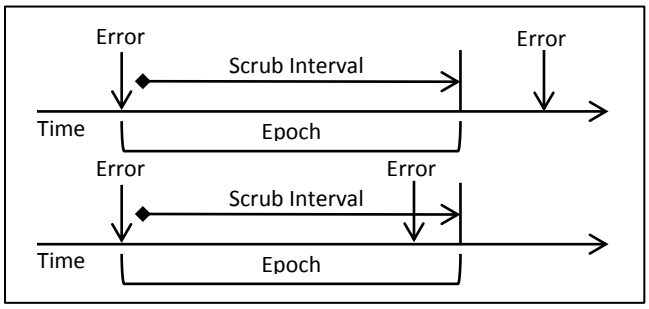


Figure 6. A DRAM that produces errors in two epochs (top) has a permanent fault. A DRAM that produces errors in one epoch (bottom) may have a transient fault.

However, Jaguar's memory scrubber guarantees that every DRAM location is written at least once per scrub interval. This enables us to place an upper bound on the rate of transient faults in the DRAM array. A DRAM that produces errors in only one scrub interval may be experiencing a transient fault. A DRAM that experiences errors in multiple scrub intervals, by contrast, is definitely not experiencing a single transient fault. The DRAM may be experiencing a single permanent fault, or it may be experiencing multiple faults. Given the low probability of multiple faults in a single DRAM device, we assume that all such cases are a single permanent fault.

Our process for making this determination is shown in Figure 6. For each DRAM, we group time into **epochs.** An epoch begins with an error and lasts for one full scrub interval. If a DRAM reports errors only within one epoch, we classify the fault as transient. If a DRAM reports errors in multiple epochs, we classify the fault as permanent.

Figure 7 plots all faulty DRAMs as a function of the number of epochs in which that DRAM produced at least one error. We find that 28.8% of all faulty DRAMs had errors in just one epoch. We classify these as potential instances of transient faults. In contrast, 10.2% of faulty DRAMs had errors in two epochs and 61.0% had errors in three or more epochs.

This is strong evidence that DRAM failures are dominated by permanent rather than transient faults. Further, our analysis is an upper bound on the rate of transient faults. If a node writes to a memory location within an epoch, it would overwrite any existing transient faults. A fault that produced errors after a write operation, even if they were confined to a single epoch, should be classified as a permanent fault. Because we do not know each node's memory access patterns, however, our analysis would classify this fault as transient. Since an epoch is several hours long, it is possible that many of the faults we identify as transient will fit this pattern.

VII. Nodes with Multiple Failing DRAMs

Some nodes in our system log errors from multiple DRAM devices. Based on our observed failure rates, we would expect several dozen nodes to experience faults in multiple DRAMs during the experiment. In general, we treat these cases as multiple independent faults, but we would like to determine if there is any relationship between DRAM failures on the same node. Therefore, we examine nodes that logged errors in multiple DRAMs to determine whether these errors were coincident in time.

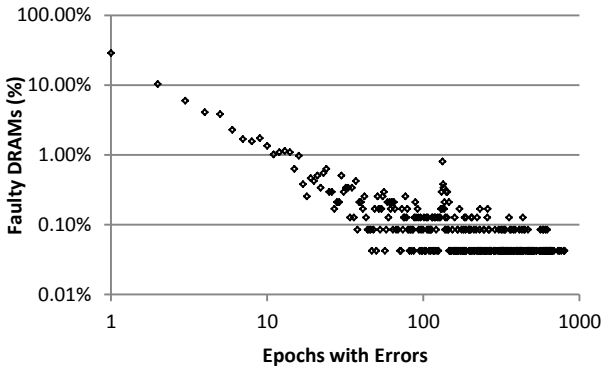


Figure 7. Faulty DRAMs as a function of the number of epochs with at least one error. 28.8% of faulty DRAMs had errors in just one epoch.

