All-Inclusive ECC: Thorough End-to-End Protection for Reliable Computer Memory

Jungrae Kim* Michael Sullivan** Sangkug Lym* Mattan Erez*

*The University of Texas at Austin
{dale40, sklym, mattan.erez}@utexas.edu

**NVIDIA
misullivan@nvidia.com

***Abstract*—Increasing transfer rates and decreasing I/O voltage levels make signals more vulnerable to transmission errors. While the data in computer memory are well-protected by modern error checking and correcting (ECC) codes, the clock, control, command, and address (CCCA) signals are weakly protected or even unprotected such that transmission errors leave serious gaps in data-only protection. This paper presents All-Inclusive ECC (AIECC), a memory protection scheme that leverages and augments data ECC to also thoroughly protect CCCA signals. AIECC provides strong end-to-end protection of memory, detecting nearly 100% of CCCA errors and also preventing transmission errors from causing latent memory data corruption. AIECC provides these system-level benefits without requiring extra storage and transfer overheads and without degrading the effective level of data protection.**

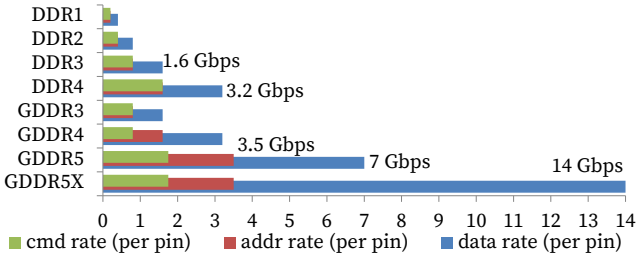
I. Introduction

Any thorough system-level protection scheme must be holistic and provide end-to-end error protection. Strong protection of any one component provides limited benefit to the overall reliability, as any unprotected component will quickly become the reliability bottleneck. Despite this, the overwhelming majority of current DRAM error protection literature is devoted towards protecting the data that are stored in memory. *Clock, control, command, and address (CCCA)* signals are left unprotected or are protected seemingly as an afterthought by ad-hoc mechanisms.

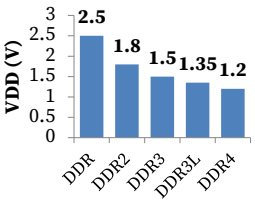
This paper presents *All-Inclusive ECC (AIECC)*, a holistic memory error protection scheme that is able to safeguard DRAM data and CCCA signals against storage and transmission errors. By leveraging the existing strong ECC schemes for the protection of CCCA errors, AIECC remedies the approach of protecting only or mostly DRAM data without discarding the extensive advancements made in the area of data protection.

With demands for both memory capacity and bandwidth constantly increasing, DRAM design evolves to provide more data bandwidth with less energy. Each recent generation of DRAM has more-than-doubled the data transfer rate of its predecessor (Figure 1a) and each generation has also decreased core and I/O voltages for better energy efficiency (Figure 1b). Increasing signal transfer rates and lowering I/O voltages each exacerbate the problem of *transmission errors*. A transmission error occurs when a signal is incorrectly transferred to or from memory; higher transfer rates and lower voltages increase the vulnerability to timing and electrical noise, respectively.

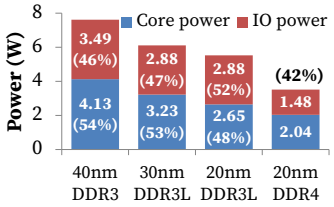
Traditionally, DRAM designs have managed the transmission error rate using sophisticated circuit techniques to improve



(a) The increasing maximum DRAM transfer rates of command, address, and data signals.



(b) The decreasing DRAM supply voltage levels.



(c) DRAM power breakdown over core and I/O [1].

Fig. 1: General DRAM trends over the past 15 years. Transmission rates continually increase each generation while supply voltages drop. Meanwhile, roughly half of the DRAM power is consumed during transmission.

signal integrity (e.g., delay-locked loops [2], [3], phase-locked loops [4], [5], on-die termination [2], [3], [4], [5], [6], differential signaling [2], [3], [4], [5], [6], and fly-by topologies [2], [3]) at the cost of burning extra power. The price paid for reliable transmission is significant, and roughly half of DRAM power is spent on I/O (Figure 1c). Rising transmission error rates and tight power constraints have made circuit techniques alone insufficient to provide high levels of reliability and efficiency. DRAM vendors have accordingly introduced ad-hoc architectural techniques to recent memory generations—such as Cyclic Redundancy Checks (CRCs)—to verify the consistency of received data and relax the circuit-level demands by tolerating some transmission errors [3], [4], [5].

While current protection techniques primarily focus on protecting data transmission, CCCA signals can be more vulnerable to transmission errors. These signals typically operate at half the transfer rates of data signals, yet they may suffer from more transmission errors due to their parallel interface (e.g. 27

1063-6897/16 $31.00 © 2016 IEEE
DOI 10.1109/ISCA.2016.60

DDR4 control/command/address signals share a clock, while 8 data signals and 1 mask signal share a data strobe) and more receivers (e.g. CCCA signals drive many chips, while data signals drive a single pin on one rank at a time). DDR4 DIMM signal and power integrity simulations show that CCCA signals have narrower time windows to reliably receive signals than data signals, limiting the overall operating frequency. Due to the lack of strong CCCA protection, current DRAMs passively limit CCCA rates for high reliability. DDR4 introduced gear-down mode which halves CCCA transfer rates for reliability [3], and GDDR5X could not scale up CCCA rates along with data rates [5] (Figure 1a). These designs trade performance for CCCA reliability by lowering command bandwidth, adding command latency, and increasing access granularity.

We present a novel and thorough protection technique for both data and CCCA errors in main memory. *All-Inclusive ECC (AIECC)* augments existing data-protection schemes to provide high coverage against CCCA errors without additional redundant storage or new signals to and from memory. AIECC is meant to be an unobtrusive addition to future memory standards. As an example, we describe AIECC as an extension to DDR4 memory, which is the most dominant main memory system design today for high-capacity servers and HPC systems. Specifically, AIECC makes the following contributions. It:

- Is the first known publication to simultaneously consider DRAM data, address, command, control, and clock protection.
- Detects nearly 100% of CCCA errors without introducing additional storage or transfer redundancy.
- Provides detailed diagnosis of CCCA errors to ease repair.
- Dovetails existing DRAM techniques (command and address parity / write CRC) to ease adoption and leverage prior effort.

In addition to the novelty and contributions of the AIECC scheme, we also provide a comprehensive literature and patent survey. While some prior art is conceptually similar, AIECC differs in important ways. We sweep a range of plausible CCCA error rates and demonstrate that ours is the first mechanism to provide complete CCCA and data error protection for DDR4 memory.

The paper proceeds as follows. We first lay out the conceptual foundations of AIECC and describe the current state-of-the-art in CCCA protection efforts in Sections II and III. Section IV describes how AIECC simultaneously protects against data and CCCA errors. Section V evaluates the reliability benefits of AIECC and shows its system-level overheads to be minimal. Finally, Section VI describes exciting future research avenues and Section VII concludes the paper.

II. Background

This section summarizes the background necessary for a complete understanding of AIECC in the context of high-capacity and high-reliability systems using DDR4 memory. Sections II-A, II-B, and II-C review DRAM memory, ECC, and CCCA errors, respectively.

Pin #	27	26	25	24	23
Signal	CK	CKE	CS	ODT	PAR
Group	**CK**	**CTRL**			**CMD/ADD**
Pin #	22	21	20	19	18…17
Signal	ACT	RAS/A16	CAS/A15	WE/A14	BG1…0
Group	**CMD/ADD**				
Pin #	16…15	14	13…11	10	9…0
Signal	BA1…0	A12/BC	A17,13,11	A10/AP	A9…0
Group	**CMD/ADD**				

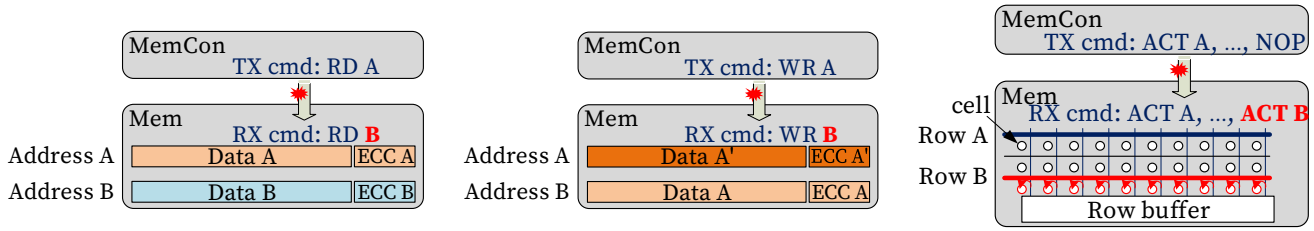
Fig. 2: The CCCA signal interface for DDR4 memory. $\overline{\text{CMD}}$ and $\overline{\text{ADD}}$ share pins by having a different context per command. $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ denote the row and column address strobes. $\overline{\text{CK}}$, $\overline{\text{CKE}}$, $\overline{\text{CS}}$, $\overline{\text{ODT}}$, $\overline{\text{PAR}}$, $\overline{\text{ACT}}$, $\overline{\text{WE}}$, $\overline{\text{BC}}$ and $\overline{\text{AP}}$ stand for clock, clock-enable, chip-select, on-die-termination, command-parity, activate, write-enable, burst-chop and auto-precharge, respectively, as described in the DDR4 spec [3].

