Approximate Computing with Unreliable Dynamic Memories

Approximate Computing with Unreliable Dynamic Memories

Shrikanth Ganapathy* Adam Teman* Robert Giterman† Andreas Burg* Georgios Karakonstantis‡
*Telecommunications Circuits Laboratory, Ecole Polytechnique Federale de Lausanne, Switzerland
† ENICS Lab, Faculty of Engineering, Bar Ilan University, Ramat Gan, Israel
‡ High Performance and Distributed Computing, Queen's University Belfast, United Kingdom
Email: *{shrikanth.ganapathy, adam.teman, andreas.burg}@epfl.ch, ‡g.karakonstantis@qub.ac.uk

Abstract—**Embedded memories account for a large fraction of the overall silicon area and power consumption in modern SoC(s). While embedded memories are typically realized with SRAM, alternative solutions, such as embedded dynamic memories (eDRAM), can provide higher density and/or reduced power consumption. One major challenge that impedes the widespread adoption of eDRAM is that they require frequent refreshes potentially reducing the availability of the memory in periods of high activity and also consuming significant amount of power due to such frequent refreshes. Reducing the refresh rate while on one hand can reduce the power overhead, if not performed in a timely manner, can cause some cells to lose their content potentially resulting in memory errors. In this paper, we consider extending the refresh period of gain-cell based dynamic memories beyond the worst-case point of failure, assuming that the resulting errors can be tolerated when the use-cases are in the domain of inherently error-resilient applications. For example, we observe that for various data mining applications, a large number of memory failures can be accepted with tolerable imprecision in output quality. In particular, our results indicate that by allowing as many as 177 errors in a 16 kB memory, the maximum loss in output quality is 11%. We use this failure limit to study the impact of relaxing reliability constraints on memory availability and retention power for different technologies.**

I. Introduction

Embedded dynamic random access memories (eDRAMs) have gained popularity due to their high-density and low retention power, as compared to static random access memory (SRAM). Replacing SRAMs with eDRAMs could provide a viable solution for managing the increasing amount of data that need to be handled by today's systems. Therefore, recent works have focused on improving the eDRAM cells and ensuring their migration to advanced process technologies [1], [2]. One of the main issues in eDRAMs is that their data is stored as charge on parasitic capacitances, which leaks away over time, thereby requiring frequent refreshes for reliable data storage. These refresh operations occupy the memory for a significant amount of time, during which it is unavailable for system access (reduced availability). In addition, during low activity (standby) periods, the refresh operations are the primary source of power consumption in these memories. The memory availability and standby-power overhead are further worsened due to the insistence of traditional approaches on determining the frequency of the refresh cycles based on a worst-case assumption for the retention time of the most leaky cell, constantly biased at rare worst-case conditions. The resulting penalties are expected to further exacerbate as silicon predictability reduces with technology scaling, putting the feasibility of traditional design approaches in doubt [3].

This reality has led to the quest for alternative design strategies and to the promising *approximate computing* paradigm, in which the error resilient nature of many applications is exploited to relax the design constraints and save power [3], [4], [5]. The approximate computing paradigm includes the development of processors and software that may not always produce 100% precise results, but their output fidelity is acceptable for human consumption with significantly reduced power consumption [3], [4]. Error resilient applications also open up possibilities in dynamic memory design, providing an opportunity to potentially reduce the frequency of required refresh cycles, while still guaranteeing minimum output quality levels under the presence of resulting failures [6], [7].

In this paper, we exploit the inherent error-resilience of applications to design a gain-cell based embedded DRAM (GC-eDRAM) memory. We evaluate the benefits of lowering refresh frequency in the context of enhancing memory availability and also study the impact on output quality for two data-mining applications executed on such an error-prone memory.

Contributions: The contributions can be summarized as:

- Analysis of a 2T gain-cell array in the context of technology scaling, revealing diminishing retention times with technology advancement.
- Evaluation of the error tolerance limits for two popular data-mining applications. For memories with a cell-failure rate as high as 10^{-3}, the observed loss in output quality is below 10%.
- Analysis of the improvement in memory availability and reduction of retention power through relaxation of the worst-case cell criterion that is traditionally used to determine the refresh rate.
- Our analysis, reveals that while the increase in availability for mature technologies is negligible, a large benefit can be achieved through the proposed refresh rate relaxation for scaled technologies.