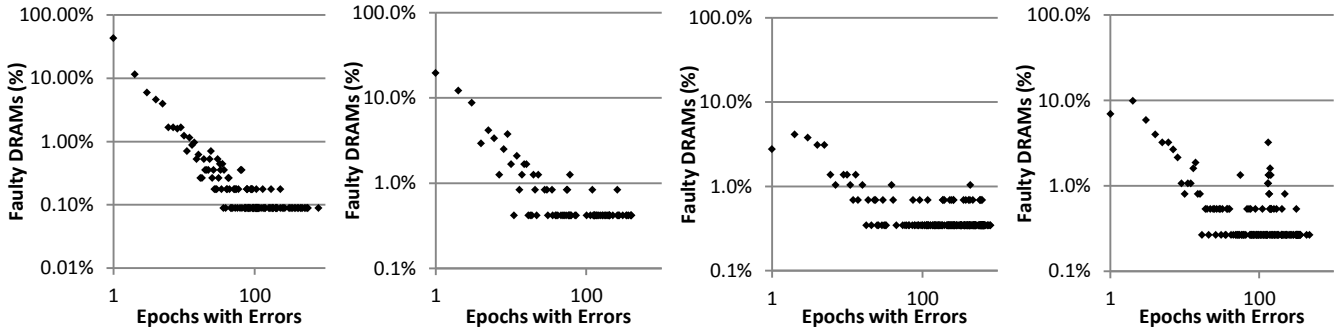
# of DRAMs with Errors	Epochs with Errors		
	1	2	3
1	26.3%	8.8%	5.2%
2	0.5%	1.0%	0.5%
3	0.1%	0.05%	0.05%

Table 2. Nodes that logged errors from 1-3 DRAMs and reported errors in 1-3 epochs, expressed as a percentage of all nodes with errors. 1.0% of nodes logged errors from two DRAMs and reported errors in two epochs, indicating two DRAMs on these nodes experienced transient faults. 0.5% of nodes logged errors in two DRAM devices but reported errors in just one epoch, indicating potential multiple-rank faults.

Table 2 shows nodes that logged errors from 1, 2, and 3 DRAMs, and reported errors from 1, 2, or 3 epochs, expressed as a percentage of all nodes with errors. We find that 26.3% of nodes with errors log errors from one DRAM device in just one epoch. These nodes experienced a single transient fault. The table also shows that 1.0% of nodes log errors from two DRAM devices in two epochs, one from each DRAM device. We conclude that two DRAMs experienced a transient fault on these nodes.

Table 2 shows that 8.8% and 5.2% of nodes log errors from one DRAM device in two and three epochs, respectively. These nodes are much more common than nodes that experience a single transient fault in two DRAMs. This provides further justification for treating these faults as single permanent faults, since we would expect the opposite to be true if these were multiple transient faults. Similarly, 0.5% of nodes logged errors from two DRAM devices in three epochs. We conclude that one DRAM on these nodes experienced a transient fault, while another DRAM experienced a permanent fault.

Table 2 highlights an interesting phenomenon. 0.5% of nodes logged errors from two DRAM devices but in just one epoch. There are two possible explanations for this. First, this node may have experienced simultaneous independent transient faults, but this type of multiple-fault event should be very rare. Alternatively, DRAMs may exhibit a failure mode that corrupts shared board-level circuitry, resulting in errors from other DRAMs sharing the same circuitry. Section VIII presents further evidence for this type of failure mode.



(a) Single-Bit Faults (b) Single-Column Faults (c) Single-Row Faults (d) Single-Bank Faults

Figure 8. Many single-column, single-row, and single-bank faults persist across epochs, indicating that these are permanent faults. A significant fraction of single-bit and single-column faults are only present in one epoch, indicating that there may also be a transient fault type for these failure modes.

Failure Mode	% Faulty DRAMs
Single-bit	49.7%
Single-word	2.5%
Single-column	10.6%
Single-row	12.7%
Single-bank	16.3%
Multiple-bank	2.5%
Multiple-rank	5.5%

Table 3. The fraction of failing DRAMs experiencing a given failure mode (for example, all errors mapped to a single DRAM bit on 49.7% of failing DRAMs).

Failure Mode	Fault Type		Failure Rate	
	Transient	Permanent	Transient	Permanent
Single-bit	43.3%	56.7%	14.2	18.6
Single-word	81.4%	18.6%	1.4	0.3
Single-column	19.7%	80.3%	1.4	5.6
Single-row	2.8%	97.2%	0.2	8.2
Single-bank	7.0%	93.0%	0.8	10.0
Multiple-bank	17.5%	82.5%	0.3	1.4
Multiple-rank	24.3%	75.7%	0.9	2.8

Table 4. The percentage of transient and permanent faults for each failure mode and the corresponding failure rate in FIT/device.