A. DRAM and CCCA Signals

DRAM memory is organized as two-dimensional arrays (banks) and requires three commands to access a fresh piece of data. An activation command (ACT) fetches a row of data into an internal row buffer. Then, a read or write command (RD or WR) uses a column address to select and transfer or overwrite a particular block from the activated row. Once a memory transfer or write completes, the DRAM bank must be restored to a ready state by issuing a precharge command (PRE). To exploit data locality and amortize command overheads, an access to DDR4 initiates a burst of 8 data transfer beats. An access to an already-activated row (a *row buffer hit*) does not require an ACT or PRE command, potentially saving command bandwidth for spatially local accesses. However, command bandwidth can be a limiting factor for programs lacking locality, especially in systems featuring fine-grained access granularities [7], [8], [9]. Furthermore, DRAM requires periodic refresh commands (REF) to prevent data loss from leakage, further taxing the available command bandwidth.

DDR4 DRAM commands use 28 non-data pins to issue and control 4 types of signals: *clock ($\overline{\text{CK}}$), control ($\overline{\text{CTRL}}$), command ($\overline{\text{CMD}}$), and address ($\overline{\text{ADD}}$)*, as shown in Figure 2. Note that signal identifiers are overlined to differentiate them from DRAM commands. Also note that there is both an activate command, ACT, and an activate signal, $\overline{\text{ACT}}$. The $\overline{\text{CMD}}$ and $\overline{\text{ADD}}$ signals time-multiplex physical pins, while $\overline{\text{CTRL}}$ and $\overline{\text{CK}}$ signals have dedicated wiring. Transmission errors over any of these pins can have a disastrous impact that is not correctable by conventional data-only ECC, as explained later in Section II-C.

A DRAM chip with an N-bit data interface is called a $\times N$ DRAM (e.g. a $\times 4$ or $\times 8$ DRAM). A rank is a set of DRAM chips that are accessed together to provide the desired data bus width. A channel is a set of ranks that time-share physical data transfer lanes. A rank/channel is comprised of *DIMMs (dual in-line memory modules)* that are built from multiple DRAM chips on a PCB board. A *memory transfer block (MTB)* is the unit of memory access and is defined by the rank width and burst length; typical MTB sizes are 64B or 128B, depending on the memory configuration.



(a) A read address error. A valid codeword is read from the wrong location ({data B, ECC B}), escaping data-only ECC.

(b) A write address error. The wrong location is overwritten and stale data is left in the intended destination. Neither error will be detected by data-only ECC.

(c) A duplicate ACT command error. The second ACT (ACT B) erroneously overwrites row B with row A. This error escapes data and address protection.

Fig. 3: Some CCCA transmission error examples. Even a small CCCA error can have disastrous consequences, and data-only ECC is useless in detecting, correcting, or diagnosing such errors.

B. Error Correcting Codes

Error checking and correcting (ECC) codes are often used by business-critical, high-availability, or high-reliability systems to protect data in memory from errors. An 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 due to errors is called a *non-codeword*.

ECC check codes are typically stored alongside data in ECC DIMMs that have some extra DRAM chips. For example, a 72-bit single-rank ECC DIMM has 18 ×4 chips to provide a 64-bit data interface and 8 bits for redundancy. This 12.5% of redundant storage is the industry standard for ECC DIMMs, and it is sufficient to provide high levels of data error protection using aggressive coding techniques [10]. AIECC extends such protection to the CCCA signals without requiring any additional redundancy such that it fits in the footprint of commodity ECC DIMMs.

A strong level of memory ECC protection that protects memory from any single chip failure is called *chipkill ECC*. Chipkill ECC on ×4 DRAM chips should therefore correct up to 32 erroneous bits per MTB to restore the full 8-beat transfer coming from the faulty chip. There are many chipkill ECC implementations; some of them use multiple codewords per memory access [11] while others use a single codeword [10] for coding efficiency.

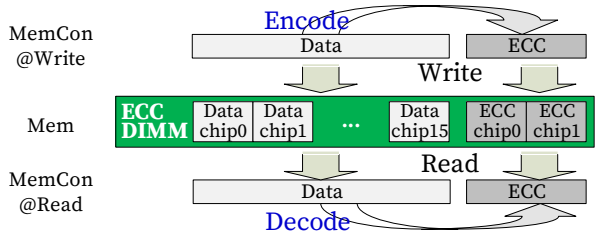
The maximum codeword size is defined by the underlying ECC code. An 8-bit symbol *Reed-Solomon (RS)* code, which is commonly used in chipkill ECC schemes, supports up to a 255-symbol (2040-bit) codeword. However, memory configurations do not typically have such coarse-grained accesses and current chipkill ECC schemes use smaller codewords (a *shortened code*) accordingly.[1] AIECC leverages the shortened nature of chipkill ECC to strongly protect address information without any additional redundancy, as we explain later.

[1]Current 8-bit RS chipkill schemes use anywhere from 18 [11] to 144 [10] symbols, depending on the memory configuration.

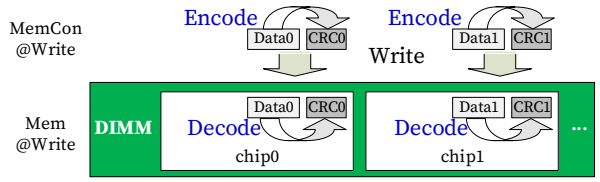
C. CCCA Errors

Transmission errors in CCCA signals are often disastrous, corrupting data storage and escaping data-only ECC. Figure 3 gives some examples of transmission errors and their associated consequences. Figure 3a shows a transmission error that changes the address of a read operation. Despite reading the wrong location, the data-only ECC codeword ({data B, ECC B}) is valid and the error causes *silent data corruption (SDC)*. An error in a write address (Figure 3b) poses an even more serious risk—not only is the wrong location (address B) updated with incorrect data, but also the data in the originally intended destination (address A) becomes obsolete. Both locations have their storage data corrupted (*memory data corruption (MDC)*) yet each location still holds a valid ECC codeword such that a following read will escape data-only ECC and result in SDC. Transmission errors in the command signals can be catastrophic, as well. Figure 3c shows a CCCA error that generates duplicate activations (for row A and B) on the same bank. The memory bit-lines are already activated with row A data and mistakenly opening a word-line copies this data into row B, destroying it and causing significant MDC. Accordingly, a later read to row B will yield valid codewords yet incorrect data, resulting in SDC.

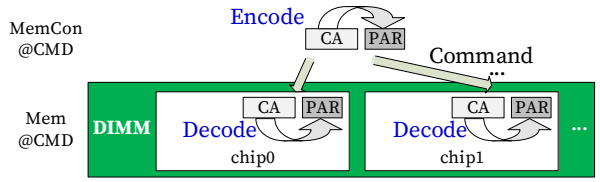
CCCA errors are growing rapidly with increasing transfer rates. A large-scale field analysis on DDR3 shows that the number of detected CA errors is as many as 72% of uncorrected data ECC errors [12]. Assuming that circuit techniques ensure a fixed *Bit Error Ratio (BER)*, DDR4 will suffer from roughly twice as many CCCA errors due to its doubled command bandwidth. With a more realistic assumption of increasing BER at higher transfer rates, the CCCA error growth will even outpace bandwidth scaling. AIECC detects and diagnoses transmission errors as they occur, preventing a high BER from translating into silent data corruption or system failures. This allows the continuing use of commodity DRAMs (which are not necessarily optimized for high reliability) in business or safety-critical systems, even in the presence of increasing transmission error rates.



