SECRET: A Selective Error Correction Framework for Refresh Energy Reduction in DRAMs

CHUNG-HSIANG LIN and DE-YU SHEN, National Taiwan University

YI-JUNG CHEN, National Chi Nan University

CHIA-LIN YANG, National Taiwan University

CHENG-YUAN MICHAEL WANG, Macronix International Co., Ltd

DRAMs are used as the main memory in most computing systems today. Studies show that DRAMs contribute to a significant part of overall system power consumption. One of the main challenges in low-power DRAM design is the inevitable refresh process. Due to process variation, memory cells exhibit retention time variations. Current DRAMs use a single refresh period determined by the cell with the largest leakage. Since prolonging refresh intervals introduces retention errors, a set of previous works adopt conventional error-correcting code (ECC) to correct retention errors. However, these approaches introduce significant area and energy overheads. In this article, we propose a novel error correction framework for retention errors in DRAMs, called *SECRET* (selective error correction for refresh energy reduction). The key observations we make are that retention errors are hard errors rather than soft errors, and only few DRAM cells have large leakage. Therefore, instead of equipping error correction capability for all memory cells as existing ECC schemes, we only allocate error correction information to leaky cells under a refresh interval. Our *SECRET* framework contains two parts: an offline phase to identify memory cells with retention errors given a target error rate and a low-overhead error correction mechanism. The experimental results show that among all test cases performed, the proposed *SECRET* framework can reduce refresh power by 87.2% and overall DRAM power up to 18.57% with negligible area and performance overheads.

Categories and Subject Descriptors: B.3.2 [Memory Structures]: Design Styles—DRAMs

General Terms: Design, Management, Performance

Additional Key Words and Phrases: DRAM-based main memory, refresh power reduction, error control codes, retention errors

ACM Reference Format:

Chung-Hsiang Lin, De-Yu Shen, Yi-Jung Chen, Chia-Lin Yang, and Cheng-Yuan Michael Wang. 2015. SECRET: A selective error correction framework for refresh energy reduction in DRAMs. *ACM Trans. Architect. Code Optim.* 12, 2, Article 19 (June 2015), 24 pages.
DOI: <http://dx.doi.org/10.1145/2747876>

Extension of a Conference Paper: C.-H. Lin, D.-Y. Shen, Y.-J. Chen, C.-L. Yang, and C.-Y. Wang, “SECRET: Selective Error Correction for Refresh Energy reduction in DRAMs” at ICCD ’12.

This work was partially supported by the National Science Council (Ministry of Science and Technology) of Taiwan under grants NSC 101-2220-E-002-017-, NSC 99-2218-E-260-004-, MOST-103-2220-E-002-005-, MOST-103-2622-E-002-034-, MOST-103-2221-E-260-034-, and MOST-102-2221-E-260-030-; Etron Technology Inc. NTU-101R7500 and NTU-104R104124; Excellent Research Projects of National Taiwan University 101R890822+ and 104R890822; and Macronix International Co., Ltd. 99-SC253 and 103-S-C05.

Authors’ addresses: C.-H. Lin, D.-Y. Shen, and C.-L. Yang, No. 1, Sec. 4, Roosevelt Rd., Taipei, Taiwan; emails: {f94040, r99118, yangc}@csie.ntu.edu.tw; Y.-J. Chen, No. 303, University Rd., Puli Township, Nantou County, Taiwan; email: yjchen@ncnu.edu.tw; C.-Y. M. Wang, No.16, Li-Hsin Road, Science Park, Hsin-chu, Taiwan; email: michaelwang@mxic.com.tw.

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 permissions@acm.org.

© 2015 ACM 1544-3566/2015/06-ART19 \$15.00

DOI: <http://dx.doi.org/10.1145/2747876>

1. INTRODUCTION

DRAMs are used as the main memory in most computing systems today. Studies show that DRAMs consume up to 40% system power in commercial servers [Lefurgy et al. 2003] and about 20% system power in mobile phones [Vargas 2005]. One of the main challenges in low-power DRAM design is the inevitable refresh process. Due to leakage in DRAM cells, periodic refresh operations to recharge DRAM cells are necessary to retain data. In commodity DRAMs, a single refresh period determined by the cell with the largest leakage is selected. A refresh operation incurs both reads and writes to DRAM cells, and refresh power contributes to a significant part of DRAM power [Ghosh and Lee 2007; Liu et al. 2012]. Therefore, to further reduce DRAM power, minimizing refresh operations is critical.

Researchers have proposed several approaches to reduce DRAM refresh power, such as disabling the refresh operations for memory blocks that have no data [Emma et al. 2008; Isen and John 2009] or are recently recharged by memory accesses [Ghosh and Lee 2007], or applying different refresh intervals to memory blocks independently according to the worst case in each block [Kim and Papaefthymiou 2003; Liu et al. 2012]. Another approach is to prolong the refresh interval and adopt conventional error-correcting code (ECC) methods, such as Hamming code or Bose-Chaudhuri-Hocquenghem (BCH), to correct retention errors [Emma et al. 2008; Wilkerson et al. 2010]. In these approaches, ECCs are applied to all DRAM cells. Therefore, these approaches not only come with significant area overheads but also incur performance and energy penalties, as decoding and encoding ECCs are required for every memory read/write. For example, in conventional (72, 64) Hamming code designs, eight DRAM chips are paired with an extra chip, which requires 12.5% area overhead and additional power consumption. Wilkerson et al. [2010] utilize BCH code to reduce the refresh power consumption of eDRAMs that are used as the last-level caches. In Wilkerson et al. [2010], all cache lines are protected by BCH, but a low-complexity decoding method can be adopted when there is no more than one error in a cache line. The area overhead of their method can be reduced by increasing the data size protected by a BCH code, but this incurs other overheads, such as bandwidth requirements. Although an optimization is proposed to solve this issue, the optimization is only suitable for caches, not for main memories.

In this article, we propose a novel error correction mechanism for retention errors in DRAMs, called *SECRET* (selective error correction for refresh energy reduction). *SECRET* is developed based on a concept called *selective error correction* (SEC). Retention errors are hard errors and are sparsely distributed over DRAM chips. According to Kim and Lee [2009], only $10^{-6}\%$ of the cells have data retention times shorter than 128ms, and $10^{-4}\%$ of the cells have data retention times shorter than 500ms. Therefore, if we have a priori knowledge of cells with high leakage rates, instead of equipping error correction capability in all memory cells as existing ECC schemes, we can allocate error-correcting bits for those cells only and utilize a refresh interval that is longer than their data retention times to reduce refresh power. This observation leads to a very different error correction design for retention faults from prior works.

The proposed *SECRET* framework contains a low-overhead error correction mechanism, SEC, tailored specifically for retention errors. SEC adopts Error-Correcting Pointers (ECP) [Schechter et al. 2010] for retention error corrections. An ECP stores the address of a faulty cell and an additional correcting bit to replace the faulty cell. ECP allows one-to-one mapping between a faulty cell and its correcting bit. Therefore, the ECP mechanism is good for correcting errors with their positions identified, such as hard errors. Moreover, ECP provides variable-strength error-correcting capability with much less overhead compared to other error correction methods, such as BCH [Schechter et al. 2010]. In the *SECRET* framework, an architect first decides the

Fig. 1. Distribution of retention times of a DRAM chip.

target retention error rates considering the relation among retention error rates, refresh intervals, and error correction overheads. To utilize SECRET at runtime, a one-time profiling process to collect actual retention time of memory chips is performed before a machine is first used. This step takes only 1 to 2 minutes for a 4GB DRAM. The addresses of leaky cells are stored in a file along with required error-correcting bits. During system booting, these data are then loaded into the main memory. To adapt to temperature variations in systems, SECRET also has the capability of adjusting the refresh interval accordingly at runtime. Through experiments, we demonstrate a systematic design space exploration method to guide an architect to derive target retention error rate and key SEC design parameters. Our experimental results show that the proposed SECRET framework can reduce refresh power by 87.2% and total DRAM power by 18.57% with negligible area and performance overheads.

The rest of this article is organized as follows. The preliminaries of DRAM refresh operations and the ECP mechanism are discussed in Section 2, and research works related to SECRET are discussed in Section 3. Section 4 describes the proposed SECRET framework. The area, power, and performance overheads of SECRET are discussed in Section 5. Section 6 shows the experimental setup. Section 7 presents the experimental results and analysis. Section 8 concludes the article.

