Bamboo ECC: Strong, Safe, and Flexible Codes for Reliable Computer Memory

Jungrae Kim Michael Sullivan Mattan Erez

{dale40, mbsullivan, mattan.erez}@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin

Abstract

Growing computer system sizes and levels of integration have made memory reliability a primary concern, necessitating strong memory error protection. As such, large-scale systems typically employ error checking and correcting codes to trade redundant storage and bandwidth for increased reliability. While stronger memory protection will be needed to meet reliability targets in the future, it is undesirable to further increase the amount of storage and bandwidth spent on redundancy. We propose a novel family of single-tier ECC mechanisms called Bamboo ECC to simultaneously address the conflicting requirements of increasing reliability while maintaining or decreasing error protection overheads.

Relative to the state-of-the-art single-tier error protection, Bamboo ECC codes have superior correction capabilities, all but eliminate the risk of silent data corruption, and can also increase redundancy at a fine granularity, enabling more adaptive graceful downgrade schemes. These strength, safety, and flexibility advantages translate to a significantly more reliable memory system. To demonstrate this, we evaluate a family of Bamboo ECC organizations in the context of conventional 72b and 144b DRAM channels and show the significant error coverage and memory lifespan improvements of Bamboo ECC relative to existing SEC-DED, chipkill-correct and double-chipkill-correct schemes.

1. Introduction

The reliability of main memory is a significant and growing concern. DRAM components may become more vulnerable to hard and intermittent faults as DRAM transistor dimensions and supply voltages decrease [1], leading to more permanent and repeated memory errors [2, 3, 4]. When also considering the growing number of memory parts in large systems and the high cost of replacing recently-introduced on-package stacked memories [5, 6, 7], it is clear that strong and long-lasting error protection for future memory systems is necessary.

Business-critical servers, datacenters, and high performance systems currently employ chipkill-correct or double-chipkill reliability mechanisms [8, 9, 10] to protect data in memory from errors. Such schemes use error coding to allow for continuous operation despite the failure of one or two DRAM chips. Field analyses report that chipkill-correct significantly improves memory reliability by correcting 99.94% of all errors [11], achieving a 42× better uncorrected DRAM error rate than single-bit correction [2].

While chipkill-correct and double-chipkill bring needed reliability improvements to high-availability systems, future memory systems will require stronger error correction and detection capabilities due to increasing error severity and levels of integration. These stronger capabilities are particularly important for memories that are integrated within a processor package because of their wide chip interfaces and high replacement costs. Meanwhile, despite this heightened need for strong memory protection to prevent potentially disastrous errors, tight energy and storage budgets make it undesirable to spend further on main memory reliability mechanisms. This paper presents and evaluates a novel family of efficient single-tier ECC mechanisms called *Bamboo ECC* codes that simultaneously address these conflicting requirements. Bamboo ECC codes provide significantly stronger protection than the current state-of-the-art ECC mechanisms, while requiring the same or less redundant storage and off-chip bandwidth.

The advantages of Bamboo ECC codes can be roughly characterized by three important improvements. **Strength:** Bamboo ECC codes have superior correction capabilities, and can correct more pin and chip errors than the state-of-the art single-tier ECC mechanism. **Safety:** The vastly superior detection capability of Bamboo ECC all but eliminates the risk of silent data corruption with currently observed fault modes, ensuring safe system operation. **Flexibility:** Bamboo ECC codes can increase redundancy at an 8b granularity, compared to 8B for the state-of-the-art chipkill technique. This fine-grained redundancy allows for more adaptive graceful downgrade schemes, further improving both reliability and system lifetime. When combined into a system context these improvements can lead to orders of magnitude fewer silent data corruptions or greatly extended system lifetime.

Three main insights lead to our innovative code design: (1) many faults manifest errors on a single data I/O due to the DRAM internal structure (e.g. a subarray fault) or the DRAM external interface (e.g. a fault in a through-silicon-via [TSV]), (2) aligning ECC symbols to frequent error patterns allows more frequent corrections, (3) while an ECC code guarantees detec-

Appears in HPCA 2015.

tion of errors up to a certain severity, careful analysis reveals that some codes provide superior detection capabilities beyond these guarantees.

Bamboo ECC applies large-symbol ECC to the burst of data received from one or more adjacent memory I/O pins or TSVs. This vertical ECC layout allows Bamboo ECC to effectively correct DRAM pin/TSV and chip errors; meanwhile, Bamboo ECC achieves the minimum possible redundancy for its level of correction by using large symbols and can detect almost all severe errors by using large ECC codewords. This large ECC word size results in a manageable increase in decoding complexity and latency while increasing ECC strength, safety, and flexibility. Current trends indicate that logic will continually get cheaper and faster than memory [12] such that it is natural for Bamboo ECC to use on-chip logic in lieu of further straining the already-overtaxed memory system.

We chose the name Bamboo ECC because the vertically-aligned layout of the codes brings to mind the vision of a bamboo forest. The strength and flexibility of young bamboo, along with its importance in safety-critical scaffolding, bespeaks the advantages of this family of codes.

The remainder of the paper proceeds as follows. Sections 2.1–2.3 review the function and terminology of DRAM, system reliability mechanisms, and ECC codes, introducing the conceptual foundations of Bamboo ECC codes. Section 3 briefly describes the rich tapestry of prior memory ECC codes, with a focus on deciphering the state-of-the-art codes employed in current systems. Section 4 describes Bamboo ECC codes, with an emphasis on how the codes can improve the reliability and memory lifespan of a large machine. Finally, Section 5 evaluates Bamboo ECC codes, demonstrating their substantial advantages.

2. Background

This section reviews the concepts and terminology that are fundamental to a full description and evaluation of Bamboo ECC. Brief introductions to DRAM, system reliability concepts, and ECC codes are presented below.

2.1. DRAM

Dynamic Random Access Memory (DRAM) is widely used due to its low cost and high density. DRAM represents each bit of memory using a single transistor and capacitor, organizing these memory cells in in two-dimensional arrays (*banks*) to amortize control overheads. Each bank is sub-divided into 512 $\times$ 512 cell *subarrays* and all data within neighboring subarrays are connected to one or more neighboring data pins for efficiency [13, 14, 15, 16].

Most DRAMs use multiple *data pins (DQs)* to provide a parallel chip interface. A DRAM with an N-bit DQ interface is called a $\times$N chip and multiple chips are accessed together in parallel to provide a wider external interface called a *rank*. A group of *ranks* that time-share the same control and data interface is called a *channel*. A DRAM access transfers a block of data bit-by-bit over multiple cycles—one *beat* of data is transferred through the rank every cycle, and the number of beats transferred during each access is called the memory *burst length*. Many systems use a burst length of 8, such that each access transfers 64/128B over a 64/128b data channel.

The deeply hierarchical structure of DRAM is depicted in Figure 1. Memory cells are stored in subarrays (1a), which are formed into banks (1b). In turn, these banks are used to form chips, ranks, and channels (1c). Recent field studies of DRAM faults indicate that memory errors follow some idiosyncratic trends due to this structure [4, 17, 18, 2, 3]. Large-scale analyses of DRAM fault patterns in Jaguar (with DDR2) [2] and Cielo (with DDR3) [3] indicate that most faults are confined to a single DQ, owing to the subarray structure of DRAM. Bamboo ECC is well suited for correcting this important fault mode because of its vertical symbol layout.

2.1.1. DRAM trends and Bamboo ECC A memory channel has several design options, including the width and number of DRAM chips used. Server systems often use narrow DRAM chips (e.g. 16 $\times$ 4 DDR4 chips per 64b channel) to increase the DRAM capacity per channel at the expense of energy consumption. Graphics and mobile systems generally use wider DRAM chips (e.g. 2 $\times$ 32 GDDR5 [19] or LPDDR3 [20] chips) to provide more bandwidth per chip and consume less energy per channel by accessing fewer chips. Recent on-package memories with massive vertical interconnections [5, 6, 21] provide a single very wide $\times$128 chip to service an entire access in an energy-efficient manner. These wider DRAM chips have increased reliability concerns as a single chip fault can potentially affect many bits within an access. The high error detection coverage of Bamboo ECC can be used to mitigate reliability concerns for wide DRAM chips, as safe operation can even be maintained in the presence of very severe errors.

In addition to the width of individual DRAM chips, designers can increase the channel width at the cost of decreasing the number of ranks in the system. While a larger channel can simplify ECC schemes (for reasons we explain later), the larger access granularity from a wider channel often results in a serious performance degradation due to over-fetching and reduced memory-level parallelism. Fujitsu reports that pairing two 64b channels together in an 8 DIMM system to form four 128b data channels degrades commercial application performance (SPECint2006 [22]) by 6%, and memory-intensive benchmark (STREAM [23]) performance by 43%. The performance hit is even more staggering on a lower capacity node—with 4 DIMMs, the performance degradation for the commercial and memory benchmarks are 28.3% and 46.3%, respectively [24]. Bamboo ECC is designed to operate on narrow DRAM channels, avoiding the performance and efficiency penalties of increasing channel width at the expense of channel count.