VIII. DRAM FAILURE MODES

In this section, we examine the error data for patterns to determine the failure modes experienced by DRAM in the Jaguar system. Our goal is to identify and understand the different ways that DRAM can fail.

A. Failure Mode Patterns

We first analyze the logged physical addresses and ECC syndromes to associate each error to a specific location in a DRAM. In this way, we create a "map" of failing locations in each DRAM, which allows us to infer a failure mode for each DRAM.

We identify several failure modes across all DRAMs:

- **Single-bit**: All errors map to a single bit.
- **Single-word**: All errors map to a single word.
- **Single-column**: All errors map to a single column.
- **Single-row**: All errors map to a single row.
- **Single-bank**: All errors map to a single bank.
- **Multiple-bank**: Errors map to multiple banks.
- **Multiple-rank**: Errors map to multiple DRAMs in the same lane.

Table 3 shows the percentage of DRAMs that exhibited each failure mode. This table does not differentiate permanent faults from transient faults.

There are several interesting results in Table 3. First, 49.7% of all DRAM faults are single-bit faults. A further 39.6% of all DRAM faults are contained to a single row, column, or bank of a DRAM, demonstrating that large multi-bit faults are a common failure mode in modern DRAMs. Rows, columns, and banks share circuitry in the DRAM, such as address decoders, control signals, and sense amplifiers. Therefore, it seems likely that these failures are due to faults in the shared circuitry rather than to faults in the DRAM array elements themselves. Table 3 also shows that 2.5% of nodes have errors that affect multiple banks in a single DRAM. We consider these single faults rather than multiple independent faults.

Finally, Table 3 shows a high incidence of multiple-rank faults. This indicates that a node experienced errors from multiple DRAMs in the same physical lane. We attribute these errors to a fault in a single DRAM that affects shared external circuitry such as a data (DQ) or strobe (DQS) pin, rather than to multiple independent faults. There are two reasons for this. First, the incidence of nodes with errors from DRAMs in the same lane is much higher than the incidence of nodes with errors in DRAMs from different lanes. We would expect the opposite if these errors were due to multiple independent faults. Second, 44% of nodes with errors in the same lane begin experiencing errors from both DRAMs in the same epoch, and many others begin experiencing errors within a few dozen epochs. This is highly improbable for independent faults.

There are two major findings from this data. First, large multi-bit (row, column, bank, and multiple-bank) faults appear to contribute more significantly to the overall failure

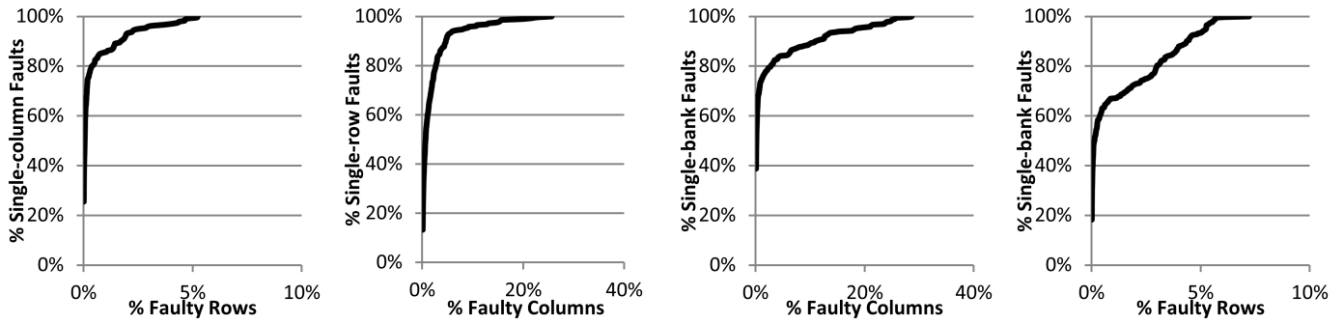
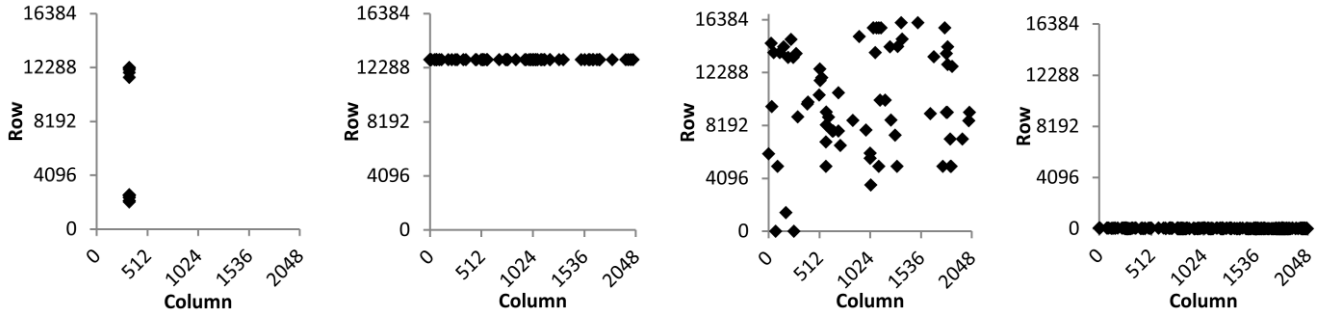