The rest of the paper is organized as follows: Section II recaps the considered 2T GC-eDRAM bitcell, followed by an analysis of the extent of errors that can be tolerated by various data mining applications in Section III. The proposed approach with the potential improvement in memory availability and power savings are presented in Section IV. Section V concludes the paper.

II. Gain-cell embedded DRAM

An increasingly popular alternative to SRAM in systems that require very dense embedded memories is the two-transistor (2T) gain cell. This circuit consists of a write transistor (MW) and a read transistor (MR), as illustrated in the inset of Fig. 1. This topology significantly reduces the transistor count compared to the six or eight transistors, required by a single-port or two-port SRAM, respectively, thereby leading to a memory core area that is more than 50% smaller than equivalent SRAMs. This is achieved with an inherent two-ported functionality, as the write and read paths are internally decoupled, providing non-ratioed operation that is naturally suitable for process scaling. In addition, the 2T cell

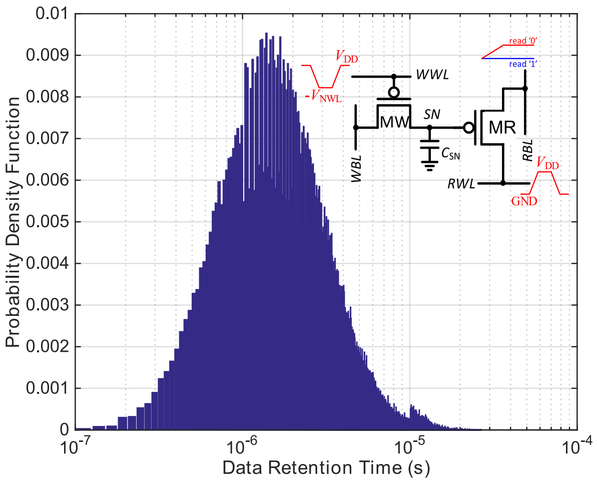

Fig. 1. Probability Density Function of 2T GC-eDRAM bitcell in a 28 nm FD-SOI process. The plot was extracted for 100k Monte Carlo samples at 25°C. *inset*: Schematic and typical operational waveforms of the 2T Gain Cell.

is characterized by very low leakage, as the cell has no direct connection between the voltage supplies through a cut-off transistor channel. However, the storage mechanism of the cell relies on dynamic charge, stored on the parasitic capacitance of the circuit (C_{SN}), which leaks away, primarily due to subthreshold conduction through MW. The time it takes for the data to leak away is known as the data retention time (DRT) of the bitcell. A refresh operation has to be initated before the time elapsed after a write isn't greater than the DRT in order to ensure data integrity. The refreshes increase the standby power consumption of the eDRAM array (better known as *retention power*), as well as limit the system availability of the array during active periods, since during refresh cycles, the memory array is accessed by a refresh controller.

For process technologies above approximately 100 nm, the DRT of the 2T GC-eDRAM bitcell was in the order of milliseconds, as the subthreshold leakage of MOS devices was relatively low and the parasitic capacitance could be significantly increased through metal stacking. However, the effectiveness of metal stacking has been impeded by the introduction of low-k interconnect dielectrics, and the subthreshold leakage has significantly increased primarily due to short-channel effects (SCE) and drain-induced barrier lowering (DIBL). Therefore, the nominal DRT for a 2T GC-eDRAM bitcell has dropped by several orders-of-magnitude. This phenomenon is exacerbated by the increasing process variations that characterize scaled technologies, as small deviations in the device threshold voltage have an exponential effect on the subthreshold leakage. To show the influence of variability at deeply scaled nodes, Fig. 1 plots the probability density function (PDF) of the DRT of a 2T gain cell, implemented in a 28 nm FD-SOI process. While the average DRT is in the microseconds range, the distribution is approximately lognormal with a far reaching left-side tail, resulting in a worst-case DRT of less than 100 ns for the depicted sample set of only 100,000 cells.

