Received 10 October 2023, accepted 1 November 2023, date of publication 3 November 2023,
date of current version 10 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3329984

SURVEY

Review of Memory RAS for Data Centers

JISEONG LEE, (Graduate Student Member, IEEE),
MIN JOON KIM, (Graduate Student Member, IEEE),
WOO-SEOP KIM, (Member, IEEE), AND YONG SIN KIM, (Senior Member, IEEE)
School of Electrical Engineering, Korea University, Seoul 02841, South Korea

Corresponding author: Yong Sin Kim (shonkim@korea.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government Ministry of Science and ICT (MSIT) under Grant NRF-2021R1A2C2014652; and in part by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) through the Open Field Smart Agriculture Technology Short-Term Advancement Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) under Grant 03322029032HD070.

ABSTRACT Multi-bit error and downtime due to uncorrectable error (UE) in a dual in line memory module (DIMM) have received great attention in data centers for its high repair or replacement cost. These problems can be alleviated by utilizing ECC (Error Correction Code) technology, which enables prompt error correction during initial occurrences and prediction of future UEs based on recurring error patterns. The technologies for addressing errors can be categorized into reliability, availability, and serviceability (RAS), and need to be optimized using various parameters such as accuracy, recall, F-measures, and cost reduction. This paper describes an overview of the current RAS technologies and trends in memory for data centers, which includes an analysis of conventional ECC technologies and their recent developments. Once UEs cannot be completely eliminated with ECCs, page offline methods based on analysis on error patterns and characterization of UE can be performed. Recent research trends for reducing memory capacity wasted by UE and page offline have been towards on-die ECC in high bandwidth memory architecture.

INDEX TERMS Correctable error (CE), error correction code (ECC), memory reliability, availability, serviceability (RAS), uncorrectable error (UE).

I. INTRODUCTION

The scale of data centers has exponentially increased with the increase in workload, hardware density, and high-performance requirements. As a result, downtime due to device failure has become one of the most serious problems in maintaining stable operation. Industry and academia have developed various solutions for downtime-related problems in terms of reliability, availability, and serviceability (RAS) [1], [2].

Reliability represents the ability of equipment to prevent or correct errors and can be quantified by averaging the time interval between failures, defined as the mean time between failures (MTBFs). For example, reliability features attempt to correct or isolate errors while stopping a related program or the entire system. Thus, reliability depends exponentially on the minus of recovery time and inversely on the MTBF [3]. Availability is a probability of a system operating normally and indicates whether users can continue to use the service. It is quantified as the duration of the downtime for a certain period, according to the number of 9 s [4], [5]. For example, 99.9% and 99.99% probability of availability indicates unavailability of a system for 526 and 53 min per year, respectively. Serviceability is the ability of a system to maintain operation without failure and repair itself without intervention. Therefore, early detection and response to potential problems can be significant for serviceability [6].

Since the early 2000s, RAS techniques have been developed for reducing the server downtime caused by system failure or data loss [7]. The downtime cost per data center has steadily increased, reaching 740,000 USD per data center based on the availability of 99.9% in 2016 [8]. If multiple datacenters are allocated to a major industry company, the downtime cost can be substantial. Once availability increases to 99.99%, the cost can be reduced to one-tenth or less. Therefore, RAS features are essential to the data center.

The associate editor coordinating the review of this manuscript and approving it for publication was Catherine Fang.

© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

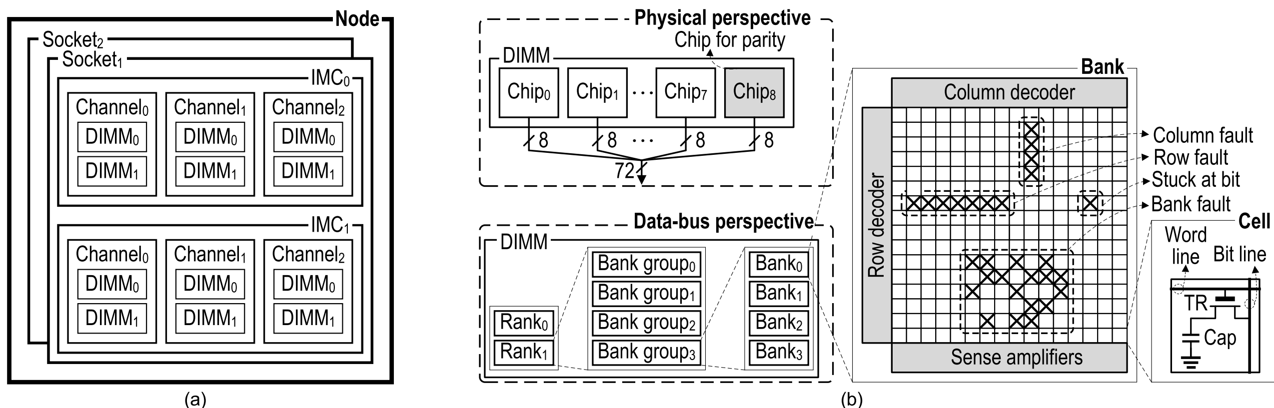

FIGURE 1. Basic memory structure (a) in a node and (b) in a DIMM.

RAS features are widely applied to most hardware in data centers. The rate of hardware failure depends on the type of data center workload. In particular, for a database and memcache, the failure rate of memories can be higher owing to the frequent access of storage devices [9]. Moreover, in the case of Web, Hadoop, and Ingest, the CPU performs massive calculation and requires frequent memory access. Therefore, the RAS in memory has played an increasingly important role for increase in capacity required by most workloads [10].

Errors in memory are divided into correctable errors (CEs), which can be corrected using an error correction code (ECC), and uncorrectable errors (UEs) [6]. Because of the significantly greater influence of UE on downtime than that of CE, various studies have been conducted for modifying or predicting UE [5], [7], [8]. Conventional methods determine UE if the CE in a single cell repeats at a certain rate or higher [10]. To achieve a higher detection rate of UE, certain patterns of CE have been utilized in several RAS techniques [11]. Because a large portion of UE in these methods is not a real UE, conventional RAS techniques cannot avoid memory loss due to the removal of memory access at the location of UE. Recently, to reduce unwanted memory loss, RAS techniques have adopted artificial intelligence in UE prediction at the expense of computational overhead [11]. To overcome this problem, on die ECC (OD-ECC) in conjunction with system ECC can be adopted for newly developed memories [12], [13], [14], [15]. ECC technology increases reliability by adding extra memory bits for error correction at the expense of overhead, which ensures data integrity and uptime.

Prior survey/review papers [16], [17], [18] regarding memories focus on architecture, neural network, and software, but none of prior works has comprehensively reviewed memory RAS, error correction techniques, and future challenge. As the size of data scale in data centers increases, it is essential to manage memory RAS techniques not only to reduce the probability to generate errors but also to lower the downtime cost. Therefore, this paper analyzes the RAS technologies used in real data centers of various companies including Intel, IBM, Dell, google, and so on and aims to provide guidance for future memory research directions based on existing studies. The rest of this paper is organized as follows: Section II provides a survey of the memories for datacenters as backgrounds. In Section III, we discuss about several error correction techniques. Section IV presents error prediction based on page offline. Section V discusses future work. Finally, Section VI concludes the study.

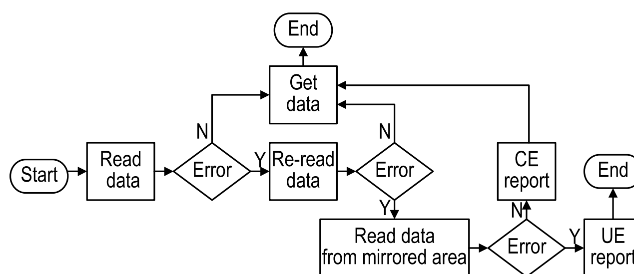

FIGURE 2. Error handling mechanism for server of a data center.

II. BACKGOURND

A. MEMORY STRUCTURE

Fig. 1(a) shows the basic memory structure in a node. A data center is composed of large number of nodes, where each node consists of two sockets for the two CPUs. Each CPU has two integrated memory controllers (IMCs) that manage data in and out of memory within multiple channels [19]. An IMC is also called a memory chip controller (MCC) or memory controller unit (MCU). A channel typically comprises up to two dual in line memory modules (DIMMs) and its advantages and disadvantages differ depending on the types of DIMM. DIMMs in laptops and PCs are unbuffered DIMMs (UDIMMs) [20] and can be expanded up to two ranks, where buffers and registers are excluded for fast response. Registered DIMMs (RDIMMs) are used in relatively small-scale servers [21]. By adding a buffer to memory, the address and command signals of the DIMM can be expanded to four ranks per DIMM. An ECC is embedded in the RDIMM. For relatively large-scale servers, an ECC can be embedded in a load-reduced DIMM (LRDIMM) with an isolated memory buffer to RDIMM [22]. Data are sequentially transferred through buffers without relying on rank. LRDIMM supports

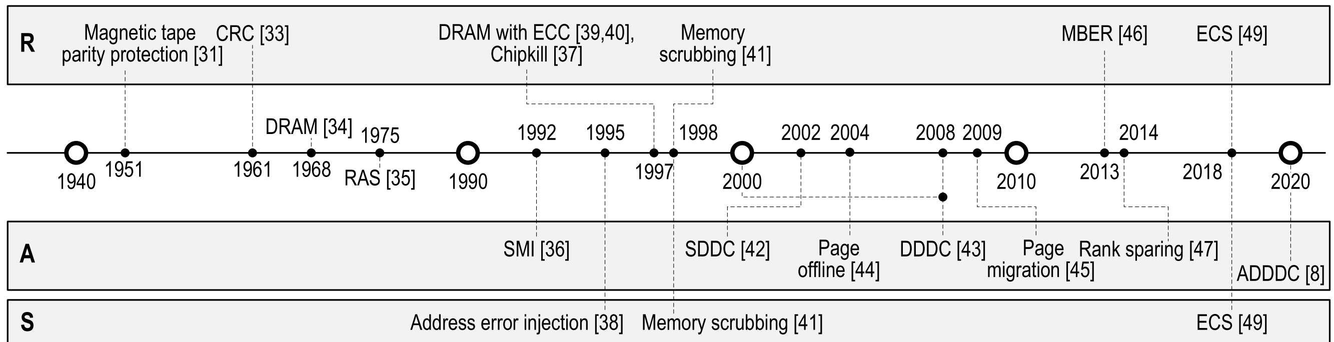

FIGURE 3. RAS feature time line from 1940 to 2020.

high-capacity memories that provide a fast response rate at the expense of a significant increase in cost.

The structure of a DIMM can be seen as the physical or the data-bus perspective view as depicted in Fig. 1(b). From the physical perspective view, a DIMM consists of eight DRAM chips and additional one for parity [23]. Each chip outputs 8-bit data, which makes total 72-bit data be one rank. Four banks compose one bank group for data-bus perspective, and four bank groups make one rank. The bank consists of cell arrays, row and column decoders, and sense amplifiers. The number of bits controlling the row or column address depends on each manufacturer [25].

Errors are classified as soft and hard [26]. Soft errors are caused by the collision of high-energy particles with the cosmic radiation, which leads to CEs at higher rate. Conversely, hard errors are UEs that keeps their values either '0' or '1' permanently. Errors are marked as '×' in the bank. If soft errors repeat frequently in a cell, they can be treated as a hard error. Faults represent a pattern on errors among cells and can be classified into row, column, bank, and stuck at the bit.

