E³CC: A Memory Error Protection Scheme with Novel Address Mapping for Subranked and Low-Power Memories

LONG CHEN, YANAN CAO, and ZHAO ZHANG, Iowa State University

This study presents and evaluates E³CC (Enhanced Embedded ECC), a full design and implementation of a generic embedded ECC scheme that enables power-efficient error protection for subranked memory systems. It incorporates a novel address mapping scheme called Biased Chinese Remainder Mapping (BCRM) to resolve the address mapping issue for memories of page interleaving, plus a simple and effective cache design to reduce extra ECC traffic. Our evaluation using SPEC CPU2006 benchmarks confirms the performance and power efficiency of the E³CC scheme for subranked memories as well as conventional memories.

Categories and Subject Descriptors: B.3.2 Primary Memory

General Terms: Measurement, Performance, Reliability

Additional Key Words and Phrases: DRAM, reliability, power, address mapping

ACM Reference Format:

Chen, L., Cao, Y., and Zhang, Z. 2013. E³CC: A memory error protection scheme with novel address mapping for subranked and low-power memories. *ACM Trans. Architec. Code Optim.* 10, 4, Article 32 (December 2013), 22 pages.

DOI: <http://dx.doi.org/10.1145/2541228.2541239>

1. INTRODUCTION

Memory reliability is increasingly a concern with the rapid improvement of memory density and capacity, as memory holds more and more data. Without error protection, a single upset of a memory cell may lead to memory corruption, which may have further consequences including permanent data corruption, program and system crashes, security vulnerabilities, and so forth. However, the majority of consumer-level computers and devices have not yet adopted any memory error protection scheme. The more data there are in unprotected memory and the longer they stay there, the higher the chance that users may experience those mishaps.

Meanwhile, memory power efficiency has become a first-order consideration in computer design. DRAM memory systems can consume more power than processors [Lin et al. 2008] on memory-intensive workloads, and it has been predicted that future systems may spend more than 70% of power in memory [Borkar 2011]. Recently proposed subranked DDRx memories [Ahn et al. 2009a, 2009b; Zheng et al. 2008] reduce memory power consumption significantly by using memory subbanks of less than 64-bit data bus and fewer numbers of devices than conventional DDRx memories, for example, with two x8 devices in a subrank of 16-bit bus width. Additionally, mobile devices such as iOS, Android, and Windows phones and tablets have started to use low-power DDRx (LPDDR, LPDDR2, LPDDR3, and the incoming LPDDR4) memory with 32-bit

Author's addresses: Z. Zhang, Iowa State University – Department of Electrical and Computer Engineering; email: zzhang@iastate.edu.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701, USA, fax +1 (212) 869-0481 or permission@acm.org.

© 2013 ACM 1544-3566/2013/12-ART32 \$15.00

DOI: <http://dx.doi.org/10.1145/2541228.2541239>

data bus, which is similar to 32-bit subranked memory. Those mobile devices have been increasingly used in applications that require reliable computing to an extent, for example, medical care, mobile banking, and construction.

These two trends lead to a conflict between memory reliability and power efficiency. Conventional ECC (error correcting code) memory employs a memory error protection scheme using a (72, 64) SECDED (single error correction, double error detection) code, of which an ECC word consists of 64 data bits and eight ECC bits. A memory rank in ECC DDR $\times$ memory may consist of eight x8 memory devices (chips) dedicated for data and one x8 device dedicated for ECC, or 16 x4 devices dedicated for data and two x4 devices dedicated for ECC. Subranked and low-power memories are incompatible with this memory rank organization, leaving the question of how to implement error protection in them.¹ The study of Mini-Rank [Zheng et al. 2008], one type of subranked memory organization, proposes a generic approach called *embedded ECC* to alleviate the concern. In embedded ECC, data bits and ECC bits of an ECC word are mixed together and, therefore, dedicated ECC devices are no longer needed. It essentially decouples memory organization and the choice of error protection code. With the mixing, however, the *effective size* of the DRAM row is no longer power-of-two, which complicates memory device-level address mapping. If an efficient address mapping is nonexistent, Embedded ECC will not be a practical solution. Embedded ECC does identify Chinese Remainder Mapping (CRM) [Gao 1993] to be a potential solution to the address mapping issue, but there lacks design and implementation details.

This article presents Enhanced Embedded ECC (E³CC), which is a full design and implementation of embedded ECC, but with its own innovations and contributions. E³CC shares the following merits with the original embedded ECC (most of which were not identified in the previous study [Zheng et al. 2008]):

- E³CC can be integrated into subranked memory, yielding a power-efficient and reliable memory system. It can also be applied to those mobile devices using low-power DDR $\times$ memory.
- Although it is proposed for subranked memories, it may also be used to provide ECC protection to non-ECC DDR $\times$ DIMMs by an extension to the memory controller, with no change to the DIMMs or devices. Such a system can be booted in either ECC memory mode or non-ECC memory mode.
- It is now possible to use other error protection codes with more extension in the memory controller; for example, a long BCH(532, 512) code that corrects two-bit error and detects three-bit error using 532-bit word size with 3.9% overhead. By comparison, the conventional ECC memory corrects one-bit error and detects two-bit error using 72-bit word size with 12.5% overhead.

This study on E³CC has its own contributions and innovations beyond the original idea of embedded ECC:

- The E³CC is a thorough design and a complete solution. Equally important, its performance and power efficiency have been thoroughly evaluated by detailed simulation of realistic memory settings. Both are critical to proving that E³CC will be a working idea when applied. By comparison, the original embedded ECC was a generic idea, with little design detail and no evaluation at all. It was a supplemental idea to the Mini-Rank design.
- A novel and unique address mapping scheme called *Biased Chinese Remainder Mapping* (BCRM) has been proposed to resolve the address mapping issue for the page-interleaving address mapping scheme. BCRM is an innovation by itself. It can be

¹If a shorter ECC word is used, ECC storage overhead will increase significantly.

used in memory systems where the address mapping is not power-of-two based and spatial locality is desired.

- The study of Mini-Rank [Zheng et al. 2008] identified CRM [Gao 1993] as a potential method of efficient address mapping, which motivated the search for BCRM. However, this study has found out that cacheline interleaving in Embedded ECC does not really need CRM; and for page interleaving, CRM may destroy or degrade the row-level locality of page interleaving.
- This study identifies the issue of extra ECC traffic in DDR3 memories that embedded ECC may cause as a result of the burst length requirement of DDR3. A simple and effective solution called *ECC cache* is used to effectively reduce this traffic overhead from embedded ECC.

E³CC and the original embedded ECC idea are related to but different from recent studies on memory ECC design. In particular, virtualized ECC [Yoon and Erez 2010] also makes ECC flexible, as E³CC does, but by storing ECC bits into separate memory pages. It relies on a processor's cache to buffer unused ECC bits to avoid extra ECC traffic. E³CC has more predictable worst-case performance than virtualized ECC, because it stores data bits and the associated ECC bits in the same DRAM rows. Accessing uncached ECC bits is merely an extra column access and additional memory bursts following those for data, with only a few nanoseconds added to memory latency, and no extra power spent on DRAM precharge and row activation. LOT-ECC [Udipi et al. 2012] provides reliability of the chipkill-correct level using nine-device rank (assuming x8 devices). LOT-ECC also stores ECC bits with data bits in the same DRAM rows; however, it is otherwise very different from E³CC in design. LOT-ECC provides stronger reliability than E³CC but at the cost of more storage overhead (26.5% vs. 14.1%). The LOT-ECC design as presented is not compatible with subranked memories, and thus LOT-ECC memory may not be as power efficient as E³CC memory with subranking. Those studies have identified the issue of extra ECC traffic and have used caches explicitly or implicitly to reduce the traffic, although the mechanism is different from that of ECC cache.