2. PRELIMINARIES

2.1. DRAM Organization and Refresh Operations

A DRAM die is composed of one or more memory arrays, which are rectangular grids of DRAM cells. Each DRAM cell is composed of a transistor and a capacitor connected to the wordline and the bitline. Once the row decoder selects a row and the wordline is charged, all transistors of the row are activated, and the charges in the capacitors of the DRAM cells are passed to the sense amplifier through the bitlines. Therefore, DRAMs are read destructive, and all data are only stored in the row buffer when a row is opened.

Due to the limitation of materials, the charges stored in capacitors leak gradually over time. Therefore, a periodic refresh of DRAM cells is necessary to guarantee data integrity. The refresh rate is typically set to be higher than the leakage rate of the fastest-leaking DRAM cells [Emma et al. 2008]. Figure 1 shows the retention time distribution measured from real DRAM chips [Kim and Lee 2009]. The refresh interval is set to T_{ref_min} as indicated by the dotted line, which is the shortest data retention time of all DRAM cells. In a conventional DRAM module, the refresh interval is usually set to 64ms. However, frequent refresh operations may cause high energy and performance overheads, as each row of a DRAM bank is read into the row buffer and then written back to recharge all DRAM cells of the row when the refresh operation is triggered for

Fig. 2. (a) Illustration of an ECP. (b) Hardware implementation of the ECP₁ decoder [Schechter et al. 2010].

the bank. Setting the refresh interval to the worst-case cell is overprovisioned for most DRAM cells. As mentioned in Section 1, Kim and Lee [2009] show that only $10^{-6}\%$ of the cells have data retention times shorter than 128ms, and $10^{-4}\%$ of the cells have data retention times shorter than 500ms. Therefore, if we can correct the retention errors of DRAM cells with a high leakage rate, we can utilize a refresh interval that is longer than their data retention times to reduce refresh power while data correctness is guaranteed.

2.2. Error-Correcting Pointer

ECP [Schechter et al. 2010] is proposed for correcting wear-out errors in phase change memories. Since wear-out errors can be known in advance, only faulty cells are equipped with ECPs. The structure of an ECP is shown in Figure 2(a). An ECP is composed of p -bit correction pointer and a replacement cell. The correction pointer indicates the position of the faulty cell. The replacement cell stores the value that should be stored in the faulty cell, as the faulty cell is corrupted and cannot be trusted to hold the value correctly. For example, in Figure 2(a), the 510th bit is known to be a faulty cell. Its ECP has the pointer set to 510; the replacement cell stored the correct bit value. Since there are 2^p data bits with p address bits, a p -to- 2^p row decoder is required to align the replacement cell with the error to perform decoding. Figure 2(b) shows the logic circuit of the ECP₁ decoder [Schechter et al. 2010] that corrects one error in 512 data bits. The ECP scheme is very flexible to correct multiple errors, as each identified error is protected by a specific ECP. In the meantime, ECP keeps its decoding simplicity for correcting multiple errors. However, ECP can only handle hard errors that are known in advance, not soft errors, as ECPs have no ability of detecting errors.

3. RELATED WORK

For reducing the number of unnecessary refresh operations, the most straightforward method is directly measuring DRAM temperature, leakage energy consumption, or voltage to adjust the refresh interval accordingly [Kim et al. 2007; Tran et al. 2011; Tsai et al. 2008]. However, to apply these techniques, the design and the manufacturing process of conventional DRAMs must be adapted accordingly [Zheng et al. 2008].

To reduce the number of refresh operations without altering the design and manufacturing process of conventional DRAMs, one set of techniques reduces refresh operations by exploiting the access pattern [Agrawal et al. 2013; Ghosh and Lee 2007], or the properties of data in DRAMs [Isen and John 2009; Liu et al. 2011; Moshnyaga et al. 2007; Patel et al. 2005]. Since DRAM cells are recharged after being read or written,

smart refresh [Ghosh and Lee 2007] employs a time-out counter for each DRAM row to eliminate unnecessary refresh on rows that are recently accessed. However, smart refresh cannot reduce DRAM refresh power when the memory is idle. Moshnyaga et al. [2007] target DRAM banks that are utilized as swap caches and propose to stop refreshing the DRAM banks if the data are clean and not accessed for a long time. This mechanism is directed by the OS, and data lost due to retention errors are recovered by retrieving back from the lower levels of the memory hierarchy. Targeting eDRAMs as on-chip caches, Agrawal et al. [2013] propose to reduce refresh operations of frequently accessed and not-to-be-accessed cache lines, as frequently accessed cache lines are recharged whenever it is accessed, and cache lines that are not accessed for a long time store the data that are no longer required.

Since only the integrity of valid or critical data matters, Isen and John [2009] propose eliminating refresh operations on memory regions that are not allocated or are not written yet after allocation. Flikker [Liu et al. 2011] lowers the refresh rate of a memory region that stores the noncritical data, which do not affect the correctness of the application. Since the leakage currents of DRAM cells are always unidirectional, Patel et al. [2005] propose to deactivate refresh operations of the blocks with clusters of zeros. However, this approach not only changes the DRAM structure but also increases the cost per bit of the DRAM chip.

Another set of methods reduces refresh energy by exploiting the characteristics of the DRAM cells. Since the retention time of each DRAM cell varies, a DRAM chip can be partitioned into several small blocks or bins, and the refresh interval is determined for each block or bin [Kim and Papaefthymiou 2003; Liu et al. 2012]. Venkatesan et al. [2006] exploit the variation of DRAM retention times and propose allocating DRAM pages to cells with long retention time first to prolong the refresh interval.

Several methods are similar to the proposed SECRET framework that utilizes ECC to correct retention errors due to prolonged refresh intervals. Katayama et al. [1999] propose using Reed-Solomon code to prevent retention errors and reduce refresh rates. They also propose adjusting the refresh interval according to the error rates. However, the decoding time of Reed-Solomon is quite long, and the mechanism incurs noticeable performance degradation. Targeting eDRAMs as on-chip caches, Emma et al. [2008] utilize the Berger code to reduce refresh rates of eDRAM caches with the assumption that the correct data can be found in the lower-level memory hierarchy when the Berger code detects retention errors. Wilkerson et al. [2010] utilize BCH code to provide multibit correction for eDRAM caches to reduce refresh power. They also propose a low-complexity decoding method that can be adopted when there is no more than one error in a cache line. All works that adopt ECC to reduce the refresh rate allocate ECC for all memory blocks without considering the properties of retention errors.

Due to the advance of technology, recent studies show that the retention time of a DRAM cell may shift randomly between multiple retention time states, a phenomenon called *variable retention time* (VRT) [Khan et al. 2014; Liu et al. 2013; Yaney et al. 1987]. The VRT phenomenon complicates retention time profiling because a cell's retention time may deviate from the profiled retention time from several times. Liu et al. [2013] present a comprehensive quantitative study of retention behavior of DRAMs with a severe VRT problem. Khan et al. [2014] analyze the efficacy of common error mitigation techniques in DRAM chips with VRT problems.

4. THE SECRET FRAMEWORK

4.1. Main Idea

Figure 3 shows the overview of the proposed SECRET framework. As mentioned in Section 1, the main design concept of SECRET is that with a priori knowledge of leaky

Fig. 3. SECRET framework.

cells under a refresh interval, we could construct a resource-efficient error correction scheme. Therefore, the center of SECRET is an SEC mechanism that equips the error correction capability only to identified leaky cells. To achieve more refresh power reduction (i.e., a longer refresh interval), SEC must be able to tolerate more retention errors, which in turn requires more resource overhead. Therefore, an architect must carefully evaluate the power savings versus overhead trade-off to decide the target error rate for which that SEC is built. A one-time profiling process to collect actual retention times of memory chips is performed before a machine is first used. Offline profiling can be performed by system providers or users if the profiling utility is built in the system. The addresses of leaky cells are stored in a file along with required error-correcting bits. During system booting, the addresses of leaky cells and their correcting bits are then loaded into the main memory. To maintain the robustness against variations in data retention times due to the change in operating temperature, our SECRET framework adjusts the refresh interval at runtime to adapt to temperature variations. Next, we describe the details of the major building blocks of SECRET.

4.2. Selective Error Correction

To design the SEC mechanism, there are two main design issues. First, what kind of error correction method should be used to correct retention errors? Second, how do we locate memory cells with retention errors at runtime?

A good error correction method for SEC should support the variable-strength error-correcting capability. The distribution of retention errors is random within a DRAM chip. In other words, the number of retention errors of a memory block protected by error-correcting bits varies. Since retention errors are hard errors, all leaky memory cells that are identified offline need to associate with error-correcting bits. Therefore, a most cost-effective error correction method for SEC is to allow memory blocks with more retention errors to have stronger error correction capability and vice versa. To achieve this, the ECP scheme mentioned in Section 2.2 is a perfect candidate for retention errors. The number of ECPs of a protected memory block is equal to the number of its leaky cells. Although BCH is also able to provide variable-strength error-correcting capability, its decoding and encoding processes are much more complex than those of ECP.