2.2. Reliability

As a matter of terminology: a *fault* is a physical phenomenon or defect that may cause an error or failure, an *error* is a discrepancy between the intended data in a system and its actual informa-

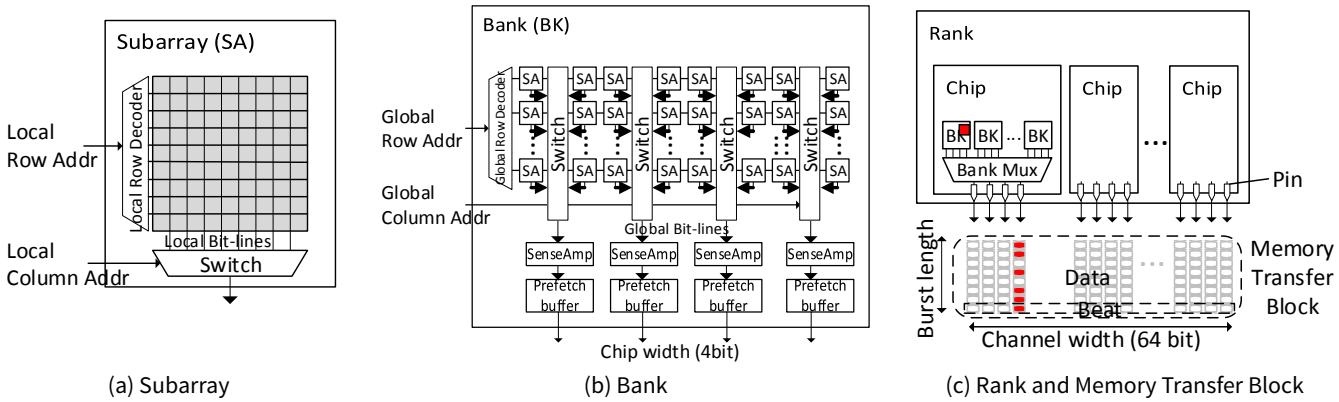


Figure 1: The internal organization of DRAM: subarrays, banks, ranks, channels, and the memory transfer block are shown.

tion, and a *failure* is an instance in time when a system displays a behavior that is contrary to its specification [25]. A fault may or may not generate an error depending on circumstances, and an error can be masked or corrected by error control systems. The aim of this research is to efficiently provide such an error control system to prevent DRAM data errors from developing into a system failure. In this paper an error indicates a memory data error, where data in memory is different from what was previously written, and a failure indicates a memory failure, where a memory data error is neither detected nor corrected by ECC and as such erroneous data are provided to the processor.

No error control system can diagnose and correct all possible errors. In general, an error control system handles an error in one of four ways. The error can be properly diagnosed and corrected, the error can be diagnosed but judged too severe or costly to correct, the error can be misdiagnosed and improperly corrected, or the error can remain completely undiagnosed. Due to the importance of this distinction, the notations *DCE*, *DUE*, *DME* and *UUE* are used to denote *Detectable and Correctable Errors*, *Detectable but Uncorrectable Errors*, *Detectable but Miscorrected Errors*, and *Undetectable and Uncorrectable Errors*, respectively. In the event of a DME or UUE, corrupted data are mistakenly provided to the processor under the auspices of the error control system. If the running application consumes this corrupted data, *silent data corruption* may occur with no indication to the system or user. The notation *SDC* is used to denote the event where corrupted data are provided to the processor by either a DME or UUE.

The implications of a DUE can vary depending on circumstances. In the case of memory, a DUE indicates that some data has been lost. This loss of data could be acknowledged and tolerated (by an error tolerant application), it may be corrected by some higher-level protection mechanism (such as checkpoint and restart or a hierarchical state preservation and restoration runtime system [26]), or it may indicate a fail-stop condition where forward progress is halted (but no silent data corruption occurs). Section 5 evaluates Bamboo ECC in the context of machines with high-level state recovery facilities, as many high-performance and high-availability machines have system-level recovery mechanisms in place. Therefore, it makes sense that the memory system reliability mechanisms should complement these system level mechanisms.

2.3. Error Correcting Codes

An *error checking and correcting (ECC)* code detects and corrects errors by adding redundant information whose value is generated algorithmically from the protected data. A data and check code pair is called an ECC word. A valid ECC word whose check bits are consistent with its data is called a *codeword*, while an invalid pair is called a *non-codeword*. The process of generating a codeword from data is called *encoding* and the process of detecting errors from a word and (possibly) restoring the original data is called *decoding*.

2.3.1. Theoretical code coverage: The *Hamming distance* between two messages of equal length is the number of positions at which their symbols differ. Correspondingly, the *code distance* of an ECC code is the minimum Hamming distance between two distinct codewords. The distance of a code is directly related to its guaranteed worst-case error detection and correction capabilities. A code with distance d, if used for detection only, can detect all errors with less-than-d erroneous symbols. Alternatively, a code with distance d can detect and correct all errors with less than $\lfloor (d-1)/2 \rfloor$ erroneous symbols by decoding a non-codeword into the nearest valid codeword.

2.3.2. Practical code coverage: Errors that exceed the theoretical coverage of a code can either be detected (resulting in a DUE) or they can lead to silent data corruption. Another motivation of Bamboo ECC is that, in practice, the error detection coverage of all error codes is not equal: in general, scaling-up ECC code word size (increasing code word length, redundancy and correction capabilities proportionally) is associated with higher error detection coverage for severe errors. For example, a two single-symbol correcting codes can be merged into a double symbol correcting one. While the latter can correct all of the errors that the former can, its larger codeword bestows stronger error detection capabilities. By maximizing its ECC word size, Bamboo ECC can safely reduce the SDC probability for severe errors down to practically zero.

We illustrate why the practical error detection coverage of an ECC code is determined by its word size using an example.

Figure 2 shows a conceptual codespace for a distance 3 (or *Single Symbol Correcting (SSC)*) code. A ball of Hamming distance d represents all words that are d symbols different from a codeword. If there is a single symbol error in a codeword, the erroneous word is on the HD=1 (innermost) ball and the SSC code can always restore the original data by finding the nearest codeword. If there are two symbol errors, the erroneous word is on the HD=2 ball and the error will be miscorrected to a neighboring codeword if and only if it also falls on the HD=1 ball of the neighbor; otherwise, the error will result in a DUE. Note that the theoretical coverage of this code is not double symbol detection as there exists this potential for miscorrection. From a practical coverage perspective, however, the ratio of the number of miscorrections to the total number of words on each ball decreases with larger ECC word sizes as the code space becomes increasingly sparse. Thus, it can be seen that the error detection of a code for errors beyond its guaranteed coverage is maximized with the ECC word length. Likewise, three symbol errors on the HD=3 (outermost) ball may be undetected (if they fall on a neighboring codeword), miscorrected (if they fall on another HD=1 ball), or they are otherwise detected. Again, it can be seen that the error detection of a code for errors beyond its theoretical coverage increases with the ECC word length.

High error detection of severe errors is fundamental for Bamboo ECC codes to ensure safe system operation. Therefore, all of our analyses in Section 5 are performed using real ECC decoding behavior in order to capture the practical error detection coverage of the codes for errors beyond their worst-case protection guarantees. This is in contrast to other studies that characterize system failure rates using only the worst-case behavior of ECC codes [27, 28]. An important observation of Bamboo ECC codes is that, in practice, silent data corruption can be eliminated without resorting to expensive ECC codes with high worst-case error detection coverage. By manipulating the size of the Bamboo ECC codeword and by matching code layout with expected fault modes, we can extensively reduce SDC rates without introducing additional redundancy or constraints on the memory channel size.

3. Prior Work

ECC codes have long been used to detect and correct DRAM errors. A brief review of prior error protection approaches is presented below. An emphasis is made on deciphering the state-of-the-art memory protection schemes used by industry. Outdated and noncompetitive approaches are also precluded from later evaluation with Bamboo ECC in order to save space in Section 5.

3.1. SEC-DED

A simple but widely-used ECC scheme for DRAM applies a *Single Error Correcting-Double Error Detecting (SEC-DED)* code to each beat of a memory transfer (Figure 3a). On a 64b data channel, 8b of redundancy are needed for SEC-DED, leading to the industry-standard 72b ECC DIMM with 12.5% redundancy.