B. DATA CENTER ERROR HANDLING MEACHANISM

Fig. 2 shows the error-handling mechanism of a server in a data center [27]. When memory starts read, an error is checked. If an error exists, the same data is read again. If the re-checked data has an error, data without an error are obtained from the mirrored area. Otherwise, get data for the cell is processed. Once the mirrored data exhibits an error, a UE is reported without getting data. If no error is presented, a CE is reported while getting data. A conventional solution for UEs is sparing and migration at the page level or higher from mirrored area. Mirroring skill can improve a system fault tolerance and error correction capabilities.

When an error occurs, an embedded code is deployed to perform real-time initial correction. Once the correction is completed, the error is considered as a CE. When the same error repeats or leads to multi-bit errors, ECC generates logs for errors and patterns over time to predict future UE and addresses potential issues. This sequential approach is crucial for maintaining system stability and minimizing unexpected failures. The commonly used RAS technologies [29] address memory errors in the following methods:

- ECC detects and corrects errors in real-time.
- Predictive failure analysis monitors the health in memory modules, predicts potential failures, and enables proactive maintenance.
- Hot swapping and redundancy allow for the replacement of faulty memory modules while the system is running and maintains availability by automatically replacing failed modules.
- Memory mirroring and RAID provides fault tolerance by duplicating data across multiple memory modules and retrieving correct data if errors occur.
- Error logging and reporting record errors.

C. DEVELOPMENTS OF RAS

Fig. 3 shows development of RAS features from 1940 to 2020 [14], [28], [29]. Each solid line represents the year in which the technology was first released or documented in a paper, patent, white paper, manual, or other similar form of documentation.

In the 1940s, erroneous data caused by computers was studied at Bell Labs. Before an error occurs, three identical sets of data are stored. If an error does occur, two other sets are used for data recovery, resulting in 66% redundancy. To solve this problem, the concept of a Hamming code that can verify hundreds of data with just a few parity bits was introduced [30]. During this time, magnetic tapes were used to store parity in 1951 [31]. Since then, various codes have been used for transmission/reception as well. As an example, Reed-Solomon (RS) code [32] was used in cyclic redundancy check (CRC) [33]. When using the magnetic tape, the focus was on reducing device failure due to errors in memory. After IBM was granted the first DRAM patent in 1968 [34], techniques such as downtime and maintenance emerged as a key area of concern, and the concept of RAS was utilized in the 1970s [35]. The capacity of DRAM started with 1 Kbit in the 1970s, increased to megabit and gigabit in 1980s and 2000s, respectively, resulting in significant scale issues and multi-bit errors between multiple devices. Scalable memory interconnection (SMI) or scalable coherent interface (SCI) have been released to correspond to scale issues [36]. To resolve and simulate the multi-bit error, chipkill [37] and address error injection that determines faulty cells by injecting a virtual or real error

have emerged [38], respectively. Multi-bit error significantly depends on environmental conditions such as space [39], [40]. Memory scrubbing techniques for detecting multi-bit or UE in advance were used by scrubbing memory data [41]. In 1990s, availability and serviceability features attained more importance than the reliability technology owing to increase in UE or multi-bit error caused by high capacity of DIMM and accelerated performance of other devices [40].

Since 2000, most of RAS techniques have focused on availability. To handle errors at the device level, single device data correction (SDDC) [42] used by Intel in 2002 can reconstruct memory contents even if there are many errors throughout the chip. Double device data correction (DDDC) allows the memory DRAM device to continue to function in an event of a hard failure [43]. An improved version of DDDC was used as DDDC+1, which reduced costs due to UE by improving uptime [29]. Page offline was developed for UE prediction and widely adopted to handle the smallest unit of capacity from 1∼4 KB to be isolated [44]. From SDDC to page offline, RAS techniques adopt sparing and migration of a specific address [45]. RAS features in early 2010 included memory-based error recovery (MBER) [46] and rank sparing [47]. As data integrity and recovery of lost data have gained more importance, data recovery techniques similar to MBER have emerged for reliability. As data centers require higher memory usage, a tradeoff between memory loss and UE rate has been strongly related. Rank sparing and adaptive double DRAM device correction (ADDDC) [8] were developed to manage larger unit compared to a page. As many techniques related to availability have been released and used since 2000, the probability of normal operation in memory has significantly increased [48]. Since 2018, the initial diagnosis of the fault for reliability has gained importance because one fault can cause significant number of errors. To reduce the initial fault, error check and scrubbing (ECS) related technologies were emerged [49]. As the generation is replaced from DDR4 to DDR5, early diagnosis methods of faults have been researched extensively to integrate ECC into memories.

III. ERROR CORRECTION SKILLS

A. BASIC ERROR CORRECTION SKILLS

Error correction skill is used in server computers that deal with data corruption in all situations from computational science to financial computing. Several ECC algorithms exist, such as Hamming, RS [50], and Bose Chadhuri Hocquenghem (BCH) codes. This section describes various ECC skills based on ECC algorithms.

1) SINGLE ERROR CORRECTION DOUBLE ERROR DETECTION

The encoding process of single error correction double error detection (SECDED) is shown in Fig. 4(a) [51], [52]. P, D, and P' denote initial parity, data, and parity bit, respectively. This example is the encoding process of only 16-bit combining data and 5-bit parity, which equals to (16,11) as a cord word. Input is random data, and all parity bits, P'_0-P'_4, in the encoding process are 0. The encoding result for parity bits can be derived as follows:

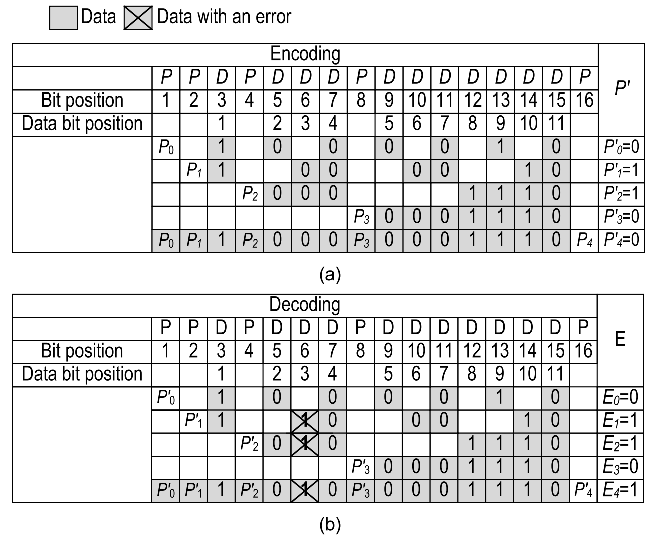

FIGURE 4. SECDED of (a) encoding and (b) decoding process with third data error.

$$P'_0 = P_0 \oplus D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7 \oplus D_9 \oplus D_{11}, \tag{1}$$

$$P'_1 = P_1 \oplus D_1 \oplus (D_3 \oplus D_4) \oplus (D_6 \oplus D_7) \oplus (D_{10} \oplus D_{11}), \tag{2}$$

$$P'_2 = P_2 \oplus (D_2 \oplus D_3 \oplus D_4) \oplus (D_8 \oplus D_9 \oplus D_{10} \oplus D_{11}), \tag{3}$$

$$P'_3 = P_3 \oplus (D_5 \oplus \cdots \oplus D_{11}), \tag{4}$$

$$P'_4 = (P_0 \oplus \cdots \oplus P_4) \oplus (D_1 \oplus \cdots \oplus D_{11}), \tag{5}$$

where P_0, P_1, P_2, P_3, and P_4 are initially set to 0. If the sum of each row data bits is even, the dedicated parity bit becomes 0. If not, the parity bit is set to 1. As a result of this encoding, $P'_4P'_3P'_2P'_1P'_0$ is 01100 with the data value of Fig. 4(a). Fig. 4(b) shows the situation where the third data bit is corrupted to 1, which makes E_1, E_2, and E_4 to be 1. SECDED is used with 64-bit data and 8-bit parity, (72, 64), resulting in 12.5% redundancy in DRAM. The redundancy can be reduced by increasing parity or data bit.

2) SINGLE SYMBOL CORRECTION DOUBLE SYMBOL DETECTION

Single symbol correction double symbol detection (SSCDSD) is a correction method that uses 4–8 bits as a symbol [53]. In most cases, all bits of a symbol can be modified using SSCDSD. Error correction codes based on the SSCDSD can provide up to 42 times better error avoidance than SECDED [54]. Fig. 5(a) shows the implementation of SSCDSD. The code word with 18 symbols, (144, 128), includes 128-bit data and 16-bit ECC with 8-bit CRC and 8-bit parity [55], [56]. Four code words are composed of one cache line. The SSCDSD can correct multiple bit errors within a single symbol, regardless of the error pattern within an 8-bit symbol, as shown in Fig. 5(b). However, if an error occurs across two symbols, it can be detected as a UE symbol, similar to

TABLE 1. Type of CDC depending on polynomial.

Name	Polynomial	Application
CRC-8	$x^8 + x^2 + x + 1$	ATM
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC

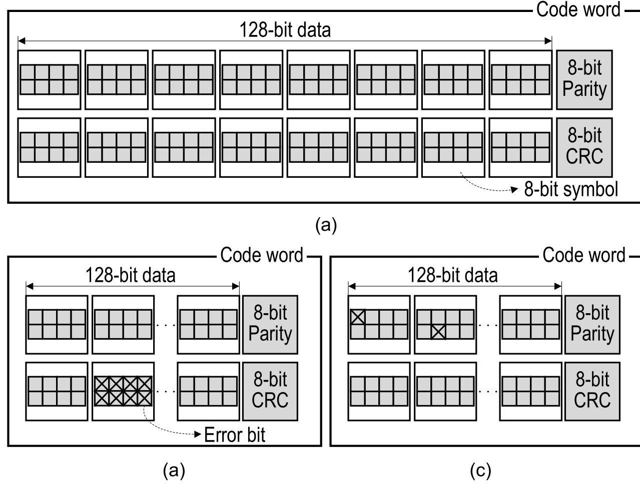

FIGURE 5. Examples of SSCDSD for (a) symbolic construction, (b) single symbolic error, and (c) multiple symbolic error [53].

SECDED. SSCDSD can also divide the data and parity by 4-bit symbol units according to the type of DRAM. The ECC type using this 4-bit symbol unit is called chipkill correction [57]. The SSCDSD depends on the CPU architecture and generation, resulting in different error-correction coverage and memory-configuration requirements to enable advanced ECC.

3) CYCLIC REDUNDANCY CHECK

CRC detects accidental errors in computer data and is typically used in digital code transmitters/receivers and storage devices [55], [56]. Various types of CRC depending on the divisor or polynomial generator exist. Table 1 shows the type of CRC depending to the order of polynomial expressed as CRC-n, where n is the highest power of the polynomial. A simple and easy way to understand CRC process is shown in Fig. 6. Fig. 6(a) shows the process of CRC transmitter, transmission, and receiver [58], [59]. The remainder of the transmitter is created by dividing the data into polynomial bits. Subsequently, the bits of the data and remainder are combined and transmitted to the receiver. The result is obtained by dividing the combined bits into the polynomial bits. No error is generated if all bits of the result are 0. Figs. 6(b) and 6(c) show examples of encoding and decoding, respectively. Polynomial p(x) can be expressed as follows:

$$p(x) = x^3 + x + 1 \rightarrow 1011. \tag{6}$$

Transmitter and receiver use polynomial bit of 1011 in (6). As the encoding process, data 1001 are divided into polynomial bits, resulting in the remaining bits 110. The combined data 1001110 are transmitted to the receiver, and the data are again divided by 1011. If all bits of the result of the division are 0, the data are accepted. Hardware implementation can be configured with shift registers and XOR gates for division and data shifting [60].