The rest of this article is organized as follows. Section 2 introduces background of the main memory system errors and related works. Section 3 presents novel address mapping and the design of embedded ECC. Section 4 describes the performance and power simulation platform for DDRx memory. The performance and power simulation results are presented and analyzed in Section 5. Finally, Section 6 concludes the article.

2. BACKGROUND AND RELATED WORK

DRAM Error Protection. Memory soft error was reported by May and Woods [May and Woods 1979] in 1979 and has been a focus of recent studies [Johnston 2000; Mukherjee et al. 2005; Borucki et al. 2008; Baumann 2005]. A recent study on Google's fleet of servers reports about 25,000 to 70,000 FIT per Mbit of DRAM systems [Schroeder et al. 2009]; FIT represents failures in billion operation hours. Some other studies [Li et al. 2010; Borucki et al. 2008] also show the increase of the memory error rate. ITRS 2010 [ITRS 2010] proposes the requirement of 1,000 FIT for future DRAM error ratio, which is a challenge to meet.

The conventional ECC DIMM is SECDED. Specifically, it employs a (72, 64) SECDED code based on Hamming code [Hamming 1950; Rao and Fujiwara 1989; Fung 2008], which encodes 64 data bits into a 72-bit ECC word with eight parity bits. A single-bit error within the 72-bit ECC word is correctable, and a double-bit error is detectable but not correctable. In DDRx memory systems, a non-ECC DDRx DIMM usually has one or two *ranks* of DRAM devices, with eight x8 or 16 x4 devices (chips) in each rank to form a 64-bit data path. An ECC DIMM has nine x8 or 18 x4 devices per rank to form

Fig. 1. Comparison of the conventional registered non-ECC DIMM and a subranked non-ECC DIMM organization. The subranked DIMM consists of four devices per rank in this example.

a 72-bit data path, with the extra one or two devices to store the parity bits. The DDR x data bus is 64-bit and 72-bit, respectively, without and with ECC, so ECC DIMMs are not compatible with a motherboard designed for non-ECC DIMMs.

Chipkill [Dell 1997; Intel 2002] has the capability of Single Device Data Correction (SDDC) in addition to SECDED. ECC DIMM does not support SDDC, because a single device failure might incur the loss of eight bits in each 72-bit ECC word, which is not recoverable. Recent chipkill design groups multiple bits from one memory device as a symbol and applies symbol correction code to recover a device failure. The downside of chipkill, however, is excessive memory power consumption because in most chipkill schemes each memory access may involve many more memory devices than conventional ECC memories.

DRAM Operations and Subranked Memory Organization. The detail of DRAM operations could be referred to previous studies [Cuppu et al. 1999; Rixner et al. 2000; Zheng et al. 2008; Udipi et al. 2010]. A memory access may involve up to three DRAM operations, namely, precharge, activation (row access), and read/write (column access). The memory controller issues commands to all devices in a rank to perform those operations. Subranked memory organization [Ware and Hampel 2006; Zheng et al. 2008; Ahn et al. 2009b] is proposed to reduce the number of devices in a rank so as to reduce DRAM operation power spent on precharge and activation. It also increases the number of subranks, which helps reduce DRAM background power because subranks may enter low-power modes more frequently than ranks in conventional DIMMs. Figure 1 illustrates two DIMM organizations, a conventional registered DIMM and a sub-ranked DIMM with 32-bit rank size. The difference is that the incoming commands and addresses are buffered and demultiplexed to the devices in the destination subrank. This difference makes them subranked; that is, a full-size rank is now divided into subranks of four, two, or even one device. Figure 1(b) shows an example of subranked DIMM with four devices per rank. For each memory access, the number of DRAM devices involved is effectively reduced, thus effectively reducing the row activation power and background power, which dominate memory power consumption.

Related Work. There have been many studies on memory system reliability [Yoon and Erez 2010; Schechter et al. 2010; Seong et al. 2010; Yoon et al. 2011] and memory power efficiency [Diniz et al. 2007; Hur and Lin 2008; Zheng et al. 2008; Udipi et al. 2010]. Most of those reliability-related studies focus on error correction for PCM and SRAM. Two studies closely related to our work are virtualized ECC [Yoon and Erez 2010] and LOT-ECC [Udipi et al. 2012], which have been discussed in Section 1.

Recent studies show another trend of adopting subranked memory architecture for power concern. The subranked memory architecture divides conventional memory rank into multiple subranks, each of a smaller number of DRAM devices, so as to reduce the activation and row access power. The threaded memory module [Ware and Hampel 2006] splits a conventional DDR x bus into subbuses to reduce the number of DRAM

devices involved in a single memory access, effectively breaking a memory rank into subranks. Mini-Rank [Zheng et al. 2008] breaks a memory rank into subranks by using a bridge chip on each DIMM, which converts data between the 64-bit channel data bus and multiple narrower buses inside the DIMM. The use of the bridge chip may improve the utilization ratio of the channel bus. The study includes a comprehensive evaluation of memory energy savings from subranking. The general idea of embedded ECC is proposed to address the concern of implementing ECC over Mini-Rank DIMM, which has been discussed in the introduction. Multicore DIMM [Ahn et al. 2009b] is similar but uses split data bus with shared command and address bus to reduce the design complexity.

The E³CC scheme is designed for commodity DRAM devices and modules. By comparison, an industry design [Leung et al. 2008] revises the data array inside memory devices to store ECC bits, which leads to noncommodity DRAM devices. Manufacturing such devices usually incurs high cost. Another industry design [Danilak 2006] assumes commodity DRAM devices and modules, as E³CC does, and it costores data and the related ECC bits in the same devices. However, the data and ECC bits are stored in different memory banks of the device, which means accessing the ECC bits requires separate DRAM precharges and activations.

A recent patent [Haertel et al. 2012] from AMD proposes to costore data and ECC bits in the same DRAM rows in commodity devices and modules, with the same motivation of avoiding separate precharges and activations. It uses a simple address mapping scheme to obtain the device addresses of data and ECC bits without using division: simply put, it works as long as the most significant three bits of the physical address are selected as part of the device column address. We have had a finding that embedded ECC with the conventional cacheline-interleaving scheme does not require any special address mapping (Section 3.2); that is actually a special case of the AMD design when the most significant three bits of the physical address become the most significant three bits of the device column address. E³CC uses the novel BCRM scheme, which is more complex than simple bit selection but works for the conventional page-interleaving mode. The AMD design may also be used with page interleaving; however, the page size has to be reduced to no more than one-eighth of the page size of the conventional page interleaving. Additionally, the AMD design was proposed for conventional memory ECC. If with a new ECC code the ratio of data bits to ECC bits is no longer 8:1 or a power of two, the AMD design may not work or may have to waste a significant ratio of storage. Note that the patent description uses two terms differently: The term “logical address space” is the physical address space in this article, and the term “physical address space” is the device address space.

3. DESIGN OF E³CC

In this section, we present the design of E³CC and discuss in detail its design issues including memory block layout, page interleaving and BCRM, ECC traffic overhead and ECC cache, and the use of long error protection code. All discussions are specific to DDR2/DDR3 DRAM memory; however, most discussions are also valid for other types of memory such as phase-change memories.