To locate memory cells with retention errors at runtime, we partition memory space into equal-sized regions, and ECPs of the same memory region are placed contiguously in the memory. To locate the ECPs of each region, the SEC mechanism utilizes an ECP directory to index the ECPs. As shown in Figure 4, the ECP directory is indexed by region number, and each ECP directory entry records the number of ECPs in the corresponding memory region and the physical address of the first ECP of the region. After an ECP is found, the memory cell with the retention error is then located. Therefore, for each memory request, the ECP directory is checked to see if the corresponding memory region contains faulty cells (i.e., a nonzero number in the field of the number

Fig. 4. Illustration of utilizing the ECP directory to index ECPs of each region and locate leaky cells by ECPs.

ECP Director Entries			ECPs	
Region Number (Tag)	ECPs (#)	ECP Address	ECPs	Dirty
0x8086	2	0xffffffff3	...	0
0xcc04	1	0xffffffff7	...	1
0x40c8	0	0xffffffffe8	...	0
⋮				
0x2240	3	0xffffffd6	...	0

Fig. 5. Architecture of the SEC cache.

of ECPs). If there are faulty cells, all ECPs of that memory region are fetched from the memory, and the memory controller compares the pointers of ECPs and the addresses of the requested data. The ECPs that match the addresses are decoded to perform error correction.

From the preceding discussion, we can see that the SEC mechanism introduces extra memory requests for fetching the ECP directory and ECPs. To minimize the performance overhead of additional memory accesses, we propose an optimization that caches the ECP directory and ECPs in the memory controller. When an ECP directory entry and ECPs of that region are fetched from memory to the memory controller, they are kept in the SEC cache in the memory controller. Figure 5 shows the architecture of the SEC cache. Each SEC cache line stores the information of one ECP directory entry and all ECPs in that region. To have an ECP directory entry and ECPs of the corresponding memory region cached in the same set, the SEC cache is indexed by region numbers instead of the physical addresses of the ECP directory or ECPs. The dirty bit is used to indicate if the replacement cells of ECPs are updated. If a replacement is required, a write-back operation of ECPs is triggered by setting the corresponding dirty bit. Please note that the ECP directory part in the SEC cache is read-only and does not have to be written back to memory.

The decision of the SEC cache configuration is crucial to system performance. The cache configuration includes the number of sets, the number of ways, and the cache line size. The numbers of sets and ways of the SEC cache affect the total number of memory regions that can be cached and the hit rate of the SEC cache. Since the SEC cache that achieves acceptable cache hit rate is totally decided by the access pattern of the memory regions, the numbers of ways and sets of the SEC cache should be decided along with the configuration of the microarchitecture during system design time where the design space exploration is performed for the target workloads.

For the SEC cache line size, it decides the number of retention errors in a region that can be cached. Thus, to decide the appropriate cache line size, an architect needs to estimate the number of retention errors in a region given an error rate. However, during system design time, the actual distribution of leaky cells of DRAM chips to be installed in a machine is not known. Therefore, to decide the SEC cache line size at design time, we assume that the geometric distribution of retention times is uniformly

Fig. 6. System architecture of the SEC mechanism.

random distribution, as the leakage currents that cause retention errors are from several different sources, such as band-to-band tunneling current and body effect [Roy et al. 2003], which may have different geometric distribution properties. A similar assumption of uniformly random geometric distribution of leaky cells is also made in Kim and Lee [2009]. Moreover, memory cells in a region do not usually locate in the same chip. In a typical DRAM system design, the requested data are interleaved in all DRAM chips in a rank. For example, a contiguous 64-byte data block is partitioned into eight 8-byte data blocks, and these 8-byte data blocks are distributed across eight DRAM chips in this rank. Therefore, we believe that strong locality among retention errors are not common cases. Extreme cases only happen when excursion occurs during DRAM manufacturing. However, to be able to accommodate variations among real DRAM chips, we could set the line size larger than what is estimated with leaky cell distribution generated based on uniformly random distribution. In this article, we set the line size to be 1.5 times of the worst-case estimated number.

Figure 6 shows the system architecture of the SEC mechanism. The memory controller contains the SEC cache and the Error-Correcting Unit. The Error-Correcting Unit is similar to the ECP₁ decoder described in Schechter et al. [2010] that decodes one ECP at one time. However, unlike phase change memories, the values of faulty DRAM cells with retention errors are not stuck at 0 or 1, and the differential encoding is not applicable. We explicitly store the replacement value in the replacement cell and slightly modify the decoder accordingly. When the memory controller receives a memory request, the memory controller issues the request to both the memory arrays and the SEC cache. If the requested ECPs are found in the SEC cache, the error correction information is provided to the Error-Correcting Unit from the SEC cache directly. In this case, no extra memory requests are issued. If it is an SEC cache miss, the memory controller first issues a memory access to fetch the ECP directory entry from memory. One or more memory accesses for the ECPs are needed only when the ECP directory entry indicates that there are errors in the requested region.

4.3. Offline Phase: ECP Directory/ECP Construction

During the offline phase, we first identify memory cells with retention errors given a target error rate and then build the ECP directory and ECPs accordingly.

We characterize data retention times of DRAM cells in a method similar to the testing process proposed in Venkatesan et al. [2006] and Katayama et al. [1999]. The process prolongs the refresh interval incrementally and checks if the DRAM cells can retain data. To test the DRAM chips, all refresh operations are disabled and the row buffer is managed by the close-page policy, where the contents of the row buffer is immediately written back to its row so that the row buffer is prepared for another access to a different row. The testing process writes all 1's to the chips, waits for a refresh interval,

and reads the data back to see if they are intact. The process stops until the number of retention errors meets the target number or meets the number of ECPs that an SEC cache line can keep. Note that in addition to the DRAM cells that cause retention errors given the target refresh interval, the set of memory cells used by the refresh interval adaption process described in Section 4.4 should be identified in the offline phases as well. Details of how to decide this set of memory cells are described in Section 4.4.

The offline profiling process is performed once before a machine is first used, and the time required for retention time testing is quite small. Since the offline profiling is performed after DRAM modules are installed in a machine, the process does not affect the manufacturing testing complexity. As mentioned earlier, the profiling process includes (1) writing data into the DRAM, (2) waiting for the target retention time, and (3) reading the data out. Assume that we perform the profiling process on a 4GB DRAM, which has 512K rows, and the tested refresh intervals range from 50ms to 1s. According to Micron [2006], 3.4 μ s and 2.8 μ s are needed for writing and reading a row, respectively. Since step (2) can be overlapped with reads/writes of other rows, one-pass profiling time can be estimated as reading/writing 512K rows, which is equal to 3.3 seconds. Therefore, with 20 refresh threshold steps, the overall profiling process takes about 66 seconds.

For modern DRAM chips, the retention time of a DRAM cell may be affected by the data pattern of its neighboring cells, or it may just shift randomly between multiple retention time states. As mentioned in Section 3, this is the VRT phenomenon [Khan et al. 2014; Liu et al. 2013]. Since chips with severe VRT problems are prone to retention errors and would need ECCs with high error correction capabilities, DRAM manufacturers tend to identify these chips as defects during in-house testings for the sake of quality and cost. However, it is possible that DRAM chips installed in a machine still have few VRT problems. To guarantee the robustness of SECRET in DRAM systems with VRT problems, we can use various data patterns to collect the retention time of a DRAM cell in the worst case to minimize the data pattern-induced VRT. For the random VRT problem, we can set a guard band during the profiling process—that is, assuming that the maximum to minimum retention time of a DRAM cell is no more than n times. For a cell that has profiled retention time x , we assume that its worst-case retention is x/n . For DRAM chips that are packaged and installed in a machine, since the chips have few VRT problems as mentioned previously, the value of n should be small (e.g., no more than 2).