Figure 9. CDFs of the fraction of faulty columns or rows that experience errors for single-row, single-column, and single-bank faults. All single-column faults affect fewer than 6% of rows in that column, while some single-row faults affect up to 30% of the columns in that row.



(a) Single-Column Fault (b) Single-Row Fault (c) "Spread" Single-Bank Fault (d) "Row-cluster" Single-Bank Fault

Figure 10. A "map" of failing locations for a single-column fault, a single-row fault, and two single-bank faults. Each graph plots failing locations for one fault. Single-column faults tend to form clusters in the row space, while single-row faults tend to be spread across the entire column space. Single-bank faults can be spread out (as in Figure 10c) or tightly clustered in the row space (as in Figure 10d).

rate than has been reported by other studies. Second, we find a significant number of multiple-rank faults that affect communication on external (board-level) wires, disrupting access to multiple DRAMs simultaneously. As far as we know, this failure mode has not been previously observed in the field. Multiple-rank faults are especially troubling because they look like the simultaneous failure of multiple DRAM devices, making it difficult to identify the actual failing component.

B. *Putting It All Together*

Figure 8 shows the number of epochs with errors for DRAMs with four different fault types. A plurality of nodes with single-bit and single-column faults have errors in one epoch. This may indicate two fault types for these failure modes: a permanent fault and a transient fault. By contrast, the number of single-row and single-bank faults with errors in one epoch is small. This suggests that all single-row and single-bank faults may be permanent, but that some of them occurred only once during the experiment.

Table 4 shows the percentage of each failure mode classified as transient and permanent faults and the corresponding failure rate for each failure mode. This again confirms that single-row and single-bank faults tend to be permanent, while all other failure modes show a high number of potential transient faults.

IX. A Closer Look at Multi-bit Faults

The large number of multi-bit failure modes warrant closer examination. In this section, we examine these faults in more detail, to determine how many sub-arrays are affected by each failure mode and identify the areas in the DRAM device corrupted by each failure mode.

Failure Mode	Failing DQs			
	1	2	3	4
Single-column	85.8%	3.3%	0.8%	10.0%
Single-row	31.1%	66.8%	1.4%	0.7%
Single-bank	55.5%	23.0%	3.8%	17.8%
Multiple-bank	17.5%	33.3%	3.5%	45.6%
Multiple-rank	7.5%	7.1%	1.8%	83.6%

Table 5. The number of failing DQ pins as a fraction of DRAMs for large multi-bit failure modes. For example, 85.8% of DRAMs with a single-column failure had errors on one DQ pin. Errors on multiple DQ pins are indicative of failures in multiple DRAM sub-arrays.

A. *Sub-arrays Affected*

Internally, a DRAM bank is divided into multiple sub-arrays. Each sub-array provides one bit of a four bit DRAM access. Therefore, we can determine the number of sub-arrays affected by a failure by examining the number of DQ pins that exhibit errors. Table 5 shows the number of unique DQ pins in error for each of the multi-bit failure modes in Table 3. The table shows that 85.8% of single-column faults were confined to a single DQ pin (i.e., one sub-array), while 10.0% of single-column faults affected all four DQ pins (i.e., all four sub-arrays). This seems to indicate the presence of two distinct single-column failure modes: one confined to a single DRAM sub-array and one that affects all four sub-arrays. DRAMs that have two or three DQ pins in error may

Failure Mode	Banks Affected						
	2	3	4	5	6	7	8
Multiple-bank	46.2%	1.9%	3.8%	7.7%	3.8%	0.0%	36.5%
Multiple-rank	4.0%	2.0%	3.0%	14.1%	7.1%	15.2%	54.5%

Table 6. The fraction of banks affected by multiple-bank and multiple-rank faults. 46.2% of multiple-bank faults affect only two DRAM banks, while 36.5% affect all eight DRAM banks. This is indicative of two different failure modes.

eventually show errors on all four DQ pins, or else this may indicate the presence of additional failure modes.

