JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.21, NO.2, APRIL, 2021
https://doi.org/10.5573/JSTS.2021.21.2.134
ISSN(Print) 1598-1657
ISSN(Online) 2233-4866

# An Approximate DRAM Design with an Adjustable Refresh Scheme for Low-power Deep Neural Networks

**Duy Thanh Nguyen[1], Hyun Kim[2], and Hyuk-Jae Lee[1]**

***Abstract*—A DRAM device requires periodic refresh operations to preserve data integrity, which incurs significant power consumption. Slowing down the refresh rate can reduce the power consumption; however, it may cause a loss of data stored in a DRAM cell, which affects the correctness of computation. This paper proposes a new memory architecture for deep learning applications, which reduces the refresh power consumption while maintaining accuracy. Utilizing the error-tolerant property of deep learning applications, the proposed memory architecture avoids the accuracy drop caused by data loss by flexibly controlling the refresh operation for different bits, depending on their criticality. For data storage in deep learning applications, the approximate DRAM architecture reorganizes the data so that these data are mapped to different DRAM devices according to their bit significance. Critical bits are stored in more frequently refreshed devices while non-critical bits are stored in less frequently refreshed devices. Compared to the conventional DRAM, the proposed approximate DRAM requires only a separation of the chip select signal for each device in a DRAM rank and a minor change in the memory controller. Simulation results show that the refresh power consumption is reduced by 66.5 % with a negligible accuracy drop on state-of-the-art deep neural networks.**

***Index Terms*—Deep learning, approximate DRAM, low power DRAM, bit-level refresh, fine-grained refresh**

## I. INTRODUCTION

Deep learning applications require large amounts of computation with excessive data traffic from the memory [1]. Therefore, DRAM energy consumption is substantial in the total system energy. Beside the power consumed by data access to DRAM, a significant portion of DRAM power is made by refresh operation. It is necessary to read out data periodically and then written them back into the same memory cells because DRAM cells cannot retain the stored data permanently. This refresh operation is necessary for all cells in a DRAM, whether they store significant data or not. Therefore, this refresh operation results in significant power consumption even though certain DRAM cells do not store the data that are accessed by an active process in the processor. As the DRAM density increases, the refresh power consumed becomes prominent. In a future 64 Gb DRAM device, the refresh operation is expected to account for up to 50% of the total power consumption [2].

Accordingly, various refresh management techniques have been proposed to reduce power consumption [2-7]. In three studies, [2-4], the OS is requested by default to figure out the retention time information for each DRAM

Manuscript received Jan. 19, 2021; accepted Mar. 4, 2021
[1]Inter-University Semiconductor Research Center, Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Korea
[2]Department of Electrical and Information Engineering and the Research Center for Electrical and Information Technology, Seoul National University of Science and Technology, Seoul 01811, Korea
E-mail : hyunkim@seoultech.ac.kr (Corresponding author)
Extended from a Conference: Preliminary results of this paper were presented at the IEIE/IEEE International Conference on Electronics, Information, and Communication (ICEIC), 2020. This paper has been accepted by the editorial board through the regular reviewing process that confirms the original contribution.

row, and a DRAM controller uses this information to selectively perform refresh operation for each row. These techniques significantly reduce the refresh power consumption by skipping unnecessary refresh commands. However, with increasing memory sizes, it is not cost-effective and not scalable to store the retention time information of all the rows of the DRAM. Moreover, profiling the retention time information of all DRAM cells requires a significant amount of effort, and incorrect results may be obtained, as it has to deal with Variable Retention Time and Data Pattern Dependencies [8]. Two studies in [5] and [6] propose another approach to reduce the refresh power consumption by allowing the possibility of occurrence of a small amount of errors in DRAM cells. In [5], Flikker provides a software solution for the approximate memory by partitioning data into critical and non-critical data. It leverages the Partial-Array Self-Refresh Mode (PASR) [9], which supports different refresh rates for the different sections of the same DRAM bank. However, since all bits of data are unprotected from errors, in the case of floating data, an erroneous most significant bit (MSB) may change the data on a very large scale, which causes overflow in computation. Providing a finer-grained refresh control than [5], Sparkk in [6], uses varying refresh periods for different bits based on their importance. Sparkk protects the critical bits of integer data, thus making the applications more robust to an error than in the method proposed by Flikker; however, it lacks in software support and does not provide power measurement and performance evaluation based on architectural simulation. Different from Sparkk and Flikker, the studies in [7], and [18] proposes a transposed approximate memory architecture. A row-level refresh scheme is applied to refresh data bits according to their criticality. This scheme saves 69.68% of the refresh power with a negligible accuracy drop. However, a block of data needs to be buffered and transposed in the memory controller before saving to the external memory. It potentially causes the long latency for data access.