3.1. DIMM Organization and IntraBlock Layout

Figures 2 and 3 contrast the difference of conventional ECC memory and E³CC memory. It shows the mapping of memory subblocks inside a memory block of cacheline size; note that a memory block may be mapped to multiple memory devices. Most memory accesses are caused by cache miss, writeback, or prefetch. The examples assume Micron DDR3-1600 x8 device MT41J256M8, which is 2G bit each and has eight internal banks, 32K rows per rank, 1K columns per row, and eight bits per column. DDR3 requires

Fig. 2. An example layout of a memory block of cacheline size (512-bit) inside a rank of a conventional ECC DDR3 DIMM using x8 devices. Each rank has eight data devices plus an ECC device. Each device has multiple banks, which is not shown. D represents a generic data supercolumn, and E represents a generic ECC supercolumn. One supercolumn is 64-bit, or eight columns in a device row. B_i represents a 64-bit subblock of the memory block, and E_i represents the corresponding eight ECC bits. The memory block occupies eight 64-bit super-columns that are distributed over the eight devices; the ECC bits are stored in a column in the ECC device. An 8K-bit row consists of 128 supercolumns. The row layouts of the second device and the ECC device are selectively highlighted, and so are the mappings of subblock B1 and the ECC bits.

Fig. 3. An example layout of a memory block of cacheline size (512-bit) inside a rank of E^3CC DDR3 DIMM of the full rank size. Other assumptions are the same as in Figure 2. The row layout of the second device is selectively highlighted, and so are the mappings of the second subblock B1 and the ECC subblock E1.

a minimum burst length of eight, so each column access involves consecutive eight columns. We call an aligned group of eight columns a *supercolumn*. The ECC is a (72, 64) SECDED code made by a (71, 64) Hamming code plus a checksum bit, so each ECC word consists of 64 data bits and eight ECC bits. In the conventional ECC DIMM, a memory block of the 64B cacheline size may occupy eight 64-bit supercolumns distributed over the eight data devices. The ECC bits occupy a single supercolumn in the ECC device.

E^3CC DIMM is physically the same as conventional non-ECC DIMM. However, the ECC bits are mixed with data bits in the same DRAM rows. To maintain the 8:1 ratio of data and ECC, there is one ECC supercolumn for every eight data supercolumns. The

Fig. 4. An example layout of a memory block of cacheline size (512-bit) for 16-bit subranked E³CC DDR3 DIMM. The other assumptions and notations are the same as in Figures 2 and 3. The row layout of the third device is selectively highlighted, and so are the mappings of subblocks B0–B3 and the ECC subblocks E0–E3.

data bits of the memory block are distributed over the eight devices, one supercolumn per device. The ECC bits are also evenly distributed, occupying one-eighth supercolumn per device. An ECC supercolumn in the row is made by ECC bits from eight memory blocks. For this particular device, each row has 112 data supercolumns and 14 ECC supercolumns. The last two supercolumns are not utilized, which are 1.6% storage overhead, in addition to the 12.5% ECC storage overhead.

The layouts can have different variants as long as the data and ECC bits of a memory block are evenly distributed over all devices in the rank and using the same row address. The intrablock layout of E³CC DIMM is designed to be compatible with the burst length of eight of the DDR3 devices.² A DDR3 device uses an internal buffer of eight bits per I/O bin to convert slow and parallel data into high-speed serial data; therefore, for a x8 device, each column access (data read/write) involves 64-bit memory data. Accessing a memory block of 64 B requires only a single column access. However, accessing the associated ECC bits requires another column access (but no extra precharge or activation), of which only one-eighth of the bits are needed. We proposed ECC cache to address this issue, which will be discussed in Section 3.4. DDR2 also supports a burst length of four (in addition to eight), for which the layout can be different.

Figure 4 shows the intrablock layout for 16-bit subranked E³CC DIMM. The DIMM organization is the same as 16-bit subranked, non-ECC DIMM. The memory block is now mapped to two x8 devices, using four data supercolumns and one-half ECC supercolumn. Each memory access involves two devices, drastically reducing the power and energy spent on device precharge and activation. Accessing the ECC bits incurs less potential waste of bandwidth than before, as two memory blocks share one ECC supercolumn instead of eight in fully ranked DIMM. The last two supercolumns are still left over.

We do not show the layouts of eight-bit and 32-bit subranked E³CC DIMM, as they are similar to that of the 16-bit, except that the memory block is mapped to one and four devices, respectively. The leftover ratio stays at 1.6% as the last two columns cannot

²DDR3 has a burst chop 4 mode, in which the burst length is reduced to four but with extra performance overhead introduced.

Fig. 5. Representative memory address mappings for cacheline interleaving and page interleaving in DDRx memory systems. Cacheline interleaving 1 may be used in systems of memory power and thermal concerns, and cacheline interleaving 2 may be used to minimize the impact of rank-to-rank switching penalty on memory throughput.

be utilized. For eight-bit subranked DIMM, the ECC bits of a memory block occupy a whole ECC supercolumn and therefore there is no potential waste of bandwidth when accessing the ECC bits.

3.2. Interleaving Schemes and Address Mapping

Memory interleaving decides how a memory block is mapped to memory channel, DIMM, rank, and bank. It is part of the memory address mapping, that is, to translate a given physical memory block address to the channel, DIMM, rank, bank, row, and column indexes/addresses. In conventional ECC DIMM, the logic design is simply a matter of splitting memory address bits. With E³CC, the number of memory blocks in the same row of the same rank and bank is no longer power-of-two, which complicates the memory device-level address mapping.

Two commonly used address mapping schemes are cacheline interleaving and page interleaving, and each has its variants. Figure 5 shows representative address mappings of the two types in DDRx memories. The main difference is that in page interleaving, consecutive memory blocks (within page boundary) are mapped to the same DRAM rows, whereas in cacheline interleaving the consecutive blocks are distributed evenly over channels, DIMMs, ranks, and banks (not necessarily in that order). Page interleaving is commonly used with open-page policy, in which the row buffer of a bank is kept open after access; cacheline interleaving is commonly used with close-page policy. If another access falls into the same memory row, the data is still in the row buffer of the bank and thus the read/write operation may start immediately. Otherwise, the bank must be precharged and then activated before the read/write operation, increasing memory access latency. Cacheline interleaving with close-page policy is generally more power efficient than page interleaving with open-page policy, because it consumes more power to keep a row buffer open. However, the latter can be more power efficient if there is a high level of page locality in memory accesses.

A major concern of E³CC is regarding the address mapping scheme, because the effective size of DRAM rows, for example, the number of data blocks/bits (excluding ECC bits) they hold, is no longer power-of-two. That means *the column address may not be extracted directly from the physical memory address bits*, and all address components to the left of the column address bits will be affected. We find, however, that for cacheline-interleaving schemes whose column address is the leftmost bits in the physical memory address, the address mapping is not an issue. Those address components can be broken down as shown in Figure 5; the difference is that a subset of high-end column addresses becomes invalid, as they would represent invalid physical memory addresses on E³CC memory. However, page interleaving, which is also commonly used, is affected because the column address appears in the middle of the physical memory address. An integer

division may have to be used, unless a unique and efficient address mapping scheme is found.