(a) Data ECC (DECC)



(b) Write CRC (WCRC)



(c) CA Parity (CAP)

Fig. 4: The DDR4 reliability features.

III. Prior CCCA Protection Work

Due to rising transmission error rates and their impact on high-capacity servers, there have been a number of recent industrial developments to detect and correct CCCA errors. Such CCCA protection efforts, while numerous, seem to be fragmented and ad-hoc—no single mechanism is able to provide the comprehensive protection of data and CCCA signals. The wealth of recent industrial interest for this issue bespeaks the importance of AIECC[2], and the relative dearth of activity in academic literature for CCCA errors is an oversight that this paper attempts to rectify.

The weak and ad-hoc CCCA protection that is making its way into DRAM standards is described below in Section III-A; other alternatives that are openly published or are hinted at via public patent records are investigated in Section III-B. AIECC is designed to fit within the footprint of DDR4 CCCA protection and it incorporates and supersedes some of the best qualities from the alternative protection mechanisms. Ultimately, however, AIECC is stronger and more efficient than any prior mechanism—no other solution is able to simultaneously protect against data, address, clock, and command errors without introducing additional redundancy or new signals to and from DRAM. The relationship between AIECC and prior CCCA protection approaches is explained in more detail in Section III-C.

[2]Note that we developed AIECC independent of this prior patent survey.

A. Current DRAM Practices

DRAM has introduced reliability features to protect data against storage and transmission errors. Large-scale systems typically employ *Data ECC (DECC)*, as described in Section II-B, to protect memory data against storage and transmission errors (Figure 4a). On a read, DECC fetches both data and its ECC check bits, detecting and correcting inconsistencies due to errors. A data transmission error on a memory write remains latent as MDC until a following read to the locations with erroneous data. Such MDC is problematic, as it is likely to cause severe data loss that is not correctable through ECC. Accordingly, DDR4 introduces a *write CRC (WCRC)* for the early detection of write data transmission errors (Figure 4b) to reduce the chance of memory data corruption. WCRC generates an 8-bit CRC checksum of the write data to each chip and transmits this CRC over 2 additional beats that trail the standard 8-beat data transfer. Each DDR4 DRAM chip checks the consistency of this checksum with the received data before writing its memory array.

DDR4 introduces two other weak ad-hoc mechanisms for dealing with CCCA errors: *CA parity (CAP)* and *Gear Down Mode*. CA parity (Figure 4c) uses a dedicated pin to transfer the even parity of the $\overline{\text{CMD}}/\overline{\text{ADD}}$ signals. Upon receiving a command, each DRAM chip computes the parity of its received $\overline{\text{CMD}}/\overline{\text{ADD}}$ and checks it against the received CAP. CAP is a weak level of CCCA protection, as it does not cover the $\overline{\text{CK}}$ and $\overline{\text{CTRL}}$ groups and cannot detect an even number of bit- errors on the $\overline{\text{CMD}}/\overline{\text{ADD}}$ signals. Gear Down Mode halves the CCCA transmission rate to trade latency and command bandwidth for signal quality, keeping the data transmission rate and the data bandwidth the same. While gear down mode reduces the CCCA error rate, it is not a viable solution for memory-intensive workloads that do not exhibit very high locality.

B. Other Approaches for CCCA Protection

The topic of the end-to-end memory protection is of great importance to the server industry, and some form of CCCA protection has been the topic of patents from Freescale [13], Fujitsu [14], IBM [15], [16], [17], Intel [18], Micron [19], [20], Sun (now Oracle) [21], [22], Azul Systems [23], [24], and others [25], [26]. The different approaches taken by these protection mechanisms (as well as a single chain of academic literature [27], [28], [29]) are described below.

1) Separate Address Protection:: Various authors describe techniques to protect the address signal using separate check bits [19], [20], [21]. While these approaches are straightforward, they are inefficient as they sacrifice transmission bandwidth and require additional signals to and from memory.

2) Combined Address and Data Protection:: Address protection can be combined with data protection while sharing check bits. It appears that two prevailing techniques exist for such protection: codeword transformation and combined ECC.

Codeword transformation uses the address information to transform the data or check bits upon a write, reversing this transformation before ECC decoding. In the case of an address transmission error, this reverse transformation will corrupt the codeword; this corruption may be detected via data ECC.

Nicholas [17] partitions a write data block into 32 sub-blocks and XORs each sub-block with a corresponding address bit. Sub-blocks should be organized to report address errors as detectable-yet-uncorrectable; no specific ECC or details are mentioned, however. Normoyle [23] and Wong et al. [22] generate checksums over the address and later XOR this information in with the DECC check bits. Normoyle generates a 4-bit CRC checksum from the address and merges it with 2 DECC nibbles, which results in a detectable-but-uncorrectable error in single-symbol-correcting double-symbol-detecting (SSC-DSD) ECC codes. Wong merges a 1-bit address parity into at least two check bits for similar capabilities in a single-error-correcting double-error-detecting (SEC-DED) code.

An alternative mechanism, combined ECC, combines the address with the data prior to ECC encoding. Chen et al. [15], [16] describe a linear code that accepts the address along with the data and encodes them together for either a 140 or 146-bit channel, providing relatively weak address protection but taking special care to correctly diagnose address errors in most cases. Vogt [18] appends the address to the data before encoding, with no consideration of ECC specifics or correct address error diagnosis. The same approach was studied by Gumpertz for checking a variety of metadata, including the storage address [27] (also mentioned in [28]). The concept is described with respect to SEC-DED ECC and is not evaluated in the context of off-chip memory. Sazeides et al. [29] describe a scheme to combine metadata with ECC, but they do not consider address errors or off-chip memory. Normoyle and Hathaway [24] first take a CRC of the address and then append it and other metadata to the data before encoding.

3) Protection of Control Signals: The protection of control signals through separate ECC has been proposed, trading transmission bandwidth for reliability [19], [21]. Partial protection of the DRAM command stream through history tracking is proposed by Wang [26] and an abandoned patent application by Romdhane [25]. These approaches check for illegal command sequences or timing violations without requiring any additional signals to or from memory, but they cannot provide complete protection against command and control errors.

C. Relationship of Prior Work with AIECC

AIECC is designed to put the CCCA reliability mechanisms present in DDR4 [3] to good use in a cohesive and comprehensive protection scheme; its relationship with this work is clear and intentional. AIECC further protects the $\overline{\text{ADD}}$ signal by concatenating it with the data before encoding, similar to [18], [27]. AIECC is the first approach that considers strong levels of protection (100% address error detection with detailed error diagnosis for address transmission errors) and this paper represents the first realistic evaluation of a combined ECC protection scheme for off-chip memory.

The Command State and Timing Checker proposed for AIECC is similar in principle to the mechanisms described by [25], [26], yet is more complete. AIECC is the first to evaluate the coverage of protocol tracking, and it augments the approach to provide the complete coverage of command errors by extending the DDR4 CA parity signal to fill gaps (such as a missing WR command).

When taken as a whole, AIECC both combines and extends the best aspects of prior CCCA protection efforts. No other approach simultaneously protects against data, address, clock, and command errors with as high of error coverage as AIECC. AIECC also performs early write error detection, is able to correctly diagnose address errors, and it does not require additional redundancy or new signals to and from DRAM.

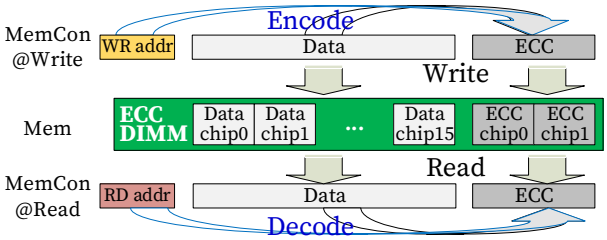
IV. All-Inclusive ECC

AIECC provides thorough and strong protection by combining 4 complementary techniques. *Extended data ECC (eDECC)* and *extended write CRC (eWCRC)* protect memory against address errors by exploiting currently unused faculties of the data ECC and write CRC. They are able to strongly protect both the data and address simultaneously with no extra storage and transfer overheads. An architectural mechanism called the *Command State and Timing Checker (CSTC)* uses memory protocol information to detect illegal command sequences and protect against errors in the $\overline{\text{CK}}$, $\overline{\text{CTRL}}$, and $\overline{\text{CMD}}$ signals. Finally, *extended CA Parity (eCAP)* strengthens this command error protection, filling some coverage holes of the CSTC.

