INVITED: Approximate Computing with Partially Unreliable Dynamic Random Access Memory - Approximate DRAM

Matthias Jung, Deepak M. Mathew, Christian Weis, Norbert Wehn
Microelectronic Systems Design Research Group
University of Kaiserslautern
Kaiserslautern, Germany
{jungma,deepak,weis,wehn}@eit.uni-kl.de

ABSTRACT

In the context of approximate computing, *Approximate Dynamic Random Access Memory* (ADRAM) enables the trade-off between energy efficiency, performance and reliability. The inherent error resilience of applications allows sacrificing data storage robustness and stability by lowering the refresh rate or disabling refresh in DRAMs completely. Consequently, it is important to know exactly the statistical DRAM behavior with respect to retention time, process variation and temperature to manage this trade-off and thereby deliberately exploiting the error resilience of different target applications.

CCS Concepts

•**Hardware** → **Dynamic memory;**

Keywords

Approximate Computing, Approxmiate DRAM, Refresh, Retention Time

1. INTRODUCTION

In the past, *Approximate* and *Probabilistic Computing* evolved as design paradigms that exploit the error resilience of applications to increase their performance and decrease their power consumption [9].

Recent advances in Approximate Computing have been highlighted through a dedicated Special Issue on the topic in the IEEE Design and Test Magazine [10]. A wide range of techniques like Approximate Register File for GPUs [11], Approximate Load Value Prediction [34], Error Prediction for approximate accelerators [16], and RRAM-Based Analog Approximate Computing [19], have been explored.

Moreover, these research activities have been extended from pure computing cores to uncore components, such as building blocks of communication systems [8] or the DRAM controller [7] and further the DRAM itself, often denoted as *Approximate DRAM* [22, 23, 1, 15, 26, 13]. The underlying motivation for an Approximate DRAM is the increasing power consumption and performance penalty caused by unavoidable DRAM refresh commands. The authors of [21] and [2] predicted that 40% to 50% of the power consumption of future DRAM devices will be caused by refresh commands. Moreover, 3D integrated DRAMs like Wide I/O or HMC worsen the scenario with respect to increased cell leakage, due to the much higher temperature. Therefore, the refresh frequency needs to be increased accordingly to avoid retention errors [27].

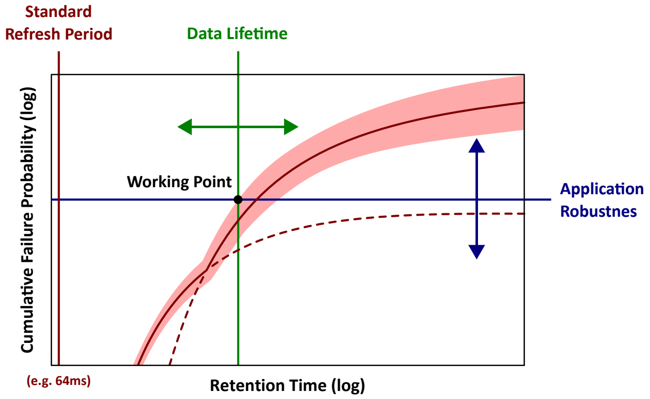

Figure 1: Qualitative Retention Error Behavior

The characteristic parameters of DRAMs, such as timings (e.g. t_{RAS}) and currents (I_{DDX}), listed in datasheets are very pessimistic due to the high process margins added by the vendors to ensure correct functionality under worst-case conditions and a high-enough yield [5, 4, 18]. Similar to these timings and currents, the DRAM refresh rate recommended by the vendors and JEDEC ($t_{REF} = 64\ ms$) adds a large guard band, as shown in Figure 1.

Approximate DRAM exploits this fact by lowering the refresh frequency (reducing the guard bands proposed by the vendors) or even disable the refresh completely and accepting the risk of data errors. Many applications have an inherent error resilience that tolerates these errors and therefore, refresh power often can be reduced with a minimal loss of the output quality.

