Memory Errors in Modern Systems

The Good, The Bad, and The Ugly

Vilas Sridharan[1], Nathan DeBardeleben[2], Sean Blanchard[2], Kurt B. Ferreira[3],
Jon Stearley[3], John Shalf[4], Sudhanva Gurumurthi[5]

[1]RAS Architecture, [5]AMD Research, Advanced Micro Devices, Inc., Boxborough, MA
[2]Ultrascale Systems Research Center, Los Alamos National Laboratory, Los Alamos, New Mexico *
[3]Scalable System Software, Sandia National Laboratories, Albuquerque, New Mexico †
[4]National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory, Berkeley, CA ‡
{vilas.sridharan, sudhanva.gurumurthi}@amd.com, {ndebard, seanb}@lanl.gov
{kbferre, jrstear}@sandia.gov, jshalf@lbl.gov

Abstract

Several recent publications have shown that hardware faults in the memory subsystem are commonplace. These faults are predicted to become more frequent in future systems that contain orders of magnitude more DRAM and SRAM than found in current memory subsystems. These memory subsystems will need to provide resilience techniques to tolerate these faults when deployed in high-performance computing systems and data centers containing tens of thousands of nodes. Therefore, it is critical to understand the efficacy of current hardware resilience techniques to determine whether they will be suitable for future systems.

In this paper, we present a study of DRAM and SRAM faults and errors from the field. We use data from two leadership-class high-performance computer systems to analyze the reliability impact of hardware resilience schemes that are deployed in current systems. Our study has several key findings about the efficacy of many currently-deployed reliability techniques such as DRAM ECC, DDR address/command parity, and SRAM ECC and parity. We also perform a methodological study, and find that counting errors instead of faults, a common practice among researchers and data center operators, can lead to incorrect conclusions about system reliability. Finally, we use our data to project the needs of future large-scale systems. We find that SRAM faults are unlikely to pose a significantly larger reliability threat in the future, while DRAM faults will be a major concern and stronger DRAM resilience schemes will be needed to maintain acceptable failure rates similar to those found on today's systems.

Categories and Subject Descriptors B.8.1 [*Performance and Reliability*]: Reliability, Testing, and Fault-Tolerance

Keywords Field studies; Large-scale systems; Reliability

1. Introduction

Current predictions are that exascale systems in the early 2020s will have between 32 and 100 petabytes of main memory (DRAM), a 100x to 350x increase compared to 2012 levels [8]. Similar increases are likely in the amount of cache memory (SRAM) in such systems. Future data centers will also contain many more nodes than existing data centers. These systems and data centers will require significant increases in the reliability of both DRAM and SRAM memories in order to maintain hardware failure rates comparable to current systems.

The focus of this work is to provide insight and guidance to system designers and operators on characteristics of a reliable system. In particular, our focus is on analyzing the efficacy of existing hardware resilience techniques, their impact on reliable system design, and whether they will be adequate for future systems.

For our analysis, we use data collected over the past several years from two leadership-class production sys-

* A portion of this work was performed at the Ultrascale Systems Research Center (USRC) at Los Alamos National Laboratory, supported by the U.S. Department of Energy contract DE-FC02-06ER25750. The publication has been assigned the LANL identifier LA-UR-14-26219.

† Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000. The publication has been assigned the Sandia identifier SAND2014-16515J

‡ A portion of this work used resources of the National Energy Research Scientific Computing Center supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

ASPLOS '15, March 14–18, 2015, Istanbul, Turkey.

http://dx.doi.org/10.1145/2694344.2694348

tems: Hopper, a 6,000-node supercomputer located at the NERSC center at Lawrence Berkeley Labs in Oakland, California; and Cielo, an 8,500-node supercomputer located at Los Alamos National Laboratory (LANL) in Los Alamos, New Mexico. Both supercomputers are based on AMD Opteron™CPUs and contain DDR3 DRAM. In aggregate, the data that we analyze (which is a subset of the full data available) comprises over 314 million CPU socket-hours and 45 billion DRAM device-hours. This scale gives us the ability to gather statistically representative data on faults observed during production lifetimes of CPU and DRAM devices. While our data is collected primarily from supercomputing centers, the results of our analysis are applicable to any large data center or compute cluster where hardware reliability is important.

This paper adds several novel insights about system reliability to the existing literature and also highlights certain aspects of analyzing field data that are critical to perform correctly in order to derive correct insights. Broad contributions include:

- A detailed analysis of DRAM and SRAM faults on Hopper. This data complements existing studies on faults in other production systems, including Jaguar, Cielo, and Blue Waters, thus furthering our understanding of DRAM and SRAM faults [12][33][34].
- The effect of altitude on DRAM fault rates. To our knowledge, this is the first study to present an analysis of altitude effects on DRAM in production systems.
- The impact of counting errors instead of faults to evaluate system reliability. Counting errors is common among researchers and data center operators (e.g., [12][30]), but no one has quantified the effect of this methodology on the accuracy of the reliability conclusions obtained.
- The effect of several hardware-based resilience techniques on system reliability, including SEC-DED ECC, chipkill ECC, and command/address parity in DRAM subsystems. We also examine SRAM resilience schemes, and analyze in-depth the impact of design choices on SRAM reliability.
- A projection of the impacts of SRAM and DRAM faults on future exascale-class systems.

Our study has several key findings that are of interest to system designers and operators. These findings are wide ranging and cover a variety of hardware resilience techniques in common use in the industry. We group these findings into three categories: the good, the bad, and the ugly.

The Good. Our study highlights several advances made in the understanding of fault behavior and system reliability. These include:

- DDR command and address parity, an addition to JEDEC's DDR-3 and DDR-4 specifications, has a substantial positive effect on system reliability, detecting errors at a rate comparable to the uncorrected error rate of chipkill ECC.
- The majority of observed SRAM faults in the field are transient faults from particle strikes. This further confirms that SRAM faults are a well-understood phenomenon in current and near-future process technologies.
- The majority of uncorrected SRAM errors are due to single-bit faults; therefore, reducing SRAM uncorrected error rates is a matter of engineering time and effort (e.g., replacing parity codes with ECCs) rather than new research and novel techniques. Furthermore, appropriate multi-bit fault protection reduces or eliminates the expected increase in uncorrected error rate from SRAM faults due to high altitude.

The Bad. Unfortunately, some of our findings point to areas where more work needs to be done, or more understanding is required. These include:

- Altitude increases the fault rate of some DRAM devices, though this effect varies by vendor. This is evidence that some (but not all) DRAM devices are susceptible to transient faults from high-energy particle strikes.
- Unlike on-chip SRAM, external memory (e.g., DRAM) of future systems will require stronger resilience techniques than exist in supercomputing systems today.

The Ugly. Finally, some of our results show potential problems in commonly-used practices or techniques:

- Performing field studies is difficult. For instance, we demonstrate that counting errors instead of faults can lead to incorrect conclusions about system reliability. We examine results from recent studies that have used error counts, and use our data to show that using this methodology can result in misleading or inaccurate conclusions.
- SEC-DED ECC, a commonly used ECC technique, is poorly suited to modern DRAM subsystems and may result in *undetected* errors (which may cause silent data corruption) at a rate of up to 20 FIT per DRAM device, an unacceptably high rate for many enterprise data centers and high-performance computing systems.

The rest of this paper is organized as follows. Section 2 defines the terminology we use in this paper. Section 3 discusses related studies and describes the differences in our study and methodology. Section 4 explains the system and DRAM configurations of Cielo and Hopper. Section 5 describes our experimental setup. Section 6 presents baseline data on faults. Section 7 examines the impact of counting errors instead of faults. Section 8 presents our analysis of existing hardware resilience techniques. Section 9 extracts lessons from our data for system designers. Section 10 presents our projections for future system reliability, and Section 11 concludes.