However, using integer division may negatively affect memory latency and throughput, increasing both the initial latency and the queueing delay of memory access. It was not acceptable in the 1980s and 1990s [Lawrie and Vora 1982; Teng 1983; Gao 1993] and is still very questionable in modern processors. An integer division takes many clock cycles; for example, 56 to 70 in a Pentium 4 processor for 32-bit division. It can be even more expensive than double-precision floating-point division (38 cycles in the Pentium 4 processor) [Takagi et al. 2005]. A recent manual from Intel [Intel Corporation] reports that the latency of IDIV (Integer Division) instruction is from 20 to 70 cycles for Intel 64 and IA-32 architectures. We did a simulation to evaluate the performance impact of adding 5ns, 10ns and 15ns for integer division delay to the memory controller overhead, using the baseline simulation setting in Section 4. The performance degradation is 3%, 5%, and 7%, respectively. The simulation assumes that the division unit is fully pipelined, that is, with no queueing delay. In reality, it is difficult to implement a pipelined division unit, so a designer may have to use multiple unpipelined division units. For example, consider a system of two DDR3-1600 memory channels and using 64-byte cache blocks, in which the peak memory throughput per channel is one 64-byte block every 5ns. An unpipelined division unit of 15ns latency can only match one-third of the memory channel throughput. Therefore, three such units are needed for each channel, six in total, to match the peak system memory throughput.³

3.3. Page Interleaving with BCRM

CRM and BCRM. We devise a novel and efficient address mapping scheme called Biased Chinese Remainder Mapping (BCRM). We start with an existing scheme called Chinese Remainder Mapping (CRM), which was proposed based on Chinese Remainder Theorem [Gao 1993], in which a memory block address d is decomposed into a pair of integers $\langle u, v \rangle$, with $0 \leq d \leq pm - 1$, $0 \leq u \leq m - 1$, and $0 \leq v \leq p - 1$, using the following formula⁴:

$$u = d \bmod m, \quad v = d \bmod p.$$

CRM ensures that the mapping from d to $\langle u, v \rangle$ is “perfect” if p and m are coprimes, where being “perfect” means all possible combinations of $\langle u, v \rangle$ are used in the address mapping. Two integers are coprimes if their greatest common divisor is 1. The formal proof of the “perfect” property is given in the previous study [Gao 1993]. Table I shows the layout of block address d under this mapping, assuming $p = 7$ and $m = 8$. As it shows, the numbers are laid out diagonally in the two-dimension table. In a prime memory system, p is intended to be the number of memory banks and m the number of memory blocks in a bank, and p is also intended to be a prime number. However, when used with page interleaving, CRM will disturb the row-level locality, as it was not designed for maintaining the locality.

BCRM adds a *bias factor* to the original CRM formula:

$$u = ((d - (d \bmod p)) \bmod m), \quad v = d \bmod p.$$

The bias factor $-(d \bmod p)$ adjusts the row position of each block; it “draws back” consecutive blocks to the same row. Table II shows an example of BCRM using the same parameters as in Table I. In our application scenario, u is the global row address and v is the superblock index (to be discussed) in the row. The global row address is

³The cost of such division unit is probably less than that of a usual one because of the fixed denominator.

⁴We have reversed the notations of u and v from the equations in Gao [1993].

Table I. An Example of Layout for the Address Mapping from d to $\langle u, v \rangle$ Based on the CRM, with $m = 8$ and $p = 7$. For example, 11 is mapped to $\langle 3, 4 \rangle$ as $11 \bmod 8 = 3$ and $11 \bmod 7 = 4$. A row represents a value of u and a column represents a value of v . The numbers in bold type highlight the mapping for the first 14 blocks. In CRM, each column is intended to represent a memory bank, and consecutive blocks should be mapped evenly to multiple banks. Row-level locality does not matter.

	0	1	2	3	4	5	6
0	0	8	16	24	32	40	48
1	49	1	9	17	25	33	41
2	42	50	2	10	18	26	34
3	35	43	51	3	11	19	27
4	28	36	44	52	4	12	20
5	21	29	37	45	53	5	13
6	14	22	30	38	46	54	6
7	7	15	23	31	39	47	55

Table II. An Example of Layout for BCRM, Which Maps d to $\langle u, v \rangle$ with $m = 8$ and $p = 7$. For example, 11 is mapped to $\langle 7, 4 \rangle$ as $(11 - (11 \bmod 7)) \bmod 8 = 7$ and $11 \bmod 7 = 4$. A row represents a value of u and a column represents a value of v . The numbers in bold type highlight the mapping for the first 14 blocks. In BCRM, each row is intended to represent a memory row (all blocks in the same row resides in a single bank, and different rows may or may not be mapped to the same bank), so maintaining row-level locality or not makes a critical difference.

	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	49	50	51	52	53	54	55
2	42	43	44	45	46	47	48
3	35	36	37	38	39	40	41
4	28	29	30	31	32	33	34
5	21	22	23	24	25	26	27
6	14	15	16	17	18	19	20
7	7	8	9	10	11	12	13

Fig. 6. The logic flow of BCRM-based address mapping with page interleaving.

further decomposed as channel, DIMM, rank, bank, and row addresses as Figure 6 shows.

A merit of those non-power-of-two mappings using modulo operations is that fast and efficient logic design exists for modulo operation $(d \bmod p)$, if p is a small integer,

using the following residue arithmetic relationships [Teng 1983]:

$$\begin{aligned}(a + b) \bmod p &= (a \bmod p + b \bmod p) \bmod p \\ (a * b) \bmod p &= ((a \bmod p) * (b \bmod p)) \bmod p.\end{aligned}$$

In general cases, assume d is an n -bit number of bits $d_{n-1} \dots d_1 d_0$; then

$$d \bmod p = \sum_{i=0}^{n-1} (d_i \cdot (2^i \bmod p)) \bmod p.$$

The logic design can be further simplified if p is a certain preset number such that $(2^i \bmod p)$ has limited outcomes. For example, a simple logic design for the case of $p = 7$ has been shown [Teng 1983]. In this case, $(2^i \bmod p)$ is either 1, 2, or 4 for any value of i . The logic design is to count the number of 1s in the address bits for each possible outcome; multiply the numbers by 1, 2, and 4; respectively, and summarize the results. Since all those operations are modulo based with p being a small number, they can be implemented by simple hardware such as small ROMs or PLAs.

Finally, with m being a power of two, $(x \bmod m)$ is simply to select the least significant $\log_2(m)$ bits of x .

Proof of Correctness and Row Locality. BCRM is a “perfect” address mapping; that is, the mapping is mathematically a one-to-one function from $d \in [0, mp - 1]$ to $\langle u, v \rangle \in [\langle 0, 0 \rangle, \langle m - 1, p - 1 \rangle]$. This can be proven as follows. CRM has been proven to be a perfect address mapping [Gao 1993]; that is, it can be formulated as a one-to-one mathematical function cr from $[0, mp - 1]$ to $[\langle 0, 0 \rangle, \langle m - 1, p - 1 \rangle]$. BCRM can be formulated as a mathematical function bcr defined as the following:

$$\begin{aligned}bcr(d) &= f(cr(d)) \\ f(\langle u, v \rangle) &= \langle (u - v) \bmod m, v \rangle.\end{aligned}$$

Function f is obviously a one-to-one function from $[\langle 0, 0 \rangle, \langle m - 1, p - 1 \rangle]$ to $[\langle 0, 0 \rangle, \langle m - 1, p - 1 \rangle]$. Therefore, bcr is a one-to-one function from $[0, mp - 1]$ to $[\langle 0, 0 \rangle, \langle m - 1, p - 1 \rangle]$.

BCRM has the property of “row locality”: assume x is an integer that satisfies $(x \bmod p) = 0$, and then $x, x + 1, \dots, x + p - 1$ will be mapped to the same row; that is, the value of u is the same for those consecutive d values.

Using BCRM in Memory Address Mapping. We let u be the global row address and m be the number of all DRAM rows in the whole system, which is a power of two. We make v be the index of the memory *superblock*, which is defined such that each row hosts exactly seven memory superblocks. Correspondingly, d is the global memory superblock address. The scheme is feasible because in E³CC the number of utilized columns in a row is a multiple of 7, as 63 columns are utilized for every 64 columns. For example, for the example in Figure 3, a memory superblock is $112/7 = 16$ blocks, and for the example in Figure 4, a superblock is $28/7 = 4$ blocks. We assume that each row has 64 or a multiple of 64 supercolumns, which is valid for today’s memory device. As device capacity continues to increase, it will continue to hold. After v is generated, the memory controller generates a sequence of column addresses from v (plus the removed bits) for the corresponding data and ECC columns.