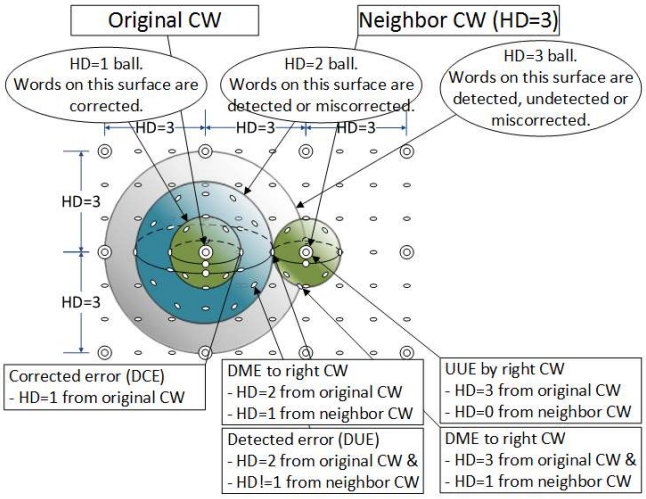


Figure 2: The conceptual codespace of an SSC code. A double circle indicates a valid codeword and a single circle indicates an erroneous non-codeword. Balls centered around a codeword represent all words that have the same Hamming distance (HD) from the codeword (e.g. words on the innermost ball are a Hamming distance of 1 away from the original codeword).

3.2. Interleaved SEC-DED Codes

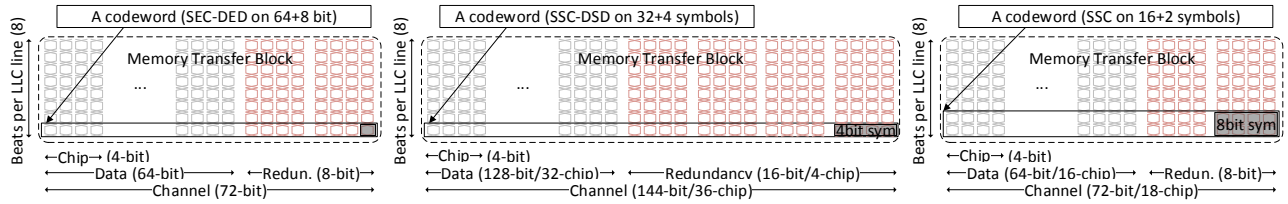
Rising system sizes and levels of integration have rendered SEC-DED insufficient for many high-performance and high-availability systems. As such, chipkill-correct protection is important to provide continuous operation despite a failing DRAM chip. One straightforward way to provide chipkill-level memory protection is to interleave four SEC-DED codewords together. By distributing data from a ×4 DRAM chip over 4 different codewords, single-chip-correct and double-chip-detect can easily be achieved.

While such an approach was employed by IBM, HP, and EMC in the past [30], it requires a 256b data channel. As was mentioned in Section 2.1, forming such wide channels is often disastrous to system performance and efficiency. For this reason, interleaved SEC-DED codes are not competitive with Bamboo ECC.

3.3. 4-bit RS Codes

Sun and older AMD chips seem to use a 4-bit symbol *Single Symbol Correcting-Double Symbol Detecting (SSC-DSD)* Reed-Solomon (RS) code [31] to provide chipkill-correct on ×4 DRAM chips (Figure 3b) [32, 33]. The symbols are aligned to chip boundaries so that a chip-fault is confined to a single symbol and can be corrected by SSC-DSD. Four symbols of redundancy are needed to provide chipkill protection because of the narrow (4-bit) symbol size. This scheme requires a 144b memory channel, and as such cannot compete with the efficiency of Bamboo ECC.

Chipkill-correct level protection is referred to as Chipkill, Single Device Data Correction (SDDC), extended ECC, and ChipSpare protection by IBM, Intel, Sun (now Oracle), and HP, respectively [8, 9, 29, 10]



(a) SEC-DED codeword ((64+8)-bit channel / 1-beat).

(b) 4-bit Symbol Chipkill (SSC-DSD) ((128+16)-bit channel / 1-beat).

(c) 8-bit Symbol Chipkill (SSC) ((64+8)-bit channel / 2-beat).

Figure 3: The codeword layout of ECC schemes that are currently in use.

3.4. 8-bit RS Codes

Newer AMD chips seem to use an SSC RS code with an 8-bit symbol to provide chipkill-correct on ×4 DRAM chips [8, 34]. The 8-bit symbols are built from two beats from the same ×4 chip so that a chip fault is confined to a single symbol. (Figure 3c). The large, 8-bit symbol size achieves SSC with 2 redundant symbols, allowing a 72b memory channel to be serviced by a single DIMM. Operating on a narrow channel makes this scheme efficient; as such it is the closest related work to Bamboo ECC codes, and is used as the baseline chipkill-correct design in Section 5.

Because this scheme does not provide *Double Symbol Detecting (DSD)* protection with a 72b memory channel, 2-chip faults may lead to silent data corruption. To mitigate these concerns, AMD uses the history of corrected symbol locations within each cache line to heighten error detection. If corrected symbol locations differ among codewords within a cache line, AMD chipkill reports a DUE. The rationale behind this decision is that errors on different chips over a single cache line are very rare so that the symptom is likely to have been generated from DMEs arising from a severe error.

3.5. Redundant Bit Steering and Double-Chipkill

Double-chipkill-correct is provided by some products for applications that demand higher levels of protection. *Redundant bit steering* (RBS) is a technique that was developed by IBM for enterprise mainframe computers to provide double-chipkill-correct protection [35]. Redundant bit steering uses part of the ECC check code as hot spare bits, remapping faulty pins through these spares. Remapping is done in the memory controller and is transparent to both the OS and the user.

It seems as if double-chipkill-correct capabilities are provided in current products through RBS. Most notably, Intel's *Double Device Data Correction (DDDC)* [10] (referred to by HP as Double ChipSpare [36]) appears to correct two sequential chip-errors by applying a chip-level protection through dynamic bit steering. If a chip fails, a spare chip is used to replace the failed chip. More recent products provide *DDDC+1*, which is able to correct an additional single bit-error on top of *DDDC* [10].

While the exact details of RBS are unknown, a sensible scheme that matches the reported redundancy requirements of commercial products follows. An RS code with 4-bit symbols and a 128b data channel requires a 3-symbol check code to provide SSC protection and an extra symbol to provide SSC-DSD. At the beginning of system operation (assuming no faults), memory uses all available pins to provide SSC-DSD protection. Upon a detected chip or pin error, the memory controller downgrades all affected memory to an SSC code and remaps the faulty chip through the fourth redundant symbol.

By downgrading affected memory locations from an SSC-DSD code to an SSC code, a memory system can tolerate up to 2 successive chip failures (DDDC-level protection). It seems likely that DDDC+1 downgrades protection to an SEC code following a second successive chip failure to provide end-of-life bit-correction capabilities. Bamboo ECC codes are designed to be highly amenable to RBS. Compared to other chipkill ECC codes, Bamboo ECC codes can provide a fine-grained retirement (such as a pin retirement) to face single-DQ faults. In coordination with retirement, Bamboo ECC can provide superior correction capabilities for sequential faults, correcting two concurrent chip-faults and up to 3 sequential chip faults and 3 pin faults on the same 128b channel as DDDC+1.

3.6. Multi-Tiered ECC Approaches

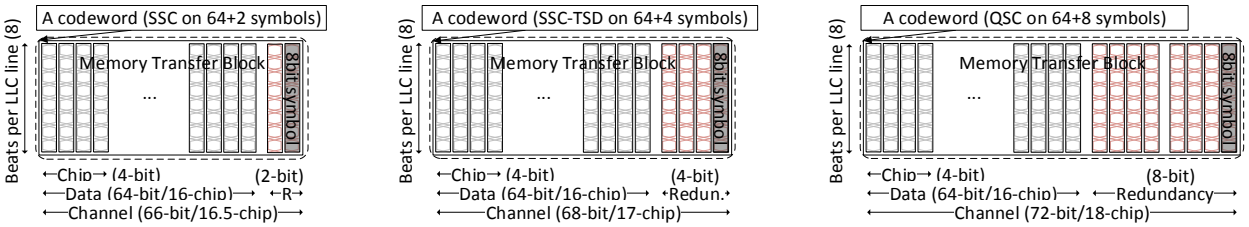
Current commercial chipkill-correct schemes utilize single-tiered ECC mechanisms, using a dedicated correction check symbol per codeword and operating in the conventional ECC DIMM memory footprint. In addition to these commercial chipkill-correct schemes, there are a number of academic approaches (including Virtualized ECC [37], LOT-ECC [38] and Multi-ECC [39]) that use additional storage to provide strong, low-redundancy error protection.