2. Terminology

In this paper, we distinguish between a fault and an error as follows [6]:

- A fault is the underlying cause of an error, such as a stuck-at bit or high-energy particle strike. Faults can be *active* (causing errors), or *dormant* (not causing errors).
- An error is an incorrect portion of state resulting from an active fault, such as an incorrect value in memory. Errors may be *detected* and possibly *corrected* by higher level mechanisms such as parity or error correcting codes (ECC). They may also go *uncorrected*, or in the worst case, completely *undetected* (i.e., silent).

Hardware faults can further be classified as *transient*, *intermittent*, or *hard* [7] [10] [11]. 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.

In this paper, we examine a variety of error detection and correction mechanisms. Some of these mechanisms include: *parity*, which can detect but not correct any single-bit error; single-error-correction double-error-detection *error correcting codes* (SEC-DED ECCs), which can correct any single-bit error and detect any double-bit error; and *chipkill* ECCs. We discuss two levels of chipkill ECC: *chipkill-detect* ECC, which can detect but not correct any error in a single DRAM chip; and *chipkill-correct*, which can correct any error in a single DRAM chip. To protect against multi-bit faults, structures with ECC or parity sometimes employ *bit interleaving*, which ensures that physically adjacent bitcells are protected by different ECC or parity words.

3. Related Work

During the past several years, multiple studies have been published examining failures in production systems. In 2006, Schroeder and Gibson studied failures in supercomputer systems at LANL [29]. In 2007, Li *et al.* published a study of memory errors on three different data sets, including a server farm of an Internet service provider [21]. In 2009, Schroeder *et al.* published a large-scale field study using Google's server fleet [30]. In 2010, Li *et al.* published an expanded study of memory errors at an Internet server farm and other sources [20]. In 2012, Hwang *et al.* published an expanded study on Google's server fleet, as well as two IBM Blue Gene clusters [16], Sridharan and Liberty presented a study of DRAM failures in a high-performance computing system [33], and El-Sayed *et al.* published a study on temperature effects of DRAM in data center environments [14].

In 2013, Siddiqua *et al.* presented a study of DRAM failures from client and server systems [31], and Sridharan *et al.* presented a study of DRAM and SRAM faults, with a focus on positional and vendor effects [34]. Finally, in 2014, Di Martino *et al.* presented a study of failures in Blue Waters, a high-performance computing system at the University of Illinois, Urbana-Champaign [12].

Parameter	Cielo	Hopper
Nodes	8568	6000
Sockets / Node	2	2
Cores / Socket	8	12
DIMMs / Socket	4	4
Location	Los Alamos, NM	Oakland, CA
Altitude	7,320 ft.	43 ft.

Table 1. System Configuration Information.

Our study contains larger scale and longer time intervals than many of the prior studies, allowing us to present more representative data on faults and errors. Among those studies with similar or larger scale to ours, many count errors instead of faults [12][14][16][30], which we show in Section 7 to be inaccurate. In addition, few of these prior studies examine the efficacy of hardware resilience techniques such as SEC-DED, chipkill, and DDR command/address parity.

There has been laboratory testing on DRAM and SRAM dating back several decades (e.g., [9][13][23][24][27]), as well as a recent study by Kim *et al.* on DRAM disturbance faults [19]. Lab studies such as these allow an understanding of fault modes and root causes, and complement field studies that identify fault modes which occur in practice.

4. Systems Configuration

Our study comprises data from two production systems in the United States: Hopper, a supercomputer located in Oakland, California, at 43 feet in elevation; and Cielo, a supercomputer in Los Alamos, New Mexico, at around 7,300 feet in elevation. A summary of relevant statistics on both systems are given in Table 1.

Hopper contains approximately 6,000 compute nodes. Each *node* contains two 12-core AMD Opteron™processors, each with twelve 32KB L1 data caches, twelve 512KB L2 caches, and one 12MB L3 cache. Each node has eight 4GB DDR-3 registered DIMMs for a total of 32GB of DRAM.

Cielo contains approximately 8,500 compute nodes. Each node contains two 8-core AMD Opteron™processors, each with eight 32KB L1 data caches, eight 512KB L2 caches, and one 12MB L3 cache. Each node has eight 4GB DDR-3 registered DIMMs for a total of 32GB of DRAM.

The nodes in both machines are organized as follows. Four nodes are connected to a *slot* which is a management module. Eight slots are contained in a *chassis*. Three chassis are mounted bottom-to-top (numerically) in a *rack*. Cielo has 96 racks, arranged into 6 *rows* each containing 16 racks.

4.1 DRAM and DIMM Configuration

In both Hopper and Cielo, each DDR-3 DIMM contains two *ranks* of 18 DRAM devices, each with four data (DQ) signals (known as an x4 DRAM device). In each rank, 16 of the

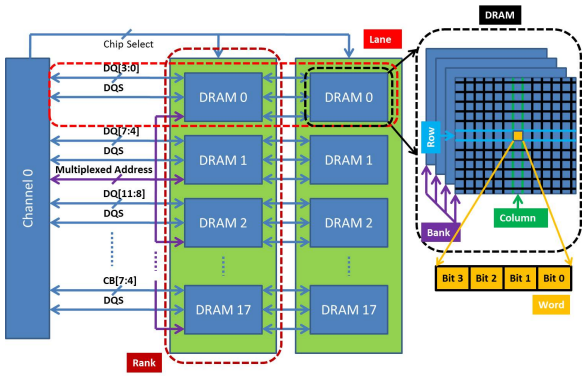

Figure 1. A simplified logical view of a single channel of the DRAM memory subsystem on each Cielo and Hopper node.

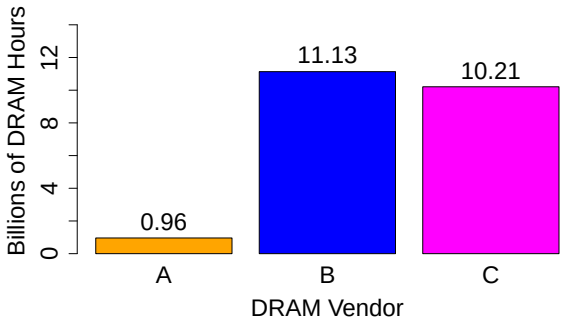

Figure 2. DRAM device-hours per vendor on Hopper. Even for the least-populous vendor (A), we have almost 1B device-hours of data.

DRAM devices are used to store data bits and two are used to store check bits. A *lane* is a group of DRAM devices on different ranks that shares data (DQ) signals. DRAMs in the same lane also share a strobe (DQS) signal, which is used as a source-synchronous clock signal for the data signals. A memory *channel* has 18 lanes, each with two ranks (i.e., one DIMM per channel). Each DRAM device contains eight internal *banks* that can be accessed in parallel. Logically, each bank is organized into *rows* and *columns*. Each row/column address pair identifies a 4-bit *word* in the DRAM device. Figure 1 shows a diagram of a single memory channel.

Physically, all DIMMs on Cielo and Hopper (from all vendors) are identical. Each DIMM is double-sided. DRAM devices are laid out in two rows of nine devices per side. There are no heatsinks on DIMMs in Cielo or Hopper.

The primary difference between the two memory subsystems is that Cielo uses chipkill-correct ECC on its memory subsystem, while Hopper uses chipkill-detect ECC.

5. Experimental Setup

For our analysis we use three different data sets - corrected error messages from console logs, uncorrected error messages from event logs, and hardware inventory logs. These three logs provided the ability to map each error message to specific hardware present in the system at that point in time.

Corrected error logs contain events from nodes at specific time stamps. Each node in the system has a hardware memory controller that logs corrected error events in registers provided by the x86 machine-check architecture (MCA) [5]. 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.

Uncorrected error event logs are similar and contain data on uncorrected errors logged in the system. These are typically not logged via polling, but instead logged after the node reboots as a result of the uncorrected error.

Both console and event logs contain a variety of other information, including the physical address and ECC syndrome associated with each error. These events are decoded further using configuration information to determine the physical DRAM location associated with each error. For this analysis, we decoded the location to show the DIMM, as well as the DRAM bank, column, row, and chip.