However, it is very difficult to characterize the weak cells in DRAMs, as they experience *Variable Retention Times* (VRT) and *Data Pattern Dependencies* (DPD) [20, 33] for their retention times. Moreover, in [33] it is shown that the temperature has a strong effect on VRT. Hence, it is

DAC '16, June 05-09, 2016, Austin, TX, USA
© 2016 ACM. ISBN 978-1-4503-4236-0/16/06...$15.00
DOI: http://dx.doi.org/10.1145/2897937.2905002

infeasible during startup of a system to determine an exact list of weak cells of a single DRAM device that considers all parameters, such as temperature, retention time, VRT and DPD to omit these cells from usage.

Thus, the key for Approximate DRAM is to conceive the DRAM device as a stochastic model that also includes process variation. Moreover, the right *Working Point* in the retention error curve for a specific application has to be selected (cf. Figure 1). Therefore, application knowledge with respect to the data lifetime and robustness, as well as knowledge about the retention error behavior of the used DRAM devices with respect to process variation and temperature is indispensable. For instance, refresh can be fully disabled if it is assured that either the lifetime of the data is shorter than the currently required DRAM refresh period or if the application can tolerate bit errors to some degree in a given time window.

Moreover, Figure 1 shows that the retention error curve is composed of two overlaying distributions (tail cells and normal cells) [12]. The distribution of the normal cells is due to the variation of junction leakage, which is influenced by the Si/SiO_2 energy (eV) band. The dominant leakage component of the tail cells is the *Gate Induced Drain Leakage* (GIDL) caused by a trap located at the drain-gate overlap region. These traps are induced by defects during high-temperature processing (high-power plasma etching, oxidation, etc.).

Since the area of the gate overlap region is very small compared with the junction region, most of the cells do not have any trap at the drain-gate overlap region, and their leakage currents solely come from the junction region. However, if a cell has a deep trap located at the gate overlap region, the trap can generate a large leakage current, and the cell becomes a tail cell [12]. Kim and Lee investigated in [17] the further scaling down of DRAM and observed that both distributions will separate, which results in a stronger bend in the retention error curve.

This paper is part of DAC 2016 Special Session "Cross-Layer Approximate Computing: Challenges and Solutions". Other papers in this special session are "Cross-Layer Approximate Computing: From Logic to Architectures" [31], "Cross-Layer Approximations for Neuromorphic Computing: From Devices to Circuits and Systems" [29], and "Programming Uncertain Things" [24].

2. RELATED WORK

Liu et al. presented the first work on Approximate DRAM, called *Flikker* [22], which reduces the number of refreshes by partitioning a DRAM bank in a critical and non-critical region. The non-critical region will be refreshed with a lower refresh rate.

A similar approach is followed by *Quality Aware Approximate DRAM* [26], which characterizes the used DRAM by extensive retention time measurements. As a result the DRAM pages are sorted into quality bins. During allocation critical data is stored in high quality bins, whereas approximate data is allocated in low quality bins. The whole DRAM is then refreshed with the same refresh rate, which makes this approach applicable to today's DRAM devices, since no changes of the internal DRAM structure are required. However, this approach is very time consuming due to the prior characterization and it is very challenging because of the VRT phenomenon. Moreover, there is an overhead to store the essential information to apply this technique (sorted page order).

The *REVA* [1] refresh scheme can be used in dedicated video applications. It refreshes only the important *region of interest* (ROI) in a video frame.

Sparkk [23] proposes the idea of permutation of the data bits on several DRAM chips that are refreshed with different rates. The most significant bits of a byte are located in a highly refreshed DRAM device and the least significant bits are stored in a less refreshed chip.

A more recent idea, *AVATAR* [25], tries to overcome VRT issues by combining an online *Error-Correcting Code* (ECC) mechanism with row selective refresh.

In [13] a holistic simulation environment for investigations on Approximate DRAM based on a *retention error model* [33], *DRAMPower* [6], *DRAMSys* [14] and *3D-ICE* [32] is shown.

Omitting Refresh (OR) [15] shows that for dedicated applications refresh can be disabled completely without or with negligible impact on the application performance. This is possible if it is assured that either the lifetime of the data is shorter than the currently required DRAM refresh period or if the application is error resilient and can tolerate bit errors to some degree in a given time window. It is shown that especially for 3D-integrated systems with Wide I/O DRAM this strategy is beneficial. The lowest 3D-DRAM layer has the highest average temperature. Hence, this layer requires a higher refresh rate than the rest of the DRAM stack [27]. Consequently, the lowest layer is a perfect candidate for applying OR, by tolerating an *unreliable* memory layer in order to save refresh power. While the upper *reliable* part of the stack is refreshed the *unreliable* region can be accessed exclusively, which is managed by the DRAM controller.