Most single-row failures, by contrast, had errors on one or two DQ pins. Again, this indicates the presence of multiple failure modes: one that corrupts a row in one sub-array and one that corrupts the same row in multiple sub-arrays. Similarly, single-bank and multiple-bank faults often affect multiple DQ pins, indicating that these failure modes often corrupt data in multiple sub-arrays.

Finally, another notable aspect of Table 5 is that 83.6% of multiple-rank failures affect four DQ pins. This may indicate that the most common multiple-rank failure mode is due to a strobe (DQS) pin fault rather than a DQ pin fault.

B. Multi-bit Fault Patterns

Another interesting question is whether a multi-bit fault affects an entire row, column, or bank. Figure 9 shows that single-column faults show errors on only up to 5-6% of the total rows in that column. By contrast, some single-row faults show errors on more than 25% of the columns in the row. The same is true of single-bank faults, which affect a small fraction of the total columns or rows in the bank.

Figure 10 plots a "map" of errors for representative single-column, single-row, and single-bank faults from our data set. Each sub-graph in Figure 10 plots an error map for a single fault. Figure 10(b) shows that single-row faults tend to be spread across the entire column space, despite showing errors on only a subset of columns (3.2% of columns for the DRAM in Figure 10(b)). This is true for most single-row faults in our data set, indicating that these faults affect all columns in the row, but errors occur only in a subset of columns due to a combination of memory access patterns and the potential intermittent nature of the faults.

By contrast, Figure 10(a) shows that single-column faults tend to manifest in discrete clusters in the row space. This pattern is representative of all single-column faults in our data set, indicating that this is a true failure mode, rather than an artifact of memory access patterns.

DRAMs with single-bank faults appear to show two unique error patterns. Figure 10(c) shows errors scattered randomly around the row and column space, while Figure 10(d) shows errors spread across the column space but confined to a small cluster of rows. These are representative patterns from our data set; it is unclear whether these are distinct failure modes or an artifact of access patterns.

DRAMs with multiple-bank faults tend to show similar error patterns to single-bank faults, but spread across multiple banks. As shown in Table 6, 46.2% of multiple-bank faults affect two banks of a DRAM device, while 36.5% affect all eight banks. This may indicate the presence of several failure modes. Table 6 also shows that multiple-rank faults are most likely to affect all eight banks, indicating

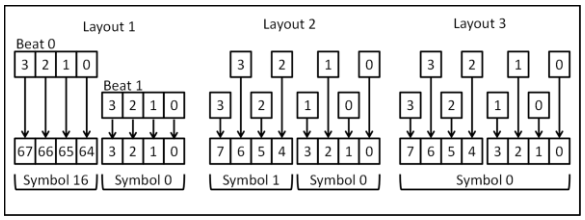


Figure 11. A single DRAM provides data bits to an ECC word over two beats **[27]**. The memory controller can route these bits to an ECC word in different ways (called an ECC layout). In this figure, only Layout 3 is capable of chipkill because all data bits from a DRAM are routed to the same symbol in the ECC word.

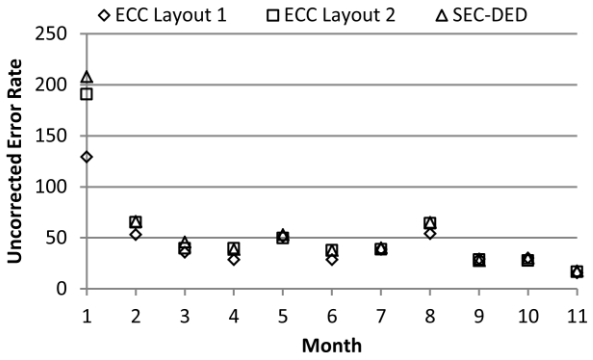


Figure 12. The uncorrected error rate for two different non-chipkill ECC layouts and SEC-DED ECC normalized to the average monthly uncorrected error rate with chipkill. Excluding the first month, chipkill reduces the uncorrected error rate by 36-42x over non-chipkill ECC.

that these faults are likely in the I/O logic of the DRAM device rather than in the array logic.