We make the assumption that each DRAM device experiences a single fault during our observation interval. The occurrence time of each DRAM fault corresponds to the time of the first observed error message per DRAM device. We then assign a specific type and mode to each fault based on the associated errors in the console log. Our observed fault rates indicate that fewer than two DRAM devices will suffer multiple faults within our observation window, and thus the error in this method is low. Prior studies have also used and validated a similar methodology (e.g., [34]).

Hardware inventory logs are separate logs and provide snapshots of the hardware present in each machine at different points in its lifetime. These files provide unique insight into the hardware configuration that is often lacking on other systems. In total we analyzed over 400 individual hardware inventory logs that covered a span of approximately three years on Hopper and two years on Cielo. Each of these files consist of between 800 thousand and 1.5 million lines of explicit description of each host's hardware, including configuration information and information about each DIMM such as the manufacturer and part number.

We limit our analysis to the subset of dates for which we have both hardware inventory logs as well as error logs. For Hopper, this includes 18 months of data from April 2011 to January 2013. For Cielo, this includes 12 months of data from July 2011 to November 2012. For both systems, we exclude the first several months of data to avoid attributing hard faults that occurred prior to the start of our dataset to the first months of our observation. Also, in the case of both systems, time periods where the system was not in a consistent state or was in a transition state were excluded.

For confidentiality purposes, we anonymize all DIMM vendor information. Because this is not possible for the CPU

vendor, we present all CPU data in arbitrary units (i.e., all data is normalized to one of the data points in the graph). While this obscures absolute rates, it still allows the ability to observe trends and compare relative rates. Since every data center and computing system is different, this is often the most relevant information to extract from a study such as this.

5.1 Methodology

Both Cielo and Hopper include hardware scrubbers in DRAM, L1, L2, and L3 caches. Therefore, we can identify permanent faults as those faults that survive a scrub operation. Thus, we classify a fault as permanent when a device generates errors in multiple scrub intervals (i.e., when the errors are separated by a time period that contains at least one scrub operation), and transient when it generates errors in only a single scrub interval. For example, Cielo's DRAM scrub interval is 24 hours. A fault that generates errors separated by more than 24 hours is classified as permanent, while a fault that generates errors only within a 24-hour period is classified as transient.

Our observed fault rates indicate that fewer than two DRAM devices will suffer multiple faults within our observation window. Therefore, similar to previous field studies, we make the simplifying assumption that each DRAM device experiences a single fault during our observation interval [33]. The occurrence time of each DRAM fault corresponds to the time of the first observed error message from that DRAM device. We then assign a specific type and mode to each fault based on the subsequent errors from that device in the console logs. We use a similar methodology (based on fault rates) for SRAM faults.

6. Baseline Data on Hopper Faults

In this section, we present our baseline data on DRAM and SRAM fault modes and rates in Hopper. We have published similar information on Cielo in prior work [34].

6.1 DRAM Faults

Figure 2 shows the number of DRAM-hours per vendor during our measurement interval on Hopper. Our data consists of approximately 1 billion DRAM-hours of operation or more for each vendor, enough to draw statistically meaningful conclusions.

Figure 3 shows the DRAM fault rate over time in Hopper. Similar to other systems, Hopper experienced a declining rate of permanent DRAM faults and a constant rate of transient faults during our measurement interval [33] [34].

Table 2 shows a breakdown of DRAM fault modes experienced in Hopper. Similar to prior studies, we identify several unique DRAM fault modes: single-bit, in which all errors map to a single bit; single-word, in which all errors map to a single word; single-column, in which all errors map to a single column; single-row, in which all errors map to a single row; single-bank, in which all errors map to a single bank; multiple-bank, in which errors map to multiple banks; and multiple-rank, in which errors map to multiple DRAMs in the same lane.

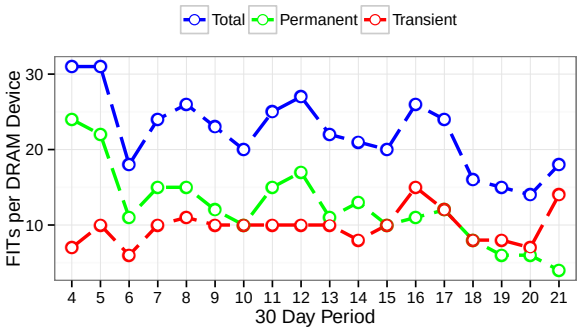

Figure 3. Hopper DRAM device fault rates per month (30-day period); 22.3 billion DRAM device hours total. The rate of permanent faults decreases over time, while the rate of transient faults remains approximately constant.

Fault Mode	Total Faults	Transient	Permanent
Single-bit	78.9%	42.1%	36.8%
Single-word	0.0%	0.0%	0.0%
Single-column	5.9%	0.0%	5.9%
Single-row	9.2%	1.8%	7.4%
Single-bank	4.3%	0.4%	3.9%
Multiple-bank	0.6%	0.0%	0.6%
Multiple-rank	1.0%	0.2%	0.8%

Table 2. DRAM fault modes in Hopper.

Similar to other DDR-2 and DDR-3 systems, a majority of DRAM faults are single-bit faults, but a non-trivial minority of faults are large multi-bit faults, including row, column, bank, and chip faults.

Figure 4 shows the transient and permanent fault rate in Hopper in FIT per DRAM device, broken down by DRAM vendor. Vendors A, B, and C in Hopper are the same as vendors A, B, and C in Cielo [34]. The figure shows substantial variation in per-vendor FIT rates, consistent with data from Cielo [34]. This implies that the choice of DRAM vendor is an important consideration for system reliability.

6.1.1 Effect of Altitude It is well-known that altitude will have an effect on modern SRAM devices due to high-energy neutrons from cosmic radiation [7], and particle strikes in DRAM were a problem in the past [23]. It is uncertain, however, whether high-energy neutrons are a significant source of faults in modern DRAM devices. We have previously published data on DRAM faults in Cielo [34]. We use this data in conjunction with Hopper data to examine the effect of altitude on DRAM faults by comparing fault rates of similar DRAM devices in Cielo and Hopper.

The Hopper and Cielo memory subsystems are extremely similar: the systems use memory from the same three

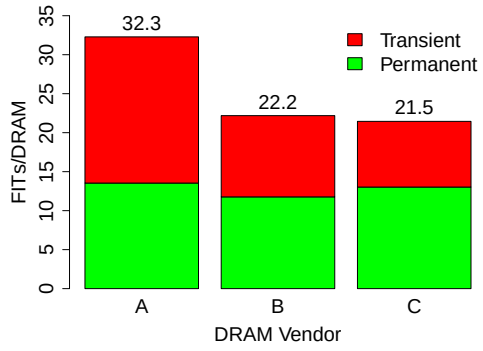

Figure 4. Fault rate per DRAM vendor. Hopper sees substantial variation in fault rates by DRAM vendor.

DRAM vendors, share the same DDR-3 memory technology, and use the same memory controllers. A major difference between the two systems is their altitude: the neutron flux, relative to New York City, experienced by Cielo is 5.53 and the relative neutron flux experienced by Hopper is 0.91[1]. Therefore, by comparing fault rates in Cielo and Hopper, we can determine what effect, if any, altitude has on DRAM fault rates.

Figure 5 plots the rate of transient faults on Cielo relative to the rate of transient faults on Hopper, using Cielo data from [34]. The figure shows that vendor A experiences a substantially higher transient fault rate on Cielo, while vendor B experiences a modestly higher transient fault rate and vendor C experiences virtually the same transient fault rate.

Table 3 breaks down this data and shows the single-bit, single-column, and single-row transient fault rates for each system by DRAM vendor. The table shows that the single-bit transient fault rates for vendors A and B are substantially higher in Cielo than in Hopper, as are the single-column and single-bank transient fault rates for vendor A. The rates of all other transient fault modes were within 1 FIT of each other on both systems. Due to the similarities between the two systems, the most likely cause of the higher single-bit, single-column, and single-bank transient fault rates in Cielo is particle strikes from high-energy neutrons.