With the knowledge of the physical positions of leaky cells, we then construct the ECP directory and ECPs. Both the ECP directory and ECPs need to be placed in nonleaky cells. Since the whole ECP directory needs to be placed in contiguous memory cells, it is possible that we are not able to find a big enough memory segment without any retention errors. In this case, we adopt the triple-modular redundancy (3-MR) ECC that duplicates the original data twice to protect the ECP directory. Each ECP directory entry needs about 36 bits, with 4 bits for the number of ECPs and 32 bits for the ECP address. Therefore, three copies of an ECP directory entry needs 108 bits, which is around 14 bytes. To avoid three separate accesses to read an ECP directory entry that is protected by 3-MR, we place the three copies of an ECP entry consecutively in the same 64B data block so that all three copies can be accessed in one memory access. A majority vote is performed at the memory controller when these three copies of the ECP directory are different. The logic circuit of the majority vote for each bit only requires three AND gates and two OR gates, so it is quite simple and very fast. The placement of ECPs is more flexible than the ECP directory. Only the ECPs of the same region need to be stored contiguously in memory, not all ECPs of the DRAM. The constructed ECP directory (along with its starting address) and ECPs are stored in a file. During system booting, this error correction information is then loaded into memory. Since the

Fig. 7. Retention times of cells in Set 1 and Set 2 versus the refresh interval ref_n .

size of the ECP directory and ECPs is rather small as discussed in Section 5, another alternative is to utilize a small flash to store the ECP directory and ECPs. Here, we assume the ECP directory and ECPs are stored in the main memory.

4.4. Refresh Interval Adaptation

The data retention capability of memory cells are subjected to various system perturbations, such as temperature. Higher temperature leads to larger leakage current. Therefore, to ensure the robustness of the proposed SECRET framework, the refresh interval needs to be adjusted dynamically to adapt to temperature variations. Our approach is similar to the runtime refresh interval adaptation method in the prototype proposed by Katayama et al. [1999]. The basic idea of the refresh interval adaptation is to adjust the refresh interval so that the number of retention errors is kept constant. In other words, when unexpected retention errors are detected, the refresh interval should be scaled down. On the other hand, when expected retention errors are not detected, the refresh interval should be scaled up. To support this, in addition to identifying the memory cells with retention errors given a target error rate, we also need to identify the memory cells that may cause retention errors when fluctuations of temperature occur and then monitor these bits periodically. Therefore, in Section 4.4.1, the process of identifying the memory cells for refresh interval adaptation is described. Moreover, we need to identify the maximum leakage variation ratio during the monitoring period. With the maximum leakage variation ratio, all memory cells that may cause retention errors due to temperature change can be guaranteed to be identified. The calculation of the maximum leakage variation ratio is described in Section 4.4.2.

4.4.1. Profiling Memory Cells for Refresh Interval Adaptation. As shown in Figure 7, ref_n indicates the refresh interval value that reaches the target error rate. The set of bits with retention times between ref_n and ref_{n+1} represents the DRAM cells that have the shortest data retention times among the DRAM cells that are not expected to have retention errors if system temperature is close to the setting of the offline profiling. This set of bits is indicated as Set 2 in Figure 7. Memory cells in Set 2 should also be protected by ECPs to prevent retention errors when the leakage current increases. On the other hand, as shown in Figure 7, Set 1 is the set of DRAM cells that have the longest data retention times among those DRAM cells that should have retention errors. Therefore, SECRET periodically checks and corrects bits in Set 1 and Set 2. When there are no retention errors in Set 1, this indicates that the length of the refresh interval can be increased to further reduce refresh power, as the number of retention errors is lower than expected. When there are retention errors in Set 2, the

length of the refresh interval should be reduced to guarantee the correctness of the DRAM system.

4.4.2. Deduction of the Worst-Case Leakage Ratio. To guarantee that all retention errors are protected by ECPs, we need to ensure that Set 2 covers all bits that may cause retention errors due to temperature change under the worst case during the monitoring period. To achieve this, we need to find out the maximum leakage variation ratio (i.e., maximum/minimum leakage current) during the monitoring period under the extreme temperature change condition. Since retention time variation is proportional to that of leakage current, the maximum leakage variation ratio is also the maximum retention time variation ratio during the monitoring period. Let t denote the monitoring period, and let R denote the maximum retention time variation ratio (leakage variation ratio) when chip temperature is T in monitoring period t . We can deduce that to guarantee Set 2 (Set 1) covers all bits under the maximum leakage variations within t , ref_{n+1} (ref_{n-1}) in Figure 7 should be set as $ref_n \times R$ (ref_n/R).

As mentioned earlier, R is proportional to the maximum variation ratio of leakage current during the monitoring period t . According to Williams [2006], when the chip temperature is T , the amount of leakage current $I_0(T)$ is modeled by the following equation:

$$I_0(T) = I_0(25^\circ C) \times (T/300)^{1.8}. \quad (1)$$

Therefore, when the chip temperature is T and the maximum temperature increase in t is ΔT , the leakage variation ratio $R(T, t)$, which is the retention time variation ratio of the period t , can be calculated by the following equation:

$$\begin{aligned} R(T, t) &= \frac{I_0(25^\circ C) \times ((T + \Delta T)/300)^{1.8}}{I_0(25^\circ C) \times (T/300)^{1.8}} \\ &= ((T + \Delta T)/T)^{1.8}. \end{aligned} \quad (2)$$

According to Lin et al. [2007], given that the chip temperature is T and monitoring period t , ΔT can be modeled by

$$\Delta T = ((TR \times P + T_{amb}) - T) \times (1 - e^{-t/\tau}), \quad (3)$$

where TR is the thermal resistance, P is the peak power consumption of the chip, T_{amb} is the ambient temperature, and τ is the time for the temperature difference between T and $(TR \times P + T_{amb})$ to be reduced by $1/e$.

Thus, assuming that the chip has a maximum and minimum working temperature of T_{max} and T_{min} , the maximum to minimum retention time ratio R in the extreme case is

$$R = \max(R(T, t)), T_{min} \leq T \leq T_{max}. \quad (4)$$

Therefore, as mentioned earlier, Set 2(Set 1) should cover all bits with retention times between ref_n and $ref_n \times R$ (ref_n and ref_n/R). Equipping ECPs for bits in Set 2 can guarantee that there is no uncorrectable retention errors even with extreme temperature change during the monitoring period.

5. DISCUSSION ON OVERHEADS OF SELECTIVE ERROR CORRECTION

5.1. ECP Directory/ECPs

The numbers of bits/bytes required by each field of the ECP directory entry and an ECP are listed in Table I. As mentioned in Section 4.3, we use the 3-MR technique to protect the ECP directory. Thus, we have three copies of each ECP directory entry, which needs a total of 108 bits (14 bytes). The memory space allocated to the ECP directory is related

Table I. Numbers of Bits/Bytes of an ECP Directory Entry and an ECP

An ECP Directory Entry	
ECPs (#)	4 bits $\times$ 3 = 12 bits
ECP address	32 bits $\times$ 3 = 96 bits
Total	14 bytes
ECP	
Correction pointer	20 bits
Replacement cell	1 bit
Total	3 bytes

to the number of regions only and independent of the number of tolerable retention errors. For ECPs, because one ECP is used to correct one retention error, the amount of memory space occupied by ECPs is related to the number of retention errors. Assume that we have a 4GB DRAM system that is partitioned into 32K regions where the size of a region is 128KB. Each ECP needs 20 bits for recording the position of the error in the region and 1 bit for the replacement cell. With 10^{-7} retention error rate, the ECP directory and ECPs occupy only 0.01% of the memory space. Even with the retention error rate up to 10^{-4} , the memory space required for the ECP directory and ECPs is still very small—only 0.24%.

5.2. Error-Correcting Unit

Our Error-Correcting Unit adopts a decoder similar to the ECP₁ decoder shown in Figure 2(b). The ECP₁ decoder has a 9-to-512 bit row decoder to align the replacement cell with the leaky cell [Schechter et al. 2010]. The ECP₁ decoder has a reasonably small latency that is no more than one processor cycle [Schechter et al. 2010], but only one ECP can be decoded at a time. The block size of single memory access is also the size of data to be processed by each operation of the Error-Correcting Unit. In a DDR3 memory, the block size is 64 bytes. The possibility of having multiple errors in such a small unit is quite low. In our experiments, we see 0.05% data blocks with one error and only $1.04 \times 10^{-5}\%$ data blocks with two errors. No data blocks have three or more errors. Therefore, one or two ECPs are decoded only in very few cases. For the area and energy overheads, our synthesis results show that the Error-Correcting Unit needs only 0.014mm² area with 45nm technology, which is negligible compared to the memory controller.

5.3. Selective Error Correction Cache

The SEC cache size is determined by three configuration parameters: associativity, the number of sets, and the cache line size. The cache line size affects how many retention errors SEC can tolerate in a region, whereas the associativity and number of sets decide the hit rate of an SEC cache. A larger SEC cache can reduce extra memory accesses and tolerate more errors at the expense of higher overheads of accessing the SEC cache itself. As mentioned in Section 4.2, the design of an SEC cache is workload related and should be performed along with the microarchitecture during the system design time. In Section 7.1, we will present a systematic way to make a right design decision for the SEC cache.