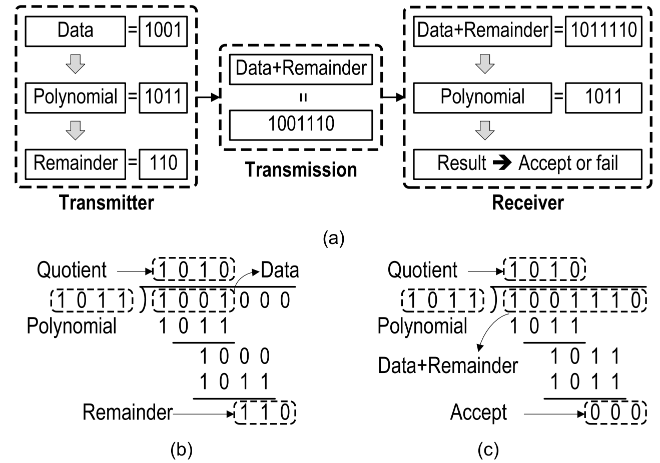

FIGURE 6. (a) Process in transmitter, transmission, and receiver of CRC and examples of (b) encoding and (c) decoding [58], [59].

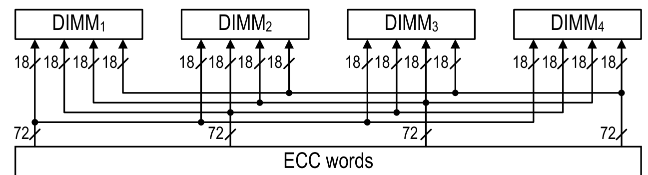

FIGURE 7. Example of chipkill correction method [61].

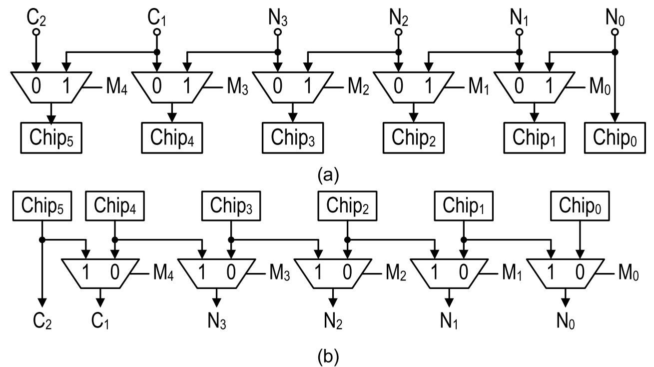

FIGURE 8. Chipkill mask method of (a) write-shift and (b) read-shift logics [62].

4) CHIPKILL CORRECTION

Because SECDED corrects one-bit error and detects two-bit error, multiple-bit errors cannot be tolerated. To overcome this problem, a correction method chipkill can be applied [61], [62]. Fig. 7 shows four 72-bit ECC words are distributed to four DIMMs evenly. Because SECDED codes are deployed simultaneously for four different words, the maximum four CEs can be corrected at a time. For the chipkill operation, the server must be simultaneously able to use four DIMMs, and the code word (72, 64), resulting in 12.5% redundancy at least.

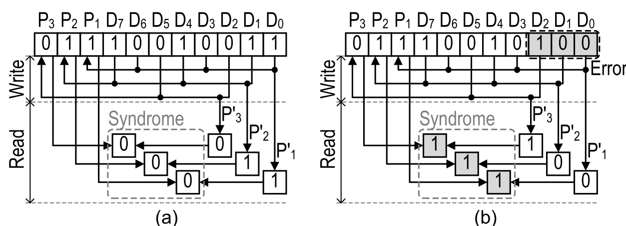

FIGURE 9. Interleaved parities and syndrome generation process (a) without errors and (b) with errors during write and read.

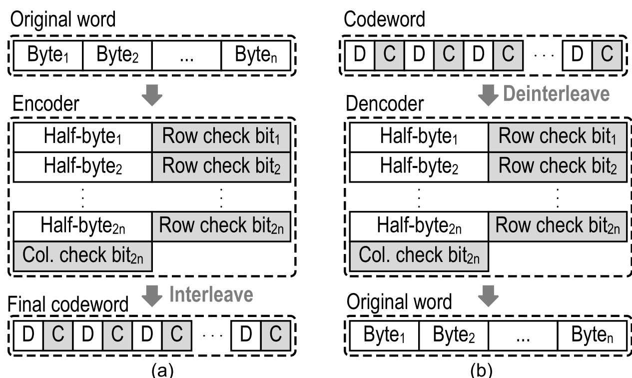

FIGURE 10. Multi-bit error correction of (a) encoding and (b) decoding process for byte original word.

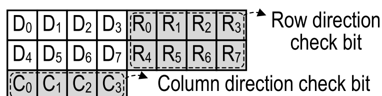

FIGURE 11. A method to correct multi-bit errors when row and column check bits are generated.

Fig. 8 shows a simplified example of using a write- and read-shift logics to mask a non-operating chip [62]. In Fig. 8(a), N_0, N_1, N_2, and N_3 denote the nibbles corresponding to each chip, and C_1 and C_2 denote the correction and detection nibbles, respectively. M_0–M_4 are the masking bits. If the chip operates normally, all masking bits are 0; otherwise, one of the register records 1 to alarm a non-operating chip. Assuming that $chip_2$ is not operating correctly, $M_4M_3M_2M_1M_0$ needs to be 11100 for bypassing $chip_2$ at the expense of losing the detection nibble C_2. Subsequently, in Fig. 8(b), the same masking bits are used as the read process. In the real application, N_0–N_{31} are the nibble bits, and C_0–C_2 are correction bits with C_3 a detection nibble.

B. ADVANCED ERROR CORRECTION SKILLS

In order to increase the capacity and the bandwidth of DRAM, the die-stacked structure is widely used. Furthermore, to minimize the impact on latency during error correction, there is a trend to implement ECC skills within DRAM chip to address multi-bit errors. However, the implementation of robust ECC for multi-bit error correction faces significant challenges due to limited area, latency, and power constraints in memory. For example, while chipkill is widely used in server environments to effectively address single or multiple errors in DRAM, its application in high-capacity memory architectures is limited due to latency inefficiencies [63]. This section provides detailed explanations of advanced error correction techniques capable of addressing and detecting multi-bit errors.

1) MULTI-BIT ERROR CORRECITON AND DETECTION SKILLS

The codes that have been used traditionally were mainly capable of correcting single error and, in some cases, detecting double errors. This is because soft errors typically only affect a single bit. However, as technology scaled, the probability of soft errors affecting more than single bit significantly increased. As a result, single error correction became insufficient. Generally, multiple errors are located in close proximity to each other. To address these, there is a method called SEC-DAEC (Single Error Correction-Double Adjacent Error Correction) that can correct up to two adjacent errors [64], [65], [66]. As one of the SEC-DAEC skills, the interleaved-parity (IP) method can be utilized. Fig. 9(a) shows the process of generating parities and syndrome during write and read operations without errors. In SECDED, parities are created during the write process and the syndrome is used to determine the presence of errors during the read process. On the other hand, parities for IP are generated separately during write and read operations. The generated parities are calculated for XOR to compute the syndrome. For example, in the case of data D_0, D_1, and D_2 without errors, parity bits P_1, P_2, and P_3 are generated during write. During read, additional parity bits P'_1, P'_2, and P'_3 are generated for the corresponding data. Syndrome is calculated by performing XOR calculations on the parity bits generated during the write and read processes. The result of obtaining 000 in this case indicates the absence of errors. If errors exist in D_0, D_1, and D_2 in Fig. 9(b), P'_1, P'_2, and P'_3 are generated differently from the write process. By obtaining a result of 111, the syndrome calculation can be used to identify the position of the data and perform error correction.

SEC-DAEC is a technique capable of correcting up to two adjacent errors, whereas Single Error Correction-Double Error Correction-Triple Adjacent Error Correction (SEC-DEC-TAEC) enhances this capability by correcting two errors and also three adjacent errors. In other words, SEC-DEC-TAEC can handle a greater number of bit errors and provides more robust error correction functionality. Thus, SEC-DEC-TAEC offers higher reliability and error correction capability than SEC-DAEC [67], [68].

Fig. 10 shows the process of generating a specific check bit for an original word in bytes using SEC-DEC-TAEC. During the encoding process in Fig. 10(a), original word, which consists of n bytes, is divided into 2n half-bytes. For each half-byte, row check bits are generated in the row direction, and column check bits are generated in the column direction. During the interleaving process, a data D and check bit C are rearranged in a crossover pattern, leading to the formation of the final codeword. During the decoding process, the final codeword generated in the encoding process is deinterleaved to arrange it into n half bytes and n check bits.

TABLE 2. Comparison of ECC skills.

Parameter		SEC [75]	SECDED [76]	DEC [77]	DEC [78]	TAEC [67]	SSEC [79]
Code		Hamming	Hamming	Spotty	Ordinary least squares	Hsiao code	RS
Technology [nm]		16	65	65	65	180	65
Number of data		64	16	16	16	16	16
Number of parity		7	3	7	12	12~24	10
Area [μm^2]	Encoder	70.8	45.2	152.4	64.0	958.0	514.0
	Decoder	148.2	83.6	442.0	1004.4	2874.0	6094.4
Power [mW]	Encoder	-	0.03	0.11	0.04	0.08	0.35
	Decoder	-	0.07	0.39	1.29	0.15	5.02
Delay [ns]	Encoder	-	0.25	0.39	0.22	0.32	3.16
	Decoder	-	0.45	0.76	0.94	1.25	5.88

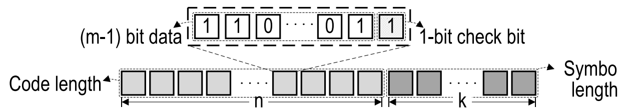

FIGURE 12. Symbol and bit arrangement in SPC-RS.

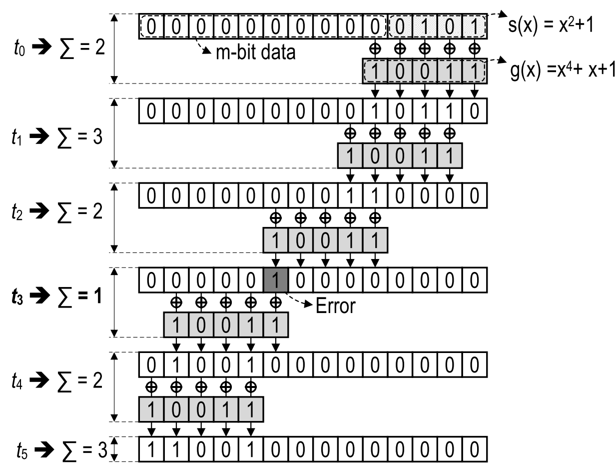

FIGURE 13. Single-error search of CRC where g(x) = x^4+x+1 with a syndrome s(x) = x^2+1 [70].