Our main observation from this data is that particle-induced transient faults remain a noticeable effect in DRAM devices, although the susceptibility varies by vendor and there are clearly other causes of faults (both transient and permanent) in modern DRAMs. Also, our results show that while altitude does have an impact on system reliability, judicious choice of the memory vendor can reduce this impact.

6.1.2 Multi-bit DRAM Faults Figure 6 shows an example of a single-bank fault from Hopper of multiple adjacent DRAM rows with bit flips. The figure shows that errors from this fault were limited to several columns in three logically adjacent rows. Multiple faults within our dataset matched this general pattern, although the occurrence rate was relatively low. All errors from this fault were corrected by ECC.

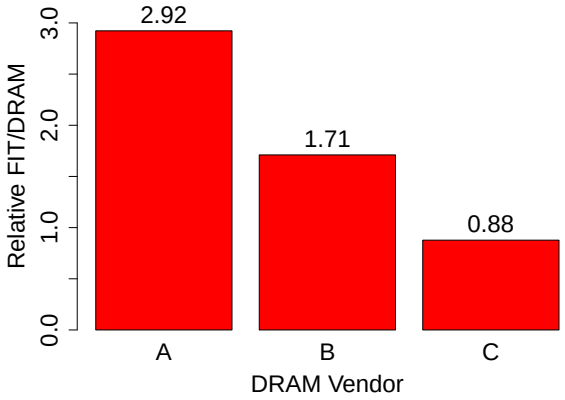

Figure 5. Cielo DRAM transient fault rates relative to Hopper (Hopper == 1.0). Vendor A shows a significantly higher transient fault rate in Cielo, likely attributable to altitude.

Fault Mode	Vendor	Cielo	Hopper
Single-bit	A	31.85	18.75
	B	15.61	9.70
	C	6.10	7.93
Single-column	A	6.05	0.0
	B	0.55	0.0
	C	0.18	0.0
Single-bank	A	14.65	0.0
	B	0.07	0.09
	C	0.18	0.10

Table 3. Rate of single-bit, single-column and single-bank transient DRAM faults in Cielo and Hopper (in FIT). The higher rate of transient faults in Cielo may indicate that these faults are caused by high-energy particles.

The pattern of errors from this fault appears to be similar to a fault mode called DRAM disturbance faults or "row-hammer" faults [19] [26]. This fault mode results in corruption of a "victim row" when an "aggressor row" is opened repeatedly in a relatively short time window. We did not do root-cause analysis on this part; therefore, we cannot say for certain that the fault in question is a disturbance fault. However, it is clear that fault modes with similar characteristics do occur in practice and must be accounted for.

Our primary observation from data in this section is that new and unexpected fault modes may occur after a system is architected, designed, and even deployed: row-hammer faults were only identified by industry after Hopper was designed, and potentially after it began service [26]. Prior work has looked at tailoring detection to specific fault modes (e.g., [35]), but the data show that we need to be able to handle a larger variety of fault modes and that robust detection is crucial.

6.2 SRAM Faults

In this section, we examine fault rates of SRAM in the AMD Opteron™ processors in Hopper.

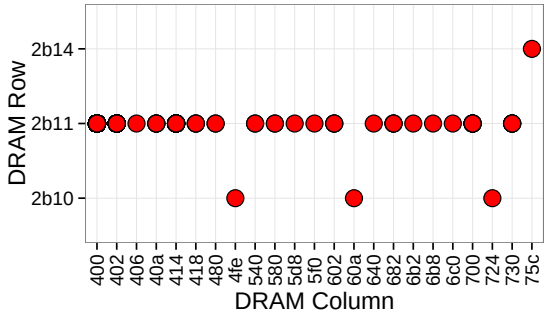

Figure 6. A single-bank fault from one Hopper node.

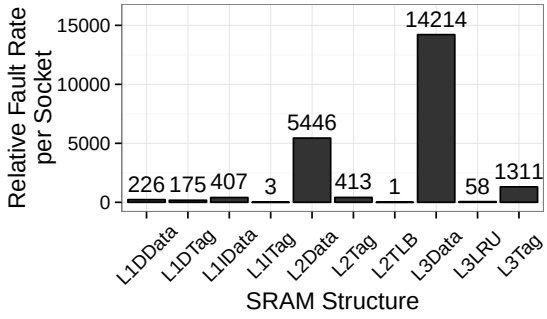

Figure 7. Rate of SRAM faults per socket in Hopper, relative to the fault rate in the L2TLB. The fault rate is affected by structure size and organization, as well as workload effects.

6.2.1 SRAM Fault Rate Figure 7 shows the rate of SRAM faults per socket in Hopper, broken down by hardware structure. The figure makes clear that SRAM fault rates are dominated by faults in the L2 and L3 data caches, the two largest structures in the processor. However, it is clear that faults occur in many structures, including smaller structures such as tag and TLB arrays.

A key finding from this figure is that, at scale, faults occur even in small on-chip structures, and detection and correction in these structures is important to ensure correctness for high-performance computing workloads running on large-scale systems. This result is significant for system designers considering different devices on which to base their systems. Devices should either offer robust coverage on all SRAM and significant portions of sequential logic, or provide alternate means of protection such as redundant execution [36].

6.2.2 Comparison to Accelerated Testing Accelerated particle testing is used routinely to determine the sensitivity of SRAM devices to single-event upsets from energetic particles. To be comprehensive, the accelerated testing must include all particles to which the SRAM cells will be exposed to in the field (e.g., neutrons, alpha particles) with energy spectra that approximate real-world conditions. Therefore, it is important to correlate accelerated testing data with field data to ensure that the conditions are in fact similar.

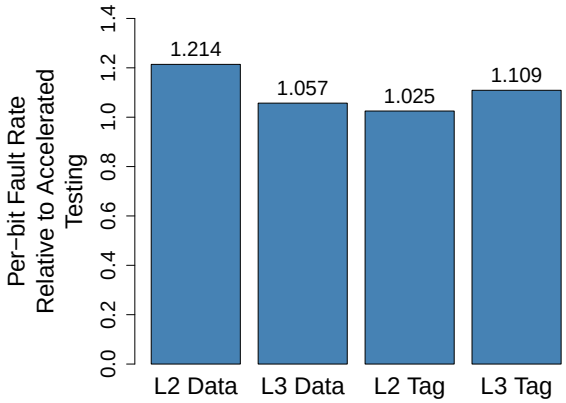

Figure 8. Rate of SRAM faults in Hopper compared to accelerated testing.

Accelerated testing on the SRAM used in Hopper was performed using a variety of particle sources, including the high-energy neutron beam at the LANSCE ICE House[22] at LANL and an alpha particle source. The testing was "static" testing, rather than operational testing. Static testing initializes SRAM cells with known values, exposes them to the beam, and then compares the results to the initial state to look for errors.

The L2 and L3 caches employ hardware scrubbers, so we expect the field error logs to capture the majority of bit flips that occur in these structures. Figure 8 compares the per-bit rate of SRAM faults in Hopper's L2 and L3 data and tag arrays to results obtained from the accelerated testing campaign on the SRAM cells. The figure shows that accelerated testing predicts a lower fault rate than seen in the field in all structures except the L2 tag array. The fault rate in the L2 tag is approximately equal to the rate from accelerated testing.

Overall, the figure shows good correlation between rates measured in the field and rates measured from static SRAM testing. Our conclusion from this correlation is that the majority of SRAM faults in the field are caused by known particles. While an expected result, confirming expectations with field data is important to ensure that parts are functioning as specified, to identify potential new or unexpected fault modes that may not have been tested in pre-production silicon, and to ensure accelerated testing reflects reality.