Overall, a major finding of this data is that DRAM failure modes are extremely varied and depend heavily on the internal DRAM organization. Since this organization can differ substantially amongst devices and vendors, DRAM failure modes are likely to vary significantly as well.

X. Uncorrected Errors and Chipkill

Uncorrected errors from DRAM can be a significant source of system downtime [8]. As a result, techniques such as chipkill have become common to reduce the rate of uncorrected errors in DRAM. However, few studies have examined the benefit of chipkill in the field. Hwang et al. noted that 17% of observed corrected errors required chipkill to correct [16], but did not examine the uncorrected error rate with and without chipkill. Furthermore, operators typically replace DIMMs that experience an uncorrected error; in the absence of chipkill, many of the DIMMs experiencing these errors would have been replaced after the first error, reducing the number of errors that require chipkill to correct. In this section, we examine the effect of chipkill on the uncorrected error rate of each node.

A. Effectiveness of Chipkill

Jaguar's memory sub-system can be configured with multiple ECC layouts, only some of which are chipkill-capable (see Figure 11) [27]. The chipkill-capable Layout 3 is the actual layout in use on Jaguar.

For each error in our data set, we can identify the specific failing bits to determine whether the error would have been

corrected with one of the non-chipkill ECC layouts as well as with SEC-DED ECC. We assume that a DIMM will be replaced after the first uncorrected error, removing the faulty DRAM from the system. Therefore, for each fault we record only the first error that requires chipkill to correct.

Figure 12 plots the monthly rate of these errors for all three non-chipkill ECC layouts, normalized to the average monthly rate of actual uncorrected errors. Excluding the first month, chipkill reduces the rate of uncorrected errors by 36x relative to ECC Layout 1, by 40x relative to ECC Layout 2, and by 42x relative to SEC-DED ECC. (We exclude the first month because it includes faults that developed before the start of the experiment.) This reduction in uncorrected error rate is smaller than the benefit shown in laboratory studies of older memory technologies [8], but is significantly larger than the benefit inferred from looking solely at corrected error rates [4] [16].

The primary finding from this data is that chipkill has a substantial reliability benefit to a DRAM sub-system. This finding has been confirmed by the experience of the Jaguar system operators, since the system did not have chipkill for the first half of 2009.

B. *Uncorrected Errors*

Because Jaguar's memory system has chipkill capability, any remaining uncorrected errors from memory are the result of faults in multiple DRAM devices on the same rank, which are unlikely to develop simultaneously. In fact, we find that more than 83% of nodes with uncorrected errors first experienced corrected errors from an existing fault.

Large multi-bit faults increase the likelihood of an uncorrected error because these faults impact more ECC words, and thus are more likely to overlap with a second fault. Figure 13 plots the probability of an uncorrected error on nodes with an existing fault, normalized to the probability of an uncorrected error on a node with no faults. The presence of a single-bit fault increases a node's probability of an uncorrected error by 17x compared to a node with no faults. The presence of a multiple-bank or multiple-rank fault, however, increases a node's probability of uncorrected error by 350x and 700x, respectively, approximately the same increase experienced by nodes with two independent DRAM faults. This implies that system operators should prioritize replacement of DIMMs whose DRAMs are experiencing multiple-bank and multiple-rank faults.

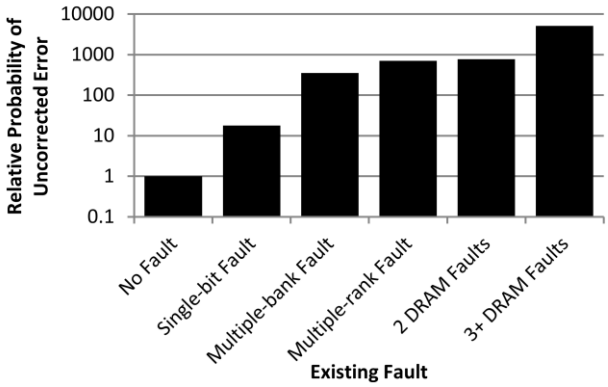


Figure 13. The relative probability of an uncorrected error on nodes with an existing fault. Existing multiple-bank and multiple-rank faults increase the probability of a subsequent uncorrected error by 350x and 700x, respectively, relative to a node with no faults.