3.4. Extra ECC Traffic and ECC Cache

The E³CC design may incur extra ECC traffic because of the minimum burst length requirement in modern DDR x memories, eight for DDR3 and four for DDR2. In a full-rank DDR3 memory, a column access (read or write operation) consumes 64 B bandwidth by default. It takes two column accesses to transfer 64 B data and their 8 B

ECC, but the consumed bandwidth is 128 B, with 56 B bandwidth spent on unneeded ECC. The overhead is reduced to 24 B and 8 B, respectively, in 32-bit and 16-bit subranked DIMMs, and is eliminated in eight-bit subranked DIMMs. DDR2 supports a burst length of four, which can be used to reduce the overhead.

To reduce this overhead, we propose to include a small ECC cache in the memory controller to buffer and reuse the extra ECC bits. The cache is a conventional cache with a block size of 8 B, and it utilizes the spatial locality in memory requests. We find that a 64-block, fully associative cache is sufficient to capture the spatial locality for most workloads for our simulation setting; set-associative cache can also be used. For best performance, the cache size may increase with the maximum numbers of channels, ranks, and banks to be supported in the system.

It is worth noting that a partial write of an ECC supercolumn in DRAM does not require reading the ECC column first. DDRx supports the use of *data mask* (DM) for partial data write. The DM has eight bits. On a write operation, each bit may mask out eight data pins from writing, and thus the update can be done by 8B granularity. Furthermore, our design utilizes the burst-chop-4 feature of DDR3 on ECC writes to reduce the memory traffic overhead and power overhead from writing ECC bits.

3.5. Reliability and Extension

E³CC can correct single random bit error and detect double random bit errors happening in DRAM cells. The study on LOT-ECC [Udipi et al. 2012] identifies other types of failures and errors, including row failure, column failure, row-column failure, pin failure, chip failure, and multiple random bit errors, which can be covered by LOT-ECC. The first four types are stuck-at errors and can be covered by flipping the checksum bit in the error protection code [Udipi et al. 2012]. If the same idea is used in the error protection code of E³CC, E³CC can also detect a single occurrence of those failures, but it may not be able to correct them. E³CC cannot guarantee the immediate detection of a pin failure, because it may cause multiple-bit errors in an ECC word on E³CC. However, a pin failure also causes stuck-at errors, which with a high probability will be quickly detected by consecutive ECC checking. Similarly, a chip failure may also be quickly detected but cannot be recovered. Note that LOT-ECC is stronger than E³CC at the cost of higher storage overhead (26.5% vs. 14.1%) and lower power efficiency. We argue that the E³CC design is suitable for consumer-level computers and devices that require memory reliability enhancement with low impact on cost and power efficiency.

The flexibility of E³CC enables the use of other error protection coded for improved storage overhead, reliability, power efficiency, and performance. Conventional ECC memory using the Hamming-based (72, 64) SECDED code has a storage overhead of 12.5%. Hamming code is actually a special case of BCH codes with a minimum distance of 3 [Morelos-Zaragoza 2006]. For memory blocks of 64 B cacheline size, one may use very long BCH codes such as BCH(522, 512), BCH(532, 512), and BCH(542, 512) codes, which are SEC, DEC, and TEC (single-, double-, and triple-error correction), respectively, for a 512-bit data block. The storage overhead ratio is 1.95%, 3.9%, and 5.9%, respectively. To use BCH code in E³CC, another set of address mapping logic may be needed.

4. EXPERIMENTAL METHODOLOGIES

4.1. Power Simulation Platform

We have built a detailed main memory simulator for the conventional DDRx and the subranked memory system and integrated it into the Marssx86 [Patel et al. 2011] simulator. In the simulator, the memory controller issues device-level memory commands based on the status of memory channels, ranks and banks, and schedules requests by

Table III. Major Simulation Parameters

Parameter	Value
Processor	4 cores, 3.2 GHz, 4-issue per core, 14-stage pipeline
Functional units	2 IntALU, 4 LSU, 2 FPALU
IQ, ROB, and LSQ size	IQ 32, ROB 128, LQ 48, SQ 44
Physical register num	128 Int, 128 FP, 128 BR, 128 ST
Branch prediction	Combined, 6 k Bimodal + 6 k Two-level, 1 K RAS, 4 k-entry and 4-way BTB
L1 caches (per core)	64 KB Inst/64 KB Data, 2-way, 16 B line, hit latency: 3-cycle Inst, 3-cycle Data
L2 cache (shared)	4 MB, 8-way, 64 B line, 13-cycle hit latency
Memory	DDR3-1600, 2 channels, 2 DIMMs/channel, 2 ranks/DIMM, 8 banks/rank
Memory controller	64-entry buffer, 15 ns overhead for scheduling and timing
DDR3 DRAM latency	DDR3-1600:11-11-11

Table IV. Parameters for Calculating Power for 2 Gbit, x8 DDR3-1600(11-11-11) Device

Parameters	Value
Normal voltage	1.5 V
Active precharge current	95 mA
Precharge power-down standby current	35 mA
Precharge standby current	42 mA
Active power-down standby current	40 mA
Active standby current	45 mA
Read burst current	180 mA
Write burst current	185 mA
Burst refresh current	215 mA

hit-first and read-first policy, using cacheline interleaving and page interleaving. We did experiments with the two cacheline-interleaving schemes in Figure 5; they are very close in the average performance. The first cacheline-interleaving scheme is used in the presented results. The basic XOR-mapping [Zhang et al. 2000] scheme is used as the default configuration for page interleaving to reduce the bank conflict.

Table III shows the major parameters for the simulation platform. The simulator models Micron device MT41J256M8 in full detail. To model the power consumption, we integrate a power simulator into the simulation platform using the Micron power calculation methodology [Micron Technology, Inc.]. It also models the DDR3 low-power modes, with a proactive strategy to put memory device in power-down modes with fast exit latency. We follow the method of a previous study [Zheng et al. 2008] to break down memory power consumption into background, operation, read/write, and I/O power. The power parameters are listed in Table IV.

We simulate a quad-core system with each core running a distinct application from the SPEC2006 [Standard Performance Evaluation Corporation] benchmark suits. We follow the method used in a previous study [Kim et al. 2010] to group those benchmarks into three categories: MEM (memory-intensive), ILP (compute-intensive), and MIX (mix of memory-intensive and compute-intensive) based on their L2 cache misses per 1,000 instructions (L2 MPKI). The MEM applications are those benchmarks with L2 MPKI greater than 10 and ILP applications are those with less than one L2 MPKI.

Table V shows 18 four-core multiprogramming workloads randomly selected from those applications. The MEM workloads consist of memory-intensive applications, the ILP workloads contain compute-intensive applications, and the MIX workloads have the MEM benchmarks and ILP benchmarks mixed together. For each workload, we create a checkpoint after all the benchmarks in one workload running to its typical phase. Then we collect the detailed simulation results from user space for 200 million

Table V. Workload Specifications

Workload	Applications	Workload	Applications
MEM-1	mcf, libquantum, soplex, milc	ILP-1	gobmk, sjeng, gcc, namd
MEM-2	sphinx3, soplex, libquantum, mcf	ILP-2	gobmk, sjeng, gcc, perlbench
MEM-3	lbm, mcf, soplex, sphinx3	ILP-3	gobmk, sjeng, deaIII, perlbench
MEM-4	mcf, libquantum, lbm, sphinx3	ILP-4	gobmk, deaIII, namd, perlbench
MEM-5	soplex, milc, lbm, milc	ILP-5	sjeng, gcc, namd, perlbench
MEM-6	libquantum, lbm, milc, sphinx3	ILP-6	gcc, deaIII, namd, perlbench
MIX-1	lbm, sphinx3, deaIII, perlbench	MIX-2	lbm, milc, deaIII, namd
MIX-3	libquantum, gcc, milc, namd	MIX-4	soplex, sjeng, libquantum, gcc
MIX-5	gobmk, sphinx3, mcf, perlbench	MIX-6	mcf, soplex, gobmk, sjeng

instructions minimum for each program. The performance is characterized using SMT weighted speedup [Snavey et al. 2002] $\sum_i^n (IPC_{multi}[i]/IPC_{single}[i])$, where n is the total number of applications running, $IPC_{multi}[i]$ is the IPC value of application i running under a multicore environment, and $IPC_{single}[i]$ is the IPC value of the same application running alone.