Note that the calculation of the DRT assumes worst-case biasing conditions [2], which have an extremely low probability of ever occurring. However, in order to ensure 100% reliable operation, the refresh rate of the entire array is traditionally set according to the worst-cell under such conditions, and therefore, the feasibility of using this circuit in the considered process is almost non-existent, even for small arrays. Not only would the resulting retention power be much higher than the leakage power of an equivalent SRAM, but, for typical array sizes operated at typical clock frequencies, such a low retention time is also insufficient for performing a single array refresh before the stored data is corrupted. Therefore, in order to ensure the integration of GC-eDRAM at scaled technologies, alternative approaches must be taken.

III. Error-Tolerance in Applications

Such an alternative approach to the design for the worst-case is to consider living with errors in the memory. Recent research in this direction has highlighted the emergence of a new class of applications with relaxed reliability requirements that can tolerate up to a certain number of memory failures with minimal loss in output quality [3], [4], [5]. This is based on the observation that these applications are inherently error resilient such that hardware errors can be masked by virtue of the inherent application characteristics. Exploiting the inherent error-resilience of such applications can help design systems that depart from the 100% error-free computing paradigm, providing the opportunity to trade-off data-integrity (application quality) for energy-efficiency and silicon real-estate.

To evaluate the benefits of relaxing the 100% reliability requirement, we study the impact of memory failures on the application output quality for two popular machine learning applications: K-nearest neighbors (KNN) and elastic net (EN). KNN and EN are used extensively in the machine learning domain for building classification and regression models, respectively. The benchmarks were developed using the popular open-source machine learning framework, Scikit-Learn [8]. Using an in-house developed simulation framework, we run the benchmarks on a functional memory model that mimics the behaviour of running the applications on a physical memory with failures. We used real world datasets for the training and testing phases of the benchmark execution [9], partitioned into training and testing inputs (0.8:0.2), and executed them on a 16 kB simulated memory. For a range of GC-eDRAM failure probabilities, we determine the number of failures in a memory sample, and accordingly inject random bit-flips into the training data before executing the benchmark. The outputs of the benchmark are then compared against the golden values of the testing data to produce quality metrics - R^2 for KNN and *score* for EN. While R^2 indicates the goodness of the fit (for EN), score computes the mean of the number of correct classifications for KNN.

Fig. 2 shows the variation in output quality for a range of GC-eDRAM failure probabilities. The measured output quality is normalised to the quality estimated when the benchmarks are executed on a fault-free memory. We observe that for a cell-failure rate of 10^{-3}, the output quality for EN is almost 90%. For a 16 kB memory, this translates to a maximum of 177 failures that can be tolerated by a single memory sample. We notice that the inherent error resilience of KNN is much better than that of EN, as evinced by the fact that for a failure probability as high as 10^{-3}, the quality is almost 97% of the quality obtained when executed on a fault-free memory.

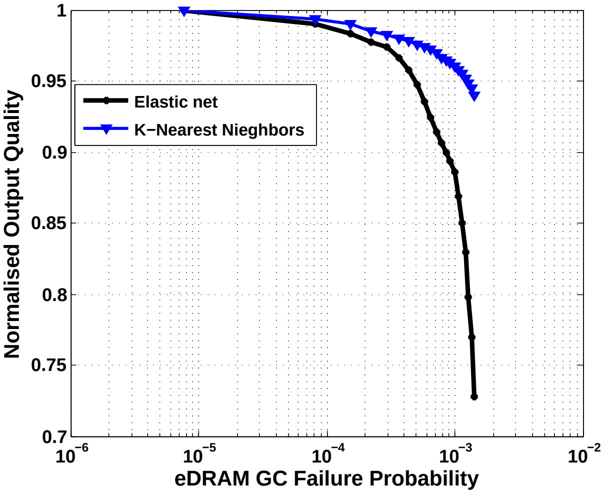

Fig. 2. Measured output quality for a range of eDRAM GC failure probabilities. All values normalised to measured quality in case of a fault-free memory.