XI. CONCLUSIONS

This study analyzed 11 months of DRAM failure data from the Jaguar high-performance computing cluster at Oak Ridge National Lab. We performed a detailed study of DRAM failure modes and fault types. We derived an upper bound on the rate of transient faults in the DRAM array. We examined multi-bit failure modes and identified several unique fault patterns. We also analyzed the impact of chipkill on the uncorrected error rate due to DRAM failures.

Prior work has shown that DRAM vendor and technology can have a large effect on failure rates [22]. Thus, more work is needed to confirm these results on other systems with different DRAM device types. Nonetheless, we believe that our findings have several implications for the architecture and design of future systems.

First, we found that multi-bit failures constituted almost 50% of all DRAM faults, and that there were a large variety of multi-bit failure modes. This implies that ECC techniques that store their data and check bits in the same DRAM devices must pay careful attention to the placement of data and check words in memory to achieve comparable reliability to traditional "parallel" ECC [28]. This also implies that memory failure rates will depend on internal device organization in addition to device count and capacity. Therefore, the performance of detection and correction techniques that target specific failure modes will depend heavily on the memory being studied [29].

We also found that permanent faults account for at least 70% of DRAM failures. This diminishes the effectiveness of memory scrubbing as a means to reduce the uncorrected error rate, since memory scrubbing is effective only against transient faults and certain intermittent faults.

We found that 8% of DRAM faults were multiple-bank and multiple-rank faults, and that these faults were most likely to lead to a future uncorrected error. This implies that OS-level memory page retirement algorithms will have only a modest impact on the uncorrected error rate, since a node would need to retire an impractical amount of its memory to eliminate these faults. Our data show that a page retirement algorithm that retires up to 6.25% of a node's memory (enough to counter single-bit, -row, -column, and -bank faults) would reduce the uncorrected error rate by only 8%.

Finally, our data shows that chipkill reduced the uncorrected error rate by 42x compared to SEC-DED ECC. However, the Jaguar system still shows uncorrected errors due to DRAM faults. Due to projected increases in DRAM device counts and capacities, future systems will likely require even stronger memory error protection techniques.

Overall, our data suggest that DRAM failures will pose an increasing concern in the future due to the projected increase in node DRAM capacity during the next decade. With current memory protection techniques, we would need several orders of magnitude improvement in DRAM failure rates to maintain current node failure rates. This implies that high-reliability systems will need significant advances in memory reliability techniques to maintain comparable failure rates to today's systems.

XII. Acknowledgments

We thank Steve Johnson, Dave Londo, and Hansi Bohnstedt from Cray, Inc. for providing data collection and configuration information on the Jaguar system, and Alan Wood and Kevin Lepak for comments on early versions of the manuscript.

XIII. References

[1] James F. Ziegler and William A. Lanford, "The Effect of Sea Level Cosmic Rays on Electronic Devices," in *IEEE International Solid-State Circuits Conference*, 1980, pp. 70-71.

[2] ITRS, "International Technology Roadmap for Semiconductors," 2010.

[3] Bianca Schroeder and Garth A. Gibson, "A Large-Scale Study of Failures in High-Performance Computing Systems," in *Proceedings of the International Conference on Dependable Systems and Networks*, June, 2006, pp. 249-258.

[4] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber, "DRAM Errors in the Wild: A Large-Scale Field Study," in *Proceedings of the 11th Internation Joint Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, Seattle, WA, June, 2009.

[5] Xin Li, Kai Shen, Michael C. Huang, and Lingkun Chu, "A Memory Soft Error Measurement on Production Systems," in *Proceedings of the USENIX Annual Technical Conference*, Santa Clara, CA, 2007, pp. 275-280.

[6] Abdallah M. Saleh, Juan J. Serrano, and Janak H. Patel, "Reliability of Scrubbing Recovery Techniques for Memory Systems," *IEEE Transactions on Reliability*, vol. 39, no. 1, pp. 114-122, April 1990.

[7] Shubhendu S. Mukherjee, Joel Emer, Tryggve Fossum, and Steven K. Reinhardt, "Cache Scrubbing in Microprocessors: Myth or Necessity?," in *Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, March, 2004, pp. 37-42.