Subsequently, the decoding process is applied for the original word.

Fig. 11 shows the process of correcting multi-bit errors using the codeword generated in Fig. 10. D_0 to D_7 represent each half byte, and R_0 to R_3 are the check bits in the first row, while R_4 to R_7 are the check bits in the second row used to detect errors in the generated rows. Similarly, in the column direction, check bits C_0 to C_3 are generated vertically. By utilizing the generated check bits, the exact positions of each data can be determined, enabling the correction of errors up to 2 bits. The correction of 3-bit errors is only performed if they are adjacent bits. This method generates one check bit for each half byte, leading to over three times the utilization of parity bits when compared to SECDED. As a result, SEC-DEC-TAEC consumes greater power, area, and delay.

2) ADVANCED CRC

Recent researches in CRC have been focusing on transforming specific bits into symbols to enable single symbol

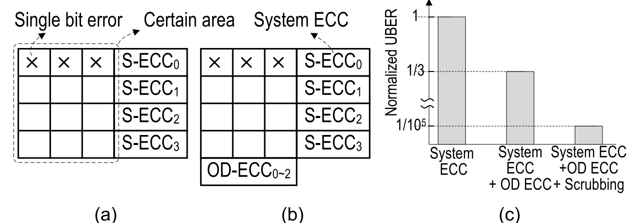

FIGURE 14. DDR5 memory: (a) structure of system ECC and OD-ECC, (b) UBER comparison when on-die scrubbing is adopted.

correction (SSC) [69]. This method combines the use of Single Parity Check (SPC) and RS code as SPC-RS code in Fig. 12. This code consists of n code lengths and k symbols, with each symbol comprising m data bits. One of these symbols is utilized as a check bit to determine the presence of errors in each symbol. Another method for error correction in CRC without the use of Look-up tables exists. Fig. 13 shows the process of single error search using a generator polynomial $g(x) = x^4 + x + 1$ when the syndrome $s(x) = x^2 + 1$ is generated [70]. By performing XOR calculations with the combined result of m-bit data and $s(x)$ using $g(x)$ from t_0 to t_5, the presence of errors is determined by the resulting value of the summation symbol $\sum$ representing the sum of all data. This allows for the detection of single error. For example, by detecting that the value of $\sum$ is greater than or equal to 2 in t_0, t_1, t_2, t_4, and t_5, but only 1 in t_3, it is possible to identify a single error. Additionally, by expanding the series of $g(x)$, double error correction can also be achieved. In addition to this method, there are other approaches such as modifying the existing forms of $g(x)$ and $s(x)$ to reduce the latency in the error correction process or attempting to correct errors of more than 2-bits [71], [72].

3) ON-DIE ECC

Fig. 14(a) shows a multi-bit error for the structure of DDR5 memory having four system ECCs and three OD-ECCs with a hamming code of (72,64) and (104,96), respectively [73], [74]. By simultaneously applying the system ECCs and OD-ECCs, the uncorrectable bit error rate (UBER) can be decreased to one third. For example, if 3-bit error occurs in a memory without OD-ECC as depicted in Fig. 14(a),

it is treated as a UE. However, with OD-ECCs orthogonal to system ECCs in Fig. 14(b), these 3-bit error can be corrected by each OD-ECC. Thus, UBER can be reduced as the number of OD-ECCs increases at the expense of additional chip area. Fig. 14(c) shows a simulated comparison of system ECC, OD-ECC, and scrubbing. System ECCs in conjunction with OD-ECCs reduces UBER to one third. The UBER is reduced to 10^{-5} with all three techniques combined, which leads to significant increase in reliability.

4) ECC COMPARISON

Table 2 shows a comparison of ECC techniques. This comparison is based on [64] and [67] and compared the required parity, area, power, and delay for 16 or 64-bit data. Single symbol error correction (SSEC) [79] uses symbol correction code to correct multi-bit errors. As the number of errors to be corrected increases, the overhead of the encoder and decoder also increases. Therefore, it is important to carefully consider the form of ECC that can be implemented in memory.

IV. ERROR PREDICTION

In general, errors are logged into a file via the Mcelog Linux kernel module [10], [80], [81], and the following contents are also stored to the log file.

1) Time stamp
2) Physical address
3) Server name
4) Sockets, channels, and banks where physical addresses are located.
5) Memory access, such as reading or writing, is performed when an error occurs.

UE causes a higher system crash rate, in general, and thus the prediction of UE with a CE had been widely adopted in the 2010s. This method was based on historical CE statistics: when the CE exceeds a certain number for a certain period, a certain capacity can be replaced before the UE occurs. In addition, statistical information was used in failure models or rates for analyzing the relationship between devices and environmental conditions or determining whether UE predictions were appropriate for low- or high-end servers [10], [82], [83], [84].

Several solutions are available for preventing predicted UE from occurring. For example, cache lines were controlled to remove errors by isolating 8 KB capacity or less at the expense of system overhead [85], [86], [87], [88]. However, page offline can remove UEs more than 94% by isolating 4 KB capacity without additional hardware expense [27], [89]. Since a page is the minimum capacity that can be controlled by the system, page offline is a method to minimize memory capacity waste.

Fig. 15 shows the conventional page offline policy. Initially, at the operating system (OS) level, a threshold for the number of CE is set to X for a given time window T. When an error occurs, the address for the error is obtained, and the number of repetitions is counted. If the error count exceeds

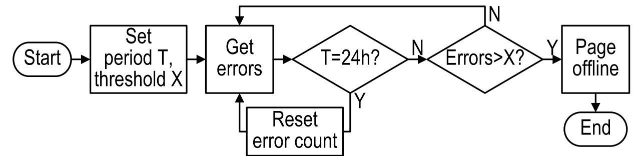

FIGURE 15. Conventional policy for page offline.

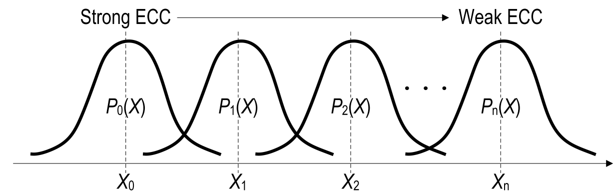

FIGURE 16. *X/T* control to determine weak or strong ECC.

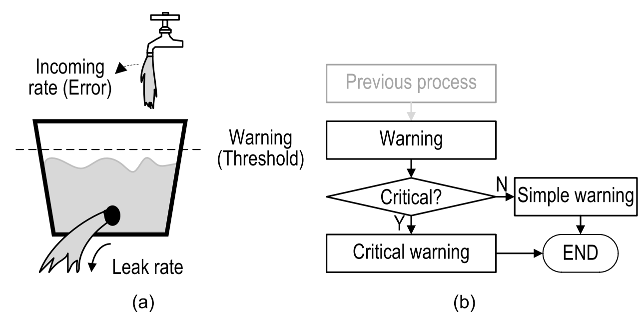

FIGURE 17. Leaky bucket (a) concept and (b) flow chart.

X in T, the page is not isolated. If not, the error counting becomes reset. X can be set differently depending on the device. For example, the degree of wear differs depending on the usage and reliability of DIMM. Therefore, the probability of failure of a DIMM goes higher as time goes on, which requires to control X for the optimum solution [90]. Fig. 16 shows the control of the strong and weak ECCs by adjusting X [91], where $P_n(X)$ is a probability density function (PDF) of error correction capability corresponding to X [92].

One method for determining X is the leaky bucket algorithm [93]. This algorithm is used to warn the system through a trigger when a certain threshold exceeds X in parts, such as UE, DIMM, Socket, page, and CE. Fig. 17(a) shows a basic concept of a leaky bucket. The incoming rate represents a water fills up rate in a certain bucket, and the water leak rate varies by the pressure to the hole. When the amount of water filling is higher than the amount of water leaked, a warning is issued, which follows the same principle as the error exceeding X. The algorithm completes after making decision for simple or critical waring in Fig. 17(b). Depending on warning level, the size of offline can be determined.

Fig. 18 shows a simple leaky bucket algorithm based on the warning level and leaky rate. In Fig. 18(a), the warning level and the leak rate are three and two, respectively. A leak rate of two indicates that the error count maintains its value during the two consecutive time sequence if no error presents. Fig. 18(b) depicts the values of count and warning for the given warning level and leak rate of three and four,

Warning level = 3, Leak rate =2			
Sequence	Error	Count	Warning
1	1	1	0
2	1	2	0
3	0	2	0
4	0	1	0
5	1	2	0
6	0	2	0
7	0	1	0
8	0	1	0
9	1	2	0

(a)

Warning level = 3, Leak rate =4			
Sequence	Error	Count	Warning
1	1	1	0
2	1	2	0
3	0	2	0
4	0	2	0
5	1	**3**	**1**
6	0	**3**	**1**
7	0	**3**	**1**
8	0	**3**	**1**
9	1	**4**	**2**

(b)

FIGURE 18. Example of simple leaky bucket algorithm according to warning and leaky rate: (a) Warning = 3, leak rate = 2, and (b) Warning = 3, leak rate = 4.

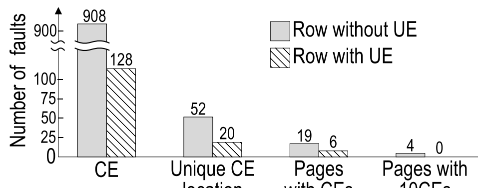

FIGURE 19. CE and UE associations for the other 2-rows from 2-servers [92].

respectively, which shows that the error count cannot quickly leak compared to the case of Fig. 18(a).

A higher leak rate can be applied to an older device that has not been replaced recently. Depending on the waring level, the mean cost to recovery (MCTR) was analyzed [94]. Random forest (RF) with deep learning performs the best in determining the optimized warning level. If the warning level is normalized to its maximum value up to 1.

Even though the number of CE works as an important role in predicting the UE for page offline, the counting of CEs within a page makes page-to-page faults prediction be difficult due to inter-page circuit connection. Fig. 19 shows the CE and UE associations by observing two different rows from two different servers [95]. A row without UE has a higher CE rate than a row with UE. For example, the number of CE, unique CE location, pages with CEs, and pages with 10 CEs for the row without UE are 908, 52, 19, and 4 respectively. These numbers indicate that CE is not related to UE. Because UE prediction based on the number of CE in a page cannot be applied based on the observation in Fig. 19, the method for predicting UE by setting a specific CE pattern has been researched as shown in Fig. 20 [11], [96], [97]. For the column-based UE prediction in Fig. 20(a), the possibility of UE occurrence is determined when a pattern is determined by the length of the column pattern, l_c. When the pattern repeats, the column fault is declared. Similar to the column fault, the row- and bank-based UE prediction of Fig. 20(b) are evaluated by the length of the row pattern, l_r, and the given area $l_c \times l_r$, respectively.

A. COLUMN-PATTERN-BASED UE PREDICTION

The UE prediction can be performed using column faults because the UE occurs more frequently and is scattered across

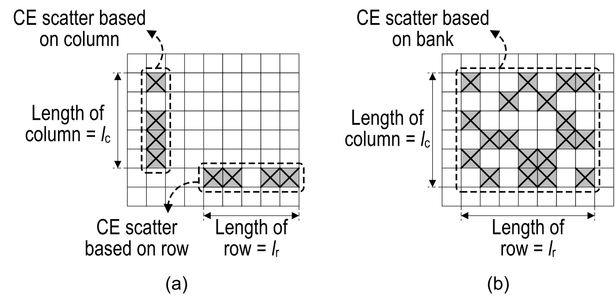

FIGURE 20. (a) Column and row and (b) bank prediction according to error pattern [11], [96], [97].