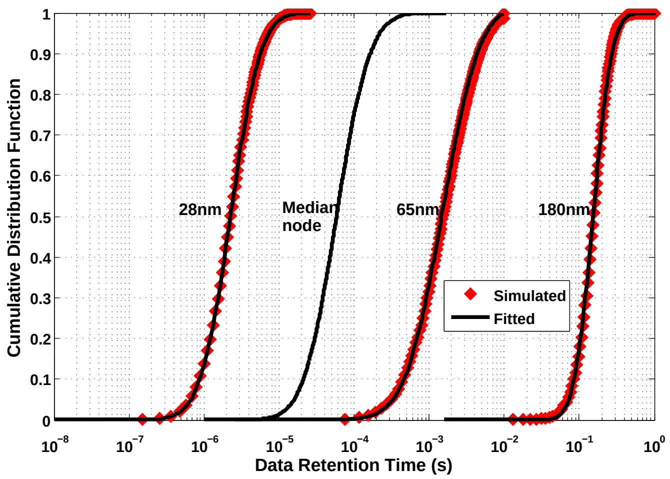

Fig. 3. Cumulative Distribution Functions of the data retention times of 2T GC-eDRAM bitcells for various process technologies: 180 nm CMOS, 65 nm CMOS, 28 nm FD-SOI, and a estimated median node.

IV. Utilizing Unreliable GC-eDRAMs

The discussion of Section II suggested that 2T GC-eDRAM is a problematic alternative to SRAM at scaled technologies due to the high refresh rate determined by traditional design for the worst-case. However, the tolerance of various applications to errors, shown in Section III, suggests that by relaxing the requirement of 100% reliable operation and reducing frequency of the refresh cycles its advantages can still be exploited.

To explore the effectiveness of this new design paradigm, the cumulative distribution functions (CDFs) of the DRT of 2T GC-eDRAM bitcells are plotted in Fig. 3 for three process technologies: mature 180 nm CMOS, scaled 65 nm CMOS, and deeply-scaled 28 nm FD-SOI. The figure clearly shows the deterioration of the retention time with technology scaling, while also emphasizing the long left-side tail of the DRT distribution, which is common to all considered processes. From this plot, we can learn that for older process technologies, the nominal DRT was in the range of tens to hundreds of milliseconds. For typical sub-arrays with a few hundred rows, this still enables a high memory availability as also shown in [10]. On the other hand, as previously seen in Fig. 1, for a deeply scaled 28 nm process, the nominal DRT is in the microsecond range. Even without considering the worst-case outliers, this requires a prohibitively high refresh rate and renders arrays with realistic access times and sizes infeasible. However, when considering a median node - in this case, 65 nm CMOS - the nominal DRT is high enough to support many array sizes and operating frequencies, but the yield requirements lead to refresh periods that may significantly impede the availability of the array for system access.

The CDFs of Fig. 3 are based on a limited number of Monte Carlo samples, and therefore do not represent the infinite left-side tail of the distributions. This infinite tail would always lead to some degree of bit failures, and therefore, in order to set a reasonable refresh time for 100% error-free operation, yield must be brought into the discussion. For a refresh period of T_{ref}, the yield of an array with N cells is defined as

$$Y = (1 - P(t_{\text{ret}} < T_{\text{ref}}))^N \qquad (1)$$

with $P(t_{\text{ret}} < T_{\text{ref}})$ representing the probability that the retention time of a cell (t_{ret}) is smaller than the refresh period. Therefore, for a given target yield of Y_{T}, for 100% error-free operation, the refresh time should be set to

$$T_{\text{ref}} \leq f^{-1}(1 - \sqrt[N]{Y_{\text{T}}}) \qquad (2)$$

where f^{-1} is the inverse CDF of the DRT. In order to obtain this probability, we observe that the distributions shown in Fig. 3 can be approximated to lognormal distributions, as shown by the fitted curves on the plot. Using this approximation, we can find the required refresh rate to obtain such a desired target yield. Furthermore, this observation enables us to estimate the DRT distribution of a *median* node that falls in between our simulated technologies to evaluate the feasibility of 2T GC-eDRAMs in such a process.