[8] Timothy J. Dell, "A White Paper on the Benefits of Chipkill-Correct ECC for PC Server Main Memory," IBM Corporation, 1997.

[9] Algirdas Avizenis, Jean-Claude. Laprie, Brian Randell, and Carl Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11-33, Jan-March 2004.

[10] Cristian Constantinescu, "Impact of Deep Submicron Technology on Dependenability of VLSI Circuits," in *International Conference on Dependable Systems and Networks (DSN)*, Bethesda, MD, 2002, pp. 205-209.

[11] Robert Baumann, "Soft Errors in Advanced Computer Systems," in *IEEE Design and Test of Computers*, May 2005, pp. 258-266.

[12] Cristian Constantinescu, "Trends and Challenges in VLSI Circuit Reliability," *IEEE Micro*, vol. 23, no. 4, pp. 14-19, July-August 2003.

[13] Christopher Weaver, Joel Emer, Shubhendu Mukherjee, and Steven Reinhardt, "Techniques to Reduce the Soft Error Rate of a High-Performance Microprocessor," in *International Symposium on Computer Architecture (ISCA)*, Munchen, 2004.

[14] Shubhendu Mukherjee, Christopher Weaver, Joel Emer, Steven Reinhardt, and Todd Austin, "A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor," in *International Symposium on Microarchitecture (MICRO-36)*, San Diego, 2003.

[15] Xin Li, Michael C. Huang, Kai Shen, and Lingkun Chu, "A Realistic Evaluation of Memory Hardware Errors and Software System Susceptibility," in *USENIX*, Boston, MA, 2010.

[16] Andy A. Hwang, Ioan Stefanovici, and Bianca Schroeder, "Cosmic Rays Don't Strike Twice: Understanding the Nature of DRAM Errors and the Implications for System Design," in *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, London, 2012.

[17] Vilas Sridharan and Dean Liberty, "A Study of DRAM Errors in the Field," in *Silicon Errors in Logic - System Effects (SELSE)*, Champaign, IL, 2012.

[18] Bianca Schroeder, Personal Communication.

[19] Timothy C. May and Murray H. Woods, "Alpha-Particle Induced Soft Errors in Dynamic Memories," *IEEE Transactions on Electron Devices*, vol. 26, no. 1, pp. 2-9, January 1979.

[20] Alan Messer et al., "Susceptibility of Commodity Systems and Software to Memory Soft Errors," *IEEE Transactions on Computers*, vol. 53, no. 12, December 2004.

[21] Ludger Borucki, Guenter Schindlbeck, and Charles Slayman, "Comparison of Accelerated DRAM Soft Error Rates Measured at Component and System Level," in *Reliability Physics Symposium (IRPS)*, Phoenix, 2008, pp. 482-487.

[22] Heather Quinn, Paul Graham, and Tom Fairbanks, "SEEs Induced by High-Energy Protons and Neutrons in SDRAM," in *Proceedings of the IEEE Radiation Effects Data Workshop (REDW)*, Las Vegas, NV, 2011, pp. 1-5.

[23] Aniruddha N. Udipi et al., "Rethinking DRAM Design and Organization for Energy-Constrained Multi-Cores," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, Saint-Malo, 2010.

[24] Richard W. Hamming, "Error Detecting and Error Correcting Codes," *Bell System Technology Journal*, vol. 29, no. 2, pp. 147-160, 1950.

[25] AMD, Inc., "AMD64 Architecture Programmer's Manual Revision 3.17," 2011.

[26] Tom Anderson and Brian Randell, *Computing Systems Reliability*. Cambridge: Cambridge University Press, 1979.

[27] AMD, Inc. (2010, April) BIOS and Kernel Developer's Guide (BKDG) For AMD Family 10h Processors. [Online]. http://support.amd.com/us/Processor_TechDocs/31116.pdf

[28] Doe Hyun Yoon and Mattan Erez, "Virtualized and Flexible ECC for Main Memory," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Pittsburgh, PA, 2010.

[29] Aniruddha N. Udipi, Naveen Muralimanohar, Rajeev Balasubramonian, Al Davis, and Norman P. Jouppi, "LOT-ECC: LOcalized and Tiered Reliability Mechanisms for Commodity Memory Systems," in *ISCA: International Symposium on Computer Architecture*, Portland, OR, 2012.