It is widely known that deep learning applications are error tolerant [10, 11]. Therefore, based on this property, this paper proposes an approximate DRAM with a fine-grained refresh scheme. The proposed scheme maps bits of data to different DRAM devices according to their bit significance. The critical bits are stored in more frequently refreshed devices to prevent the accuracy drop, whereas the non-critical bits are mapped to less frequent refreshed device thereby saving refresh power considerably. The simulation results show that the proposed approximate DRAM can significantly reduce the refresh power consumption by up to 66.5% with negligible degradation in accuracy. It is worth mentioning that this paper is extended from the previous work in [28]. In this paper, the hybrid memory architecture and evaluation methodology are elaborated in more detail. In addition, the evaluation of VGGNet inference with the real approximate memory on the FPGA platform is added to verify the correctness of the assumed error injection model.

The rest of this paper is organized as follows. Section II presents the proposed approximate architecture. Section III describes the evaluation methodology. In Section IV, the evaluation results are provided, and finally the conclusions are presented in Section V.

## II. PROPOSED APPROXIMATE DRAM

The main goal of the proposed approximate memory architecture is to apply different refresh rates to the different bits of data depending on the significance of the bits. The critical bits (*i.e.*, sign bit and exponent bits) should be placed in the error-free zone with the normal refresh rate. On the other hand, the non-critical bits (*i.e.*, mantissa bits) are placed in the erroneous zone with a slower refresh rate to achieve power saving during refresh operations.

A deep learning application, in general, accesses 32-bit floating-point data [12]. The effectiveness of the approximate memory architecture depends on separating important data from the other data and refreshing them at the normal rate, while the other insignificant data are refreshed at a much slower rate, which may cause errors. The more fined-grained the critical data partitioning is, the more refresh reduction it can achieve. This paper proposes an approximate DRAM architecture, as shown in Fig. 1. This architecture is refreshed with different fined-grained schemes while protecting the 9 critical bits. This architecture, so called *APPROX2*, refreshes data at granularity of 2 bits. The least significant bits (LSBs) of the data are refreshed at slower rate than the MSBs. To reduce the design effort, hardware overhead, and

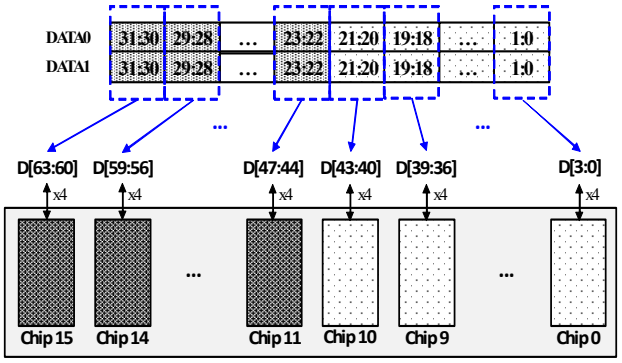



**Fig. 1.** Approximate memory architecture: *APPROX2*. Dark device: precise, dotted white device: approximate.

performance maintenance, this paper proposes a refresh scheme at DRAM device granularity. The DRAM devices for critical data are called precise devices and are marked with a dark color in Fig. 1. The precise devices are refreshed at the normal rate. On the other hand, the DRAM devices for non-critical data are called approximate devices and are marked with a dotted white color in Fig. 1. It should be noted that, for simplicity, the refresh period of approximate devices should be a multiple of the normal refresh period (*tRF* = 64 ms). In a conventional DRAM module, when the memory controller issues the refresh command, all the DRAM devices in a rank are refreshed simultaneously, and are thus refreshed at the same rate. To skip refresh operations for the approximate devices, the proposed DRAM architecture requires separate chip select signals for devices in the same rank, but this separation incurs a small area overhead. Furthermore, the design of the memory controller for the proposed architecture does not change much from the conventional design. To this end, in the new memory controller design, besides the refresh interval counter (*tRFI* = 7.8 µs) and row counter, a retention time round counter (multiple of 64 ms) is needed to decide whether a specific device needs to be refreshed during the current 64-ms round.

In most computing systems, the cache line size is 64 bytes. To optimize the latency when reading or writing cache lines from/to memory, one cache line should be fit in one DRAM full burst access (*BL* = 8). Since this paper focuses on the approximate memory for the class of deep learning applications, it is assumed that the cache line that is evicted/loaded to/from the approximate DRAM contains only single-precision floating-point data. Therefore, a 64-byte cache line contains 16 data words.

Since the refresh period is different for each group of 2 nearby bits, 32/2=16 devices are needed to form a rank. To keep the bit width of the DRAM rank unchanged (64 bits), the ×4 DRAM device is used. The data mapping of *APPROX2* is also shown in Fig. 1. Two corresponding bits of each data are merged and stored in a proper device. For example, {DATA1[31:30] and DATA0[31:30]} are stored in chip 15, {DATA1[29:28] and DATA0[29:28] are stored in chip 14, and so on. To keep the 9 critical bits (*i.e.*, DATA[31:23]) protected, the chips 11, 12, 13, 14, and 15 must be error-free, and therefore refreshed at the normal rate. Other chips can contain erroneous bits, and are refreshed at a slower frequency. The refresh policy for *APPROX2* is expressed as follows:

$$RP(n) = \begin{cases} 64 & for\ 11 \le n \le 15 \\ (10 - \mathrm{n}) * incr + offset & for\ 0 \le n \le 10 \end{cases} \quad (1)$$

where *RP(n)* represents the refresh period of *n-th* device and (*incr, offset*) are chosen experimentally. Therefore, a maximum refresh power saving of 11/16=68.75% can be achieved when the refresh operation of the approximate devices is turned off completely. Further evaluation with simulation results are provided in Section IV.