5.4. Refresh Interval Adaption

The monitoring overheads of the refresh interval adaptation process mainly come from the extra memory accesses for reading the bits in Set 1 and Set 2 along with their ECP directory and ECPs. Assuming the monitoring performed for every second and 85°C peak temperature, we can infer that the maximum leakage variation ratio R is

Table II. System Configuration

Parameters	Value
Processor	4 cores, 2GHz, 1 hardware context per core
L1 caches (per core)	64KB Inst/64KB Data, 2-way, 64B line
L2 cache (shared)	8MB, 16-way, 64B line
DRAM	4GB, 2 channels, 2 DIMMs/channel, 1 rank/DIMM
Memory controller	32-entry request buffer
Memory rank	1GB, 1,333MHz DDR3-SDRAM, 8 banks/rank
Memory bank	16,384 rows/bank, 1,024 columns/bank
Channel bandwidth	8-byte/channel, 10.66GB/s
Row buffer management	open-page policy
Refresh configuration	64ms, one-channel, one-rank, all-bank policy
Major power parameters	$I_{DD5B} = 240\text{mA}$ (Burst Refresh Current) $I_{DD3N} = 62\text{mA}$ (Active Standby Current) $I_{DD2P1} = 40\text{mA}$ (Precharge-Power down current, fast-exit) $I_{DD2P0} = 12\text{mA}$ (Precharge-Power down current, slow-exit)
Major timing parameters	$t_{REFI} = 7.8\mu\text{s}$, $t_{RFC} = 110\text{ns}$
The SECRET framework	Region size of 128KB, 32K regions in a 4GB DRAM
ECP directory	32K entries

equal to 1.003607 according to the method described in Section 4.4. Thus, if we have a 4GB DDR3-1333 SDRAM system partitioned into 32K regions, with the retention time distribution taken from Kim and Lee [2009] and 10^{-6} target error rate, Set 1 and Set 2 contain 694 and 719 bits, respectively. Therefore, reading Set 1 and Set 2 incurs about 91KB/s bandwidth overheads, and accessing the ECP directory and ECPs incurs about 4MB/s bandwidth overheads. Since the channel bandwidth of a DDR3-1333 SDRAM is 10.66GB/s, the bandwidth overheads incurred by the adaptation process take no more than 0.1%.

6. EXPERIMENTAL SETUP

Our simulation framework is composed of three components: Wind River SIMICs [Magnusson et al. 2002], Ruby of Gems [Martin et al. 2005], and DRAMsim [Wang et al. 2005]. SIMICs is a full-system simulator that can execute target benchmarks on unmodified operating systems. To simulate memory and cache in details, Ruby that is integrated with DRAMsim is loaded into SIMICs. We simulate a Sun virtual machine called *Abisko* that runs a version of Solaris 10. All benchmarks are executed on a four-core CMP system sharing an 8MB L2 cache with 4GB DDR3 SDRAM. The power parameters of the DRAM system are set according to Micron MT41J128M8JP 1Gb DDR3 SDRAM [Micron 2006], assuming that the working temperature is no more than 85°C. We assume that when the target DDR3 system enters the idle state, it can be in the fast-exit mode or slow-exit mode. The detailed system configurations and DRAM power parameters are listed in Table II. For SECRET, the size of a region is set to 128KB, and the DRAM is partitioned into 32K regions. In the SECRET framework, the row buffer is closed as frequently as the baseline configuration, which is the DRAM system without SECRET and uses a 64ms refresh interval. We evaluate SECRET on three categories of workloads: SpecJBB for online transaction processing, PARSEC for multithreaded applications, and mixtures of Spec2006 for multiprogramming workloads. Table III lists the details of each workload. Each set of benchmarks is simulated for 1 billion cycles after fast forwarding the first 0.5 billion cycles.

For the experiments presented later in Section 7, the characteristic of retention times in a DRAM chip is taken from Kim and Lee [2009], which presents measurements from

Table III. Workloads

Workload Suite	Details
SpecJBB	4 warehouses
Spec2006	
Mix1	401.bzip2, 464.h264ref, 453.povray, and 447.dealII
Mix2	410.bwaves, 456.hmmmer, 400.perlbenc, and 471.omnetpp
Mix3	454.calculix, 416.gamess, 435.gromacs, and 450.soplex
Mix4	482.sphinx3, 444.namd, 434.zeusmp, and 437.leslie3d
Mix5	470.lbm, 445.gobmk, 473.astar, and 403.gcc
Mix6	462.libquantum, 429.mcf, 433.milc, and 458.sjeng
PARSEC	blackscholes, bodytrack, ferret, fluidanimate streamcluster, swaptions, vips, x264, canneal

Fig. 8. Retention error rates of utilizing various refresh intervals.

real DRAM chips. Figure 8 shows the relation between error rates and refresh intervals deduced from the relation between cumulative failure probability and retention time shown in Kim and Lee [2009]. As mentioned in Section 4.2, we assume that the geometric distribution of leaky cells of a DRAM chip is uniformly and randomly distributed. We generate five different geometric distributions of leaky cells for evaluations. In the five distributions, with 10^{-6} retention error rate, the average number of retention errors of a region with 128KB size for all five distributions is one error. The maximum number of retention errors in a region of the five distributions are seven, seven, eight, eight, and nine, respectively.

7. EXPERIMENTAL RESULTS

In this section, we first demonstrate how to decide the target error rate and the SEC cache configuration. For this set of experiments, we use the leaky cell distribution with a maximum of eight retention errors in a region when the retention error rate is 10^{-6} . We then discuss the energy and performance behavior of SECRET on the five generated distributions.

7.1. Design Space Exploration: Deciding Target Error Rate and Selective Error Correction Cache Configuration

The target error rate affects both refresh power reduction and error correction overheads. With higher target error rates, we can increase the refresh interval, which is inversely proportional to refresh power consumption. Therefore, the refresh power reduction can be correlated with refresh intervals using the following formula:

$$RP_{T_{ref}} = 1 - (T_{ref\_min}/T_{ref}), \quad (5)$$

Fig. 9. Refresh power reduction achieved by utilizing refresh intervals of various target retention error rates.

Fig. 10. Average cache miss rate of SEC caches with various set/way configurations.

Fig. 11. SEC cache line size versus refresh power reduction normalized to the baseline DRAM under varied retention error rates.

where $RP_{T_{ref}}$ represents the percentage of refresh power reduction achieved by the operating refresh interval T_{ref} , and $T_{ref,min}$ represents the worst-case refresh interval. Thus, based on Figure 8, we can derive the relation between refresh power reduction and error rates as shown in Figure 9. We can observe that refresh power reduction saturates when the retention error rate is larger than 10^{-4} . Therefore, we only have to consider a retention error rate that is smaller than 10^{-4} .

Target error rates also affect SEC overheads, as an SEC cache line needs to store all ECPs in a region as discussed in Section 5. To estimate the SEC cache access overheads of various cache line sizes, we first need to determine its associativity and number of sets, which affects the hit rate of the SEC cache. With higher SEC cache hit rates, we can minimize extra memory accesses to fetch ECP/ECP directory entries. Our simulation indicates that the effect of a 1% SEC cache miss rate on the overall performance is negligible. The average SEC cache miss rates of all tested workloads with different numbers of sets and ways are shown in Figure 10. We observe that only the 256-set/four-way, 512-set/four-way, and 1,024-set/four-way cache configurations can achieve a miss rate that is below 1%. Therefore, we select 256-set/four-way as our SEC cache configuration.

With the associativity and set numbers decided, we now examine how target error rates affect the SEC cache line size. Figure 11 shows the refresh power reduction and the various cache line size of the 256-set/four-way SEC cache under different retention error rates. The results are normalized to that of the baseline DRAM, and each of the points is the average of all tested workloads' results. Refresh power reduction for a given error rate is derived from Figure 9. We can observe that going beyond a 10^{-6}

Fig. 12. Normalized power consumption of the DRAM system with SECRET.

error rate only brings little improvement in refresh power reduction, whereas the SEC cache line size increases steadily. Therefore, we choose 10^{-6} as the target retention error rate, which has a 500ms refresh interval according to Figure 9. Recall that we leave margins to tolerate distribution variation among DRAM chips by setting the line size to 1.5 times the estimated size as discussed in Section 4.2. Since the maximum number of retention errors of a region in the leaky cell distribution studied in the design space exploration is eight with a 10^{-6} error rate, the SEC cache line size is set to 36 bytes, which is able to store 12 ECPs and the ECP directory entry of a region.