The authors of [28] and [30] introduce programming techniques to exploit unreliable memories by distinguishing reliable and unreliable data types.

3. ERROR MODEL

A retention error aware DRAM model is key to analyse the impact of lower refresh rates on the executed application. In [33] we created a model, which is developed in C++ and therefore usable in full system level simulations. The model was calibrated to the measurement results of a WIDE I/O DRAM. In this work we extend the input of the model for DDR3 based DRAMs. It is integrated in our advanced SystemC-TLM2.0 virtual-platform setup [14]. However, it can also be integrated in other simulation environments like gem5 [3].

To calibrate the error model with respect to retention time variations we measured 40 identical 4 Gbit DDR3 chips from the same vendor. Each single device has been measured ten times at four different temperatures and five retention times, resulting in a total of 8000 measurement points, shown in Figure 2. We plot the retention times versus the normalized and averaged number of errors obtained during each measurement step. The bars mark the minimum and the maximum measured number of errors. We find here a quite prominent variation in the order of 20% (max. number of errors), which shows a large temperature dependency. This needs to be considered as realistic guardband in approximate computing platforms utilizing the Approximate DRAM approach (cf. the sphere in Figure 1). Additionally, the figure shows a histogram of the absolute number of bit errors (be-

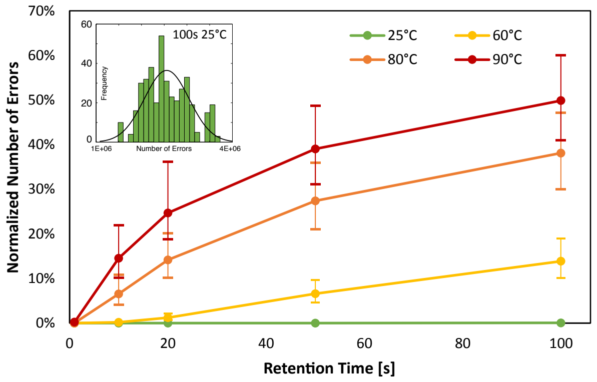

Figure 2: Process Variation of 40 4Gbit DRAM Devices with a `0xFF` Data Pattern

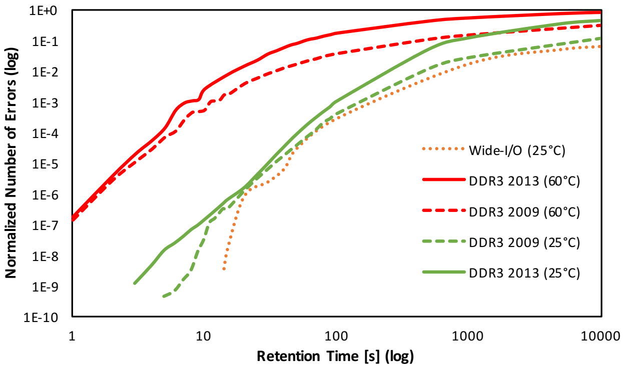

Figure 3: Scaling Trend of DRAM Retention Time

tween $1 \cdot 10^6$ and $4 \cdot 10^6$) measured at the data point with 100s retention time and a temperature of 25°C.

Furthermore, we confirm by our experimental results plotted in Figure 3 the simulative prediction made in [28, 17]. Especially for a temperature of 60°C, we see a strong bend in the curve that is located in the range of 4s to 20s for the DDR3 2013 device (30nm). For the 50nm DDR3 2009 and Wide I/O devices that we measured, we observe a bend in the curve as well. As predicted, this bend is not so distinct and occurs at later retention times compared to the 30nm devices. Further scaling down of DRAM technology will potentially separate more these two distributions.

4. CONCLUSION

When Approximate DRAM is used the DRAM must be conceived as a stochastic model. Thus it is important to quantify today's DRAM's reliability, temperature dependency, and process variation. For the first time we confirmed with the measurement of recent DDR3 SO-DIMMs the scaling trend of DRAMs predicted by [28, 17]. Approximate computing systems using Approximate DRAMs will benefit from the accurate analysis results in terms of lower guardbanding and better prediction possibilities. In the future, we will extend these measurements and our models to the recent DRAM generations, such as DDR4 and LPDDR4/5 to estimate the latest trends of DRAM scaling.