Such multi-tiered ECC schemes use a check code per codeword to detect errors and apply a second stronger error correction mechanism at a larger granularity. The goal of Bamboo ECC is to provide strong single-tier error protection for memory. As such, Bamboo ECC codes are orthogonal to these multi-tiered approaches—in fact, it is possible that Bamboo ECC could be put to good use as a component of such schemes, but such an evaluation is outside the scope of this paper.

4. Bamboo ECC

Bamboo ECC codes protect the burst of data from one or more DRAM DQs/TSVs to provide stronger correction and detection with equal or less redundancy than existing ECC codes. The following describes the motivation, operation, and overheads of Bamboo ECC in greater detail. Section 4.1 describes some

Now referred to as IBM *Memory ProteXion*.



(a) SPC on a (64+2)-bit Channel. (b) SPC-TPD on a (64+4)-bit Channel. (c) QPC on a (64+8)-bit Channel.

Figure 4: Bamboo ECC layouts on a 64b data channel (with an 8b burst length).

useful DRAM Bamboo ECC organizations. Section 4.2 shows how the flexibility of Bamboo ECC can be utilized to provide superior levels of protection over the lifetime of a system. Finally, Section 4.3 describes the cost that Bamboo ECC schemes pay to achieve their high levels of error protection and storage efficiency.

4.1. Bamboo ECC Organizations

To protect against DRAM errors, Bamboo ECC groups per-pin data as ECC symbols and uses an 8-bit symbol RS code to provide strong pin and chip protection. We describe some of efficient Bamboo ECC organizations below. Other organizations can be used to meet different memory system constraints and reliability requirements.

4.1.1. SPC: The simplest Bamboo ECC is the *Single Pin Correcting (SPC)* (Figure 4a) organization, which can correct a bit or a pin error with 2 redundant pins. SPC requires just a quarter of the redundant storage of SEC-DED on a 64b data channel (3.1% vs. 12.5%) yet it provides a better uncorrectable error rate. One issue with SPC concerns the fixed granularity of commercial DRAM chips—while memories such as embedded DRAM (eDRAM) have the data width flexibility to support SPC, off-the-shelf memory chips are typically ×4 or ×8 DDR. Employing SPC with these chips will result in an inefficient use of pins and storage. Even on commodity DRAM chips, however, SPC can be efficiently employed as a component of a graceful degradation scheme, as described in Section 4.2.

4.1.2. SPC-TPD: An extra ×4 DDR chip can provide 4 redundant Bamboo ECC symbols (Figure 4b). This redundancy can be used as either a *Double Pin Correcting (DPC)* or *Single Pin Correcting - Triple Pin Detecting (SPC-TPD)* scheme. We prefer SPC-TPD usage because it has a very high detection coverage, detecting 100% of up-to-3-pin errors and virtually all (99.9996%) errors beyond this point. The stronger correction capability of DPC, on the contrary, is less helpful as faults affecting exactly 2 pins are infrequent (a field measurement on 2-bit symbol correction [2] showed little improvement over SEC-DED) while it can increase the SDC probability by aggressively miscorrecting severe errors. SPC-TPD can be configured in a (64+4)-DQ configuration over a 64-bit data channel, halving the redundancy of SEC-DED (6.25% vs. 12.5%).

SPC misses some SEC-DED-correctable error patterns. However, our evaluation (Section 5.1) shows that SPC has better uncorrectable error rates due to the rarity of these patterns and the fact that SPC has a lower raw error rate due to its lesser redundancy.

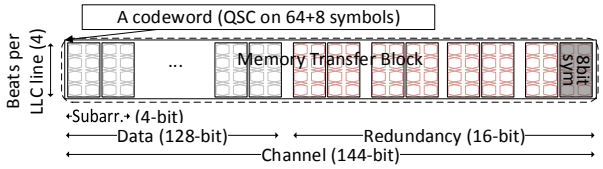


Figure 5: QDPC on a 128b data channel (2-pin × 4-beat symbols).

4.1.3. QPC: Bamboo ECC can provide chipkill-level protection with equal-or-less redundancy than state-of-the-art schemes by using a *Quadruple Pin Correcting (QPC)* organization. With two redundant ×4 chips, QPC has 8 redundant symbols (Figure 4c) and can correct up to 4 symbol errors or any single chip-error. QPC can be configured in a (64+8)-DQ manner over a 64-bit data channel, in which case the redundant storage needs match those provided by conventional ECC DIMMs (12.5%). This storage efficiency for single-tier, narrow-channel chipkill-correct is only paralleled by the AMD chipkill scheme used in recent processors. QPC enjoys a stronger correction capability than AMD chipkill by correcting pin errors that are scattered over different chips. If there are two pin faults on two chips, QPC can correct both of them while AMD chipkill must report a DUE (or, in some cases, AMD chipkill results in silent data corruption). In addition, QPC has a stronger detection capability due to its large codeword size. This leads to QPC detecting virtually 100% of all errors. A more in-depth comparison of the two ECC schemes is evaluated section 5.

One optimization to QPC is to limit its correction capability to reduce SDC rate even further, in a manner similar to the history mechanism used by AMD chipkill. While QPC can correct any 4 independent pin errors, the likelihood of having four different chips each suffer from a pin error is very low. Therefore, when such very rare errors are reported, they can be pessimistically considered to result from a severe error miscorrection. In Section 5, QPC pessimistically reports a DUE if the diagnosed pin errors belong to neither a single chip nor two separate pins.

4.1.4. OPC: Bamboo ECC can correct errors on two ×4 chips or one ×8 chip by correcting 8 pin symbols. *Octuple Pin Correcting (OPC)* Bamboo ECC achieves 8 pin correction with 16 redundant pins, resulting in a 25% overhead on a 64-bit data channel or a 12.5% overhead on a 128-bit data channel. We optimize OPC in a manner similar to that of QPC and limit its correction capability to 2 chip errors or 4 independent pin errors to reduce the SDC rate.

4.1.5. DDPC and QDPC for Wide-IO: While conventional off-package memories (e.g. DDR4, LPDDR3, and GDDR5) have a large enough burst length to build an 8-bit symbol from each pin, recent on-package memories, such as Wide-IO memory [5], use shorter 4b burst lengths to maintain a 64B access granularity over a 128-bit data interface. Another way to configure Bamboo ECC for these memories is to build 8-bit symbols over 2-pin $\times$ 4-beat blocks (Figure 5). These double-pin symbols still achieve the minimum redundancy with an 8-bit symbol without requiring a large burst length.

Double Double-Pin Correcting (DDPC) can correct a $\times$4 error by correcting two 2-pin symbols and *Quadruple Double-Pin Correcting (QDPC)* can correct two $\times$4 errors or one $\times$8 error by correcting four 2-pin symbols. DDPC and QDPC use 8/16 redundant pins, requiring 6.3%/12.5% storage overheads on the 128-bit data channel that is typical of wide on-package memories. We also limit the correction capability of DDPC and QDPC to reduce the SDC rate further, in a manner similar to QPC and AMD chipkill. For DDPC, if the corrected symbol locations do not belong to the same nibble position, we discard the correction and report it as a DUE. While this will miss opportunities for correcting two independent pin errors on different nibbles, it strongly reduces SDC events from other serious errors. Similarly, QDPC limits its correction capability to up to 2 nibbles or 3 double-pins.

4.2. Graceful Downgrade

As was described in Section 3.5, redundant bit steering can be used to gracefully downgrade memory as it accumulates permanent or intermittent faults, increasing both system lifetime and error resilience. The state-of-the-art RBS-based double-chipkill-correct scheme is DDDC+1, from Intel, which can tolerate two sequential chip failures and a bit failure.

The plethora of attractive Bamboo ECC codes above allows a Bamboo ECC-based system to more fully utilize RBS than existing ECC schemes. A gracefully downgrading Bamboo ECC-based system can correct more initial chip faults, more sequential chip faults, more end-of-life faults and has a better end-of-life detection capability than DDDC+1. In addition, a gracefully degrading Bamboo ECC system can diagnose and remap errors at the pin granularity, offering slower degradation for accumulating pin errors than a system that operates on coarse-grained symbols. Such a gracefully degrading Bamboo ECC scheme is described below.

OPC over a 128-bit data channel uses the same amount of redundancy (12.5%, or 16 pin symbols) as DDDC+1. After a pin or chip retirement, the available redundancy decreases to 12 pins, but Bamboo ECC can still operate in *hextuple pin correcting* mode to correct up to 6 pin errors. Successive pin errors can be diagnosed and retired at the pin granularity—a luxury that non-Bamboo ECC codes do not enjoy.