7. Fallacies in Measuring System Health

All data and analyses presented in the previous section refer to fault rates, not error rates. Some previous studies have reported system error rates instead of fault rates [12][14][16][30]. Error rates are heavily dependent on a system's software configuration and its workloads' access patterns in addition to the health of the system, which makes error rates an imperfect measure of hardware reliability. In this section, we show that measuring error rates can lead to erroneous conclusions about hardware reliability.

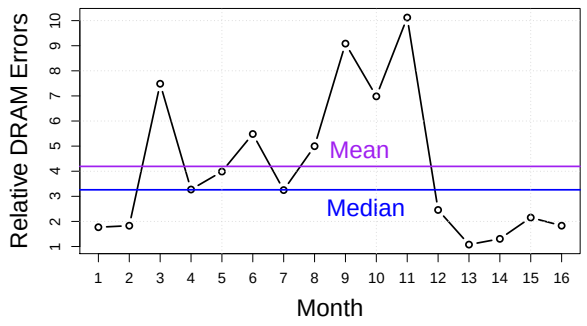

Figure 9. Hopper's memory error rate relative to Cielo. Hopper has a memory error rate 4x that of Cielo, but a memory fault rate 0.625x that of Cielo. Because error counts are confounded by other factors such as workload behavior, they are not an accurate measure of system health.

7.1 Error Logging Architecture

In most x86 CPUs, DRAM errors are logged in a register bank in the northbridge block [4] [31]. Each register bank can log one error at a time; x86 architecture dictates that the hardware discard subsequent corrected errors until the bank is read and cleared by the operating system [5]. The operating system typically reads the register bank via a polling routine executed once every few seconds [2]. Therefore, on a processor which issues multiple memory accesses per nanosecond, millions of errors may be discarded between consecutive reads of the register bank.

The error logging architecture described above means that the console logs represent only a sample of all corrected errors that have occurred on a system. Moreover, the number of logged corrected errors is highly dependent on the polling frequency set by the operating system. For instance, if the operating system is using a 5-second polling frequency, only one error per node can be reported per 5-second interval.

Uncorrected errors in x86 processors, on the other hand, are collected through a separate machine check exception mechanism [5]. These exceptions are individually delivered to the operating system, and thus uncorrected error counts do not suffer this sampling problem.

7.2 Counting Errors vs. Counting Faults

Using our data, we provide two concrete examples of how counting errors can lead to incorrect conclusions about system reliability. Our first example is a study by Di Martino *et al.* on the Blue Waters system [12]. The second example is a study by Schroeder *et al.* on memory errors in Google data centers [30].

Example 1. First, we examine the Di Martino *et al.* claim that the chipkill ECCs used on Blue Waters' DRAM corrected 99.997% of all logged errors. Because corrected errors are sampled while uncorrected errors are not, the chipkill codes on Blue Waters almost certainly corrected a much larger fraction of the total errors experienced by the system. More importantly, however, this type of analysis can lead to incorrect conclusions about the efficacy of ECC. For instance, using the authors' methodology, we find that Hopper's chipkill-detect ECC corrected 99.991% of all logged errors, while Cielo's chipkill-correct ECC corrected 99.806% of all logged errors. This implies that Hopper's ECC performs better than Cielo's ECC. However, Cielo's ECC actually had a 3x lower uncorrected error rate than that on Hopper, leading to the opposite conclusion: the chipkill-correct code on Cielo performs substantially better than the chipkill-detect code on Hopper.

Example 2. Our second example comes from a paper by Schroeder *et al.*, where the authors quote a memory error rate of 25,000-75,000 FIT/Mbit, and claim that DRAM is orders of magnitude less reliable than previously thought [30]. The rate measured by the authors is the rate of logged errors, not the rate of faults [28]. Using this methodology, we find Hopper's memory error rate was slightly more than 4x that of Cielo, again giving the impression that Hopper's DRAM is less reliable than Cielo's (see Figure 9). As stated in Section 6, however, Hopper has a fault rate of 25 FIT/device, compared to Cielo's 40 FIT/device [34], demonstrating that Hopper's DRAM is actually more reliable than Cielo's.

7.3 Importance of Counting Faults

The key observation in this section is, in order to obtain an accurate picture of system reliability, it is imperative to analyze the error logs to identify individual faults, rather than treating each error event as separate. Error event counts: (a) are more indicative of the OS's polling frequency than of hardware health; and (b) overemphasize the effects of permanent faults, which can lead to thousands or millions of errors, while underreporting the effects of transient faults, which typically only result in a few errors but can be equally deleterious to system reliability.

8. Analysis of Hardware Resilience Schemes

In this section, we use our data to examine various hardware resilience schemes in both DRAM and SRAM. Our goal is to understand the effectiveness of current hardware schemes. This data is useful for silicon and system designers who must consider a range of resilience techniques.

8.1 Comparing DRAM ECC Schemes

Many DRAM ECC schemes are used in the industry. The most common schemes are SEC-DED ECCs, chipkill-detect ECCs, and chipkill-correct ECCs [18]. SEC-DED ECC corrects single-bit errors and detects double-bit errors. Chipkill-detect ECC detects any error from a single DRAM device, while chipkill-correct ECC corrects any error from a single device. Prior studies show that chipkill-correct reduces the uncorrected error rate by 42x relative to SEC-DED ECC [33]. However, no prior study has quantified the difference in undetected errors between SEC-DED and chipkill.

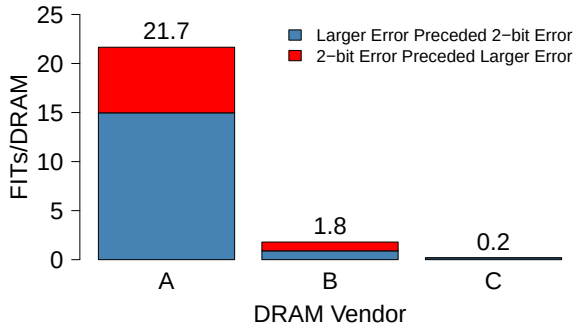

Figure 10. Rate of faults that generate errors which are potentially undetectable by SEC-DED ECC.

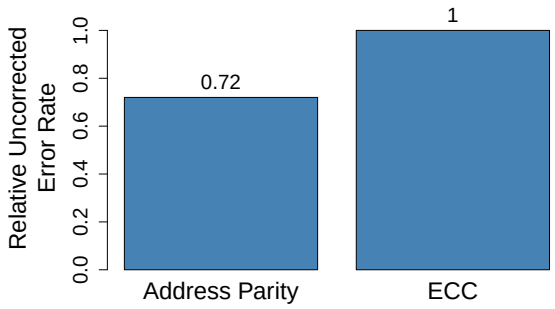

Figure 11. Rate of DRAM address parity errors relative to uncorrected data ECC errors.

Both Cielo and Hopper log the bits in error for each corrected error. By analyzing each error, we can determine whether a given error would be undetectable by SEC-DED ECC. For the purposes of this study, we assume that errors larger than 2 bits in a single ECC word are undetectable by SEC-DED (an actual SEC-DED ECC typically can detect some fraction of these errors). We call a fault that generates an error larger than 2 bits in an ECC word an *undetectable-by-SECDED* fault. A fault is undetectable-by-SECDED if it affects more than two bits in any ECC word, and the data written to that location does not match the value produced by the fault. For instance, writing a 1 to a location with a stuck-at-1 fault will not cause an error.

Not all multi-bit faults are undetectable-by-SECDED faults. For instance, many single-column faults only affect a single bit per DRAM row [33], and manifest as a series of ECC words with a single-bit error that is correctable by SEC-DED ECC.

Figure 10 shows the rate of undetectable-by-SECDED faults on Cielo, broken down by vendor. The rate of these faults exhibits a strong dependence on vendor. Vendor A has an undetectable-by-SECDED fault rate of 21.7 FIT per DRAM device. Vendors B and C, on the other hand, have much lower rates of 1.8 and 0.2 FIT/device, respectively. A Cielo node has 288 DRAM devices, so this translates to 6048, 518, and 57.6 FIT per node for vendors A, B, and C, respectively. This translates to one undetected error every 0.8 days, every 9.5 days, and every 85 days on a machine the size of Cielo.