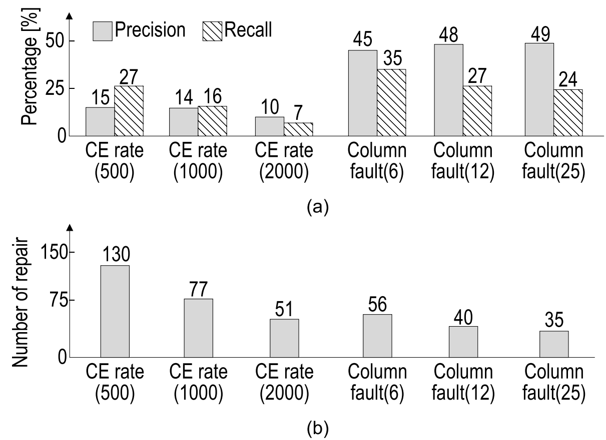

FIGURE 21. (a) Precision, recall, and (b) the number of repair values for CE rate and column fault [98].

the column [98]. When a signal is transmitted to the gate of a transistor, the effect is insignificant. However, data move along the bit lines connected to either a source or a drain. Moreover, the charge flow is interrupted owing to the resistance or capacitance of the bit line, which causes a higher error rate. For high-density capacity memory, the problem can be more significant owing to the thin bit line and many surrounding signal lines [99], [100].

The conventional prediction method based on column faults [98] was evaluated using two metrics: precision and recall as

$$Precision = \frac{TP}{TP + FP}, \tag{7}$$

$$Recall = \frac{TP}{TP + FN}, \tag{8}$$

where TP, FP, and FN are the true positive, false positive, and false negative, respectively. TP and FP indicate the cases that UE does and does not occur, respectively, when UE is predicted. FN indicates that no prediction is performed when UE occurs. Therefore, precision refers to how good the UE prediction is when the prediction is made. Low precision causes unnecessary memory isolation and thus results in memory loss. Recall indicates the ratio between predicted UE and the total occurrence of UE. A low recall indicates that many UEs still exist on the server and suffer from crashes because the UEs cannot be distinguished.

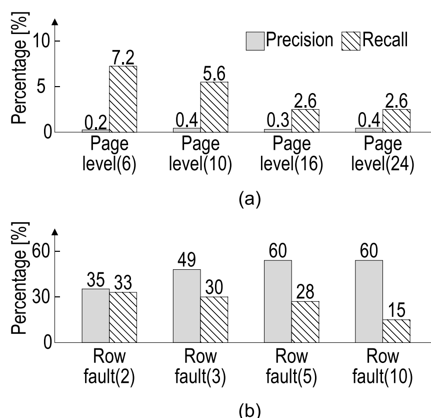

FIGURE 22. UE prediction comparison in terms of precision and recall between (a) page level and (b) row fault [95].

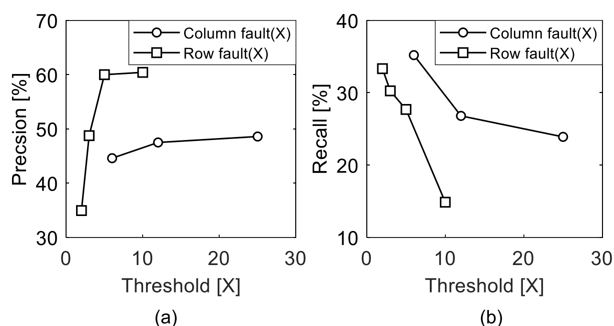

FIGURE 23. Comparison in terms of (a) precision and (b) recall of row and column fault for UE prediction [95], [98].

Fig. 21 shows the precision, recall, and number of repair values for CE rate and column fault. On the x-axis, X in CE rate(X) and column fault(X) are the number of repetitions, which are the threshold values. For the page offline based on CE rate, the best performance in precision and recall is CE rate (500), as shown in Fig. 21 (a). For the CE rate (2000), the precision and recall were 10 and 7%, respectively. When the page offline was performed owing to a column fault, precision, recall, and repair were better than the CE rate. At the lowest X value of 6, precision and recall were 45 and 35%, respectively. However, a low X can result in a higher repair, as shown in Fig. 21(b), which results in higher maintenance or replacement costs for the waste of DIMMs. Compared to the high repair value due to the low CE rate(X), even with the low threshold, the repair due to column faults was much lower. When X had the highest value of 25, the performance was better than the CE rate in terms of precision, recall, and repair. As a result, UE prediction using a column fault, rather than the CE rate, is much better [98].

B. ROW-PATTERN-BASED UE PREDICTION

In Fig. 22, a row predictor, which analyzes CE spread in the row of a specific length l_r, is used for the page offline method [95]. Similar to the column predictor shown above, if the number of fault repetitions of a specific X exceeds a row fault (X), the error can be treated as UE. Fig. 22(a) shows that the page is offline when X was set to 6, 10, 16, and 24. The highest values for precision and recall were 7.2 and 0.4%, respectively. When using the row fault, as shown in Fig. 22(b), the highest precision and recall values were 60 and 33%, respectively. Eventually, a specific pattern of rows and columns is better for precision and recall than using CE rate for predicting UE.

Fig. 23 shows a comparison of the precision and recall of the row- and column-fault UE prediction. In both methods, the precision tends to increase and recall decreases with an increase in the threshold. The row fault had a higher precision value than the column. Therefore, row faults are advantageous for increasing the accuracy of UE prediction. Conversely, the columns are more advantageous in terms of recall. The results obtained may depend on various parameters including environment, state, and workload of a DIMM.

C. HYBRIDE UE PREDICTION

As can be observed above, column and row type errors appear frequently, 15–30% of errors occur in the column/row, and 40% of them are repeated [101]. However, determining UE with only one predictor cannot guarantee high precision because errors occur randomly in DRAM based on the workload or data type. For a specific ECC skill, UE prediction is possible in the row or column using a pattern of error correction or detection. In [102], temporal and spatial localities were used instead. Temporal locality is a characteristic with a high possibility of referencing recently used data. At the moment of an additional symbol error presents after the first symbol error is observed, the probability of the chipkill triggering within 1 min is more than 90%. The timing of the UE is predicted by associating the temporal characteristics with the UE. Spatial locality is a characteristic in which adjacent data points are highly likely to be referenced. When an error occurs in a cell or specific area, the surrounding error is examined. Using these two characteristics, the prediction coverage can be increased by up to 80% and repeated errors can be prevented by 76%. This means that 63% of the failures can be prevented, but chipkill requires 38% more overhead than SECDED [103].

A combination of various types of predictors can be used to solve this problem. In [104], deep learning can be used for multiple predictors to be combined as a hybrid including logistic regression (LR) models [105], support vector machines (SVM) [106], classification and regression trees (CART) [107], backward propagation (BP) [108], gradient boosting decision trees (GBDT) [109], RF [110], [111], and extreme gradient boosting (XGBoost) [109]. Recently, boost-related deep learning has been widely used for UE prediction [11], [112]. In [11], column, row, and bank predictors were used as hybrid predictors at appropriate ratios of 0.474, 0.246, and 0.219, respectively.

D. PERFORMANCE EVALUATION OF PAGE OFFLINE

Fig. 24 shows an analysis of various types of predictors [11], where data obtained from four different servers. The results

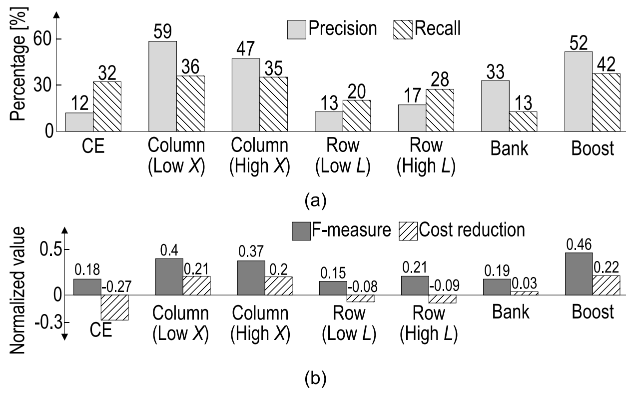

FIGURE 24. Performance evaluation of predictors: (a) precision, recall, (b) F-measure, and cost reduction [11].

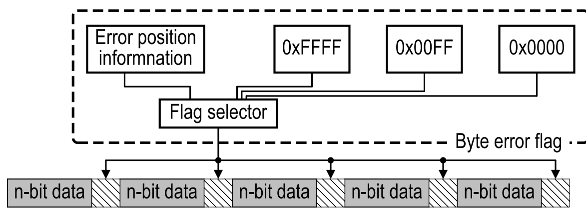

FIGURE 25. Hardware block diagram of on-byte-error-tracking [113].

were described in terms of *precision*, *recall*, *F-measure*, and cost reduction. Two figures indicate that the hybrid method, boost, tends to outperform other single pattern predictor. For the lower value of X, better performances can be expected compared to higher value of X. Similarly, the longer the length is, the better the performances are. Fig. 24(b) shows the analysis of the predictors in terms of the *F-measure* and cost reduction. The *F-measure* is the balance score between *precision* and *recall* and can be obtained as follows:

$$F - measure = \frac{2 \cdot precision \cdot recall}{precision + recall}. \quad (9)$$

A high F-measure can be obtained with a high percentage of precision and recall. For example, in the case of CE, the F-measure can be low even though the recall is relatively high due to low precision. However, in the case of row pattern presented with high L, the F-measure is higher than CE because both precision and recall are relatively well balanced. For cost reduction, the cost (>0) can be reduced by predicting UE correctly, and a negative value indicates more loss through UE prediction. Cost reduction can be expressed as follows:

$$Costreduction = \frac{C_c \cdot TP - C_r \cdot FP - C_m(TP + FP)}{C_c\,(TP + FN) + C_r(TP + FN)}, \quad (10)$$

where C_{c} is the unit cost incurred by the UE, C_{r} is the unit cost of DIMM replacement, and C_{m} is the unit cost incurred due to migration. (10) is a value for determining whether UE prediction is beneficial in terms of cost reduction. In the case of CE in Fig. 24(b), because the cost caused by frequent replacement or migration is larger than the cost that is beneficial for UE prediction, the cost reduction is negative. Except for Boost, the column showed the best performance.

E. ERROR CHECK AND SCRUB

The early diagnosis and identification of memory faults can reduce the cost of UEs. Many companies have intensively researched methods without consuming hardware areas. Conversely, in academia, many studies have been conducted on methods of accurately predicting a UE even if the hardware area and power are consumed. For example, Fig. 25 shows a hardware block diagram of OBET, in which on-byte-error-tracking (OBET) technology has an error flag that indicates existence of a fault between specific data units at the expense of 1.6 and 3% of area and power, respectively [113]. The byte error flag has three states: 0, no error, 0×00FF, that is multiple errors, and 0xFFFF, that is parity error. The remainder of the situation can be a single-bit error or other errors. The greatest benefit of adding flags between data is scrubbing. In conventional ECS methods, the entire specific data are scrubbed to detect CE or UE in memory at larger power or longer latency in extracting errors. Because OBET uses selective scrubbing by adding flags and utilizing the information, less overhead and power can be expected. As a result, OBET is useful for detecting multiple errors or UEs rather than a single error. Techniques similar to OBET include DUO [114] and XED [115]. DUO uses RS code for small redundancies but with long latency. XED is beneficial for handling multi-bit errors rather than simple single-bit correction by storing catchwords with error information in the 9th chip that stores parity. XED also needs to use a small hardware area because of the use of a register.