Acknowledgment

The authors thank Martin Schultheis and Vitor Kieling for their support. This work was partially funded by the German Research Foundation (DFG) as part of the priority program Dependable Embedded Systems SPP 1500[1], the DFG grant no. WE2442/10-1 and the Carl-Zeiss Stiftung.

5. REFERENCES

[1] S. Advani, N. Chandramoorthy, K. Swaminathan, K. Irick, Y. Cho, J. Sampson, and V. Narayanan. Refresh Enabled Video Analytics (REVA): Implications on power and performance of DRAM supported embedded visual systems. In *Computer Design (ICCD), 2014 32nd IEEE International Conference on*, pages 501–504, Oct 2014.

[2] I. Bhati, Z. Chishti, S.-L. Lu, and B. Jacob. Flexible auto-refresh: enabling scalable and energy-efficient DRAM refresh reductions. In *Proceedings of the 42nd Annual International Symposium on Computer Architecture*, pages 235–246. ACM, 2015.

[3] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood. The gem5 simulator. *SIGARCH Comput. Archit. News*, 39(2):1–7, Aug. 2011.

[4] K. Chandrasekar, S. Goossens, C. Weis, M. Koedam, B. Akesson, N. Wehn, and K. Goossens. Exploiting Expendable Process-margins in DRAMs for Run-time Performance Optimization. In *Proceedings of the Conference on Design, Automation & Test in Europe*, DATE '14, pages 173:1–173:6, 3001 Leuven, Belgium, Belgium, 2014. European Design and Automation Association.

[5] K. Chandrasekar, C. Weis, B. Akesson, N. Wehn, and K. Goossens. Towards Variation-Aware System-Level Power Estimation of DRAMs: An Empirical Approach. In *Proc. 50th Design Automation Conference*, Austin, USA, June 2013.

[6] K. Chandrasekar, C. Weis, Y. Li, B. Akesson, O. Naji, M. Jung, N. Wehn, and K. Goossens. DRAMPower: Open-source DRAM power & energy estimation tool.

[7] H. Cho, C. Cher, T. Shepherd, and S. Mitra. Understanding Soft Errors in Uncore Components. *CoRR*, abs/1504.01381, 2015.

[8] C. Gimmler-Dumont and N. Wehn. A Cross-Layer Reliability Design Methodology for Efficient, Dependable Wireless Receivers. *ACM Transactions on Embedded Computing Systems*, 2013.

[9] J. Han and M. Orshansky. Approximate computing: An emerging paradigm for energy-efficient design. In *Test Symposium (ETS), 2013 18th IEEE European*, pages 1–6, May 2013.

[10] J. Henkel. Approximate Computing: Solving Computing's Inefficiency Problem? *IEEE Design and Test*, 33(1), February 2016.

[11] D. Jeong, Y. H. Oh, J. W. Lee, and Y. Park. An eDRAM-Based Approximate Register File for GPUs. *IEEE Design & Test*, 33(1):23–31, February 2016.

[1] http://spp1500.itec.kit.edu

[12] S. Jin, J.-H. Yi, J. H. Choi, D. G. Kang, Y. J. Park, and H. S. Min. Prediction of data retention time distribution of DRAM by physics-based statistical Simulation. *IEEE Transactions on Electron Devices*, 52(11):2422–2429, Nov 2005.

[13] M. Jung, D. M. Mathew, C. Weis, and N. Wehn. Efficient Reliability Management in SoCs - An Approximate DRAM Perspective. In *21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016.

[14] M. Jung, C. Weis, and N. Wehn. DRAMSys: A flexible DRAM Subsystem Design Space Exploration Framework. *IPSJ Transactions on System LSI Design Methodology (T-SLDM)*, August 2015.

[15] M. Jung, E. Zulian, D. Mathew, M. Herrmann, C. Brugger, C. Weis, and N. Wehn. Omitting Refresh - A Case Study for Commodity and Wide I/O DRAMs. In *1st International Symposium on Memory Systems (MEMSYS 2015)*, Washington, DC, USA, October 2015.