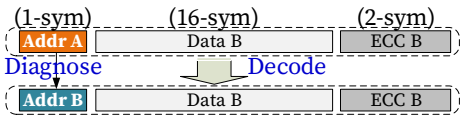
A. Extended Data ECC (eDECC)

Extended data ECC (eDECC) augments chipkill data ECC to protect address information without extra storage/transfer overheads. A $\times 4$ DRAM transfers 32 bits of data per access; eDECC leverages the strength of chipkill ECC to to detect and precisely diagnose up to 32 bits of address information. The 32-bit address used by AIECC is an MTB address that includes the rank, bank, row, and (partial) column address for the given 64B block of physical memory (Figure 5e). By using the MTB address, AIECC can protect up to 256GB per channel (compared to 4GB if it uses the full byte address); a capacity of 256GB per channel is larger than any published DRAM standard and it should be sufficient for even the highest-capacity servers.

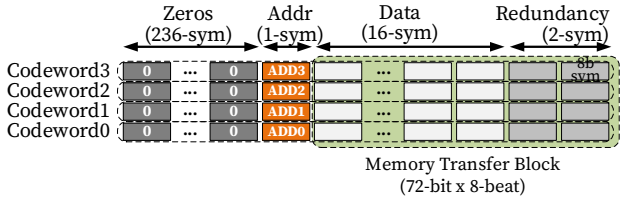
Commonly used chipkill codes can protect longer codewords than those that are called for by conventional memory access granularities. These underlying capabilities are used by eDECC to embed DRAM address information without adding additional redundancy, similar to their use for embedding other types of metadata [27], [28]. Figure 5c shows eDECC with AMD chipkill [11]. The AMD chipkill data ECC codeword has 16 data symbols and 2 check symbols, though it could potentially protect 237 more symbols without either extra redundancy or compromising its correction capability. AIECC adds one extra write address symbol to the eDECC encoding using this auxiliary protection. On a read, the returned data and redundancy are decoded together with the read address (Figure 5a). If a read address error fetches data and redundancy from a wrong address (address B instead of address A in Figure 5b), the inconsistent tuple ({address A, data B, redundancy B}) will be detected by eDECC. Further decoding of the word will reconstruct the address symbol from data and redundancy (address B in Figure 5b), diagnosing the faulty address pins



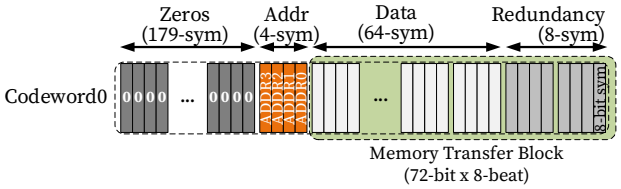
(a) An overview of Extended Data ECC (eDECC). Both the write address and data are ECC protected. On a read, the check bits verify both the read address and data.



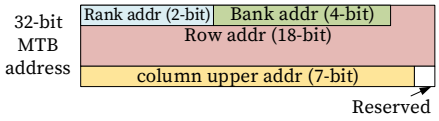
(b) Up to 32-bit address errors are correctable via chipkill ECC. An address symbol can therefore completely diagnose address errors.



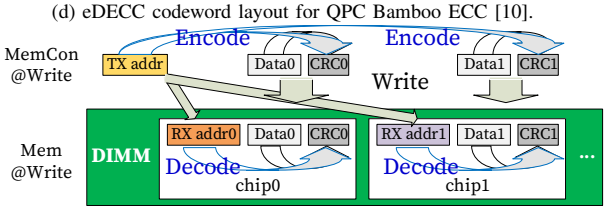
(c) eDECC codeword layout for AMD Chipkill ECC [11].



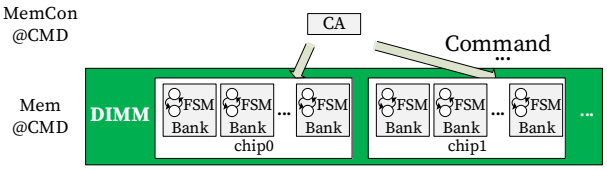
(d) eDECC codeword layout for QPC Bamboo ECC [10].



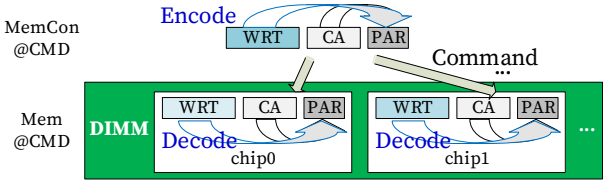
(e) MTB address in eDECC and eWCRC. A 32-bit MTB address can index up to 256GB of memory per channel.



(f) An overview of Extended Write CRC (eWCRC). A CRC checksum is generated from both the address and data (instead of just the data as with WCRC).



(g) An overview of the Command State and Timing Checker. Commands that violate timing or that cause an illegal state transition are detected by logic within each bank.



(h) Extended CA Parity (eCAP) overview. Even parity is generated from both CA and write toggle (WRT) information, detecting lost write commands.

Fig. 5: A visualization of the AIECC CCCA protection mechanisms.

by revealing the erroneous address that DRAM received. Figure 5d shows another example eDECC organization using Quadruple-Pin-Correcting Bamboo ECC [10]. The original Bamboo codeword with 64 data symbols and 8 check symbols is extended to hold 4 extra address symbols. This eDECC organization is also able to detect and diagnose any 32-bit address error without additional redundancy.

B. Extended Write CRC (eWCRC)

Extended write CRC (eWCRC) extends the write CRC of DDR4 to protect the write address as well as the data, in a similar manner to how eDECC extends a chipkill data ECC. To implement eWCRC, the DRAM controller generates an 8-bit CRC checksum from both the write data and its MTB address as shown in Figure 5f. DRAM receives the WCRC along with a write command and validates its data and address prior to changing the contents of memory. The detection coverage of eWCRC is 100% for any error that affects 8 or fewer contiguous address or data bits, and 99.6% for more severe errors. eWCRC detects write address errors prior to memory data corruption, allowing cheap common-case correction through write retry. In the rare case that an address error escapes the eWCRC, an MDC occurs; erroneously overwritten data can be diagnosed by eDECC, but the stale data that is left behind can result in SDC if read.

C. Command State and Timing Checker

The first tier of AIECC command protection is to detect illegal commands by tracking DRAM state transitions and command arrival times. AIECC adds a *Command State and Timing Checker (CSTC)* to the DRAM alongside each bank for this purpose; this CSTC checks the validity of received commands based on the memory protocol (Figure 5g). At any given time, DRAM has a predetermined bank context (ACT/REF on an idle bank and RD/WR/PRE on an open bank) and a valid context-breaking command (e.g., ACT on an open bank) can be easily detected by monitoring bank states. Additionally, command

Command	Bank state	Timing parameters
ACT	Idle	tRC, tRRD, tFAW, tRP, tRFC
REF	Idle	tRRD, tFAW, tRP, tRFC
RD	Open	tRCD, tCCD, tWTR
WR	Open	tRCD, tCCD
PRE	Open	tRAS, tRTP, tWR
NOP	Any	

TABLE I: DRAM commands with their allowed bank state and timing constraints.

arrival times are monitored to detect erroneous commands that violate the DRAM timing guarantees. Each CTSC is implemented as a small state machine using DRAM commands from the JEDEC standard and binned timing parameters that are known by the vendor. Table I shows the bank state and timing constraints for DDR4 DRAM, taken from the JEDEC specification [3]. Commands for DRAM initialization (mode register set and ZQ calibration) and for power saving modes (self-refresh and power down) are excluded for simplicity, but these commands are later included in our experimental evaluation.

There are some command errors that are not necessarily caught by the CSTC; most of these errors either do not compromise the functionality of the system or they will be detected through the address checking mechanisms. An extra refresh operation with valid timing may not be detected by the CSTC, but it does not affect correct operation. A missing refresh is not detected; we assume DRAM has some retention time margins so that a lost refresh operation does not corrupt data. Missing or duplicate activations (ACT/PRE) are not immediately detected by the CSTC, but they change the bank state so that the next command to the bank will trigger a CSTC error and data integrity is not compromised.