After the available spare pins are exhausted due to further pin or chip retirement, the redundancy of the Bamboo ECC-based system decreases down to 8 pins, which is sufficient for QPC. Finally, following a third round of pin or chip retirement, the system can downgrade to SPC-TPD using the remaining 4 redundant (non-spare) pins. Due to the storage efficiency of SPC-TPD, the Bamboo ECC-based system will still be able to correct a pin error and detect virtually 100% of errors. As a result, this graceful downgrade scheme can correct two concurrent chip errors in its initial OPC phase and then can correct up to 1 sequential chip error and 1 pin error. It can also handle finer grained errors, retiring faulty bits or pins as they accumulate.

Bamboo ECC-based graceful downgrade enjoys superior flexibility, and can be modified to work on a narrower channel. For a system with a 64-bit data channel, QPC (using 8 redundant pins) can be gracefully downgraded to SPC-TPD (with a 4 pin redundancy), correcting a sequence of 1 chip error and 1 pin error or up to 5 sequential pin errors while detecting virtually all end-of-life errors.

4.3. Overheads

The large ECC symbols and codewords of Bamboo ECC provide strong and efficient error protection at the expense of decoding complexity and latency. This section examines these costs, showing that the additional implementation overheads of Bamboo ECC codes are modest and well-aligned to current technological trends. We demonstrate later (in Section 5.3) that Bamboo ECC codes incur little performance overhead, and we expect that their memory bandwidth savings (for the same level of protection) will outweigh their costs.

RS codes with 8-bit symbols have a wide range of commercial applications, ranging from satellites to CDs, due to their simple encoding and decoding schemes. AMD chipkill uses an 18-symbol RS code with 8-bit symbols (as described in Section 3.4); a corresponding fully parallel encoder requires 6 XOR2 gates of delay and consume about 1,600 NAND2 gates' worth of chip area. With larger codewords, Bamboo ECC codes have larger encoding/decoding overheads. Specifically, a fully parallel encoder for QPC (72-symbol) is 8 XOR2 gates deep, which is 2 gates more than AMD chipkill and is still easily implementable within a single memory cycle. The QPC encoder consumes about 25,000 NAND2 gates of area, a $\times$16 increase over that of AMD chipkill. This does not represent a large amount of chip real estate considering the billions of gates available in recent processors.

Reed-Solomon decoding consists of two phases. Error detection, which happens on every memory read, is as simple as encoding and its additional Bamboo ECC overhead is acceptable (as described above). Error correction, which happens only in the rare case of a memory error, can be done by software or by sequential hardware to save area through circuit reuse. In the case of transient errors, this recovery cost will be negligible relative to the overall cost of the system, but it may become prohibitive for permanent errors. In this case, a retirement scheme should

These delay and area estimates are found through standard-cell synthesis using the Synopsys toolchain and the the 40nm TSMC standard cell library [40, 41], but are presented in a technology-independent manner

be used to mask the error and avoid the need for frequent RS correction. Such an approach (relying on a retirement scheme for permanent errors) seems consistent with the direction that industry has been moving, as is evidenced by Intel's DDDC/DDDC+1 scheme. It has also been employed by other recent academic ECC papers that have expensive correction procedures [39]. The flexible nature of Bamboo ECC codes is well suited to RBS-based retirement, as is demonstrated in Section 5.2.

Bamboo ECC codes may introduce some additional latency in the memory controller due to the alignment of their pin-based symbols. Until the completion of a transfer, per-pin symbols are not fully available, which may delay encoding and decoding. While encoding is on the less latency-critical write path (and data are usually buffered before writes), decoding is on the latency-critical read path. One way to waive this additional latency is to use an asynchronous ECC check. With asynchronous ECC checking, data are speculatively forwarded to the processing unit before decoding is complete. Later, if an error is detected, the forwarded data and any dependent calculations are discarded and corrected data are sent throughout the system. Section 5.3 evaluates the performance cost of Bamboo ECC, finding the impact to be minor. To be conservative in our evaluation, we do not apply an asynchronous ECC check—data are forwarded only after they are determined to be error-free.

5. Evaluation

We measure the error coverage, system failure rate, and performance impact of Bamboo ECC schemes and compare them with the state-of-the-art single-tier ECC below. The range and flexibility of Bamboo ECC leads to a design space of error control schemes that vary in their error coverage, redundancy, and expected system lifetime. We examine this design space below, demonstrating the substantive strength, safety, and flexibility advantages of Bamboo ECC.

5.1. Error Coverage Evaluation

The error coverage of each ECC scheme is evaluated using Monte Carlo error injection experiments. Errors based on 5 fault models (bit/pin/word/chip/rank faults) are generated and injected into a cache line. These models represent faults in different memory structures (e.g. cells, subarrays, chips and ranks). A bit-fault indicates that the cache line has a single bit-error at a random position, and a pin-fault represents a cache line that has a single corrupted DQ pin. Similarly, a word/chip/rank fault corrupts all bits within a single-chip/single-beat, a single-chip/all-beats and all-chips/all-beats, respectively. It is assumed that each bit within a corrupted region has a 50% switching probability (but the error-free pattern is excluded from evaluation). To model Wide-IO memory, we introduce additional TSV, *data strobe* (*DQS*), and subarray fault models. We assume that a DQS captures 16 DQs [5] and that a subarray is responsible for 4 DQs (the 4Kb row buffer configuration in [16]). A DQS fault corrupts a 16-bit group within a single beat, while a subarray fault corrupts 4-bits $\times$4-bursts of data.

		SEC-DED	Bamboo SPC-TPD	AMD chipkill	Bamboo QPC
Codeword (bits x beats)		72 x 1	68 x 8	72 x 2	72 x 8
CWs per 64B		8	1	4	1
Redundancy %		12.5%	6.25%	12.5%	12.5%
1 bit/pin (%)	DCE	100.0000	100.0000	100.0000	100.0000
1 word fault (%)	DCE	26.6770	26.6711	100.0000	100.0000
	DUE	55.5496	73.3289	0.0000	0.0000
	SDC	17.7733	0.0000	0.0000	0.0000
1 chip fault (%)	DCE	0.0142	0.0000	100.0000	100.0000
	DUE	98.8388	99.9996	0.0000	0.0000
	SDC	1.1470	0.0004	0.0000	0.0000
1 bit fault + 1 bit fault (%)	DCE	87.4929	0.0000	0.0000	100.0000
	DUE	12.5071	100.0000	98.9267	0.0000
	SDC	0.0000	0.0000	1.0733	0.0000
1 bit fault + 1 pin fault (%)	DCE	49.7906	0.0000	0.0000	100.0000
	DUE	50.2094	100.0000	99.9409	0.0000
	SDC	0.0000	0.0000	0.0591	0.0000
1 bit fault + 1 word fault (%)	DCE	23.3320	0.0000	0.0000	26.6716
	DUE	57.4549	100.0000	98.6451	73.3284
	SDC	19.2131	0.0000	1.3549	0.0000
1 bit fault + 1 chip fault (%)	DCE	0.0030	0.0000	0.0000	0.0000
	DUE	99.3183	99.9996	100.0000	100.0000
	SDC	0.6787	0.0004	0.0000	0.0000
1 pin fault + 1 word fault (%)	DCE	13.2878	0.0000	0.0000	26.6647
	DUE	63.1743	100.0000	99.9322	73.3353
	SDC	23.5380	0.0000	0.0678	0.0000
1 chip fault + 1 chip fault (%)	DCE	0.0000	0.0000	0.0000	0.0000
	DUE	99.9539	99.9996	100.0000	100.0000
	SDC	0.0461	0.0004	0.0000	0.0000
1 rank fault (%)	DCE	0.0000	0.0000	0.0000	0.0000
	DUE	99.9957	99.9996	100.0000	100.0000
	SDC	0.0043	0.0004	0.0000	0.0000

Table 1: A comparison of ECC error correction and detection coverage (using DDR3 with a burst length of 8).

Once errors are generated and injected based on a fault scenario (i.e. how many and what kinds of faults) we apply each ECC scheme to determine whether the error results in a DCE, DUE or SDC. If any word within a cache line reports a DUE, the cache line is reported as DUE. Similarly, if any word in a non-DUE line reports an SDC, the cache line is classified as SDC. Finally, if all the words are corrected to their original data then the cache line is marked as DCE.