Figure 10 also shows that 30% of the undetectable-by-SECDED faults on Cielo generated 2-bit errors (which are detectable by SEC-DED) before generating a larger (e.g., 3-bit) error, while the remaining 70% did not. We emphasize, however, that this detection is not a guarantee - different workloads would write different data to memory, and thus potentially exhibit a different pattern of multi-bit errors.

Our main conclusion from this data is that SEC-DED ECC is poorly suited to modern DRAM subsystems. The rate of undetected errors is too high to justify its use in very large scale systems comprised of thousands of nodes where fidelity of results is critical. Even the most reliable vendor (vendor C) has a high rate of undetectable-by-SECDED faults when considering multiple nodes operating in parallel.

Hopper and Cielo use chipkill-detect ECC and chipkill-correct ECC, respectively, and therefore exhibit much lower undetected error rates than if they used SEC-DED ECC.

8.2 DDR Command and Address Parity

A key feature of DDR3 (and now DDR4) memory is the ability to add parity-check logic to the command and address bus. The wires on this bus are shared from the memory controller to each DIMM's register. Therefore, errors on these wires will be seen by all DRAM devices on a DIMM. Though command and address parity is optional on DDR2 memory systems, we are aware of no prior study that examines the potential value of this parity mechanism.

The on-DIMM register calculates parity on the received address and command pins and compares it to the received parity signal. On a mismatch, the register signals the memory controller of a parity error. The standard does not provide for retry on a detected parity error, but requires the on-DIMM register to disallow any faulty transaction from writing data to the memory, thereby preventing data corruption.

The DDR3 sub-system in Cielo includes command and address parity checking. Figure 11 shows the rate of detected command/address parity errors relative to the rate of detected, uncorrected data ECC errors. The figure shows that the rate of command/address parity errors was 72% that of the rate of uncorrected ECC errors.

The conclusion from this data is that command/address parity is a valuable addition to the DDR standard. Furthermore, increasing DDR memory channel speeds will likely cause an increase in signaling-related errors. Therefore, we expect the ratio of address parity to ECC errors to increase with increased DDR frequencies.

8.3 SRAM Error Protection

In this section, we examine uncorrected errors from SRAM in Hopper and Cielo.

Figure 12 shows the rate of SRAM uncorrected errors on Cielo in arbitrary units. The figure separately plots uncorrected errors from parity-protected structures and uncor-

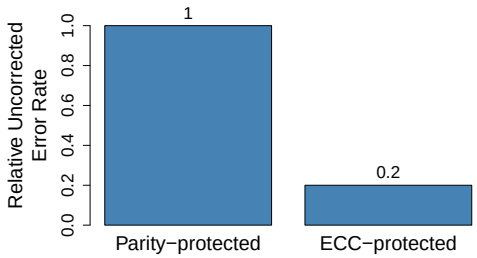

Figure 12. Rate of SRAM uncorrected errors in Cielo from parity- and ECC-protected structures.

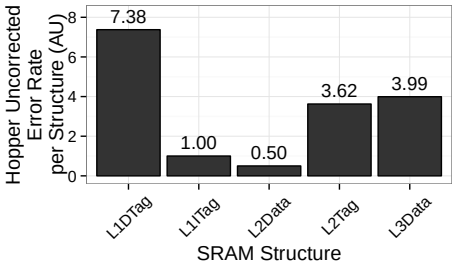

Figure 13. Rate of SRAM uncorrected errors per structure in Hopper from cache data and tag arrays. Each Hopper socket contains 12 L1 instruction and data caches, 12 L2 caches, and 2 L3 caches.

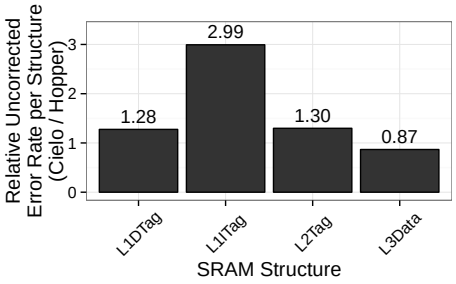

Figure 14. Rate of SRAM uncorrected errors in Cielo relative to Hopper. ECC and aggressive interleaving seem to mitigate the expected increase in uncorrected error rate due to altitude.

rected errors from ECC-protected structures, and includes structures in the core, all caches, and a variety of non-core arrays and FIFOs. ECC-protected structures include the L1, L2, and L3 caches, which comprise the majority of the die area in the processor.

The figure shows that the majority of uncorrected errors in Cielo come from parity-protected structures, even though these structures are far smaller than the ECC-protected structures. Parity can detect, but cannot correct, single-bit faults, while ECC will correct single-bit faults. Therefore, the majority of SRAM uncorrected errors in Cielo are the result of single-bit, rather than multi-bit, faults.

Our primary conclusion in this section is that the best way to reduce SRAM uncorrected error rates simply is to extend single-bit correction (e.g., SEC-DED ECC) through additional structures in the processor. While this is a non-trivial effort due to performance, power, and area concerns, this solution does not require extensive new research or novel technologies.

Once single-bit faults are addressed, the remaining multi-bit faults may be more of a challenge to address, especially in highly scaled process technologies in which the rate and spread of multi-bit faults may increase substantially [17]. In addition, novel technologies such as ultra-low-voltage operation will likely change silicon failure characteristics [37]. Current and near-future process technologies, however, will see significant benefit from increased use of on-chip ECC.

8.4 Analysis of SRAM Errors

In this section, we delve further into the details of observed SRAM uncorrected errors, to determine the root cause and potential avenues for mitigation.

Figure 13 shows the rate of uncorrected errors (in arbitrary units) from several representative SRAM structures in Hopper. In the processors used in Hopper, the L1 data cache tag (L1DTag) is protected by parity, and all errors are uncorrectable. The L2 cache tag (L2Tag), on the other hand, is largely protected by ECC. However, a single bit in each L2 tag entry is protected by parity. Because the L2 cache is substantially larger than the L1, there are approximately half as many parity-protected bits in the L2 tag as there are bits in the entire L1 tag. This is reflected in the observed uncorrected error rate from the L2 tag, which is approximately half the rate of uncorrected errors from the L1 tag[1].

Our observation from this data is that seemingly small microarchitectural decisions (e.g., the decision to exclude a single bit from ECC protection) can have a large impact on overall system failure rates. Therefore, detailed modeling and analysis of the microarchitecture (e.g., AVF analysis [25]) is critical to ensuring a resilient system design.

Prior field studies have shown that SRAM fault rates have a strong altitude dependence due to the effect of atmospheric neutrons [34]. True to this trend, Cielo experiences substantially more corrected SRAM faults than Hopper. While first principles would predict a similar increase in SRAM uncorrected errors at altitude, prior studies have not quantified whether this is true in practice.

Figure 14 shows the per-bit SRAM uncorrected error rate in Cielo relative to Hopper in the L1 instruction tag, L2 tag, and L3 data arrays. The figure shows that Cielo's error rate in the L2 and L3 structures is not substantially higher than Hopper's error rate, despite being at a significantly higher altitude. We attribute this to the error protection in these structures. These structures have both ECC and aggressive bit interleaving; therefore, a strike of much larger than two bits is required to cause an uncorrected error. Therefore, the

[1] This L2 Tag behavior has been addressed on more recent AMD processors.

data points to the conclusion that the multi-bit error rate from high-energy neutrons in these structures is smaller than the error rate from other sources, such as Alpha particles, and a variety of hard fault mechanisms.

Therefore, the data in this section suggest that appropriate error protection mechanisms can successfully offset the increase in raw fault due to altitude. We conclude that an appropriately-designed system need not be less reliable when located at higher elevations.