[16] D. S. Khudia, B. Zamirai, M. Samadi, and S. Mahlke. Quality Control for Approximate Accelerators by Error Prediction. *IEEE Design & Test*, 33(1):43–50, February 2016.

[17] K. Kim and J. Lee. A New Investigation of Data Retention Time in Truly Nanoscaled DRAMs. *Electron Device Letters, IEEE*, 30(8):846–848, Aug 2009.

[18] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang, and O. Mutlu. Adaptive-latency DRAM: Optimizing DRAM timing for the common-case. In *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*, pages 489–501, Feb 2015.

[19] B. Li, P. Gu, Y. Wang, and H. Yang. Exploring the Precision Limitation for RRAM-Based Analog Approximate Computing. *IEEE Design & Test*, 33(1):51–58, February 2016.

[20] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu. An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms. *SIGARCH Comput. Archit. News*, 41(3):60–71, June 2013.

[21] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *Proceedings of the 39th Annual International Symposium on Computer Architecture*, ISCA '12, pages 1–12, Washington, DC, USA, 2012. IEEE Computer Society.

[22] S. Liu, K. Pattabiraman, T. Moscibroda, and B. G. Zorn. Flikker: Saving DRAM Refresh-power Through Critical Data Partitioning. *SIGPLAN Not.*, 46(3):213–224, Mar. 2011.

[23] J. Lucas, M. Alvarez-Mesa, M. Andersch, and B. Juurlink. Sparkk: Quality-Scalable Approximate Storage in DRAM. In *The Memory Forum*, June 2014.

[24] T. Mytkowicz. Programming Uncertain Things. In *ACM/IEEE Design Automation Conference (DAC), (Presentation Only)*, 2016.

[25] M. K. Qureshi, D.-H. Kim, S. Khan, P. J. Nair, and O. Mutlu. AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems. *Memory*, 2(4Gb):20, 2015.

[26] A. Raha, H. Jayakumar, S. Sutar, and V. Raghunathan. Quality-aware Data Allocation in Approximate DRAM. In *Proceedings of the 2015 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, CASES '15, pages 89–98, Piscataway, NJ, USA, 2015. IEEE Press.

[27] M. Sadri, M. Jung, C. Weis, N. Wehn, and L. Benini. Energy Optimization in 3D MPSoCs with Wide-I/O DRAM Using Temperature Variation Aware Bank-Wise Refresh. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–4, March 2014.

[28] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman. EnerJ: Approximate Data Types for Safe and General Low-power Computation. In *Proceedings of the 32Nd ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '11, pages 164–174, New York, NY, USA, 2011. ACM.

[29] S. Sarwar, G. Srinivasan, S. Venkataramani, A. Sengupta, A. Raghunathan, and K. Roy. Cross-Layer Approximations for Neuromorphic Computing: From Devices to Circuits and Systems. In *ACM/IEEE Design Automation Conference (DAC)*, 2016.

[30] F. Schmoll, A. Heinig, P. Marwedel, and M. Engel. Improving the Fault Resilience of an H.264 Decoder Using Static Analysis Methods. *ACM Trans. Embed. Comput. Syst.*, 13(1s):31:1–31:27, Dec. 2013.

[31] M. Shafique, R. Hafiz, S. Rehman, W. El-Harouni, and J. Henkel. Cross-Layer Approximate Computing: From Logic to Architectures. In *ACM/IEEE Design Automation Conference (DAC)*, 2016.

[32] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza. 3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling. In *Proc. of ICCAD 2010*, 2010.

[33] C. Weis, M. Jung, P. Ehses, C. Santos, P. Vivet, S. Goossens, M. Koedam, and N. Wehn. Retention Time Measurements and Modelling of Bit Error Rates of WIDE I/O DRAM in MPSoCs. In *Proceedings of the IEEE Conference on Design, Automation & Test in Europe (DATE)*. European Design and Automation Association, 2015.

[34] A. Yazdanbakhsh, B. Thwaites, H. Esmaeilzadeh, G. Pekhimenko, O. Mutlu, and T. C. Mowry. Mitigating the Memory Bottleneck With Approximate Load Value Prediction. *IEEE Design & Test*, 33(1):32–42, February 2016.