We evaluate bit-level protection schemes (SEC-DED and SPC-TPD) and single chipkill-correct schemes (AMD chipkill and QPC) on a DDR 64-bit data channel (Table 1). On a Wide-IO 128-bit data channel, single chipkill-correct schemes (AMD chipkill and DDPC) and double chipkill-correct schemes (doubled AMD chipkill and QDPC) are evaluated to protect 4-bit subarrays (Table 2). Doubled AMD chipkill uses the same ECC layout as AMD chipkill with doubled word size, redundancy, and correction capabilities. The number of experimental runs is 2^{24}, with corresponding 99.9% confidence intervals of $\pm$0.0001% for probabilities near 0.0001%/99.9999% and $\pm$0.008% for probabilities near 1%/99%.

Table 1 compares the error coverage and redundancy of the

		D-AMD	Bamboo QDPC	AMD Chipkill	Bamboo DDPC
Codeword (bits x beats)		144 x 2	144 x 4	136 x 2	136 x 4
CWs per 64B		2	1	2	1
Redundancy %		12.5%	12.5%	6.25%	6.25%
1 bit/TSV/SA (%)	DCE	100.0000	100.0000	100.0000	100.0000
1 bit fault + 1 bit fault (%)	DCE	100.0000	100.0000	0.0000	0.0000
	DUE	0.0000	0.0000	94.1879	100.0000
	SDC	0.0000	0.0000	5.8121	0.0000
1 bit fault + 1 TSV fault (%)	DCE	100.0000	100.0000	0.0000	0.0000
	DUE	0.0000	0.0000	97.5613	100.0000
	SDC	0.0000	0.0000	2.4387	0.0000
1 bit fault + 1 SA fault (%)	DCE	100.0000	100.0000	0.0000	0.0000
	DUE	0.0000	0.0000	99.9519	99.9519
	SDC	0.0000	0.0000	0.0481	0.0481
1 TSV fault + 1 TSV fault (%)	DCE	100.0000	100.0000	0.0000	0.0000
	DUE	0.0000	0.0000	98.4214	100.0000
	SDC	0.0000	0.0000	1.5786	0.0000
1 TSV fault + 1 SA fault (%)	DCE	100.0000	100.0000	0.0000	0.0000
	DUE	0.0000	0.0000	99.9512	99.9515
	SDC	0.0000	0.0000	0.0488	0.0485
1 DQS fault (%)	DCE	2.1597	3.4627	1.3888	1.3876
	DUE	96.9478	96.5373	85.5720	98.5727
	SDC	0.8925	0.0000	13.0249	0.0397
1 DQS fault + 1 bit fault (%)	DCE	0.0923	0.4206	0.0000	0.0000
	DUE	99.4099	99.5793	93.2116	99.9491
	SDC	0.4978	0.0000	6.7804	0.0509
3 ×1 TSV fault	DCE	0.0000	100.0000	0.0000	0.0000
	DUE	99.9800	0.0000	99.5291	99.9523
	SDC	0.0200	0.0000	0.4709	0.0477
1 rank fault (%)	DCE	0.0000	0.0000	0.0000	0.0000
	DUE	100.0000	100.0000	99.9484	99.9482
	SDC	0.0000	0.0000	0.0516	0.0518

Table 2: A comparison of ECC scheme error correction and detection coverage (Wide-IO with a 128-bit data interface and a burst length of 4). 16 DQs share a data strobe (DQS) signal.

ECC schemes on the DDR 64-bit data channel. SEC-DED and SPC-TPD can all correct a bit or a pin fault. The single pin correction capability of SPC and SPC-TPD shows worse correction in some fault scenarios as it cannot correct multiple 1-bit-error-per-beat errors with different DQ positions over multiple beats (a situation that is expected to be rare). However, SPC-TPD requires only half the redundant storage of SEC-DED and its detection capability is very strong, detecting virtually 100% (99.9996%) of all errors in all scenarios, compared to the lacking error detection capabilities of SEC-DED (up to 23.5% SDC).

Among single chipkill-correct schemes, QPC has a better correction capability than AMD chipkill because of its ability to correct two independent pin errors (e.g. 1 bit fault + 1 bit fault). In addition, QPC has a very strong detection capability of 100.0000% in all scenarios, compared to the strong-yet-incomplete detection capability of AMD chipkill (up to 1.4% SDC in some scenarios). While the detection coverage of QPC is not a perfect 100% (a few pathological error patterns can result in SDC), it applies not only to two-chip faults but also to many-chip faults that represent very severe errors.

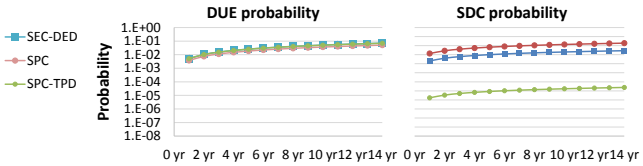


Figure 6: The failure probability of a DDR 64-bit data channel (2 ranks, 18 x4 chips per rank) over time with bit-level protection.

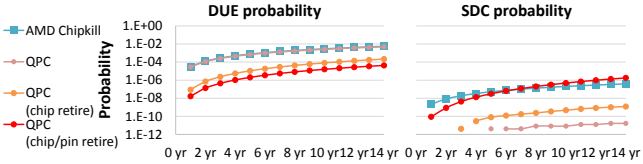


Figure 7: The failure probability of a DDR 64-bit data channel (2 ranks, 18 x4 chips per rank) over time with chipkill-correct protection.

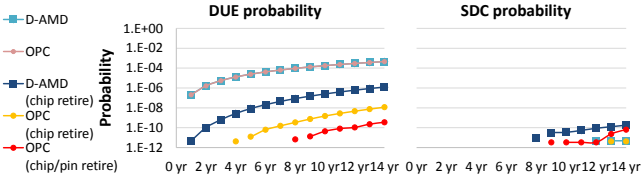


Figure 8: The failure probability of a DDR 128-bit data channel (2 ranks, 36 x4 chips per rank) with double-chipkill-correct protection.

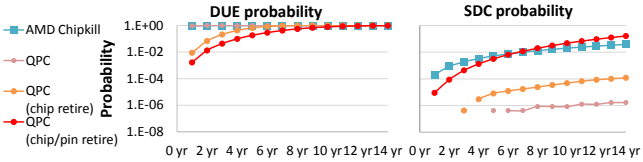


Figure 9: System-level failure probability of 100,000 DDR 64-bit data channels over time with chipkill-correct protection.

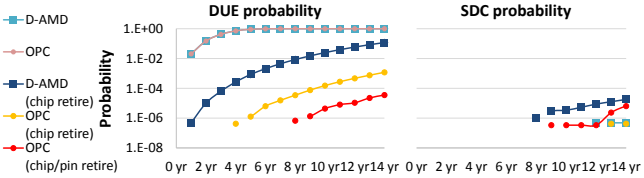


Figure 10: System-level failure probability of 100,000 DDR 128-bit data channels over time with double-chipkill-correct protection.

On a Wide-IO 128-bit data channel (Table 2), both doubled AMD chipkill and QDPC can correct any concurrent faults on up to 2 sub-arrays. QDPC correction capability is slightly better as it can correct concurrent 3 double-pin errors. The detection capability of QDPC is 100.0000% in all scenarios (even stronger than that of QPC due to its bigger word size) while doubled AMD chipkill misses up to 0.89% errors in some scenarios.

5.2. System Failure Rate Evaluation

The error coverage results from Section 5.1 show that Bamboo ECC provides superior error correction and vastly improves error detection relative to prior single-tier chipkill-correct solutions. The true mettle of an error control scheme,

however, is tested by how much it improves the failure rate of a large system at-scale.

We use a two-stage Monte Carlo simulation to evaluate the failure rates of different ECC codes and error control schemes over time. The first stage injects faults into a simulated DDR3 memory channel based on observed fault modes and rates [2]. This first-stage fault injection methodology is similar to that of FaultSim [28], but differs in two ways to more fully evaluate Bamboo ECC. First, we optimize and parallelize the simulation in order to run 200 billion simulations per configuration; this large number is needed to actually observe any failures for the strongest Bamboo ECC configurations. Second, we model the accumulation of up to four concurrent faults per memory location. Once overlapping faults are identified, the second stage of Monte Carlo simulation maps the fault modes into one of the fault models described in Section 5.1 based on number of failing DQs, generates error patterns based on the footprint of each fault model and passes each error through ECC decoding in order to judge its outcome. This is to realistically model the practical error coverage of each ECC code, as described in Section 2.3.2.

Due to their size and importance, large systems need a comprehensive system-level error control scheme; as part of this system-level control scheme, many machines have high-level state preservation and restoration facilities (e.g., checkpoint/restart). Accordingly, the analyses for RBS retirement assume some higher-level data recovery mechanisms—a DUE is recovered if there is still hot-swappable redundant space left, and a simulation is only terminated after all redundant storage is exhausted or after an SDC.