Errors that corrupt a command to or from a RD result in an extra or missing read operation, respectively. An extra or missing read command corrupts the write pointer in the read data FIFO of the DDR PHY (physical interface) such that the memory controller receives a wrong entry from the PHY. This wrong entry is then detected by eDECC as it will not validate the address of the codeword taken from the read FIFO (this address is produced and stored within the memory controller itself and is not subject to transmission errors). An extra write command will attempt to interpret the I/O at the data pins as a value and will write this value back to the open row. The data interpreted from the undriven I/O pins can be random (if they are fully undriven) or all-ones (if they are partially driven by termination resistors); in either case, the erroneous write will be handled by the eWCRC and eDECC like any write data error.

D. Extended CA Parity (eCAP)

The only command error that is not covered by either the CSTC or the AIECC address checking mechanisms is a missing write. In this case, as DRAM never receives the command, it would not report an error and the memory controller would assume that an error-free write completes. To detect this erroneous situation, we extend the CA parity of DDR4 to cover missing WR commands (Figure 5h). To enforce *extended CA parity (eCAP)*, the memory controller and memory maintain synchronized *write toggle (WRT)* bits that flip upon sending/receiving a WR command. The CA parity is then generated across both the 24 CA signals and the WRT bit. If there is a missing WR, the WRT values in the memory controller and memory disagree on the next command and the error is detected.

E. $\overline{CK}$ and $\overline{CTRL}$ Protection

Errors in the $\overline{\text{CK}}$ and $\overline{\text{CTRL}}$ signals can compromise commands as well. An additional toggle on $\overline{\text{CK}}$ results in the reception of an erroneous command, while a missing toggle causes a command to be lost. Errors in $\overline{\text{CKE}}$ (clock enable) and $\overline{\text{CS}}$ (chip select) can also incur an extra/missing command. These extra/missing commands are all detected by eDECC, eWCRC, CSTC, and eCAP as explained above. Errors in $\overline{\text{ODT}}$ (on die termination) deteriorate the data signal quality, which can be detected by eWCRC and eDECC.

F. Precise Diagnosis

eDECC not only detects address errors but can also pinpoint faulty address pin(s) by restoring the original address and comparing it against the erroneous address. We expect such knowledge to be valuable to repair techniques, such as selectively tuning delay and drive parameters of the reported pin. Without this knowledge, extensive diagnostic routines are required or repeated CCCA errors may impact system reliability and availability.

G. Correction Details

AIECC correction is more straightforward than data correction via ECC. Because AIECC generally detects CCCA errors early (before possible data corruption), correction simply entails retrying the faulty command. In the rare case that AIECC detects an error late (after possible data corruption) then a detectable-uncorrectable error must be flagged to higher system levels. This is no different from current systems that encounter an uncorrectable error, and the uncorrectable AIECC error rate is shown in Section V to be low.

V. Evaluation

We investigate the reliability and efficiency of AIECC below. Section V-A analyzes the level of CCCA reliability provided by AIECC and prior protection techniques. Section V-B investigates the impact of AIECC on the strength of the chipkill data protection, finding it to be negligible. Section V-C estimates the system-level reliability of systems with transmission errors, showing the impact of AIECC to be significant in many scenarios. Finally, Section V-D describes the modest design changes (and correspondingly low overheads) required to extend a DDR4 memory to support AIECC.

A. CCCA Reliability

The CCCA error coverage of AIECC is evaluated using error injection simulations, as depicted by Figure 6. A CCCA error has different consequences with different DRAM commands; we test 5 dominant command patterns: ACT (followed by WR), ACT (followed by RD), WR, RD, and PRE. ACT

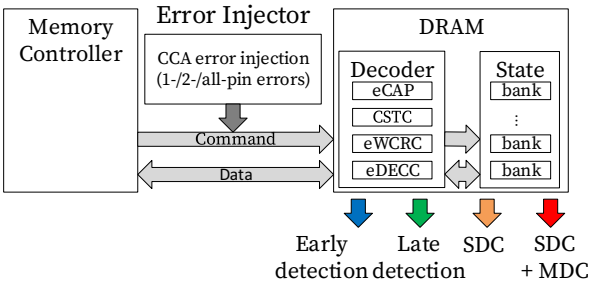


Fig. 6: The CCCA evaluation environment. The impact of injected CCCA errors is determined through AIECC and DRAM simulation.

Erroneous Pin		DRAM Command in Error			
#	Name	ACT	WR	RD	PRE
27	CK	Clock error			
26,25	CKE, CS	ACT–	WR–	RD–	PRE–
24	ODT	On-die termination error			
23	PAR	No error			
22	ACT	ACT→?	ACT+		
21	RAS (A16)		WR→MRS	RD→REF	PRE→ZQC
20	CAS (A15)	$ACT_{rowaddr}$	WR→ZQC	RD→NOP	PRE→MRS
19	WE (A14)		WR→RD	RD→WR	PRE→RFU
18∼15	BG1,0,BA1,0	$ACT_{bankaddr}$	$WR_{bankaddr}$	$RD_{bankaddr}$	$PRE_{bankaddr}$
14	A12 (BC)		$WR_{burstchop}$	$RD_{burstchop}$	No error
13∼11	A17,13,11		No error		
10	A10 (AP)	$ACT_{rowaddr}$	ACT–		
9∼0	A9∼0		$WR_{coladdr}$	$RD_{coladdr}$	No error

SDC | SDC and MDC | SDC and conditional MDC

TABLE II: The impact of 1-pin CCCA errors across pin locations and commands. CMD-/CMD+/CMD_A →CMD_B indicate missing, extra, and altered commands (changed from CMD_A to CMD_B), respectively. A transition to MRS, ZQC, and RFU indicate that the DRAM was erroneously given a mode register set command, ZQ calibration command, or a reserved-for-future-use command, respectively.

is sub-categorized because the consequence of an activate error depends heavily on the following command—a missing ACT followed by WR results in memory data corruption, while a missing ACT followed by RD reads arbitrary data but it does not corrupt storage. Other commands do not vary significantly with the following command, and we use a single test sequence for each. Before each erroneous command, the simulated DRAM is set to have all banks open (except for erroneous ACTs where the target bank is closed).

Transmission errors are modeled as 1-pin, 2-pin, and all-pin errors in the CCCA signals of the target command. These models represent sources of transmission noise such as inter-symbol interference, crosstalk with 2 victims, and clock/power noise, respectively. A $\overline{\text{CK}}$ error is modeled as a source of all-pin errors, rather than individual 1-pin error. We cover all combinations of 1-pin, 2-pin, and all-pin errors on the 27 $\overline{\text{CTRL}}$, $\overline{\text{CMD}}$, and $\overline{\text{ADD}}$ signals. In the no-protection configuration, errors are not injected on the $\overline{\text{PAR}}$ pin because it is assumed to be non-existent or disconnected.

1) Impact of undetected CCCA errors: The effects of

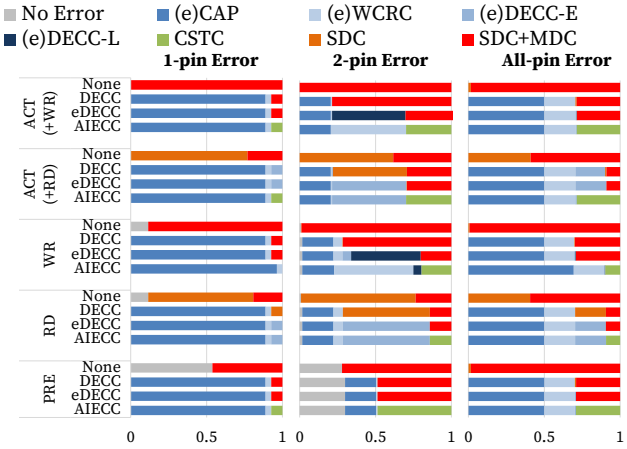


Fig. 7: The CCCA error detection coverage of an unprotected DDR4 DIMM, DDR4+DECC (DECC), DDR4+eDECC (eDECC), and DDR4+AIECC (AIECC).

undetected CCCA errors depend strongly on the DRAM command and in some cases also on the following command. To understand these effects, we implement a DRAM model that tracks DRAM states, decodes erroneous CCCA signals, and interprets the impact of each error based on the decoded command and resultant DRAM state. We tabulate our 1-pin error results in Table II and summarize the findings below. (Results of 2-pin and all-pin errors appear in later experiments, but they are not included in this analysis for simplicity.) **ACT:** Any undetected error during an ACT causes SDC+MDC (if followed by WR) or SDC (if followed by RD). **WR:** Three pins (A11, A13, and A17) do not participate in the WR operation and manifest no error. Errors on $\overline{\text{CKE}}$, $\overline{\text{CS}}$, $\overline{\text{ODT}}$, $\overline{\text{ACT}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{WE}}$, burst-chop ($\overline{\text{BC}}$), auto-precharge ($\overline{\text{AP}}$), bank address, and column address manifest as SDC+MDC. **RD:** Three pins (A11, A13, and A17) manifest no error. Errors on $\overline{\text{CKE}}$, $\overline{\text{CS}}$, $\overline{\text{CAS}}$, bank address, $\overline{\text{BC}}$, and column address manifest as SDC in the read data, while errors on other signals generate SDC+MDC. **PRE:** Fourteen pins (A17, A13∼A11, A9∼A0) manifest no error, while errors on $\overline{\text{CKE}}$, $\overline{\text{CS}}$, $\overline{\text{ACT}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{WE}}$, A10 and bank address manifest SDC+MDC.