F. COST ANALYSIS

The cost of running a datacenter is represented by total cost ownership (TCO) with many factors varying under different situation. In this section, the relationship between UE prediction and cost saving is investigated followed by methods for reducing downtime costs through accurate UE prediction.

1) TOTAL COST OWNERSHIP

At a large scale, such as a datacenter, TCO can be determined by the number of parameters: the five models in [116], [117], [118], and [119] define TCO as:

$$TCO = C_{infra} + C_{server} + C_{network} + C_{power} + C_{main}, \quad (11)$$

where C_{infra} is the infrastructure, C_{server} server acquisition, C_{network} network acquisition, C_{power} power, and C_{main} maintenance costs. The TCO varies largely based on failure rates or environment in which servers are operated. Eight models can be rarely used due to large dependency of environment, where parameters include energy, DIMM cost, DRAM fault in time (FIT), availability/mean time to failure (MTTF), silent data corruption (SDC) derating, performance, and thermal [103].

2) PREDICTION COST EVALUATION

If the error data according to the server cannot be identified, the efficiency of UE prediction is significantly reduced. To express the efficiency of UE prediction in a sever, (10) can be re-written in the form of *prediction P* and *recall R* as follows:

$$Cost\ reduction = \left[1 - \frac{C_m + C_r}{(C_c + C_r)\,P}\right] R. \tag{12}$$

Similar to (10), P can determine whether UE prediction is beneficial. If $P > (C_m + C_r) / (C_c + C_r)$, the cost reduction becomes positive. Commonly, simultaneous optimization of precision and recall can hardly be achieved [11]. According to IBM, the cost of server downtime per minute is over \$167, which is more than the cost of repairing a DIMM [48]. Therefore, UE prediction with low accuracy needs to be avoided because the inaccurate downtime cost causes lower or negative cost reduction. In hard disks, when UE prediction is inaccurate, more replacement can be considered instead of the accuracy of prediction by further investing in MCTR [994], where replacing both failed and healthy devices simultaneously as:

$$MCTR = FN \times n_F + FP \times n_H, \tag{13}$$

where n_F and n_H are the numbers of failed and healthy devices, respectively. To express the cost and time for recovery, the mean time to repair (MTTR) is widely used:

$$MTTR = 0.1 + \frac{0.9FN}{P}. \tag{14}$$

Putting (13) into (14) gives

$$MTTR = 0.1 + \frac{0.9(MCTR - FP{\cdot}n_H)}{P \cdot n_F}. \tag{15}$$

Because MCTR is linear to MTTR, their optimization cannot be easily accomplished. When UE precision P is low, both MCTR and MTTR increase. As a result, a method either to increase P or to replace devices rapidly requires to be conducted.

V. FUTURE WORK

Regarding the recently observed bit error rate in studies [120], [121], SDC occurrences can be estimated at around every 300,000 SECDED decoding cycles in a single DRAM chip system incorporating on-die (136,128) SEC and system SECDED (72,64). SDC occurs when two data bits are simultaneously corrupted and are treated as a single data. Fig. 26 shows the situation where SDC occurs in the on-die ECC and DRAM controller. If D_0 and D_1 are recognized as a single error data and D_3 is incorrectly corrected in the on-die ECC, the DRAM controller also misinterprets it and erroneously modifies the i-th data, D_i, resulting in SDC.

High bandwidth memory 2 extension (HBM2E) [73] applied SECDED simultaneously to OD-ECC and the system ECC in Fig. 27. Double errors that cannot be resolved by System ECC_1 can be prevented by OD-ECC1,2 in Fig. 27(a). However, if there are two in the OD-ECC 0 region, and these errors are included in the System ECC1 region, they are treated as UE in Fig. 27(b). This problem can be alleviated by using SSCDSD in HBM3 [122]. However, similar to SECDED, double symbol errors occur, and there are cases where neither System ECC nor On-die ECC can correct them. As a result, research need to focus on minimizing UE through the application of existing ECC technologies or advanced techniques in On-die ECC.

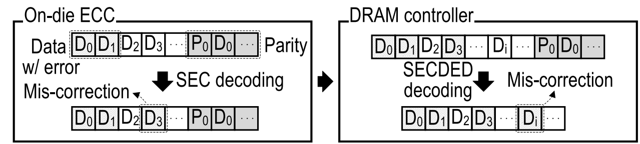

FIGURE 26. Process of generating SDC in on-die ECC and DRAM controller.

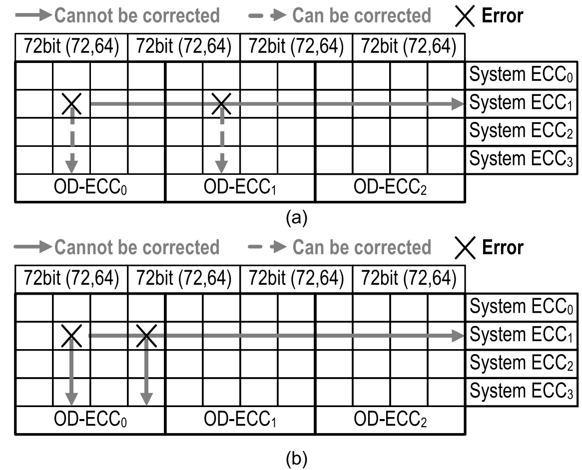

FIGURE 27. In HBM2E on-die ECC, (a) errors can be corrected and (b) cannot be corrected.

On-die ECC with SECDED can provide an advantage in terms of latency compared to SSCDSD, but it may have a disadvantage in terms of multi-bit error correction. Therefore, combining suitable ECC techniques for multi-error correction and appropriate ECS techniques can greatly enhance RAS, emphasizing the need for relevant research in future memory technologies.

VI. CONCLUSION

This paper provides an overview of the RAS features used in data centers and introduces various methods for improving memory RAS by referencing research papers, patents, and websites. Conventional ECC techniques have been widely discussed for lowering the probability of error generation. Further reduction in UEs can be accomplished with page offline methods in a system level to avoid system faults or damages at the expense of an extra hardware. Recently, starting from DDR5, on-die ECC effectively decreases the error rate by several orders of magnitude within a memory chip. However, further advancements are needed to address downtime costs in on-die ECC techniques, including the mitigation of SDC and improvements in hardware efficiency, power consumption, and latency.

REFERENCES

[1] *R. Dell PowerEdge R930 RAS [White Paper]*, Dell Technol., Round Rock, TX, USA, 2016.

[2] *Dell EMC PowerMax: Reliability, Availability, and Serviceability [White Paper]*, Dell Technol., Round Rock, TX, USA, 2020.

[3] A. B. Tucker, *Computer Science Handbook*, 2nd ed. London, U.K.: Chapman & Hall, 2004.

[4] Samsung Open Source. (Sep. 2013). *Reliability, Availability and Serviceability on Linux*. [Online]. Available: https://www.slideshare.net/SamsungOSG/ras-presentation-linuxconna

[5] R. Griffith, R. Virmani, and G. E. Kaiser, "The role of reliability, availability and serviceability (RAS) models in the design and evaluation of self-healing systems," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Rep. CUCS-021-07, 2007.

[6] J.-W. Yoon and B. Jang, "Cloud system security technology trend," *J. Korea Soc. Comput. Inf.*, vol. 20, no. 7, pp. 49–56, Jul. 2015.

[7] S. M. Kelly and J. B. Ogden, "An investigation into reliability, availability, and serviceability (RAS) features for massively parallel processor systems," Sandia Nat. Lab., Livermore, CA, USA. Tech. Rep. SAND2002-3164, 2002.

[8] A. Yao, J. Li, F. Wang, J. Zhao, H. Liu, J. Zhang, J. Zhang, A. Zhou, Y. Song, J. Xu, P. Sun, K. Zhu, N. Ahuja, D. Zhu, and S. Kuo, "A memory RAS system design and engineering practice in high temperature ambient data center," in *Proc. 19th IEEE Intersociety Conf. Thermal Thermomechanical Phenomena Electron. Syst. (ITherm)*, Orlando, FL, USA, Jul. 2020, pp. 1379–1388.

[9] Puget System. (Aug. 2019). *What is the Most Reliable Hardware in Our Puget Systems Workstations?* [Online]. Available: https://www.pugetsystems.com/labs/articles/What-is-the-most-reliable-hardware-in-our-Puget-Systems-workstations-1550/

[10] J. Meza, Q. Wu, S. Kumar, and O. Mutlu, "Revisiting memory errors in large-scale production data centers: Analysis and modeling of new trends from the field," in *Proc. 45th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2015, pp. 415–426.

[11] X. Du and C. Li, "Predicting uncorrectable memory errors from the correctable error history: No free predictors in the field," in *Proc. Int. Symp. Memory Syst.*, Sep. 2021, pp. 1–10.

[12] *Reliability, Availability, and Serviceability (RAS) for the Solaris8 Operating Environment [White Paper]*, Sun Microsyst., Santa Clara, CA, USA, 2000.

[13] M. L. Fair, C. R. Conklin, S. B. Swaney, P. J. Meaney, W. J. Clarke, L. C. Alves, I. N. Modi, F. Freier, W. Fischer, and N. E. Weber, "Reliability, availability, and serviceability (RAS) of the IBM eServer z990," *IBM J. Res. Develop.*, vol. 48, no. 3.4, pp. 519–534, May 2004.

[14] I. Krutov, "Reliability, availability, and serviceability features of the IBM eX5 portfolio," IBM Corp., New York, NY, USA, Rep. REDP4864, 2012.

[15] J. H. Laros III, "A software and hardware architecture for a modular, portable, extensible reliability availability and serviceability system," in *Proc. Workshop High Perform. Comput. Rel. Issues*, 2006.

[16] S. Mittal and J. S. Vetter, "A survey of software techniques for using non-volatile memories for storage and main memory systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 5, pp. 1537–1550, May 2016.

[17] S. Mittal, "A survey of ReRAM-based architectures for processing-in-memory and neural networks," *Mach. Learn. Knowl. Extraction*, vol. 1, no. 1, pp. 75–114, Apr. 2018.

[18] S. Umesh and S. Mittal, "A survey of spintronic architectures for processing-in-memory and neural networks," *J. Syst. Archit.*, vol. 97, pp. 349–372, Aug. 2019.

[19] X. Wang, Y. Li, Y. Chen, S. Wang, Y. Du, C. He, Y. Zhang, P. Chen, X. Li, W. Song, Q. Xu, and L. Jiang, "On workload-aware DRAM failure prediction in large-scale data centers," in *Proc. IEEE 39th VLSI Test Symp. (VTS)*, Apr. 2021, pp. 1–6.

[20] J. Brown, S. Woodward, B. Bass, and C. Johnson, "IBM power edge of network processor: A wire-speed system on a chip," *IEEE Micro*, vol. 31, no. 2, pp. 76–85, Mar./Apr. 2011.