5. EXPERIMENTAL RESULTS

In this section, we present and analyze the evaluation results of E³CC, including performance, memory traffic overhead, and power efficiency. We have conducted experiments with the full rank, 32-bit subranked, and 16-bit subranked memories, and with cacheline interleaving and page interleaving.

5.1. Overall Performance of Full-Rank Memories

It is straightforward to use a conventional DDR3-1600 non-ECC memory as a baseline for performance comparison. However, we also simulated a pseudo-DDR3-1422 ECC DRAM with a 72-bit width data bus running at 711MHz, since it provides the same raw data bandwidth as E³CC with a 64-bit data bus running at 800MHz. Note that the data rate of 1,422MHz is an artificial setting not used in practice.

Figure 7(a) compares the performance of the full-rank E³CC DDR3-1600 memory, without and with ECC cache, against the full-rank DDR3-1600 baseline and the full-rank pseudo-DDR3-1422 memory. We first focus on the memory-intensive workloads. Without ECC cache, E³CC incurs an average performance loss of 9.1% and 13.7% with cacheline and page interleaving, respectively. The performance loss comes from extra column access for ECC bits, which fetches more ECC bits than necessary and thus incurs bandwidth overhead. It will be analyzed in Section 5.3. ECC cache effectively reduces the bandwidth overhead. With ECC cache, the performance loss is reduced to 5.1% with cacheline interleaving and 7.4% with page interleaving. When E³CC is compared with pseudo-DDR3-1422, the performance overhead is 3.4% and 4.5% for cacheline and page interleaving, respectively.

For mixed workloads, E³CC without ECC cache incurs an average performance overhead of 3.2% and 5.7% with the cacheline and page interleaving, respectively. With ECC

(a) Full-rank E³CC memory vs. full-rank DDR3-1600 memory, with full-rank DDR3-1422 included. (b) 32-bit sub-ranked E³CC memory vs. 32-bit sub-ranked DDR3-1600 memory, with 32-bit sub-ranked DDR3-1422 included. (c) 16-bit sub-ranked E³CC memory vs. 16-bit sub-ranked DDR3-1600 memory, with 16-bit sub-ranked DDR3-1422 included.Fig. 7. Performance of the E³CC DDR3-1600 memories and the baseline DDR3-1600 memories of different rank sizes. E3CC denotes E³CC.

cache, the performance overhead is reduced to 2.0% for cacheline interleaving and 3.4% for page interleaving. When E³CC is compared with DDR3-1422, the performance loss is 1.0% and 2.0%, respectively. For ILP workloads, the average performance overhead is under 0.3% for both interleaving schemes, as ILP workloads are not sensitive to memory performance from the beginning.

5.2. Overall Performance of Subranked Memories

Figure 7(b) compares the performance of 32-bit subranked E³CC DDR3-1600 memories with subranked DDR3-1600 baseline and pseudo-DDR3-1422 memories. With sub-ranking, the bandwidth overhead from fetching ECC bits is reduced significantly, as discussed in Section 3.4. For memory-intensive workloads and without ECC cache, the average performance overhead is 5.6% and 9.5% with cacheline and page interleaving, respectively. With ECC cache, the overhead is reduced to 3.7% and 5.5%, respectively. When E³CC is compared with DDR3-1422, the overhead is further reduced to 1.3% and 1.6%, respectively. For mixed workloads and without ECC cache, the performance overhead is 2.3% and 4.0% from DDR3-1600 with cacheline and page interleaving,

Fig. 8. The extra read memory traffic caused by E³CC when ECC cache is used.

respectively; and with ECC cache, the overhead is reduced to 1.7% and 2.5%, respectively. When E³CC is compared with DDR3-1422, the performance difference is almost negligible.

Figure 7(c) compares the performance of 16-bit subranked memories of the E³CC DDR3-1600 type against the subranked DDR3-1600 baseline and the pseudo-DDR3-1422 type. For memory-intensive workloads and without ECC cache, the performance overhead is 4.3% and 6.5% for cacheline and page interleaving, respectively. With ECC cache, the overhead is reduced to 2.9% for cacheline interleaving and 3.6% for page interleaving. When compared to DDR3-1422, E³CC even improves the performance on page interleaving. Although this scenario seems to be counterintuitive, it is possible because the fetch of extra ECC bits has an effect of prefetch.

5.3. Memory Traffic Overhead and ECC Cache

Figure 8 presents memory traffic overhead caused by E³CC when ECC cache is used. As memory read is critical to the system performance and only memory-intensive workloads are sensitive to the memory traffic overhead, the figure presents only the memory read traffic for the memory-intensive workloads. The average overhead is 29.0%, 17.1%, and 6.5% for full-rank, 32-bit subranked, and 16-bit subranked memories, respectively, with cacheline interleaving. And with page interleaving, the overhead is 9.8%, 7.3%, and 4.1% for those three rank configurations, respectively. The overhead is less with page interleaving than with cacheline interleaving because page interleaving retains row-level spatial locality in the program address space and the ECC cache can well capture the spatial locality. ECC cache is still effective for cacheline interleaving, as it still captures the spatial locality within multiple blocks sharing the same ECC column.

Figures 9(a) and 9(b) show the ECC cache read hit rate for cacheline and page interleaving, respectively. Overall, ECC cache has a high read hit rate for memory-intensive workloads. The average hit rates are 62.5% and 87.2% for x64-rank DIMM with cacheline interleaving and page interleaving, respectively. Cacheline interleaving distributes the memory requests among all the ranks, which improves the accessing parallelism and reduces the row buffer hit rate. Therefore, the ECC cache hit rate for cacheline interleaving is smaller than that of the page-interleaving mode. As the rank size decreases to x32 and x16, the cache hit rate is reduced because the program locality is further broken as row buffer size shrinks. On average, the row buffer hit rates are 46.4% and 40.3% with cacheline interleaving with x32 and x16 rank size, respectively, and 78.1% and 62.5% with page interleaving, respectively.

5.4. Power Efficiency of E³CC Memories

As memory-intensive workloads are power-hungry applications, we focus on the power efficiency of memory-intensive workloads. Figure 10 compares the power breakdown of

Fig. 9. ECC cache read hit rate for mixed and memory-intensive workloads with cacheline and page interleaving.

Fig. 10. Memory power breakdown into operation, read/write, I/O and background power for full-rank and sub-ranked memories with and without ECC. E³CC denotes E³CC; ECC-cache is used. Only memory-intensive workloads are shown. Sub-ranked memories without ECC are used as reference baseline, with power calculated as that of regular sub-ranked memories plus 12.5% overhead. Both the baseline and E³CC are DDR3-1600 type.

full-rank and subranked memory without and with ECC. Subranked memories without ECC are shown to give a baseline reference for the corresponding subranked E³CC memories, so as to show the power efficiency of E³CC in each type of subranked memory. Both are DDR3-1600 type. Consistent with previous studies [Zheng et al. 2008; Ahn et al. 2009b], subranked memories reduce memory operation power significantly as the subrank size decreases. The background power is also reduced moderately because of the increased number of subranks over the number of ranks, as more subranks can be put into fast power-down mode to save background power. For page interleaving, operation power is not a significant component, because the row-buffer hit rate of those workloads is high and therefore a lesser number of precharges and activations are needed.