2) Detection coverage: To examine the error coverage of different protection mechanisms, we check CCCA errors against four increasing levels of protection: ① no protection, ② DDR4 + data ECC (DECC), ③ DDR4 + eDECC, and ④ DDR4 + AIECC. Figure 7 shows the fraction of 1-pin/2-pin/all-pin CCCA errors that DECC, eDECC, and AIECC can detect. **1-PIN ERRORS:** CA parity detects 1-pin errors on the 24 CA signals but not on the 3 $\overline{\text{CTRL}}$ signals, two of which are problematic ($\overline{\text{CKE}}$ and $\overline{\text{CS}}$—see Section V-A1). A missing RD command manifests as SDC with data-only DECC, yet it can be detected by eDECC. AIECC can detect all 1-pin errors. **2-PIN ERRORS:** The detection strength of CA parity is limited for 2-pin errors, resulting in large coverage holes in DDR4+DECC and DDR4+eDECC. AIECC fills the holes in CAP, avoiding all SDC and MDC. **ALL-PIN ERRORS:** CA parity actually performs

better in an all-pin scenario than with 2 erroneous pins, as it has a 50% chance of detecting the error. However, as undetected command and control errors are likely to severely corrupt data, extra coverage is still needed. AIECC provides thorough command and control protection using CSTC and eCAP, and only errors that escape eWCRC may cause SDC+MDC.

Figure 8 shows which AIECC components detect different CCCA errors. It is apparent that the most effective CCCA protection mechanism heavily depends on the specific error scenario, such that all four AIECC mechanisms are required for robust CCCA error coverage. Address protection (eWCRC and eDECC) is crucial for protecting write and read commands. All-pin errors during activation are best detected by CSTC, as the errors frequently change the command into another (invalid) command. However, eCAP is the most effective mechanism for 1-pin activation errors, and 2-pin errors are protected by either address protection or CSTC, depending on whether the address or command is affected. Only through a combination of eDECC, eWCRC, CSTC, and eCAP is AIECC able to provide complete coverage.

B. Data Reliability

Despite its strong level of CCCA protection, AIECC has minimal impact on the levels of existing data protection. Table III compares the level of data protection between data-only chipkill ECC using QPC Bamboo ECC [10], a checksum-based address protection mechanism that is fully described in a patent disclosure[3] [23], and two eDECC variants for AIECC: one based on combined ECC (as described in Section IV-A) and one based on codeword transformation. Prior work in the area of address protection that does not require redundant storage or bandwidth falls into these two camps (Section III-B). The combined ECC approach provides precise diagnostics whereas codeword transformation does not. However, it is possible that error coverage of the two techniques is not equivalent and may be lower with combined codewords [29]. To analyze these effects we create a strong codeword transformation scheme for comparison purposes. We adapt the approach taken by Nicholas/IBM [17] to work with Bamboo ECC for our strong transformation-based eDECC variant.[4] Codeword transformation eDECC decomposes a 64B data MTB into 32 16b sub-blocks, aligning these sub-blocks orthogonally to the Bamboo ECC symbols. Data is transformed by XORing each sub-block by an associated bit of the address. Any address error is guaranteed detection, and the scheme profits from the very high error detection coverage of Bamboo ECC.

We inject errors into the data and address symbols using Monte-Carlo simulation (4 billion trials), determining the results of each error through a simulated Bamboo ECC decoder. The strong detection capability of the underlying data ECC yields virtually 100% detection of serious full-rank errors, with the Monte Carlo analysis indicating $<10^{-6}$% undetected errors. While adding the address information to eDECC may degrade its data error detection capability, our experiments conclude that the degradation is negligible and the protection level is equivalent. Any single-chip errors are still corrected (preserving chipkill) and AIECC can detect virtually 100% of concurrent data and address errors.

C. System-Level Reliability Improvements

Section V-A shows that AIECC provides thorough and strong protection against CCCA errors on any and all pins. Its implication on system-level reliability, however, depends on the underlying CCCA error rate. Due to a lack of publicly available DDR4 CCCA error rates, we perform a sensitivity sweep on the *BER (Bit Error Ratio)* and multiply it with command bandwidths and signal counts to estimate CCCA error rates. Other approaches for CCCA errors, such as simulation-based modeling and lab measurements, are excluded due to factors like confidential process and layout information and low confidence from limited samples.

Due to the dependence of CCCA error manifestation on the DRAM command stream, we first characterize the DRAM behavior of 56 benchmarks from the NPB, SPEC2006, Parsec, and SPLASH2X benchmark suites[5] [30], [31], [32], [33] using the Xeon E5 and E7 v3 memory controller performance counters [34]. Four representative clusters are identified by hierarchically clustering across the memory bandwidth utilization, read to write ratio, CAS to ACT ratio, and ACT→RD to ACT→WR ratio of each program. Three of the clusters differ mainly in their data bandwidth utilization, and the other (SPLASH2X's wat-ns) is an outlier with an extreme read-to-write ratio.[6] Table 9a gives the data and command bandwidth for the median centroids of these clusters.

For a given BER, we estimate the CCCA FIT rate of each representative centroid using Equation 1. This equation accumulates the FIT contribution from each CCCA-sensitive command—ACT (+WR), ACT (+RD), WR, RD, and PRE–over all 1-pin errors. Apart from a $\overline{\text{CK}}$ error, which affects all pins, no multi-pin errors are modeled due to a lack of data on their rates and distributions.

$$\begin{aligned} FIT_{CCCA} = BER \times \sum_{i \in \text{CMD}} \sum_{j \in \text{ERR}} \{ & \{\text{Command Bandwidth}\}_i \\ & \times \{\text{Signal Count}\}_j \\ & \times \{\text{Undetected Probability}\}_{i,j} \\ & \times 3.6 \times 10^{12} \} \end{aligned} \quad (1)$$

Our evaluation is swept from 10^{-16} BER, which is the minimum design specification for data in the JEDEC DDR4 standard [3], up to 10^{-22}. The minimum JEDEC BER of 10^{-16} is very weak and corresponds to 2.8×10^6 unprotected FIT_{CCCA} with the high-bandwidth centroid. This BER is likely to be

[3]The approach taken by Normoyle/Azul is modified slightly for QPC—the 4-bit address checksum it uses is triplicated and merged into first beats of 3 chips in order to avoid miscorrection from 2-pin correcting QPC.

[4]Note that Nicholas/IBM [17] does not describe any specific implementation of their general idea.

[5]The C, ref, and native input sets are used for these suites, with one thread per core. SPEC2006 is run with 10 replicated processes in order to fit in a 16GB memory footprint.

[6]NPB's cg is also an outlier with a high but less-extreme read-to-write ratio; it is omitted for brevity.