9. Lessons for Reliable System Design

There are several lessons about reliable system design to be gleaned from our study. We believe this study both confirms and rebuts many widely-held assumptions and also provides new insights valuable to system designers and researchers.

- **Faults are unpredictable.** Some faults may occur much more often than expected (e.g., DDR address parity) and some fault modes may not be known at design time (e.g., DRAM disturbance faults). Therefore, providing a robust set of error detectors is key.
- **Details matter.** For instance, simply describing a cache as "ECC-protected" does not convey adequate information, since excluding even a single bit from this protection can have a large negative effect on system reliability. Careful analysis and modeling (e.g., AVF analysis) are required to predict the expected failure rate of any device.
- **The ability to diagnose is critical.** This includes both ensuring that hardware logs have adequate diagnosis information as well as having access to appropriate software tools. For example, knowing exactly which bits are in error is critical to understanding fault modes and determining what went wrong. Adding this level of diagnosability to the hardware can require a much larger investment of time and effort than adding ECC to a structure in the first place.
- **Analysis is tricky.** Qualitatively, our experience is that this type of field study is difficult to perform correctly. These studies require understanding the details of hardware and software behavior (some of which may be undocumented), mining extremely large data sets to separate the signal from the noise, and carefully interpreting the results. Quantitatively, our data shows that not following these steps can lead to incorrect conclusions about system reliability.
- **Scale changes everything.** In large systems, even very small structures require protection due to the sheer number of components in a modern-day supercomputer or data center. Components that neglect this type of protection run the risk of corrupting data at non-trivial rates.

10. Projections to Future Systems

In this section, we examine the impact of DRAM and SRAM faults on a potential exascale supercomputer. Our goal is to understand whether existing reliability mechanisms can cope with the challenges presented by likely increases in system capacity and fault rates for future large-scale systems and data centers. To accomplish this, we scale our observed DRAM and SRAM error rates to likely exascale configurations and system sizes. We also model the impact of different process technologies such as FinFET transistors [15].

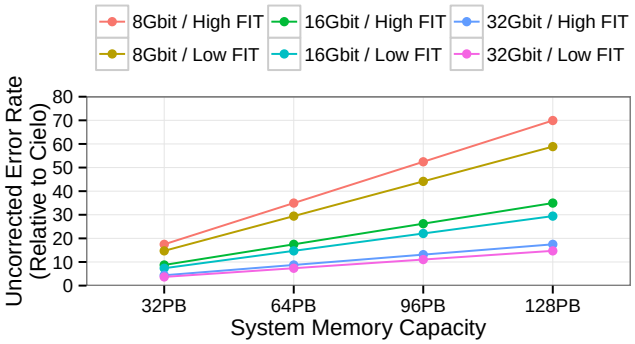

Figure 15. Rate of DRAM uncorrected errors in an exascale supercomputer relative to Cielo.

All analysis in this section refers to detected errors. Our projections are for currently-existing technologies and do not consider the impact of new technologies such as die-stacked DRAM, ultra-low-voltage CMOS, or NVRAM.

10.1 DRAM

Exascale supercomputers are predicted to have between 32PB and 128PB of main memory. Due to capacity limitations in die-stacked devices [32], much of this memory is likely to be provided in off-chip memory. The interface to these off-chip devices is will likely resemble current DDR memory interfaces. Therefore, it is critical to understand the reliability requirements for these off-chip sub-systems.

Prior work has shown that DRAM vendors maintain an approximately constant fault rate per device across technology generations, despite reduced feature sizes and increased densities [9]. Therefore, we expect the per-device fault rates in an exascale computer will be similar to those observed in todays DRAM devices. Our goal is to determine whether current error-correcting codes (ECCs) will suffice for an exascale supercomputer. Therefore, we assume that each memory channel in an exascale system will use the same single-chipkill code in use on Cielo. As a result, we project that the per-device uncorrected error rate in an exascale supercomputer will be the same as the per-device error rate in Cielo.

Because we do not know the exact DRAM device capacity that will be in mass production in the exascale timescale, we sweep the device capacity over a range from 8Gbit to 32Gbit devices. A larger DRAM device can deliver a specified system memory capacity with fewer total devices.

Figure 15 shows the results of this analysis. The figure plots the system-wide DRAM uncorrected error rate for an exascale system relative to the DRAM uncorrected error

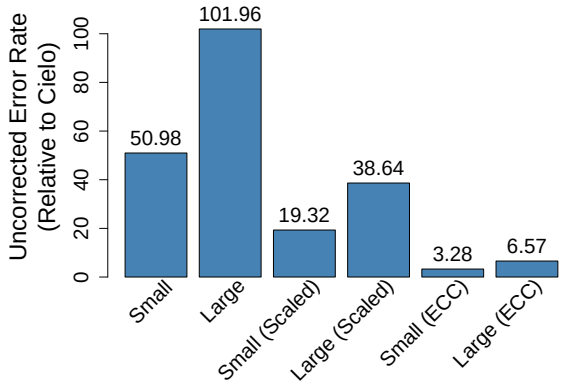

Figure 16. Rate of SRAM uncorrected errors relative to Cielo in two different potential exascale computers composed of CPU and GPU/APU nodes.

rate on Cielo. The figure plots two per-device error rates at each point, each taken from field data on different systems. The figure shows that uncorrected error rate for an exascale system ranges from 3.6 times Cielos uncorrected error rate at the low end to 69.9 times Cielos uncorrected error rate at the high end.

At a system level, the increase in DRAM uncorrected error rates at the high end of memory capacity is comparable to the 40x reduction in DRAM uncorrected errors achieved when upgrading from SEC-DED ECC to chipkill ECC [33]. If we assume that SEC-DED ECC provides insufficient reliability for todays memory subsystems, our conclusion from these results is that higher-capacity exascale systems may require stronger ECC than chipkill.

10.2 SRAM

We now turn our attention to SRAM failures on future systems. A socket in Cielo contains 18MB of SRAM in the L2 and L3 cache data arrays. Exascale-class processors are projected to see a substantial increase in processing power per socket, and thus will contain significantly more SRAM per node. Taking into account technology trends and reported structure sizes for CPU and GPU processors [3], we project that an exascale socket will contain over 150MB of SRAM, or an 8-10x increase in SRAM per socket over current supercomputers. The increase in SRAM relative to todays systems is less than the increase in DRAM because of the switch to general-purpose graphical processing units (GPGPUs) and/or GPGPUs integrated with CPU cores in an accelerated processing unit (APU), both of which rely less on large SRAM caches than traditional CPU cores.

Assuming that SRAM fault rates remain constant in future years, the per-socket fault rates will increase linearly with SRAM capacity. Therefore, an exascale processor will see 8-10x the number of SRAM faults experienced by a current processor, and potentially 8-10x the rate of uncorrected SRAM errors. This translates to a system-level uncorrected error rate from SRAM errors of 50-100 times the SRAM uncorrected error rate on Cielo, depending on the system size. This is shown in the first group of bars of Figure 16, labeled Small for a low node-count system (50k nodes) and Large for a high node-count system (100k nodes).

Per-bit SRAM transient fault rates have trended downwards in recent years [17]. If this trend continues, the SRAM uncorrected error rate per socket will be lower than our projections. For instance, according to Ibe et al. the per-bit SER decreased by 62% between 45nm and 22nm technology. If we see a corresponding decrease between current CMOS technologies and exascale technologies, the exascale system SRAM uncorrected error rate decreases to 19-39 times the uncorrected error rate on Cielo, shown in the second group of bars of Figure 16.

Finally, as noted previously, the majority of uncorrected SRAM errors are due to single-bit faults. If these errors are eliminated in an exascale processor (e.g. by replacing parity with ECC), the exascale system SRAM uncorrected error rate would be only 3-6.5 times Cielos uncorrected error rates, shown in the third group of bars in Figure 16.