With a 256-set/four-way set-associative SEC cache and the target retention error rate set to 10^{-6} , the total SEC cache size is 36KB. According to CACTI 5.3 [Thoziyoor et al. 2008], assuming 45nm technology, the area of the SEC cache is 0.327mm^2 ; the access latency is 0.683ns, which is about two processor cycles when the clock rate is 2GHz; the leakage power is 25mW; and the energy consumption per access is 0.054nJ.

7.2. Energy Analysis

We first discuss the energy results of SECRET on DRAMs executing the workloads mentioned in Section 6. All results reported here take all energy overheads discussed in Section 5 into consideration. Figure 12 shows the power consumption of the DRAM system with SECRET normalized to the baseline. The breakdown shows the power consumption of the SEC cache and the DRAM system, respectively. The results show that the DRAM power consumption of the baseline architecture is about 1W on the average, whereas the maximum power consumption is 1.07W and minimum power consumption is 0.87W. From Figure 12, we can observe that even with an average of 2.6% more power overheads of the SEC cache, SECRET still achieves up to 18.57% DRAM power reduction on the average.

To see how SECRET affects the major parts of the DRAM power consumption, we break down the DRAM power consumption into DRAM peripheral leakage, dynamic power, and refresh power, as shown in Figure 13. This set of results is normalized to the baseline. For each workload, the bar on the left and right are the results of baseline and SECRET, respectively. The DRAM refresh power is reduced by 87.2% for all workloads, as the refresh interval is increased from 64ms to 500ms. In our

Fig. 13. DRAM power breakdown. DRAM peripheral leakage power, dynamic power, and refresh power of the baseline (left) and the SECRET framework (right) for each workload.

experiments, when the power overheads are not considered, the refresh power reduction achieved by SECRET contributes an average of 20.10% of total DRAM power reduction. SECRET also achieves an average of 1.48% DRAM power reduction by reducing DRAM peripheral leakage power, as infrequent refresh operations lead to long idle times. The dynamic power increases slightly, about 0.43% on average, due to additional memory accesses for fetching ECP directory entries and ECPs.

To see how much DRAM power reduction can be achieved by SECRET when the DRAM bandwidth utilization varies, we project the percentage of total DRAM power consumption that refresh operations take under various DRAM bandwidth utilizations by the Micron DDR3 power calculator [Micron Technology 2007]. According to the estimations, when the DRAM bandwidth utilization is under 7%, the refresh power consumption still takes more than 10% of DRAM power. When the DRAM bandwidth utilization is greater than 70%, the refresh power takes no more than 1.5% of DRAM power. However, please note that in a realistic system setup as discussed in Section 6, the memory bandwidth utilization usually is low. With the 8MB L2 cache, the memory bandwidth utilization is close to only 1%. To see the performance of SECRET in real systems with increasing DRAM accesses, we also perform a set of experiments with the last-level cache sizes reduced to 4MB and 2MB, respectively. The detailed discussion is in Section 7.4.

7.3. Performance Analysis

As mentioned in Section 5, the performance overheads come from additional memory accesses caused by SEC cache misses. Figure 14 shows the breakdown of the three types of additional memory accesses, and the results are normalized to the number of data accesses issued by the workload. We can see that SECRET introduces at most 2.87% and an average of 1.62% more data accesses in our tested cases. On average, fetching the ECP directory introduces 0.84% more memory accesses, whereas fetching ECPs incurs 0.55% additional memory accesses. Since the peak memory bandwidth demand of the baseline architecture is 174.56 MB/s, SECRET causes no more than

Fig. 14. Percentage of additional memory accesses introduced by SECRET.

3.5 MB/s bandwidth. On an SEC cache miss, fetching the ECP directory entry is necessary, but fetching ECPs only happens when the requested region has retention errors. Therefore, the number of memory accesses due to ECP directory fetching is slightly more than that for ECPs. Writing ECPs back introduces only an average of 0.23% more memory accesses. The extra memory accesses do not incur noticeable penalty on overall performance. The average IPC values of the baseline and SECRET are 13.472 and 13.475, respectively. Among all test cases, x.264 has the most performance degradation when SECRET is applied. However, even in this case, x.264 only has 1.3% performance degradation compared to the baseline. For the workload Mix6, SECRET achieves 1.4% performance improvement, as modules are less likely to be occupied by refresh operations.

7.4. Evaluating SECRET with Reduced Last-Level Cache Size

In this set of experiments, we reduce the L2 cache size from 8M to 4MB and 2MB. The results show that compared to the 8MB L2 cache, the 4MB and 2MB L2 caches have an average of 56.5% and 192.6% more DRAM accesses, respectively. For 8MB, 4MB, and 2MB last-level caches, the average bandwidths are 97.02MB/s, 150.60MB/s, and 276.41MB/s, respectively, whereas the total available memory bandwidth is 10.66GB/s. As the last-level cache size reduces, the percentage of time that DRAM ranks spent in the active standby mode also increases. For the 8MB L2 cache size, only 1.19% of time is spent in the active standby mode. For the 4MB and 2MB L2 cache sizes, the percentage of time spent in the active standby mode is increased to 1.82% and 3.36%, respectively. The 8MB, 4MB, and 2MB L2 caches have average of 23.05%, 21.20% and 18.79% refresh power overheads, respectively.

As mentioned in Section 4.2, the design of the SEC cache is decided during system design time along with the configuration of the microarchitecture. Thus, for the 4MB and 2MB L2 cache configurations, we respectively perform the process of design space exploration for the SEC cache as described in Section 7.1. The results show that the settings of 256-set/four-way SEC cache and 10^{-6} retention error rate are also adequate for systems with an 4MB L2 cache. For the 2MB L2 cache configuration, since more DRAM accesses are introduced and the spatial locality of these accesses increases, the settings of 128-set/four-way SEC cache and 10^{-6} retention error rate are adequate.

Figure 15 shows the power breakdown of the DRAM systems that utilize 8MB, 4MB, and 2MB L2 caches. We show the average DRAM system power consumption of the 16 workloads both with and without SECRET. The results are normalized to the power

Fig. 15. Normalized DRAM system power breakdown. All results are normalized to the power consumption of the baseline system that has an 8MB L2 cache.

Fig. 16. Percentage of additional memory accesses introduced by SECRET with 8MB, 4MB, and 2MB L2 cache sizes.

consumption of the DRAM system with an 8MB L2 cache without SECRET. Since the refresh interval is increased from 64ms to 500ms, the DRAM refresh power is reduced by 87.2% for all cases. However, with the decreasing L2 cache size and the increasing number of DRAM accesses, the percentages of dynamic power and peripheral leakage power also increase. Therefore, the percentage of DRAM power reduction achieved by SECRET also diminishes when reducing the L2 cache size. For the 8MB L2 cache configuration, SECRET achieves 18.75% DRAM power consumption. With the 4MB L2 cache configuration, the DRAM power consumption achieved by SECRET is reduced to 16.71%. However, even with the 2MB L2 cache, which has 0.6% more peripheral leakage power and 5.1% more dynamic power than the 8MB L2 cache size, SECRET can still achieve an average of 13.5% DRAM power reduction with 1.4% power overheads of the SEC cache.

Figure 16 shows the additional memory accesses introduced by SECRET when utilizing the three L2 cache sizes. For each L2 cache size, the result is normalized to the system with the same L2 cache size and no SECRET. As mentioned earlier, the 256-set/four-way SEC cache size is utilized for both 8MB and 4MB L2 caches. The 128-set/four-way SEC cache size is utilized by the 2MB L2 cache. Although smaller L2 caches introduce more accesses to DRAMs and the SEC cache, the spatial locality of

the accesses actually increases. Therefore, for the 4MB L2 cache configuration, it has a lower SEC cache miss rate and fewer additional DRAM accesses than the 8MB L2 cache configuration even if systems with 4MB and 8MB L2 caches both utilize the same SEC cache configuration. Our experimental results show that SECRET introduces very negligible performance degradation (less than 1%) for all L2 cache sizes.

7.5. Evaluation of Distributing Leaky Cells with Spatial Locality

As mentioned in Section 4.2, we assume that the leaky and normal cells are uniformly and randomly distributed in a DRAM chip. Based on this assumption, the SEC cache line size is decided. However, when the clustering of leaky cells happens in a DRAM chip installed in a machine with SECRET, it is possible that the maximum number of retention errors given the target refresh interval is larger than the number of ECPs that an SEC cache line size can hold. In this case, the SECRET framework has to choose a refresh interval shorter than the target interval so that the maximum number of retention errors in a region can be fit in the selected SEC cache line size. As a result, the amount of refresh power reduction may be reduced. Thus, in this set of experiments, we create a set of retention time distributions with various degrees of spatial locality to evaluate SECRET. Distributions with high spatial locality indicate that leaky cells are more likely to be clustered in the neighboring area.