[21] H. David, C. Fallin, E. Gorbatov, U. R. Hanebutte, and O. Mutlu, "Memory power management via dynamic voltage/frequency scaling," in *Proc. 8th ACM Int. Conf. Autonomic Comput.*, Jun. 2011, pp. 31–40.

[22] D. H. Yoon, J. Chang, N. Muralimanohar, and P. Ranganathan, "Boom: Enabling mobile memory based low-power server DIMMs," in *Proc. 39th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2012, pp. 25–36.

[23] S. Wang, H. Hu, H. Zheng, and P. Gupta, "MEMRES: A fast memory system reliability simulator," *IEEE Trans. Rel.*, vol. 65, no. 4, pp. 1783–1797, Dec. 2016.

[24] D. H. Yoon and M. Erez, "Virtualized and flexible ECC for main memory," in *Proc. 15th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, Mar. 2010, pp. 397–408.

[25] *HP Advanced Memory Error Detection Technology*. Accessed: Nov. 7, 2023. [Online]. Available: https://www.hpe.com/psnow/doc/c02878598

[26] *Reliability, Availability and Serviceability (RAS) for Memory Interfaces, White Paper*, Synopsys, Mountain View, CA, USA, 2014.

[27] N. Cui, *Demonstrating the Memory RAS Features of Lenovo Think System Servers*. Hong Kong: Lenovo Press, 2017.

[28] J. Fruehe, "AMD EPYC brings new RAS capability," Moor Insights Strategy, White Paper, Jun. 2017. [Online]. Available: https://www.amd.com/system/files/2017-06/AMD-EPYC-Brings-New-RAS-Capability.pdf

[29] *Xeon Processor E7 Family: Reliability, Availability, and Serviceability*, Intel Corp., Santa Clara, CA, USA, 2011.

[30] T. M. Thompson, *From Error-Correcting Codes Through Sphere Packings to Simple Groups*. Mathematical Association of America, 1983, pp. 1–60.

[31] *Parity Bit*. Accessed: Nov. 7, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Parity_bit

[32] *RS Code*. Accessed: Nov. 7, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Reed-Solomon_error_correction

[33] W. W. Peterson and D. T. Brown, "Cyclic codes for error detection," *Proc. IRE*, vol. 49, no. 1, pp. 228–235, Jan. 1961.

[34] G. Sideris, "The Intel 1103: The MOS memory that defied cores," *Electronics*, vol. 46, no. 9, pp. 108–113, 1973.

[35] G. R. Ahearn, Y. Dishon, and R. N. Snively, "Design innovations of the IBM 3830 and 2835 storage control units," *IBM J. Res. Develop.*, vol. 16, no. 1, pp. 11–18, Jan. 1972.

[36] *IEEE Standard for Scalable Coherent Interface (SCI)*, IEEE Standard 1596-1992, Apr. 2012, pp. 1–255.

[37] T. J. Dell, "A white paper on the benefits of chipkill-correct ECC for PC server main memory," *IBM Microelectron. Division*, vol. 11, nos. 1–23, pp. 5–7, Nov. 1997.

[38] G. A. Kanawati, N. A. Kanawati, and J. A. Abraham, "Ferrari: A flexible software-based fault and error injection system," *IEEE Trans. Comput.*, vol. 44, no. 2, pp. 248–260, Feb. 1995.

[39] *ECC Memory*. Accessed: Nov. 7, 2023. [Online]. Available: https://en.wikipedia.org/wiki/ECC_memory

[40] G. M. Swift and S. M. Guertin, "In-flight observations of multiple-bit upset in DRAMs," *IEEE Trans. Nucl. Sci.*, vol. 47, no. 6, pp. 2386–2391, Dec. 2000.

[41] R. K. Burek, "The near solid-state data recorders," *Johns Hopkins APL Tech. Dig.*, vol. 19, no. 2, pp. 235–240, Apr./Jun. 1998.

[42] *Intel E7500 Chipset MCH Intel X4 Single Device Data Correction (X4 SDDC) Implementation and Validation*, Intel Corp., Santa Clara, CA, USA, Aug. 2002.

[43] E. Delano, "Tukwila: A quad-core Intel Itanium processor," in *Proc. Hot Chips*, vol. 20, Aug. 2008, pp. 1–29.

[44] T. M. Chalfant, *Solaris Operating System Availability Features*. Santa Clara, CA, USA: Sun Microsystems, 2004.

[45] W. Zhang and T. Li, "Exploring phase change memory and 3D die-stacking for power/thermal friendly, fast and durable memory architectures," in *Proc. 18th Int. Conf. Parallel Archit. Compilation Techn.*, Raleigh, NC, USA, Sep. 2009, pp. 101–112.

[46] S. Yu, S. Keil, and J. Edmondson, "Memory-based error recovery," U.S. Patent 8 365 015, Jan. 29, 2013.

[47] L. Warnes, M. B. Calhoun, D. Carr, T. Lee, D. Vu, and R. E. Espinoza-Ibarra, "Rank sparing system and method," U.S Patent 8 892 942, Nov. 18, 2014.

[48] Information Technology Intelligence Consulting Corporation. (Apr. 2020). *ITIC 2020 Global Server Hardware, Server OS Reliability Report*. [Online]. Available: https://www.ibm.com/downloads/cas/DV0XZV6R

[49] J. B. Halbert and K. S. Bains, "Memory device error check and scrub mode and error transparency," U.S. Patent 10, Nov. 13, 2018, p. 101, vol. 127.

[50] *Reed_Solomon*. [Online]. Available: https://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed_solomon_codes.html

[51] A. R. Alameldeen, I. Wagner, Z. Chishti, W. Wu, C. Wilkerson, and S.-L. Lu, "Energy-efficient cache design using variable-strength error-correcting codes," in *Proc. 38th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2011, vol. 39, no. 3, pp. 461–471.

[52] P. Reviriego, J. Martínez, S. Pontarelli, and J. A. Maestro, "A method to design SEC-DED-DAEC codes with optimized decoding," *IEEE Trans. Device Mater. Rel.*, vol. 14, no. 3, pp. 884–889, Sep. 2014.

[53] *Memory Errors and Dell EMC PowerEdge YX4X Server Memory RAS Features [White Paper]*, Dell Technol., Round Rock, TX, USA, 2020.

[54] V. Sridharan and D. Liberty, "A study of DRAM failures in the field," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Nov. 2012, pp. 1–11.

[55] J.-H. Lee, "CRC (cyclic redundancy check) implementation in high-speed semiconductor memory," in *Proc. 8th Int. Conf. Control Autom. (CA)*, Nov. 2015, pp. 17–20.

[56] J. Lee, "Matrix type CRC and XOR/XNOR for high-speed operation in DDR4 and GDDR5," *J. Inst. Electron. Inf. Eng.*, vol. 50, no. 8, pp. 136–142, Aug. 2013.

[57] J. Kim, M. Sullivan, and M. Erez, "Bamboo ECC: Strong, safe, and flexible codes for reliable computer memory," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 101–112.

[58] S. Sheng-Ju, "Implementation of cyclic redundancy check in data communication," in *Proc. Int. Conf. Comput. Intell. Commun. Netw. (CICN)*, Dec. 2015, pp. 529–531.

[59] L. Bunker and E. Hargan, "Memory malfunction prediction system and method," U.S. Patent 7 773 441, Aug. 10, 2010.

[60] D. C. Wyland, "Combined cyclic redundancy check (CRC) and Reed–Solomon (RS) error checking unit," U.S. Patent 6 836 869, Dec. 28, 2004.

[61] C. H. Lee, "A study on analysis of error correction code in server system," *J. Korea Inst. Mil. Sci. Technol.*, vol. 8, no. 3, pp. 42–50, 2005.

[62] S. K. Vemula, "System and method for memory chip kill," U.S. Patent 7 360 132, Apr. 15, 2008.

[63] S. Gurumurthi, K. Lee, M. Jang, V. Sridharan, A. Nygren, Y. Ryu, K. Sohn, T. Kim, and H. Chung, "HBM3 RAS: Enhancing resilience at scale," *IEEE Comput. Archit. Lett.*, vol. 20, no. 2, pp. 158–161, Jul. 2021.

[64] S. Liu, P. Reviriego, and F. Lombardi, "Codes for limited magnitude error correction in multilevel cell memories," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 5, pp. 1615–1626, May 2020.

[65] J. Li, P. Reviriego, L. Xiao, and H. Wu, "Protecting memories against soft errors: The case for customizable error correction codes," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 2, pp. 651–663, Apr. 2021.

[66] Y. Song, S. Park, M. B. Sullivan, and J. Kim, "SEC-BADAEC: An efficient ECC with no vacancy for strong memory protection," *IEEE Access*, vol. 10, pp. 89769–89780, 2022.

[67] M. Dong, W. Pan, Z. Qiu, X. Qi, L. Zheng, and H. Liu, "A universal, low-delay, SEC-DEC-TAEC code for state register protection," *IEEE Access*, vol. 10, pp. 57665–57673, 2022.

[68] A. Radonjic, "Integer codes correcting double errors and triple-adjacent errors within a byte," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 8, pp. 1901–1908, Aug. 2020.

[69] J. Gao, W. Zhang, Y. Liu, H. Wang, and J. Zhao, "High-performance concatenation decoding of Reed–Solomon codes with SPC codes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 9, pp. 1670–1674, Sep. 2021.

[70] V. Boussard, S. Coulombe, F.-X. Coudoux, and P. Corlay, "Table-free multiple bit-error correction using the CRC syndrome," *IEEE Access*, vol. 8, pp. 102357–102372, 2020.

[71] V. Boussard, S. Coulombe, F.-X. Coudoux, and P. Corlay, "CRC-based correction of multiple errors using an optimized lookup table," *IEEE Access*, vol. 10, pp. 23931–23947, 2022.

[72] T. Gendron, E. Boutillon, C. A. Nour, and D. Gnaedig, "Forward backward syndrome computation: A reduced complexity CRC code decoder," *IEEE Commun. Lett.*, vol. 27, no. 5, pp. 1267–1271, May 2023.

[73] K. C. Chun et al., "A 16-GB 640-GB/s HBM2E DRAM with a data-bus window extension technique and a synergetic on-die ECC scheme," *IEEE J. Solid-State Circuits*, vol. 56, no. 1, pp. 199–211, Jan. 2021.

[74] S.-I. Pae, V. Kozhikkottu, D. Somasekar, W. Wu, S. G. Ramasubramanian, M. Dadual, H. Cho, and K.-W. Kwon, "Minimal aliasing single-error-correction codes for DRAM reliability improvement," *IEEE Access*, vol. 9, pp. 29862–29869, 2021.

[75] G. Tshagharyan, G. Harutyunyan, S. Shoukourian, and Y. Zorian, "Experimental study on Hamming and Hsiao codes in the context of embedded applications," in *Proc. IEEE East-West Design Test Symp. (EWDTS)*, Novi Sad, Serbia, Sep. 2017, pp. 1–4.

[76] Y. Cassuto, M. Schwartz, V. Bohossian, and J. Bruck, "Codes for asymmetric limited-magnitude errors with application to multilevel flash memories," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1582–1595, Apr. 2010.

[77] S. Liu, P. Reviriego, K. Namba, S. Pontarelli, L. Xiao, and F. Lombardi, "Low redundancy double error correction spotty codes combined with gray coding for 64 data bits memories of 4-bit multilevel cells," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT)*, Oct. 2019, pp. 1–4.