Figure 6 shows the failure rate of a single 64b data channel over time using different bit-level ECC mechanisms. SPC-TPD has a slightly lower DUE+SDC probability and a $10,000\times$ lower SDC probability than SEC-DED, despite requiring only half as much redundancy. The lower overall failure rate of SPC-TPD is due to this lower redundancy, as the correspondingly lower raw fault rate is able to compensate for the slightly weaker correction capability of the code. Using only a quarter redundancy, SPC shows an even more substantive (DUE+SDC) improvement over SEC-DED as it has an even lower raw fault rate than SPC-TPD. However, the weaker error detection capabilities of SPC results in a $10\times$ higher SDC probability than SEC-DED.

The failure rate of a 64b data channel using chipkill-correct protection is shown in Figure 7. QPC has 12% lower overall failure probability than AMD, as it can correct two independent single-DQ faults. In addition, the strong error detection coverage of QPC results in a $1000\times$ lower SDC probability. Graceful downgrade using QPC can correct a sequence of one chip and one pin (QPC to SPC-TPD with a chip retirement) or a sequence of one chip and up to three pins (QPC to SPC-TPD to SPC with chip/pin retirement). The stronger correction of downgrade-based schemes lowers the overall failure probability of chip and pin-based retirement to 0.04 and 0.008 that of AMD, respectively. The SDC probability increases with downgrade schemes, however, as they continue operation with reduced-strength codes; despite this, downgraded QPC still demonstrates SDC rates that are comparable to, or better than that of AMD.

Figure 8 demonstrates the failure rate of a DDR 128b data channel using different double-chipkill-correct schemes. OPC has a 2% lower failure probability than doubled AMD chipkill due to its ability to correct independent single-DQ faults. We were not able to observe any SDC occurrence with OPC during the 200B runs, while doubled AMD chipkill shows a 10^{-10} probability of SDC. Doubled AMD chipkill can correct a sequence of up to 3 faults by gracefully degrading through chip retirement to AMD chipkill. The superior flexibility of OPC can correct a sequence of up to 3 chip faults and 1 pin fault by downgrading down to SPC-TPD with chip retirement or a sequence of up to 3 chip and 3 pin faults by downgrading down to SPC with pin retirement.

5.2.1. Full-system estimates and lifetime considerations Figures 9 and 10 illustrate the failure probabilities of a large system with 100,000 memory channels(a comparable number of channels to the Jaguar system analyzed by Sridharan and Liberty [2]). Several findings are readily apparent. First, the strength and safety of QPC and OPC relative to AMD (double) chipkill is demonstrated through greatly reduced SDC rates. Also, it can be seen that without some higher-level repair mechanism, the memory system will quickly reach a state where some locations report uncorrectable errors and must be repaired or replaced. For remote systems or systems where the replacement of failing memory is expensive, it is desirable to extend this time to repair as long as possible.

In addition to reducing the correction rate for permanent faults (as discussed in Section 4.3), RBS-based graceful degradation can be used to combat this rapid wearout of the memory system. For instance, by employing chip retirement, AMD can decrease the probability of requiring a system repair at 10 years by $40\times$ relative to OPC. Because of its flexibility, Bamboo ECC is able to extend the lifetime by a much greater amount, decreasing the chance of repair by roughly $6,500\times$ using chip retirement. Using fine-grained retirement at a pin granularity, the probability of irreparable failures within 10 years is negligible, measuring $230,000\times$ less than that of OPC alone.

There is a natural tradeoff between lifetime extension and guaranteed safety—an aggressive lifetime-extending scheme, such as Bamboo ECC with per-pin retirement, will spend the majority of its time in a degraded mode by design; as such, its safety may suffer compared a code that conservatively reports a DUE. Through per-chip and per-pin retirement, Bamboo ECC codes present a range of options that trade off safety and lifetime. Importantly, even the most aggressively degrading Bamboo ECC schemes maintain SDC rates that are comparable with those of AMD (double) chipkill, such that even a

Failure rates for systems with Wide I/O memory is not attempted due to a lack of empirical information on fault modes and rates.

Out of necessity, this experimental methodology is based on observed errors by ECC mechanisms in-the-field; in reality, other error modes (and undiagnosed errors) may exist.

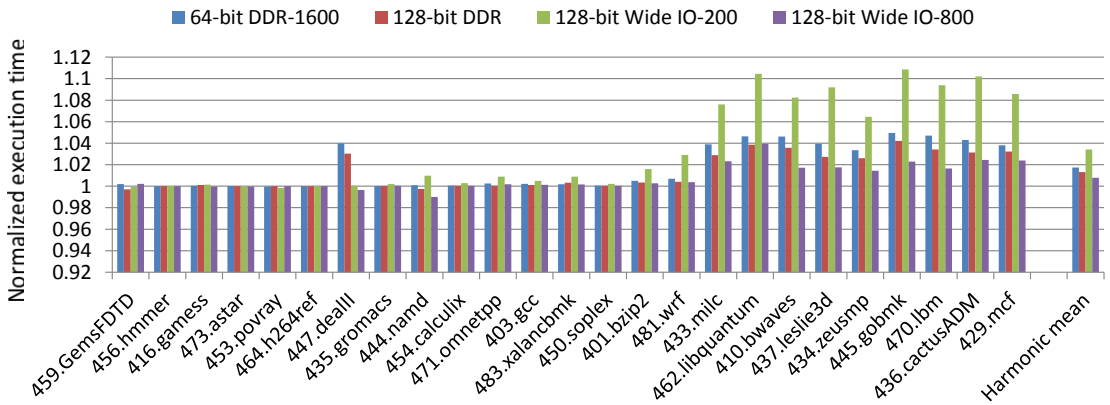


Figure 11: The execution time of Bamboo ECC schemes (normalized to AMD chipkill). Benchmarks are sorted by memory traffic.

long-lifetime Bamboo ECC organization will not compromise system safety relative to the current state-of-the-art.

5.3. Performance Evaluation

We model the performance cost of Bamboo ECC using cycle-based simulation of the SPECcpu 2006 benchmark suite [22] on the Gem5 simulator [42] (version 2.0). A 2GHz single-core processor with a 32KB I-cache, 64KB D-cache, and a 2MB L2 cache is used with 2GB of DRAM and an L2 cache stride prefetcher. The latency overhead of ECC schemes is modeled in the read data queue and write request queue of the memory controller. Existing ECC schemes are given a +1 (memory) cycle penalty for both reads and writes, while Bamboo ECC schemes have a +4 cycle read and a +1 cycle write penalty. This 3 additional cycle increase models the waiting time for all beats of each symbol to be transferred. There are 4 memory configurations: DDR3-1600 (800MHz DDR) 64b data channel, DDR3-1600 (800MHz DDR) 128b data channel, Wide-IO-200 (200MHz SDR) 128b data channel and a hypothetical Wide-IO-800 (400MHz DDR) 128b data channel. For the DDR3-1600 128b channel, we increase the cache line size from 64B to 128B to match the DRAM access granularity. The measurement period is 0.2 billion instructions using detailed simulation after 0.5 billion instructions of functional simulation (to warm up the caches). We run 25 of the 29 SPEC-cpu2006 benchmarks; the remaining 4 benchmarks (perlbench, sjeng, tonto and sphinx3) fail to run due to simulator issues.

Figure 11 shows the execution time comparison between existing ECC schemes and Bamboo ECC. The benchmarks are sorted based on their memory traffic. Bamboo ECC shows 1.7%, 1.3%, 3.4% and 0.8% execution cycle increases compared to AMD chipkill on DDR3-1600 64b, DDR3-1600 128b, Wide-IO-200 128b and Wide-IO-800 128b channels, respectively. The larger increase in Wide-IO-200 comes from its longer transfer time. Wide-IO-200 has a longer burst transfer time for an access (20ns) than DDR3-1600 (5ns), which increases the waiting time of Bamboo ECC, hurting performance. However, with the increasing trend of Wide-IO bandwidth and clock frequencies, this waiting time should decrease. The 5ns burst transfer time of our hypothetical Wide-IO-800 decreases this performance overhead down to 0.8%. Some benchmarks report execution time reductions with Bamboo ECC, which are likely to be experimental noise from irregular benchmarks as the numbers are small. The execution cycle increases in Bamboo ECC will be much smaller than the performance degradation from increasing the data width from 64b to 128b for stronger ECC protection, which is reported to be very performance limiting by Fujitsu [24] (see our discussion in Section 2.1).