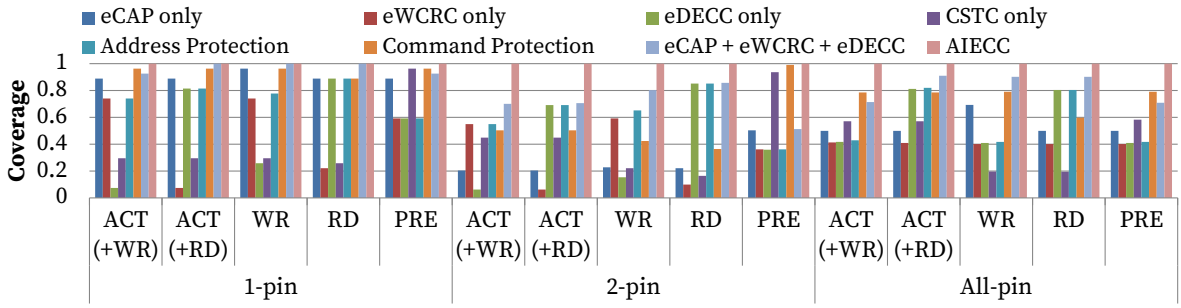


Fig. 8: A quantitative evaluation of the different AIECC components: eDECC (Section IV-A), eWCRC (Section IV-B), address protection (eDECC+eWCRC), CSTC (Section IV-C), eCAP (Section IV-D), command protection (CSTC+eCAP), eDECC+eWCRC+eCAP (for completeness), and AIECC (eDECC+eWCRC+CSTC+eCAP). All four components of AIECC are necessary to protect against different CCCA errors. Note that the test scenario is ACT-WR-RD-PRE so that most row address errors are first detected by eWCRC; this may not always be the case.

Data Error	Address Error	Protection			
		QPC	QPC+Azul [23]	QPC+eDECC-t [17]	QPC+eDECC-c
None	1 bit	100.0% SDC	CE-R	CE-R	CE-R+
	32 bits	100.0% SDC	6.3% SDC	CE-R	CE-R+
1 bit	None	CE-D	CE-D	CE-D	CE-D
	1 bit	100.0% SDC	0.14% SDC	CE-RD	CE-RD+
	32 bits	100.0% SDC	6.4% SDC	$<10^{-6}$% SDC	$<10^{-6}$% SDC
1 chip	None	CE-D	CE-D	CE-D	CE-D
	1 bit	100.0% SDC	$<10^{-6}$% SDC	$<10^{-6}$% SDC	$<10^{-6}$% SDC
	32 bits	100.0% SDC	6.3% SDC	$<10^{-6}$% SDC	$<10^{-6}$% SDC
1 rank	None / 1 bit / 32 bits	$<10^{-6}$% SDC			

TABLE III: Data and address reliability comparison. eDECC-t and eDECC-c denote eDECC with codeword transformation and combined ECC, respectively. SDC stands for silent data corruption; CE-D, CE-R, and CE-R+ stand for a corrected error through data ECC, retry following a DUE, and retry following accurate diagnosis. Similarly, CE-RD and CR-RD+ stand for a corrected error that uses both retry and data ECC (with the retry following a DUE and accurate diagnosis, respectively).

higher than that in a real system—the FIT of a DDR2/DDR3 x4 DRAM device (including both storage and transmission errors) is around 25–66 [12], [35], [36]. The stronger BER of 10^{-22} corresponds to 2.8 FIT_{CCCA}, 3.4×10^{6} system-FIT and a 12-day MTTF on a system with 1.2M DRAM devices. We believe this number to be within an order of magnitude of the CCCA error rate exhibited by the Cielo system [12], which has a similar number of DDR3 chips as our modeled system; however, the actual measurements on Cielo are not public.

Figure 9b shows FIT_{CCCA} from 10^{-22} BER with and without protection. Applications with more data bandwidth have a higher FIT_{CCCA} because they issue more commands. DDR4+DECC reduces the CCCA SDC and MDC rates by an order of magnitude using the DDR4 reliability mechanisms. eDECC reduces the SDC rate further by detecting read address errors. AIECC improves the unprotected CCCA failure rate by four orders of magnitude because it detects all read errors and nearly all write errors (as was shown in Section V-A2).

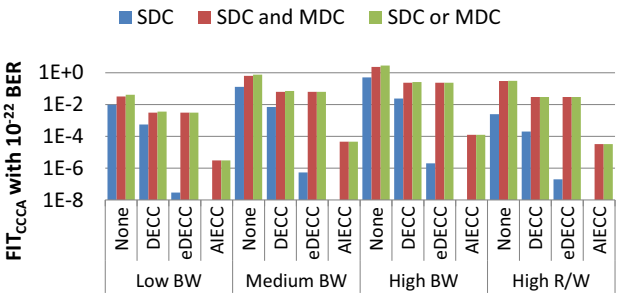
We expect that large-scale systems utilizing DDR4 will suffer from increased CCCA error rates due to its doubled transfer rate, and test higher BER values of 10^{-21} and 10^{-20} accordingly. These BERs change the Y-axis scale of Figure 9b yet its shape remains the same. Table 9c shows the estimated SDC MTTF on a 1.2M-DRAM system with high bandwidth utilization. AIECC provides an MTTF of 8 years, even with the more severe BER. In contrast, other schemes have orders of magnitude worse reliability with an MTTF of just 1 or 2 days.

D. Hardware Overheads

AIECC has negligible hardware overhead and it represents a straightforward upgrade to existing DDR4 features and chipkill data ECC. It requires no additional pins or bandwidth to and from memory as it reuses the existing WCRC and CAP mechanisms and it does not require any additional ECC storage. We implement a Verilog model of AIECC to estimate its logic overheads and synthesize it with the Synopsys toolchain and the TSMC 40nm LP standard cell library [37], [38]. ePAR/eWCRC/eDECC+AMD/eDECC+QPC have area overheads equivalent to 30/180/140/2200 NAND2 gates and dynamic+static power increases of 0.01/0.1/0.05/0.8mW, respectively. The logic depth of the DECC decoder increases by 1 XOR gate and cycle latency is not impacted. On the DRAM side, ePAR/eWCRC/CTSC require the area equivalent of 30/180/9000 NAND2 gates per chip and consume 30/180/0.8mW, respectively. CTSC is off the critical path and it does not affect memory latency. The AIECC correction procedure requires support in the memory controller for

# Apps	Major Feature	Data BW	Command BW ($\times 10^6$ cmds/sec)				
			ACT (+WR)	ACT (+RD)	WR	RD	PRE
33	Low Data BW	0.50%	0.64	0.39	0.69	2.22	1.03
10	Med. Data BW	7.90%	9.18	16.7	8.57	33.3	25.9
11	High Data BW	22.0%	39.4	76.2	29.2	90.1	116
wat-ns	High RD/WR	4.31%	0.15	6.13	0.17	23.6	6.28

(a) Representative benchmark clusters and their bandwidths.



(b) $\times 4$ DRAM CCCA FIT rates after protection with 10^{-22} BER.

BER	Protection			
	None	DECC	eDECC	AIECC
10^{-22}	12 days	4 months	5 months	768 years
10^{-21}	1 day	13 days	15 days	77 years
10^{-20}	3 hours	32 hours	35 hours	8 years

(c) CCCA SDC Mean-Time-To-Failure on a system with 1.2M DRAM chips and high bandwidth utilization.

Fig. 9: AIECC system-level reliability evaluation.

command replay. We note that most systems already support for on-demand scrubbing[7] [11], [39] (writing back corrected data to DRAM to eliminate transient bit-flips), and expect the further additions for AIECC correction to be modest.

While eDECC does not impact performance with the QPC ECC used in this paper, it could potentially increase read latency using AMD chipkill and other chipkill-correct schemes. These schemes can detect data errors before a read transfer is complete, but would have to wait until the end of a block transfer for precise eDECC address error diagnosis. Prior simulation results show this performance overhead to be modest (an average of 1.7% on a 64-bit DDR3-1600 data channel [10]). Alternatively, a different eDECC organization could sacrifice precise diagnosis to enable early data error detection, or asynchronous ECC checking [40] could avoid this penalty in the common case; a detailed exploration of AIECC performance with alternative ECC schemes is left for future work.

VI. Discussion

Applicability to Other Memories: AIECC targets DDR4, as it is the dominant memory for large-scale systems and can provide both high capacity and performance. The principles that empower AIECC are general, however, as is its methodology for targeting and evaluating complete end-to-end protection. As an example of this generality, AIECC can be applied to GDDR5 with modest changes. eDECC can be supported in GDDR5 with some tailoring to the ECC codes in use (it is unlikely that GPUs use AMD chipkill or QPC). GDDR5 includes an EDC pin that can be reused for eWCRC by incorporating both the address and data for writes. While GDDR5 does not have a dedicated CA parity pin, missing writes and other command errors could be detected by incorporating WRT and CA parity into the GDDR5 read CRC over the same EDC pin. CSTC can be implemented in GDDR5 a similar manner to DDR4, only using the GDDR5 commands and timing parameters.

The fundamental concepts and design choices behind AIECC are also congruent with 3D stacked DDR memory such as HBM [41]. The CRC of packetized HMC memory [42], meanwhile, protects against CCCA transmission errors between the memory controller and the base layer, but the end-to-end protection of AIECC could also protect against internal base layer errors. Future work should extend AIECC to these memory organizations once it is clear how transmission errors and data protection operate in the stacked domain.