E³CC achieves good power efficiency when compared with the baseline. The full-rank baseline is actually the conventional full-rank ECC DIMM. The subranked baselines correspond to a design that extends subranked memories ideally to implement ECC; for example, for 32-bit subranked memory of four x8 device per subrank, it extends the subrank bus to 36-bit and adds a x4 device to store ECC bits, assuming that the x4 device consumes half power of a x8 device.

We first focus on the full-rank memory. For cacheline interleaving, E³CC increases the overall power rate of full-rank memory by 5.6% from the baseline. Regarding each power component, the read/write power increases by 52.4% and I/O power by 23.7%, but the background power decreases by 9.9% and operation power by 4.5%. For page interleaving, E³CC is almost the same as the baseline, with the overall power rate decreased by 0.1%. It increases the read/write power by 34.0% and I/O power by 9.1% but decreases the background power by 10.8% and operation power by 25.1%. The read/write and I/O power increases come from accessing extra ECC bits. ECC cache is more effective on page interleaving than on cacheline interleaving, and therefore the increases are less on the page interleaving than on the cacheline interleaving. The background power is reduced because, compared with the baseline, there are eight devices per rank instead of nine. Operation power is reduced partially for a similar reason, as there are eight devices per rank to precharge and activate instead of nine. Additionally, we have found that the average row buffer miss rate of page interleaving is 15.9% with E³CC and 17.5% with the baseline, which means less precharge and fewer activations with E³CC. For all settings, because of the BCR mapping, E³CC tends to have a lower row-buffer miss rate than the baseline.

The power efficiency of E³CC vs. the baseline increases with subranked memory because of decreasing memory traffic overhead. For 32-bit subranked memory with cacheline interleaving, E³CC on average reduces the overall power rate by 1.1%, with a 10.7% decrease of background power, 9.9% decrease of operation power, 22.3% increase of read/write power, and 8.0% increase of I/O power. With page interleaving, on average, it reduces the overall power rate by 4.6%, with a 10.9% decrease of background power, 20.2% decrease of operation power, 13.6% increase of read/write power, and 0.6% increase of I/O power. For 16-bit subranked memory with cacheline interleaving, E³CC on average reduces the overall power rate by 7.7%, with changes of -11.0%, -14.7%, +1.8%, and -5.2% on background, operation, read/write, and I/O power, respectively. With page interleaving, it on average reduces the overall power rate by 6.1%, with changes of -10.5%, -13.4%, +5.4%, and -2.1% on background, operation, read/write, and I/O power, respectively.

We have also found that while ECC cache reduces the memory ECC read traffic efficiently, it does not reduce the memory ECC write traffic as well because there is much less spatial locality in the write traffic than in the read traffic. The memory ECC write traffic is caused by last-level cache writebacks, whose spatial locality is mostly lost because the relative timing of writebacks is different from the relative timing of

Fig. 11. The memory system MTTF for a 4GB memory system with various ECC schemes.

demand misses on the same set of memory blocks. Although the extra write traffic only has a moderate impact on the performance of those workloads, it does increase the read/write and I/O power. Our design uses the burst-chop-4 to reduce the ECC write traffic and power consumption. The use of a different processor cache writeback policy, for example, eager writeback [Lee et al. 2000] or a revised scheme, may restore the spatial locality in cache writebacks and therefore may help reduce this power increase. If so, E³CC will be even more power efficient.

5.5. Evaluation of Using Long BCH Code

As discussed in Section 3.5, the flexibility of E³CC may allow the use of very long BCH codes for improved storage efficiency and reliability. Because the storage overhead is reduced, power efficiency and performance will also be improved. We present the extra reliability by using long BCH codes in this section. We have applied a statistical MTTF model suggested by a previous study [Noorlag et al. 1980] and built a Monte-Carlo simulator that assumes that the generation of failure bit is independent and random and follows the Poisson process.

We simulated and evaluated the MTTF of a 4GB memory system with ECC capability grows from SEC to DEC and TEC (single-, double-, and triple-error correction) for the typical 64-bit data word length and a 512-bit data word length for long BCH codes, respectively. Figure 11 depicts the simulation results for a 4GB memory system with ECC capability grows from SEC to DEC and TEC. First, without ECC protection, the MTTF for the 4GB memory system is 0.014 year. With BCH(71,64), BCH(78, 64), and BCH(85, 64), which are SEC, DEC, and TEC codes for a 64-bit data block, the MTTF increases to 4.6, 164.7, and 1094.9 years with a low failure rate, and 1.6, 41.6, and 228 years with a higher failure rate. However, the overhead ratio increases from 10.9% with the SEC code to 21.2% and 32.8% with the DEC and TEC codes, respectively. The BCH(522,512), BCH(532, 512), and BCH(542, 512) codes can improve the MTTF to 3.2, 119.6, and 777.8 years with a low failure rate, and to 1.2, 29.8, and 164 years with a high failure rate, respectively. Note that in this discussion we only consider random errors in memory cells, which are on the rise with the increase of memory density. A pin failure may cause multibit errors, which may not be detected by the long BCH codes.

6. CONCLUSION

We have presented E³CC, a complete solution of memory error protection for sub-ranked and low-power memories. A novel address mapping scheme called BCRM has been found to implement page interleaving on E³CC without using expensive integer division. A simple and effective solution called ECC cache is presented to reduce extra ECC traffic. We have thoroughly evaluated the performance and power efficiency of E³CC. E³CC is suitable for consumer-level computers and mobile devices used in applications that require a certain degree of reliable computing but desire a low impact on cost and power efficiency. E³CC does not require any changes to memory devices or modules, further reducing the cost of manufacturing. Furthermore, using ECC or not can be configured at system booting time, giving users the flexibility to make a tradeoff between reliability, performance, and power efficiency.

ACKNOWLEDGMENTS

The authors appreciate the constructive comments from the anonymous reviewers. This work is supported in part by the National Science Foundation under grants CNS-0834475 and CNS-1117604.

REFERENCES

- AHN, J. H., JOUPPI, N. P., KOZYRAKIS, C., LEVERICH, J., AND SCHREIBER, R. S. 2009a. Future scaling of processor-memory interfaces. In *Proceedings of the International Conference on High Performance Computing Networking, Storage and Analysis*.
- AHN, J.-H., LEVERICH, J., SCHREIBER, R. S., AND JOUPPI, N. P. 2009b. Multicore DIMM: An energy efficient memory module with independently controlled DRAMs. *Computer Architecture Letters* 8, 1.
- BAUMANN, R. 2005. Soft errors in advanced computer systems. *IEEE Des. Test Comput.* 22, 3, 258–266.
- BORKAR, S. 2011. The exascale challenge. Keynote speech, PACT.
- BORUCKI, L. AND SCHINDLBECK, G. 2008. Comparison of accelerated DRAM soft error rates measured at component and system level. In *Proceedings of the IEEE International Symposium Reliability Physics*. 482–487.
- CUPPU, V., JACOB, B., DAVIS, B., AND MUDGE, T. 1999. A performance comparison of contemporary DRAM architectures. In *Proceedings of the 26th International Symposium on Computer Architecture (ISCA99)*. 222–233.
- DANILAK, R. 2006. Transparent error correction code memory system and method. US Patent No. 7,117,421.
- DELL, T. J. 1997. A white paper on the benefits of chipkill-correct ECC for PC server main memory.
- DINIZ, B., GUEDES, D., MEIRA, W. JR., AND BIANCHINI, R. 2007. Limiting the power consumption of main memory. In *Proceedings of the 37th International Symposium on Computer Architecture (ISCA07)*. 290–301.
- FUNG, F. Y. C. 2008. *A Survey of the Theory of Error-Correcting Codes*. Harvard-Radcliff Math Bulletin.
- GAO, Q. S. 1993. The Chinese remainder theorem and the prime memory system. In *Proceedings 20th of the International Symposium on Computer Architecture*. 337–340.
- HAERTEL, M. J., POLZIN, R. S., KOCEV, A., AND STEINMAN, M. B. 2012. ECC implementation in non-ECC components. US Patent No. 8,135,935.
- HAMMING, R. W. 1950. Error detecting and error correcting codes. *Bell System Technical J.* 29, 2, 147–160.
- HUR, I. AND LIN, C. 2008. A comprehensive approach to DRAM power management. In *Proceedings of the 14th International Symposium on High Performance Computer Architecture (HPCA08)*. 305–316.
- INTEL. 2002. Intel E7500 chipset MCH Intel x4 single device data correction (x4 SDDC) implementation and validation. <http://www.intel.com/content/www/us/en/chipsets/e7500-chipset-mch-x4-single-device-data-correction-note.html>.
- INTEL CORPORATION. *Intel 64 and IA-32 Architectures Optimization Reference Manual*. <http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-optimization-manual.html>.
- ITRS. 2010. International technology roadmap for semiconductors. <http://www.itrs.net/links/2010itrs/home2010.htm>.
- JOHNSTON, A. H. 2000. Scaling and technology issues for soft error rates. In *Proceedings of the 4th Annual Topical Research Conference on Reliability*.