A. Improving Memory Availability

As previously described, memory availability is the percentage of time a given array is available for system access, i.e., when it is not pre-occupied by the refresh controller. We calculate the memory availability (A_{mem}) as

$$A_{\text{mem}} = \frac{T_{\text{ref}} - T_{\text{busy}}}{T_{\text{ref}}} \cdot 100\%, \qquad (3)$$

where T_{ref} is the refresh period selected for the target yield, as defined in (2), and T_{busy} is the time it takes to refresh the entire array, during which it is unavailable for system access. Assuming both memory read and write operations are achieved in a single cycle with an operating frequency, f, and each refresh operation requires reading and subsequently writing each row of an array with W words, the time required for an array refresh can be written as $T_{\text{busy}} = 2W/f$.

For each of the considered process technologies, a typical access time was assumed, and an array size (number of words) was chosen to accommodate this access time. Thereafter, the worst-case DRT required for a target yield of $Y_{\text{T}} \geq 95\%$ was calculated based on the extrapolations shown in Fig. 3 for each technology node. The resulting memory availabilities are plotted in Fig. 4 (circular markers). While the mature node shows almost no impact on availability, the resulting memory bandwidth decreases for advanced technologies, reaching a

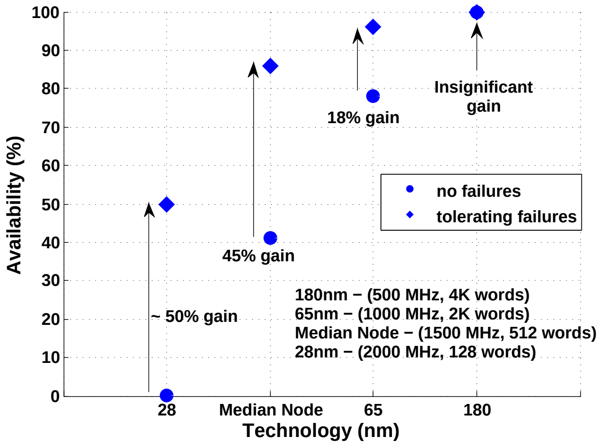

Fig. 4. Memory availability of GC-eDRAM arrays in various technology nodes: 180 nm CMOS, 65 nm CMOS, and 28 nm FD-SOI, and an estimated median node. Availability is shown for traditional worst-case design ("No Failures") and for error-tolerance of $P_{\text{err}} < 10^{-3}$ for memory bank sizes and operating frequencies typical for each node.

point where at 28 nm, the array is almost continuously in refresh, despite the small number of rows (128) chosen for this node. However, by relaxing the 100% error-free requirement to a tolerable error probability, such as the $P_{\text{err}} < 10^{-3}$ considered in Section III, a significant increase in availability can be achieved. For the mature 180 nm process this is unnecessary, but for the 65 nm node, the availability is increased to over 95%, and for the virtual median node, we can expect an availability increase of approximately 45%. However, for the deeply-scaled 28 nm node, while bringing the availability to almost 50%, this is still too low for most applications.

B. Retention Power Savings

The importance of memory availability is at a premium, when discussing high activity factors, such as those required by the applications considered in Section III. However, many applications temporarily shut down or are characterized by long periods of low activity, during which the dynamic power consumption of the memory blocks becomes inferior to their static power consumption. When considering dynamic memory solutions, refresh is required to ensure data integrity during retention, and in the case of 2T GC-eDRAM, this refresh power dominates the very low leakage power of this topology. In this case, the retention power of the memory is inversely proportional to the refresh rate, as

$$P_{\text{ret}} = P_{\text{leakage}} + P_{\text{refresh}} \approx E_{\text{refresh}}/T_{\text{ref}} \tag{4}$$

where the leakage power (P_{leakage}) is assumed to be much lower than the refresh power (P_{refresh}) and E_{refresh} is the energy consumed during a single array refresh operation. Therefore, by trading off data integrity with refresh time, in a similar fashion to the above proposal targeting availability, a significant amount of retention power can be saved.