Enriching the DRAM Design Space: Circuit-level techniques and frequency margins are used in order to achieve current DRAM transfer rates without violating transmission error rate targets. In addition, large-capacity servers often employ memory buffers to isolate each DIMM from the capacitive load of all the ranks on each channel [43], [44]. Such techniques come at a frequency, power, and latency cost. AIECC provides high reliability for a memory system that was designed a priori to achieve an industry-standard bit error ratio. However, a system using AIECC could perhaps also tolerate a relaxed BER target without reliability degradation, exposing the system designer to a richer set of design tradeoffs and achieving superior efficiency. This potential use for AIECC is left for future exploration.

VII. Conclusion

All-inclusive ECC is a readily-implementable suite of complementary error protection mechanisms for DRAM data and CCCA signals. Our analyses demonstrate that data ECC, the DDR4 reliability mechanisms, and address protection similar to that which may be used by industry are insufficient for the complete end-to-end protection of DRAM; AIECC supplements these current practices to provide strong holistic protection. Despite its apparent advantages, AIECC has minimal associated costs and requires no new signals to or from memory, no additional storage, negligible hardware real-estate, and it does not significantly affect the level of data protection.

Acknowledgment

The authors acknowledge the Texas Advanced Computing Center for providing HPC resources and the support of the Department of Energy under contract LLNS B609478 and the National Science Foundation under Grant #0954107, which partially funded this research.

[7] On-demand scrubbing is also called redirect scrubbing by AMD.

REFERENCES

[1] Samsung Electronics Co., "Samsung DDR4 SDRAM," http://www.samsung.com/global/business/semiconductor/file/media/DDR4_Brochure-0.pdf, 2013.

[2] *DDR3 SDRAM STANDARD, JESD79-3F*, Joint Electron Device Engineering Council, July 2012.

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

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

[5] *Graphics Double Data Rate (GDDR5X) SGRAM Standard, JESD232*, Joint Electron Device Engineering Council, Nov. 2015.

[6] *DDR2 SDRAM Specification, JESD79-2F*, Joint Electron Device Engineering Council, Nov. 2009.

[7] J. Ahn, M. Erez, and W. Dally, "The Design Space of Data-Parallel Memory Systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, 2006.

[8] D. H. Yoon, M. K. Jeong, M. Sullivan, and M. Erez, "The Dynamic Granularity Memory System," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 2012, pp. 548–559.

[9] M. Rhu, M. Sullivan, J. Leng, and M. Erez, "A locality-aware memory hierarchy for energy-efficient gpu architectures," in *Proceedings of the International Symposium on Microarchitecture (MICRO)*, December 2013.

[10] J. Kim, M. Sullivan, and M. Erez, "Bamboo ECC: Strong, Safe, and Flexible Codes for Reliable Computer Memory," in *Proceedings of the International Symposium on High Performance Computer Architecture (HPCA)*, 2015.

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

[12] V. Sridharan, N. DeBardeleben, S. Blanchard, K. B. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi, "Memory Errors in Modern Systems: The Good, The Bad, and The Ugly," in *Proceedings of the International Symposium on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. New York, NY, USA: ACM, 2015, pp. 297–310.

[13] W. C. Moyer, "Selective masking for error correction," U.S. Patent US8 990 657 B2, Mar, 2015. [Online]. Available: http://www.google.com/patents/US8990657

[14] M. Kuramoto, M. Koyabu, J. Tsuiki, and J. Inagaki, "Storage control circuit, and method for address error check in the storage control circuit," U.S. Patent US7 555 699 B2, Jun, 2009. [Online]. Available: http://www.google.com/patents/US7555699

[15] C.-L. Chen, M.-Y. Hsiao, P. J. Meaney, and W. W. Shen, "Detecting address faults in an ECC-protected memory," U.S. Patent US6 457 154 B1, Sep, 2002. [Online]. Available: http://www.google.com/patents/US6457154

[16] C.-L. Chen, R. B. Tremaine, and M. E. Wazlowski, "(146,130) error correction code utilizing address information," U.S. Patent US6 751 769 B2, Jun, 2004. [Online]. Available: http://www.google.com/patents/US6751769

[17] R. Nicholas, "Address error detection," U.S. Patent US8 949 694 B2, Feb, 2015. [Online]. Available: http://www.google.com/patents/US8949694

[18] P. D. Vogt, "Combined command and data code," U.S. Patent US7 827 462 B2, Nov, 2010. [Online]. Available: http://www.google.com/patents/US7827462

[19] W. D. Parkinson and E. J. Heitzeberg, "Parity and error correction coding on integrated circuit addresses," U.S. Patent US5 173 905 A, Dec, 1992. [Online]. Available: http://www.google.com/patents/US5173905

[20] A. Troia, "Integrity of an address bus," U.S. Patent US9 009 570 B2, Apr, 2015. [Online]. Available: http://www.google.com/patents/US9009570

[21] A. Phelps, "Memory subsystem including an error detection mechanism for address and control signals," U.S. Patent US6 941 493 B2, Sep, 2005. [Online]. Available: http://www.google.com/patents/US6941493

[22] S. S. Wong, K. Aravinthan, G. N. Levinsky, S. Dor, R. T. Van, and J. Lu, "Methods and systems for detecting memory address transfer errors in an address bus," U.S. Patent US7 293 221 B1, Nov, 2007. [Online]. Available: http://www.google.com/patents/US7293221

[23] K. B. Normoyle, "Address error detection by merging a polynomial-based CRC code of address bits with two nibbles of data or data ECC bits," U.S. Patent US7 203 890 B1, Apr, 2007. [Online]. Available: http://www.google.com/patents/US7203890

[24] K. B. Normoyle and R. G. Hathaway, "Encoding 64-bit data nibble error correct and cyclic-redundancy code (CRC) address error detect for use on a 76-bit memory module," U.S. Patent US7 398 449 B1, Jul, 2008. [Online]. Available: http://www.google.com/patents/US7398449

[25] K. Fekih-Romdhane, "Illegal commands handling at the command decoder stage," U.S. Patent US20 070 245 036 A1, Oct, 2007. [Online]. Available: http://www.google.com.tr/patents/US20070245036

[26] D. Wang, "Protocol checking logic circuit for memory system reliability," U.S. Patent US8 966 327 B1, Feb, 2015. [Online]. Available: http://www.google.com/patents/US8966327

[27] R. H. Gumpertz, "Error Detection with Memory Tags," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, USA, 1981.

[28] ——, "Combining Tags with Error Codes," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 1983, pp. 160–165.

[29] Y. Sazeides, E. zer, D. Kershaw, P. Nikolaou, M. Kleanthous, and J. Abella, "Implicit-storing and redundant-encoding-of-attribute information in error-correction-codes," in *Proceedings of the International Symposium on Microarchitecture (MICRO)*, 2013, pp. 160–171.

[30] "The Nas Parallel Benchmarks." [Online]. Available: http://www.nas.nasa.gov/publications/npb.html

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

[32] C. Bienia and K. Li, "Fidelity and scaling of the PARSEC benchmark inputs," in *Proceedings of the International Symposium on Workload Characterization (IISWC)*, 2010.

[33] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, 1995.

[34] "Intel xeon processor e5 and e7 v3 family uncore performance monitoring reference manual," Intel Corporation, Tech. Rep. 331051-002, 2015.

[35] 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.

[36] 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.

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

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

[39] M. Demshki and R. Shiveley, "Advanced reliability for intel xeon processor-based servers," Intel Corporation, Tech. Rep., 2010.

[40] M. Nicolaidis, T. Bonnoit, and N.-E. Zergainoh, "Eliminating speed penalty in ECC protected memories," in *Proceedings of Design, Automation, and Test in Europe (DATE)*, 2011, pp. 1–6.

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

[42] H. M. C. Consortium, "Hybrid memory cube specification 2.1," 2015.

[43] J. Haas and P. Vogt, "Fully-buffered DIMM technology moves enterprise platforms to the next level," *Technology*, p. 1, 2005.

[44] B. Mutnury, M. Cases, N. Pham, D. De Araujo, E. Matoglu, P. Patel, and B. Hermann, "Analysis of fully buffered DIMM interface in high-speed server applications," in *Electronic Components and Technology Conference, 2006. Proceedings. 56th*, 2006.