According to Zhang and Li [2009], the retention time of a DRAM cell, denoted by $T_{retention}$, can be decided by two kinds of effects—random effects and the systematic effects—where the systematic effects refer to the layout-dependent variation through which nearby devices share similar parameters. Therefore, $T_{retention}$ is modeled by

$$T_{retention} = (1 - W) \cdot T_{rand} + W \cdot T_{sys}, \quad (6)$$

where T_{rand} denotes the retention time decided by random effects and T_{sys} denotes the retention time decided by systematic effects. W is the weight value to adjust the proportion between random and systematic effects. When W is set to zero, this indicates that we consider the random effect only and no spatial locality among leaky cells when generating the retention time distribution. For this set of experiments, we randomly generate T_{rand} . For generating T_{sys} , we adopt the multiple-level quad tree approach [Agarwal et al. 2002] to model the correlated within-die variation effect among retention times of DRAM cells. The smallest quadrant in the multiple-level quad tree is set to 16K DRAM cells [Zhang and Li 2009].

In this set of experiments, we set W to 0.0, 0.1, 0.2, 0.3, 0.4, and 0.5 to model the six distributions with various degrees of spatial locality. For each W value, we utilize Equation (6) to generate 10 sets of retention time distributions. Therefore, there are 60 different retention time distributions generated for this set of experiments. The L2 cache size is set to 8MB. The settings of SEC cache and the target retention error rate are the same as the ones obtained in Section 7.1. In other words, the SEC cache line size is set to 36 bytes, which is able to store at most 12 ECPs of a region, and the target error rate is set to 10^{-6} .

Figure 17 shows the DRAM power reduction of various distributions generated by various W values. Moreover, it also shows the maximum number of retention errors in a region when the target error rate is set to 10^{-6} . We can observe that when the value of W increases, the spatial locality among retention times slightly increases and the maximum number of errors in a region also increases. However, for the 60 retention time distributions, only four distributions have the maximum number of retention errors larger than 12 when the target error rate is set to 10^{-6} in our test cases. To accommodate the selected SEC cache design, instead of utilizing the 500ms refresh interval, the refresh intervals of the four cases needs to be shortened so that the maximum number of retention errors in a region is no more than 12. The refresh

Fig. 17. DRAM power reduction of various distributions. The maximum number of retention errors in a region is measured when the target error rate is set to 10^{-6} .

interval that meets the requirement of each of the four cases is also marked in Figure 17, and the shortest refresh interval is 450ms. Although a shorter refresh interval indicates less refresh power reduction, the 450ms refresh interval achieves only 0.8% less power reduction than the best case. This shows that the proposed SECRET framework can still achieve significant DRAM power reduction when the retention time distribution has strong spatial locality.

7.6. Comparison with Traditional Error-Correcting Code Approaches

For systems with a reliability requirement, DRAMs are commonly protected by Hamming code to meet the target mean time to failures (MTTF). As mentioned in Section 1, retention errors due to a prolonged refresh interval can be treated as soft errors and corrected by ECCs. Here, we evaluate the refresh power reduction that can be achieved by Hamming code and observe if SECRET can achieve more refresh power reduction than with built-in ECC when SECRET is applied in such a DRAM system.

In this set of experiments, we assume that the target DRAM systems are equipped with (72, 64) Hamming code, and the characteristic of retention times and distribution of leaky and normal cell are the same as the one described in Section 6. We assume a 64GB DRAM, where the failure in time (FIT) of soft errors is 2,000/Mb [Tezzaron Semiconductor 2004] and the reliability requirement is set to a 10-year MTTF [Mukherjee et al. 2005]. For ECC-only systems, a failure that cannot even be detected happens when there are more than two errors in a 72-bit data block. When we prolong the refresh intervals of DRAMs with ECCs, the failure may be due to (1) soft error or (2) retention errors in the same data block. Given the refresh interval or retention error rate, we can identify the cells that have retention errors since leaky cells have been profiled. With the FIT setup, we can obtain the probability of a memory cell having a soft error, as FIT indicates the number of soft errors that a DRAM chip would have during a time unit. Therefore, combining the results of retention errors and probability of having soft errors, we can estimate the probability of having more than two errors for each block given a refresh interval. The MTTF of the whole DRAM system can be estimated accordingly. Figure 18 shows the relation between retention error rates and

Fig. 18. Mean time to failure versus retention error rate.

the MTTF. We can observe that to tolerate retention errors caused by prolonging the refresh interval, the MTTF of the protected memory system is shortened. To meet the 10-year MTTF requirement, the retention error rate should not exceed 10^{-8} , which corresponds to a 128ms refresh interval as indicated in Figure 8. In other words, under the MTTF constraint, DRAMs with Hamming code can still increase the refresh interval to 128ms and achieve 50% refresh power reduction compared to the baseline with a 64ms refresh interval.

When SECRET is applied to DRAMs with ECCs, as discussed in Section 7.2, the refresh interval is prolonged to 500ms and up to 87.2% refresh power is reduced. Therefore, the benefit of SECRET is still quite substantial in a DRAM system protected with Hamming code. Moreover, SECRET is totally compatible with DRAMs with ECCs. Providing that the MTTF requirement is met, it is possible for DRAM systems with SECRET and ECC to further prolong the target refresh interval designated by SECRET, as the ECC can also correct the retention errors. However, with a smaller FIT, DRAMs have higher probabilities of having soft errors, and the refresh interval that can be prolonged by Hamming code must be shortened so that the MTTF requirement can be met. In this case, SECRET is substantial for refresh power reduction.

8. CONCLUSION

In this article, we propose SECRET—a novel error correction framework for retention errors in DRAMs to prolong the refresh interval and achieve refresh power reduction. SECRET only allocates error-correcting bits to leaky cells that cause retention errors under a refresh interval. This framework does not require any modification in the interface between the memory controller and the DRAM. All additional hardware components are in the memory controller. The experimental results show that SECRET can reduce refresh power by 87.2% and an average of 18.57% of DRAM power consumption with negligible area and performance overheads. We also show that for DRAM systems protected by Hamming code, the benefit of SECRET is substantial.

REFERENCES

- Aseem Agarwal, David Blaauw, Vladimir Zolotov, Savithiri Sundareswaran, Min Zhao, Kaushik Gala, and Rajendran Panda. 2002. Path-based statistical timing analysis considering inter-and intra-die correlations. In *Proceedings of the International Workshop on Timing Issues in the Specifications and Synthesis of Digital Systems (TAU'02)*. 16–21.
- Aditya Agrawal, Prabhat Jain, Amin Ansari, and Josep Torrellas. 2013. Refrint: Intelligent refresh to minimize power in on-chip multiprocessor cache hierarchies. In *Proceedings of the 19th International Symposium on High-Performance Computer Architecture (HPCA'13)*.
- Phillip G. Emma, William R. Reohr, and Mesut Meterellioz. 2008. Rethinking refresh: Increasing availability and reducing power in DRAM for cache applications. *IEEE Micro* 28, 6, 47–56. DOI: <http://dx.doi.org/10.1109/MM.2008.93>
- Mrinmoy Ghosh and Hsien-Hsin S. Lee. 2007. Smart refresh: An enhanced memory controller design for reducing energy in conventional and 3D die-stacked DRAMs. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'07)*. IEEE, Los Alamitos, CA, 134–145. DOI: <http://dx.doi.org/10.1109/MICRO.2007.38>