Fig. 5 shows the potential power savings obtained by adopting a refresh rate lower than the 100% error free requirement. In this case, even the mature nodes show a significant benefit, as opposed to the case of availability, where the improvement was negligible. For the 65 nm process, the reduction of retention power approaches 5× by allowing an error probability of $P_{\text{err}} < 10^{-3}$, and all processes benefit by at least 2×. For the estimated median node, power savings similar to the 65 nm node are expected.

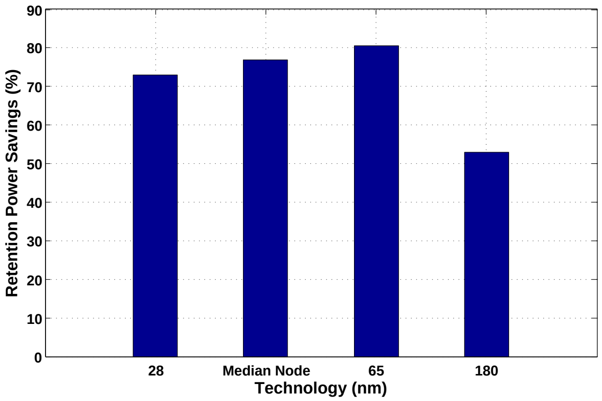

Fig. 5. Retention power savings during low activity periods for GC-eDRAM arrays at different nodes, operated with an error tolerance of $P_{\text{err}} < 10^{-3}$

V. Conclusions

In this paper, we considered a paradigm shift towards approximate computing with unreliable memories. We study the scalability of gain-cell embedded DRAMs, which provide advantageous benefits as compared to SRAM, but suffer from reduced memory availability and high refresh power as retention time deteriorates due to process scaling. Our analysis shows that by relaxing the worst-case assumption for setting the refresh rate, which is traditionally used in order to achieve 100% error-free operation, a significant benefit in both availability and power can be achieved. For mature process nodes, the increase in availability is negligible, but with process scaling, the tolerance of a small number of errors can lead to a significant increase in availability. On the other hand, the standby power reduction achieved through such an approach is significant across all nodes. These observations provide a means for improving the effectiveness of GC-eDRAM in scaled technologies.

References

[1] D. Somasekhar *et al.*, "2 GHz 2 Mb 2T gain cell memory macro with 128 GBytes/sec bandwidth in a 65 nm logic process technology," *JSSC*, vol. 44, no. 1, pp. 174–185, 2009.

[2] A. Teman *et al.*, "Replica technique for adaptive refresh timing of gain-cell-embedded DRAM," *IEEE TCAS-II*, vol. 61, no. 4, pp. 259–263, April 2014.

[3] P. Gupta *et al.*, "Underdesigned and opportunistic computing in presence of hardware variability," *IEEE TCAD*, vol. 32, no. 1, pp. 8–23, 2013.

[4] S. Venkataramani *et al.*, "Quality programmable vector processors for approximate computing," in *IEEE/ACM ISM 2013*, 2013, pp. 1–12.

[5] S. Ganapathy *et al.*, "Mitigating the impact of faults in unreliable memories for error-resilient applications," in *DAC'15*, 2015.

[6] S. Liu *et al.*, "Flikker: Saving DRAM refresh-power through critical data partitioning," *SIGPLAN Not.*, vol. 46, no. 3, pp. 213–224, Mar. 2011.

[7] A. Teman *et al.*, "Energy versus data integrity trade-offs in embedded high-density logic compatible dynamic memories," in *DATE'15*, 2015.

[8] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[9] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[10] P. Meinerzhagen *et al.*, "Exploration of sub-VT and near-VT 2T gain-cell memories for ultra-low power applications under technology scaling," *MDPI JLPEA*, vol. 3, no. 2, pp. 54–72, 2013.