5.3.1. Energy Consumption We estimate DRAM energy consumption based on a Micron model [43] with Samsung DDR3-1600 parameters [44]. The average difference between the existing ECC and Bamboo ECC schemes are 1.0% and 0.8% on DDR3-800 64b and DDR-800 128b channels, respectively. This increased energy consumption is mostly due to the increased execution times, as the number of activations and data transfers are almost the same. As power information for Wide-IO memory is not publicly available, we use LPDDR2-800 parameters from Micron [45] to estimate the energy consumption of Wide-IO. Bamboo ECC shows only a 0.6% average increase in energy on Wide-IO-200 with LPDDR2-800 parameters, as the small background power of LPDDR2 (and possibly Wide-IO memory) makes the longer execution time less costly.

6. Conclusion

This paper presents and analyzes a family of strong error checking and correcting mechanisms for DRAM called Bamboo ECC. Bamboo ECC codes provide superior efficiency, operating as single-tier mechanisms that offer up to chipkill-level protection over a 64b/128b data DRAM channel. Meanwhile, Bamboo ECC codes are strong and safe, delivering increased correction capabilities relative to the state-of-the-art single-tier DRAM schemes while simultaneously decreasing the silent data corruption rate. We show that Bamboo ECC codes are amenable to graceful downgrade using redundant bit steering and are able to offer an unprecedented level of accumulating error protection. Bamboo ECC codes with RBS also demonstrate superior flexibility, and are able to retire finer grained errors and operate on narrower channels than the current state-of-the-art ECC mechanisms, potentially extending the memory system lifetime.

Acknowledgements

The authors acknowledge the Texas Advanced Computing Center for providing HPC resources and the support of the Depart-

ment of Energy under Award #B599861 and the National Science Foundation under Grant #0954107, which partially funded this research.

References

[1] 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 *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 2010.

[2] V. Sridharan and D. Liberty, "A Study of DRAM Failures in the Field," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, 2012.

[3] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi, "Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM Faults," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, November 2013.

[4] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM Errors in the Wild: a Large-Scale Field Study," in *Proceedings of the International Joint Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, 2009.

[5] *Wide I/O Single Data Rate (Wide I/O SDR), JESD229*, Joint Electron Device Engineering Council, Dec. 2011.

[6] *Hybrid Memory Cube Specification 1.0*, Hybrid Memory Cube Consortium, 2013.

[7] Hynix Semiconductor Inc., "Blazing A Trail to High Performance Graphics," http://sites.amd.com/us/Documents/TFE2011_006HYN.pdf, 2011.

[8] Advanced Micro Devices (AMD), Inc., "BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 00h-0Fh Processors," Jan 2013.

[9] Intel Corp., "Intel Xeon Processor E7 Family: Reliability, Availability, and Serviceability," 2011.

[10] Hewlett-Packard, "How memory RAS technologies can enhance the uptime of HP ProLiant servers," 2013.

[11] International Business Machines Corp. (IBM), "Chipkill Memory," http://www-05.ibm.com/hu/termekismertetok/xseries/dn/chipkill.pdf, Tech. Rep., 2012.

[12] J. Huh, D. Burger, and S. W. Keckler, "Exploring the design space of future CMPs," in *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques (PACT)*. IEEE, 2001, pp. 199–210.

[13] A. Udipi, N. Muralimanohar, N. Chatterjee, R. Balasubramonian, A. Davis, and N. Jouppi, "Rethinking DRAM Design and Organization For Energy-Constrained Multi-Cores," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 2010.

[14] Y. Kim, V. Seshadri, D. Lee, J. Liu, and O. Mutlu, "A case for exploiting subarray-level parallelism (salp) in dram," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 2012.

[15] *DDR4 SDRAM STANDARD, JESD79-4*, Joint Electron Device Engineering Council, Sep. 2012.

[16] B. Giridhar, M. Cieslak, D. Duggal, R. Dreslinski, H. M. Chen, R. Patti, B. Hold, C. Chakrabarti, T. Mudge, and D. Blaauw, "Exploring DRAM Organizations for Energy-efficient and Resilient Exascale Memories," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, 2013, pp. 23:1–23:12.

[17] X. Li, M. C. Huang, K. Shen, and L. Chu, "A Realistic Evaluation of Memory Hardware Errors and Software System Susceptibility," in *Proceedings of the USENIX Annual Technical Conference (USENIX)*, 2010.

[18] 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 *Proceedings of the International Symposium on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2012.

[19] *Graphics Double Data Rate (GDDR5) SGRAM Standard, JESD212B.01*, Joint Electron Device Engineering Council, Dec. 2013.

[20] *Low Power Double Data Rate 3 (LPDDR3), JESD209-3B*, Joint Electron Device Engineering Council, Aug. 2013.

[21] *High Bandwidth Memory (HBM) DRAM, JESD235*, Joint Electron Device Engineering Council, Oct. 2013.

[22] Standard Performance Evaluation Corporation, "SPEC CPU 2006," 2006. [Online]. Available: http://www.spec.org/cpu2006

[23] High Productivity Computing Systems (HPCS), "HPCchallenge benchmarks." [Online]. Available: http://icl.cs.utk.edu/hpcc/hpcc_results.cgi

[24] Fujitsu Technology Solutions, 2014. [Online]. Available: http://globalsp.ts.fujitsu.com/dmsp/Publications/public/wp-ivy-bridge-ex-memory-performance-ww-en.pdf

[25] I. T. C. on Reali-Time Systems", "Terminology and Notations," http://tcrts.org/education/terminology-and-notation/, 2014.

[26] J. Chung, I. Lee, M. Sullivan, J. H. Ryoo, D. W. Kim, D. H. Yoon, L. Kaplan, and M. Erez, "Containment Domains: A Scalable, Efficient, and Flexible Resilience Scheme for Exascale Systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, 2012, pp. 58:1–58:11.

[27] X. Jian, N. DeBardeleben, S. Blanchard, V. Sridharan, and R. Kumar, "Analyzing reliability of memory sub-systems with double-chipkill detect/correct," in *Proceedings of IEEE Pacific Rim International Symposium on Dependable Computing*. IEEE, 2013, pp. 88–97.

[28] D. Roberts and P. Nair, "FAULTSIM: A fast, configurable memory-resilience simulator," in *The Memory Forum: In conjunction with ISCA*, vol. 41.

[29] Oracle Corp., "Oracle SPARC Server RAS Comparison." [Online]. Available: http://www.oracle.com/us/products/servers-storage/servers/sparc-enterprise/sparc-ras-comparison-190946.pdf

[30] T. J. Dell, "A white paper on the benefits of chipkill-correct ECC for PC server main memory," *IBM Microelectronics Division*, pp. 1–23, 1997.

[31] I. Reed and G.Solomon, "Polynomial codes over certain finite fields," in *J. Soc. for Industrial and Applied Math*, 1960, pp. 300–304.

[32] Sun Microsystems, Inc., "T2 core microarchitecture specification." [Online]. Available: http://www.oracle.com/technetwork/systems/opensparc/t2-06-opensparct2-core-microarch-1537749.html

[33] Advanced Micro Devices (AMD), Inc., "Kernel developer's guide for AMD NPT family 0Fh processors," 2007. [Online]. Available: http://developer.amd.com/wordpress/media/2012/10/325591.pdf

[34] Hewlett-Packard, "HP Advanced Memory Error Detection Technology," 2011.

[35] M. T. Chapman, "Introducing IBM Enterprise X-Architecture Technology," Tech. Rep., 2001.

[36] Hewlett-Packard, "Servers and Storage Technology for the Adaptive Infrastructure," 2006.

[37] D. H. Yoon and M. Erez, "Virtualized and Flexible ECC for Main Memory," in *Proceedings of the International Symposium on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2010.

[38] A. N. Udipi, N. Muralimanohar, R. Balasubramonian, A. Davis, and N. P. Jouppi, "LOT-ECC: Localized and tiered reliability mechanisms for commodity memory systems," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 2012, pp. 285–296.

[39] X. Jian, H. Duwe, J. Sartori, V. Sridharan, and R. Kumar, "Low-power, low-storage-overhead chipkill correct via multi-line error correction," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*. ACM, 2013, p. 24.

[40] Synopsys Inc., "Design Compiler I-2013.12-SP5-2," September 2014.

[41] Taiwan Semiconductor Manufacturing Company, "40nm CMOS Standard Cell Library v120b," 2009.

[42] "The gem5 Simulator System: A Modular Platform for Computer System Architecture Research," http://www.gem5.org.

[43] Micron Technology Co., "Calculating Memory System Power for DDR3," http://www.micron.com/-/media/Documents/Products/Technical%20Note/DRAM/TN41_01DDR3_Power.pdf, 2007.

[44] Samsung Electronics Co., "2Gb D-die DDR3 SDRAM," 2011.

[45] Micron Technology Co., "Mobile LPDDR2 SDRAM," http://www.micron.com, 2010.