Our conclusion from this analysis is that, vendors should focus aggressively on reducing the rate of uncorrected errors from SRAM faults. However, large potential reductions are available through reductions in SER due to technology scaling, and much of the remainder may be possible through expanding correction of single-bit faults. Once practical limits are reached, however, more advanced techniques to reduce the rate of multi-bit faults may be needed.

11. Summary

Reliability will continue to be a significant challenge in the years ahead. Understanding the nature of faults experienced in practice can benefit all stakeholders, including processor and system architects, data center operators, and even application writers, in the quest to design more resilient large-scale data centers and systems.

In this paper, we presented data on DRAM and SRAM faults, quantified the impact of several hardware resilience techniques, and extracted lessons about reliable system design. Our findings demonstrate that, while systems have made significant strides over the years (e.g., moving from SEC-DED ECC to chipkill on the DRAM subsystem), there is clearly more work to be done in order to provide a robust platform for future large-scale computing systems.

Acknowledgments

We thank R. Balance and J. Noe from Sandia, and K. Lamb and J. Johnson from Los Alamos for information on Cielo, and T. Butler and the Cray staff at NERSC for data and information from Hopper.

References

[1] Flux calculator. http://seutest.com/cgi-bin/FluxCalculator.cgi.

[2] mcelog: memory error handling in user space. `http://halobates.de/lk10-mcelog.pdf`.

[3] AMD. AMD graphics cores next (GCN) architecture. `http://www.amd.com/us/Documents/GCN\_Architecture\_whitepaper.pdf`.

[4] AMD. Bios and kernel developer guide (BKDG) for AMD family 10h models 00h-0fh processors. `http://developer.amd.com/wordpress/media/2012/10/31116.pdf`.

[5] AMD. AMD64 architecture programmer's manual volume 2: System programming, revision 3.23. `http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2012/10/24593_APM_v21.pdf`, 2013.

[6] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, Jan.-Mar. 2004.

[7] R. Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *IEEE Transactions on Device and Materials Reliability*, 5(3):305–316, Sept. 2005.

[8] K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R. S. Williams, and K. Yelick. Exascale computing study: Technology challenges in achieving exascale systems, Peter Kogge, editor & study lead, Sep. 2008.

[9] L. Borucki, G. Schindlbeck, and C. Slayman. Comparison of accelerated DRAM soft error rates measured at component and system level. In *IEEE International Reliability Physics Symposium (IRPS)*, pages 482–487, 2008.

[10] C. Constantinescu. Impact of deep submicron technology on dependability of VLSI circuits. In *International Conference on Dependable Systems and Networks (DSN)*, pages 205–209, 2002.

[11] C. Constantinescu. Trends and challenges in VLSI circuit reliability. *IEEE Micro*, 23(4):14–19, Jul.-Aug. 2003.

[12] C. Di Martino, Z. Kalbarczyk, R. K. Iyer, F. Baccanico, J. Fullop, and W. Kramer. Lessons learned from the analysis of system failures at petascale: The case of blue waters. In *International Conference on Dependable Systems and Networks (DSN)*, pages 610–621, 2014.

[13] A. Dixit, R. Heald, and A. Wood. Trends from ten years of soft error experimentation. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2009.

[14] N. El-Sayed, I. A. Stefanovici, G. Amvrosiadis, A. A. Hwang, and B. Schroeder. Temperature management in data centers: why some (might) like it hot. In *International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, pages 163–174, 2012.

[15] X. Huang, W.-C. Lee, C. Kuo, D. Hisamoto, L. Chang, J. Kedzierski, E. Anderson, H. Takeuchi, Y.-K. Choi, K. Asano, V. Subramanian, T.-J. King, J. Bokor, and C. Hu. Sub 50-nm FinFET: PMOS. In *International Electron Devices Meeting (IEDM)*, pages 67–70, 1999.

[16] A. A. Hwang, I. A. Stefanovici, and B. Schroeder. Cosmic rays don't strike twice: understanding the nature of DRAM errors and the implications for system design. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 111–122, 2012.

[17] E. Ibe, H. Taniguchi, Y. Yahagi, K. i. Shimbo, , and T. Toba. Impact of scaling on neutron-induced soft error in SRAMs from a 250 nm to a 22 nm design rule. In *IEEE Transactions on Electron Devices*, pages 1527–1538, Jul. 2010.

[18] X. Jian, H. Duwe, J. Sartori, V. Sridharan, and R. Kumar. Low-power, low-storage-overhead chipkill correct via multi-line error correction. In *International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, pages 24:1–24:12, 2013.

[19] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *International Symposium on Computer Architecture (ISCA)*, pages 361 – 372, 2014.

[20] X. Li, M. C. Huang, K. Shen, and L. Chu. A realistic evaluation of memory hardware errors and software system susceptibility. In *USENIX Annual Technical Conference (USENIX-ATC)*, pages 6–20, 2010.

[21] X. Li, K. Shen, M. C. Huang, and L. Chu. A memory soft error measurement on production systems. In *USENIX Annual Technical Conference (USENIXATC)*, pages 21:1–21:6, 2007.

[22] P. W. Lisowski and K. F. Schoenberg. The Los Alamos neutron science center. In *Nuclear Instruments and Methods*, volume 562:2, pages 910–914, June 2006.

[23] T. May and M. H. Woods. Alpha-particle-induced soft errors in dynamic memories. *IEEE Transactions on Electron Devices*, 26(1):2–9, Jan. 1979.

[24] A. Messer, P. Bernadat, G. Fu, D. Chen, Z. Dimitrijevic, D. Lie, D. Mannaru, A. Riska, and D. Milojicic. Susceptibility of commodity systems and software to memory soft errors. *IEEE Transactions on Computers*, 53(12):1557–1568, Dec. 2004.

[25] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin. A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor. In *International Symposium on Microarchitecture (MICRO)*, pages 29–40, 2003.

[26] J. T. Pawlowski. Memory errors and mitigation: Keynote talk for SELSE 2014. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2014.

[27] H. Quinn, P. Graham, and T. Fairbanks. SEEs induced by high-energy protons and neutrons in SDRAM. In *IEEE Radiation Effects Data Workshop (REDW)*, pages 1–5, 2011.

[28] B. Schroeder. Personal Communication.

[29] B. Schroeder and G. Gibson. A large-scale study of failures in high-performance computing systems. In *International Conference on Dependable Systems and Networks (DSN)*, pages 249–258, 2006.

[30] B. Schroeder, E. Pinheiro, and W.-D. Weber. DRAM errors in the wild: a large-scale field study. *Commun. ACM*, 54(2):100–107, Feb. 2011.

[31] T. Siddiqua, A. Papathanasiou, A. Biswas, and S. Gurumurthi. Analysis of memory errors from large-scale field data collection. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2013.

[32] J. Sim, G. H. Loh, V. Sridharan, and M. O'Connor. Resilient die-stacked DRAM caches. In *International Symposium on Computer Architecture (ISCA)*, pages 416–427, 2013.

[33] V. Sridharan and D. Liberty. A study of DRAM failures in the field. In *International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, pages 76:1–76:11, 2012.

[34] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi. Feng shui of supercomputer memory: Positional effects in DRAM and SRAM faults. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 22:1–22:11, 2013.

[35] A. N. Udipi, N. Muralimanohar, R. Balsubramonian, A. Davis, and N. P. Jouppi. LOT-ECC: Localized and tiered reliability mechanisms for commodity memory systems. In *International Symposium on Computer Architecture (ISCA)*, pages 285–296, 2012.

[36] J. Wadden, A. Lyashevsky, S. Gurumurthi, V. Sridharan, and K. Skadron. Real-world design and evaluation of compiler-managed GPU redundant multithreading. In *International Symposium on Computer Architecture (ISCA)*, pages 73–84, 2014.

[37] C. Wilkerson, A. R. Alameldeen, Z. Chishti, W. Wu, D. Somasekhar, and S.-l. Lu. Reducing cache power with low-cost, multi-bit error-correcting codes. In *International Symposium on Computer Architecture (ISCA)*, pages 83–93, 2010.