- KIM, Y., HAN, D., MUTLU, O., AND HARCHOL-BALTER, M. 2010. ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers. In *Proceedings of the 16th International Symposium on High Performance Computer Architecture (HPCA)*. 1–12.
- LAWRIE, D. AND VORA, C. 1982. The prime memory system for array access. *IEEE Trans. Comput.* 435–442.
- LEE, H. H.-S., TYSON, G. S., AND FARRENS, M. K. 2000. Eager writeback—a technique for improving bandwidth utilization. In *Proceedings of the 33rd International Symposium on Computer Architecture*. 11–21.
- LEUNG, W., TAM, K. S., AND TWOREK, M. 2008. Transparent error correcting memory. US Patent No. 7,353,438 B2.
- LI, X., HUANG, M. C., SHEN, K., AND CHU, L. 2010. A realistic evaluation of memory hardware errors and software system susceptibility. In *Proceedings of the 2010 USENIX Annual Technical Conference*.
- JIANG LIN, J., ZHENG, H., ZHU, Z., GORBATOV, E., DAVID, H., AND ZHANG, Z. 2008. Software thermal management of DRAM memory for multicore systems. In *Proceedings of the ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*. 337–348.
- MAY, T. AND WOODS, M. 1979. Alpha-particle-induced soft errors in dynamic memories. *IEEE Trans. Electron Devices* 26, 1, 2–9.
- MICRON TECHNOLOGY, INC. DDR3 SDRAM system-power calculator. <http://www.micron.com/>.
- MORELOS-ZARAGOZA, R. H. 2006. *The Art of Error Correcting Coding*. 2nd ed. Wiley, Hoboken, New Jersey.
- MUKHERJEE, S. S., EMER, J., AND REINHARDT, S. K. 2005. The soft error problem: An architectural perspective. In *Proceedings of the 11th International Symposium on High Performance Computer Architecture (HPCA'05)*. 243–247.
- NOORLAG, D. J. W., TERMAN, L. M., AND KONHEIM, A. G. 1980. The effect of alpha-particle-induced soft errors on memory systems with error correction. *IEEE J. Solid-State Circuits* 15, 3, 319–325.
- PATEL, A., AFRAM, F., CHEN, S., AND GHOSE, K. 2011. MARSSx86: A full system simulator for x86 CPUs. In *Proceedings of the 48th Design Automation Conference*. 1050–1055.
- RAO, T. R. N. AND FUJIWARA, E. 1989. *Error Control Coding for Computer Systems*. Prentice Hall, Upper Saddle River, New Jersey.
- RIXNER, S., DALLY, W. J., KAPASI, U. J., MATTSON, P., AND OWENS, J. D. 2000. Memory access scheduling. In *Proceedings of the 27th International Symposium on Computer Architecture (ISCA'00)*. 128–138.
- SCHECHTER, S., LOH, G. H., STRAUS, K., AND BURGER, D. 2010. Use ECP, not ECC, for hard failures in resistive memories. In *Proceedings of the 37th International Symposium on Computer Architecture (ISCA'00)*. 141–152.
- SCHROEDER, B., PINHEIRO, E., AND WEBER, W.-D. 2009. DRAM errors in the wild: A large-scale field study. In *Proceedings of the ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*. 193–204.
- SEONG, N. H., WOO, D. H., SRINIVASAN, V., RIVERS, J. A., AND LEE, H.-H. S. 2010. Safer: Stuck-at-fault error recovery for memories. In *Proceedings of the 43rd International Symposium on Microarchitecture (MICRO'10)*. 115–124.
- SNAVELY, A., TULLSEN, D. M., AND VOELKER, G. 2002. Symbiotic job scheduling with priorities for a simultaneous multithreading processor. In *Proceedings of the ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*. 66–76.
- STANDARD PERFORMANCE EVALUATION CORPORATION. *SPEC CPU2006*. Standard Performance Evaluation Corporation. <http://www.spec.org>.
- TAKAGI, N., KADOWAKI, S., AND TAKAGI, K. 2005. A hardware algorithm for integer division. In *Proceedings of the 17th IEEE Symposium on Computer Arithmetic*.
- TENG, M.-H. 1983. Comments on 'the prime memory system for array access'. *IEEE Transaction on Computers* C-32, 1072.
- UDIPI, A. N., MURALIMANO HAR, N., CHATTERJEE, N., BALASUBRAMONIAN, R., DAVIS, A., AND JOUPPI, N. P. 2010. Rethinking DRAM design and organization for energy-constrained multi-cores. In *Proceedings of the 37th International Symposium on Computer Architecture (ISCA'10)*. 175–186.
- UDIPI, A. N., MURALIMANO HAR, N., BALASUBRAMONIAN, R., DAVIS, A., AND JOUPPI, N. P. 2012. LOT-ECC: Localized and tiered reliability mechanisms for commodity memory systems. In *Proceedings of the 39th International Symposium on Computer Architecture (ISCA'12)*. 285–296.
- WARE, F. A. AND HAMPEL, C. 2006. Improving power and data efficiency with threaded memory modules. In *Proceedings of IEEE International Conference on Computer Design*.
- YOON, D. H., MURALIMANO HAR, N., CHANG, J., RANGANATHAN, P., JOUPPI, N. P., AND EREZ, M. 2011. FREE-p: Protecting non-volatile memory against both hard and soft errors. In *Proceedings of the 17th International Symposium on High Performance Computer Architecture (HPCA'11)*. 466–477.

- YOON, D. H. AND EREZ, M. 2010. Virtualized and flexible ECC for main memory. In *Proceedings of the 15th International Conference on Architecture Support for Programming Languages and Operating Systems (ASPLOS)*. 397–408.
- ZHANG, Z., ZHU, Z., AND ZHANG, X. 2000. A permutation-based page interleaving scheme to reduce row-buffer conflicts and exploit data locality. In *Proceedings of the 33rd International Symposium on Microarchitecture (MICRO'00)*. 32–41.
- ZHENG, H., LIN, J., ZHANG, Z., GORBATOV, E., DAVID, H., AND ZHU, Z. 2008. Mini-rank: Adaptive DRAM architecture for improving memory power efficiency. In *Proceedings of the 41st International Symposium on Microarchitecture (MICRO'08)*. 210–221.

Received November 2012; revised March 2013; accepted July 2013