- Ciji Isen and Lizy John. 2009. ESKIMO: Energy savings using semantic knowledge of inconsequential memory occupancy for DRAM subsystem. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'09)*. ACM, New York, NY, 337–346. DOI: <http://dx.doi.org/10.1145/1669112.1669156>
- Yasunao Katayama, Eric J. Stuckey, Sumio Morioka, and Zhao Wu. 1999. Fault-tolerant refresh power reduction of DRAMs for quasi-nonvolatile data retention. In *Proceedings of the International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'99)*. 311–318. DOI: <http://dx.doi.org/10.1109/DFTVS.1999.802898>
- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa R. Alameldeen, Chris Wilkerson, and Onur Mutlu. 2014. The efficacy of error mitigation techniques for DRAM retention failures: A comparative experimental study. In *Proceedings of the 2014 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'14)*. ACM, New York, NY, 519–532. DOI: <http://dx.doi.org/10.1145/2591971.2592000>
- Chan-Kyung Kim, Jae-Goo Lee, Young-Hyun Jun, Chil-Gee Lee, and Bai-Sun Kong. 2007. CMOS temperature sensor with ring oscillator for mobile DRAM self-refresh control. *Microelectronics Journal* 38, 10–11, 1042–1049. DOI: <http://dx.doi.org/10.1016/j.mejo.2007.08.003>
- Joohee Kim and Marios C. Papaefthymiou. 2003. Block-based multiperiod dynamic memory design for low data-retention power. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 11, 6, 1006–1018. DOI: <http://dx.doi.org/10.1109/TVLSI.2003.817524>
- Kinam Kim and Jooyoung Lee. 2009. A new investigation of data retention time in truly nanoscaled DRAMs. *IEEE Electron Device Letters* 30, 8, 846–848. DOI: <http://dx.doi.org/10.1109/LED.2009.2023248>
- Charles Lefurgy, Karthick Rajamani, Freeman Rawson, Wes Felter, Michael Kistler, and Tom W. Keller. 2003. Energy management for commercial servers. *Computer* 36, 12, 39–48. DOI: <http://dx.doi.org/10.1109/MC.2003.1250880>
- Jiang Lin, Hongzhong Zheng, Zhichun Zhu, Howard David, and Zhao Zhang. 2007. Thermal modeling and management of DRAM memory systems. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA'07)*. ACM, New York, NY, 312–322. DOI: <http://dx.doi.org/10.1145/1250662.1250701>
- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu. 2013. An experimental study of data retention behavior in modern DRAM devices: Implications for retention time profiling mechanisms. In *Proceedings of the 40th International Symposium on Computer Architecture (ISCA'13)*. 60–71.
- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu. 2012. RAIDR: Retention-aware intelligent DRAM refresh. In *Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA'12)*. IEEE, Los Alamitos, CA, 1–12. <http://dl.acm.org/citation.cfm?id=2337159.2337161>
- Song Liu, Karthik Pattabiraman, Thomas Moscibroda, and Benjamin G. Zorn. 2011. Flicker: Saving DRAM refresh-power through critical data partitioning. In *Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'11)*. ACM, New York, NY, 213–224. DOI: <http://dx.doi.org/10.1145/1950365.1950391>
- Peter S. Magnusson, Magnus Christensson, Jesper Eskilson, Daniel Forsgren, Gustav Hållberg, Johan Högberg, Fredrik Larsson, Andreas Moestedt, and Bengt Werner. 2002. Simics: A full system simulation platform. *Computer* 35, 2, 50–58. DOI: <http://dx.doi.org/10.1109/2.982916>
- Milo M. K. Martin, Daniel J. Sorin, Bradford M. Beckmann, Michael R. Marty, Min Xu, Alaa R. Alameldeen, Kevin E. Moore, Mark D. Hill, and David A. Wood. 2005. Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset. *ACM SIGARCH Computer Architecture News* 33, 4, 92–99. DOI: <http://dx.doi.org/10.1145/1105734.1105747>
- Micron. 2006. DDR3 SDRAM. Retrieved May 8, 2015, from http://download.siliconexpert.com/pdfs/2013/4/2/5/1/30/39/mic_manual/571gb_ddr3_sdram.pdf
- Micron Technology. 2007. System Derating for 2Gb DDR3 SDRAM (x8, -107 Speed Grade). Retrieved May 8, 2015, from http://www.micron.com/~media/documents/products/power-calculator/ddr3_power_calc.xlsm
- Vasily G. Moshnyaga, Hua Vo, Glenn Reinman, and Miodrag Potkonjak. 2007. Reducing energy of DRAM/flash memory system by OS-controlled data refresh. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'07)*. 2108–2111. DOI: <http://dx.doi.org/10.1109/ISCAS.2007.378515>
- Shubhendu S. Mukherjee, Joel Emer, and Steven K. Reinhardt. 2005. The soft error problem: An architectural perspective. In *Proceedings of the 11th International Symposium on High-Performance Computer Architecture (HPCA'05)*. 243–247. DOI: <http://dx.doi.org/10.1109/HPCA.2005.37>
- Kimish Patel, Luca Benini, Enrico Macii, and Massimo Poncino. 2005. Energy-efficient value-based selective refresh for embedded DRAMs. In *Proceedings of the 15th International Conference on Integrated Circuit*

- and System Design: Power and Timing Modeling, Optimization, and Simulation (PATMOS'05). 466–476. DOI: http://dx.doi.org/10.1007/11556930_48
- Kaushik Roy, Saibal Mukhopadhyay, and Hamid Mahmoodi-Meimand. 2003. Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. *Proceedings of the IEEE* 91, 2, 305–327. DOI: <http://dx.doi.org/10.1109/JPROC.2002.808156>
- Stuart Schechter, Gabriel H. Loh, Karin Straus, and Doug Burger. 2010. Use ECP, not ECC, for hard failures in resistive memories. In *Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA'10)*. ACM, New York, NY, 141–152. DOI: <http://dx.doi.org/10.1145/1815961.1815980>
- Tezzaron Semiconductor. 2004. Soft Errors in Electronic Memory—A White Paper. Retrieved May 8, 2015, from http://www.tezzaron.com/media/soft_errors_1_1_secure.pdf.
- Shyamkumar Thoziyoor, Naveen Muralimanohar, Jung Ho Ahn, and Norman P. Jouppi. 2008. *CACTI 5.1*. HP Laboratories.
- Le-Nguyen Tran, Fadi J. Kurdahi, and Ahmed M. Eltawil. 2011. Adjustable supply voltages and refresh cycle for process variations and temperature changing adaptation in DRAM to minimize power consumption. In *Proceedings of the 2011 IEEE Workshop on Microelectronics and Electron Devices (WMED'11)*. 1–4. DOI: <http://dx.doi.org/10.1109/WMED.2011.5767277>
- Tung-Han Tsai, Chin-Lin Chen, Ching-Li Lee, and Chua-Chin Wang. 2008. Power-saving nano-scale DRAMs with an adaptive refreshing clock generator. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'08)*. 612–615. DOI: <http://dx.doi.org/10.1109/ISCAS.2008.4541492>
- Odilio Vargas. 2005. Minimum power consumption in mobile-phone memory subsystems. *Portable Design Magazine* 26, 31–38.
- Ravi K. Venkatesan, Stephen Herr, and Eric Rotenberg. 2006. Retention-aware placement in DRAM (RAPID): Software methods for quasi-non-volatile DRAM. In *Proceedings of the 12th International Symposium on High-Performance Computer Architecture (HPCA'06)*. 155–165. DOI: <http://dx.doi.org/10.1109/HPCA.2006.1598122>
- David Wang, Brinda Ganesh, Nuengwong Tuaycharoen, Kathleen Baynes, Aamer Jaleel, and Bruce Jacob. 2005. DRAMsim: A memory system simulator. *ACM SIGARCH Computer Architecture News* 33, 4, 100–107. DOI: <http://dx.doi.org/10.1145/1105734.1105748>
- Chris Wilkerson, Alaa R. Alameldeen, Zeshan Chishti, Wei Wu, Dinesh Somasekhar, and Shih-lien Lu. 2010. Reducing cache power with low-cost, multi-bit error-correcting codes. In *Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA'10)*. ACM, New York, NY, 83–93. DOI: <http://dx.doi.org/10.1145/1815961.1815973>
- Barry W. Williams. 2006. *Principles and Elements of Power Electronics: Devices, Drivers, Applications, and Passive Components*. University of Strathclyde, Glasgow, Scotland.
- David S. Yaney, Chih-Yuan Lu, Rudeger A. Kohler, Michael J. Kelly, and James T. Nelson. 1987. A metastable leakage phenomenon in DRAM charge storage—variable hold time. In *Proceedings of the 1987 International Electron Devices Meeting*. 336–339.
- Wangyuan Zhang and Tao Li. 2009. Characterizing and mitigating the impact of process variations on phase change based memory systems. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'09)*. ACM, New York, NY, 2–13. DOI: <http://dx.doi.org/10.1145/1669112.1669116>
- Hongzhong Zheng, Jiang Lin, Zhao Zhang, Eugene Gorbato, Howard David, and Zhichun Zhu. 2008. Mini-rank: Adaptive DRAM architecture for improving memory power efficiency. In *Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'08)*. 210–221. DOI: <http://dx.doi.org/10.1109/MICRO.2008.4771792>

Received July 2014; revised March 2015; accepted March 2015