[78] A. Das and N. A. Touba, "Limited magnitude error correction using OLS codes for memories with multilevel cells," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Nov. 2017, pp. 391–394.

[79] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2004.

[80] X. Du and C. Li, "Combining error statistics with failure prediction in memory page offlining," in *Proc. Int. Symp. Memory Syst.*, Sep. 2019, pp. 127–132.

[81] D. Das, K. Cheng, and J. C. Jasper, "Mechanism for achieving high memory reliability, availability and serviceability," U.S. Patent 9 229 828, Jan. 5, 2016.

[82] S. Levy, K. B. Ferreira, N. DeBardeleben, T. Siddiqua, V. Sridharan, and E. Baseman, "Lessons learned from memory errors observed over the lifetime of cielo," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Nov. 2018, pp. 554–565.

[83] G. Wang, L. Zhang, and W. Xu, "What can we learn from four years of data center hardware failures?" in *Proc. 47th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2017, pp. 25–36.

[84] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM errors in the wild: A large-scale field study," in *Proc. 11th Int. Joint Conf. Meas. Model. Comput. Syst.*, vol. 37, no. 1, pp. 193–204, 2009.

[85] X. Du and C. Li, "DPCLS: Improving partial cache line sparing with dynamics for memory error prevention," in *Proc. IEEE 38th Int. Conf. Comput. Design (ICCD)*, Oct. 2020, pp. 197–204.

[86] D. W. Kim and M. Erez, "Balancing reliability, cost, and performance tradeoffs with FreeFault," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 439–450.

[87] D. E. Galbi, W. Heirman, and D. Ziakas, "Translation cache and configurable ECC memory for reducing ECC memory overhead," U.S. Patent 17/483 536, Jan. 13, 2022.

[88] J. Park, H. Yeom, and Y. Son, "Page reusability-based cache partitioning for multi-core systems," *IEEE Trans. Comput.*, vol. 69, no. 6, pp. 812–818, Jun. 2020.

[89] J. H. Yoo, S. H. Lee, D. Kang, S. Kim, H. Yu, J. Youn, and H. Choi, "DRAM controller having DRAM bad page management function and bad page management method thereof," U.S. Patent 9 348 679, May 24, 2016.

[90] Q. Hoang, W. G. Liu, A. Butcher, and M. Shutt, "System and method for improved handling of memory failures," U.S. Patent 11 016 835, May 25, 2021.

[91] S. Mittal and M. S. Inukonda, "A survey of techniques for improving error-resilience of DRAM," *J. Syst. Archit.*, vol. 91, pp. 11–40, Nov. 2018.

[92] Y. Cai, Y. Wu, and E. F. Haratsch, "Error correction code (ECC) selection using probability density functions of error correction capability in storage controllers with multiple error correction codes," U.S. Patent 10 153 782, Dec. 11, 2018.

[93] A. Kleen. *Mcelog: Memory Error Handling in User Space*. Accessed: Nov. 7, 2023. [Online]. Available: https://halobates.de/lk10-mcelog.pdf

[94] T. Jiang, P. Huang, and K. Zhou, "Cost-efficiency disk failure prediction via threshold-moving," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 14, p. e5669, 2020.

[95] X. Du, C. Li, S. Zhou, X. Liu, X. Xu, T. Wang, and S. Ge, "Fault-aware prediction-guided page offlining for uncorrectable memory error prevention," in *Proc. IEEE 39th Int. Conf. Comput. Design (ICCD)*, Oct. 2021, pp. 456–463.

[96] S. Zhou, C. Li, K. S. Bains, X. Du, and M. Oriol, "Lifetime telemetry on memory error statistics to improve memory failure analysis and prevention," U.S. Patent 17/317 745, Sep. 9, 2021.

[97] S. Zhou, X. Du, C. Li, and K. S. Bains, "Page offlining based on fault-aware prediction of imminent memory error," U.S. Patent 17 512 432, Sep. 9, 2021.

[98] X. Du, C. Li, S. Zhou, M. Ye, and J. Li, "Predicting uncorrectable memory errors for proactive replacement: An empirical study on large-scale field data," in *Proc. 16th Eur. Dependable Comput. Conf. (EDCC)*, Sep. 2020, pp. 41–46.

[99] M. O'Connor, N. Chatterjee, D. Lee, J. Wilson, A. Agrawal, S. W. Keckler, and W. J. Dally, "Fine-grained DRAM: Energy-efficient DRAM for extreme bandwidth systems," in *Proc. 50th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2017, pp. 41–54.

[100] S. Khan, D. Lee, and O. Mutlu, "PARBOR: An efficient system-level technique to detect data-dependent failures in DRAM," in *Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun./Jul. 2016, pp. 239–250.

[101] 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 *Proc. Int. Conf. Archit. Support Program, Lang. Oper. Syst.*, 2012, pp. 111–122.

[102] C. H. A. Costa, Y. Park, B. S. Rosenburg, C.-Y. Cher, and K. D. Ryu, "A system software approach to proactive memory-error avoidance," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Nov. 2014, pp. 707–718.

[103] P. Nikolaou, Y. Sazeides, L. Ndreu, and M. Kleanthous, "Modeling the implications of DRAM failures and protection techniques on datacenter TCO," in *Proc. 48th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Dec. 2015, pp. 572–584.

[104] X. Du and C. Li, "Memory failure prediction using online learning," in *Proc. Int. Symp. Memory Syst.*, Oct. 2018, pp. 38–49, doi: 10.1145/3240302.3240309.

[105] J. Hilbe, *Logistic Regression Models*. Boca Raton, FL, USA: CRC Press, 2009.

[106] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis," *IEEE Intell. Syst. Their Appl.*, vol. 13, no. 4, pp. 18–28, Jul./Aug. 1998, doi: 10.1109/5254.708428.

[107] W. Loh, "Classification and regression trees are machine learning models used to predict continuous or categorical values, respectively," *WIREs Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 14–23, Jan. 2011, doi: 10.1002/widm.8.

[108] Y. Liu, S. Liu, Y. Wang, F. Lombardi, and J. Han, "A stochastic computational multi-layer perceptron with backward propagation," *IEEE Trans. Comput.*, vol. 67, no. 9, pp. 1273–1286, Sep. 2018.

[109] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.

[110] I. Boixaderas, D. Zivanovic, S. Moré, J. Bartolome, D. Vicente, M. Casas, P. M. Carpenter, P. Radojkovic, and E. Ayguadé, "Cost-aware prediction of uncorrected DRAM errors in the field," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Nov. 2020, pp. 1–15.

[111] M. Pal, "Random forest classifier for remote sensing classification," *Int. J. Remote Sens.*, vol. 26, no. 1, pp. 217–222, Jan. 2005.

[112] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Mach. Learn.*, 1996, pp. 148–156.

[113] D.-T. Nguyen, N.-M. Ho, W.-F. Wong, and I.-J. Chang, "OBET: On-the-fly byte-level error tracking for correcting and detecting faults in unreliable DRAM systems," *Sensors*, vol. 21, no. 24, p. 8271, Dec. 2021.

[114] S.-L. Gong, J. Kim, S. Lym, M. Sullivan, H. David, and M. Erez, "Duo: Exposing on-chip redundancy to rank-level ECC for high reliability," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2018, pp. 683–695.

[115] P. J. Nair, V. Sridharan, and M. K. Qureshi, "XED: Exposing on-die error detection information for strong memory reliability," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 341–353.

[116] D. Hardy, M. Kleanthous, I. Sideris, A. G. Saidi, E. Ozer, and Y. Sazeides, "An analytical framework for estimating TCO and exploring data center design space," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, Apr. 2013, pp. 54–63.

[117] J. Karidis, J. E. Moreira, and J. Moreno, "True value: Assessing and optimizing the cost of computing at the data center level," in *Proc. 6th ACM Conf. Comput. Frontiers*, May 2009, pp. 185–192.

[118] K. V. Vishwanath, A. Greenberg, and D. A. Reed, "Modular data centers: How to design them?" in *Proc. 1st ACM Workshop Large-Scale Syst. Appl. Perform.*, Jun. 2009, pp. 3–10.

[119] Y. Cui, C. Ingalz, T. Gao, and A. Heydari, "Total cost of ownership model for data center technology evaluation," in *Proc. 16th IEEE Intersociety Conf. Thermal Thermomechanical Phenomena Electron. Syst. (ITherm)*, May 2017, pp. 936–942.

[120] I. Alam and P. Gupta, "COMET: On-die and in-controller collaborative memory ECC technique for safer and stronger correction of DRAM errors," in *Proc. 52nd Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2022, pp. 124–136.

[121] M. Patel, J. S. Kim, T. Shahroodi, H. Hassan, and O. Mutlu, "Bit-exact ECC recovery (BEER): Determining DRAM on-die ECC functions by exploiting DRAM data retention characteristics," in *Proc. 53rd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2020, pp. 282–297.

[122] Y. Ryu et al., "A 16 GB 1024 GB/s HBM3 DRAM with source-synchronized bus design and on-die error control scheme for enhanced RAS features," *IEEE J. Solid-State Circuits*, vol. 58, no. 4, pp. 1051–1061, Apr. 2023.

JISEONG LEE (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering from the Kumoh National Institute of Technology, Gumi, South Korea, in 2018. He is currently pursuing the M.S. and Ph.D. degrees with Korea University, Seoul, South Korea. His research interests include current regulators for system-on-chip (SoC) applications, dc/dc control techniques, low-power switched mode power supplies, memories, and power management ICs. He received the American Engineering Experiential Work with Purdue University, in 2016.

MIN JOON KIM (Graduate Student Member, IEEE) received the B.S. degree from the School of Electrical and Electronics Engineering, Chung-Ang University, Seoul, South Korea, in 2023. He is currently pursuing the M.S. degree with Korea University, Seoul. His research interests include dc/dc converters, power management ICs, low-power IC design, and memories.

WOO-SEOP KIM (Member, IEEE) received the B.S. degree (Hons.) in applied physics from Inha University, South Korea, in 1988, the M.S. degree in electronics engineering from Sungkyunkwan University, South Korea, in 1995, and the Ph.D. degree in electronics engineering from Korea University, Seoul, South Korea, in 2004. From 1990 to 2021, he was with Samsung Electronics Company, where he was involved in the development of DRAM. Since March 2021, he has been a Professor of industry-academia collaboration with the School of Electrical Engineering, Korea University. His research interests include testing for memories, high-speed memory interfaces, and quantum random number generators.

YONG SIN KIM (Senior Member, IEEE) received the B.S. and M.S. degrees in electronics from Korea University, Seoul, South Korea, in 1999 and 2003, respectively, and the Ph.D. degree in electrical engineering from the University of California at Santa Cruz, USA, in 2008. From 2008 to 2012, he was with the University of California Advanced Solar Technologies Institute (UC Solar), where he researched optimizing power in distributed photovoltaic systems. From 2012 to 2014, he was with the School of Electrical and Electronics Engineering, Chung-Ang University, Seoul, where he was involved in the development of sensors for human–machine interfaces. Since March 2014, he has been with the School of Electrical Engineering, Korea University. His current research interests include the cross-disciplinary integration of circuits and systems for energy harvesting, sensors, automotive applications, and memories.