The instruction set architecture (ISA), operating system (OS), and compiler supports for systems with approximate memory are well described in [5] and [13]. For the experiments in this study, a custom memory allocator is built to allocate the approximate data to the approximate memory. Similar to that in [5], the critical data partitioning is done by the programmer. Non-critical data are allocated in the approximate memory space. On the other hand, critical data such as instruction codes are stored in the precise memory space. In deep learning applications, if slight errors occur in the weights and feature maps, the performance might not degrade significantly. Therefore, they are non-critical data and can be stored in the approximate memory. This study uses the aforementioned memory allocator to allocate these data to the approximate memory. A pintool is written based on the Pin dynamic-instrumentation tool [14] to calculate the number of approximate pages over the total number of pages accessed by the program. It monitors all the memory accesses of the program to the approximate memory space and accumulates the total

number of accessed approximate pages. The memory footprint profiling results of some deep networks, AlexNet [21], VGGNet [15] and GoogleNet [16], listed in Table 1 show that the portion of critical data in the total memory access is very small. Therefore, the refresh power can be considerably reduced by using the approximate memory. A practical hybrid memory system may contain precise devices and approximate devices, as depicted in Fig. 2.

## III. CIRCUIT DESCRIPTION

### 1. Architectural Simulation

To estimate the performance and power consumption of the proposed system when running deep learning applications by simulation, a cycle-accurate integrated simulator [17] based on Pin [14] binary instrumentation is used for fast simulation, detailed micro-architecture, and detailed DRAM memory subsystem. The memory bandwidth and detailed power consumption information of the memory subsystem can be obtained, which calculates the DRAM power consumption by following the methodology described in [19]. The front-end Pin-based simulator is enhanced to emulate the OS kernel functionality for custom memory allocation. The non-critical data of the CNN programs are allocated to approximate virtual pages. These approximate virtual pages are implicitly assigned to the approximate physical page area. The system configuration is described in Table 2. The system has 4 out-of-order processor cores sharing the L2 cache, and each core has its own L1 instruction cache and data cache. A single memory channel is connected to a 16 GB Micron DDR3-1333 memory module consisting of ×4 and ×8 4 Gb devices.

Based on the pre-trained models from Caffe library [20], this study builds the inference code of some state-of-the-art CNNs such as AlexNet [21], VGGNet [15], and GoogLeNet [16] for performing architectural simulation and power measurement. The separation of approximate data from the normal data may cause performance overhead, because it potentially impacts locality and parallelism. How this separation affects the system performance is presented in Section IV.

**Table 1.** Memory footprint profiling from [18]

| Deep Networks | AlexNet | VGGNet | GoogLeNet |
|---|---|---|---|
| Proportion of approximate pages | 99.78% | 99.85% | 99.67% |

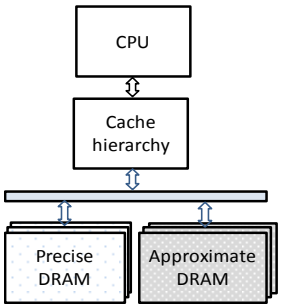



**Fig. 2**. Hybrid memory system.

**Table 2**. System configuration

| Parameters | Values |
|---|---|
| Processor | 4 out-of-order cores 2.53 GHz, 8-issue per core, 128 instruction queue, 64 ROB size, ×86 ISA support |
| L1I, L1D cache | 16 KB per core, 4-way associative |
| L2 cache | 1 MB shared cache, 4-bank, 16-way associative |
| Memory Controller | Single channel, open page, FR-FCFS, 64-entry queues per rank |
| DRAM | 16 GB DDR3-1333 using 4 Gb ×4, ×8 Micron devices |

### 2. Simulation of Deep Learning Inference with Bit Error Injection

The most recent research [22] has shown that the retention time is not equal for all DRAM cells. Instead, most cells have high retention time (called strong cells) while only a few cells (called leaky cells) have low retention time. This research also shows that DDR3 DRAM cells can hold their values longer than 64 ms. At extremely high temperatures such as 80 °C and 90 °C, the bit flip rate increases exponentially. The bit error probability of customized refresh periods used in this paper is referred to [7]. For each bit in the approximate data, it is injected with an error having a corresponding probability depending on its criticality. The simulation of the proposed architecture described in Section II is injected at a bit granularity.

This study uses the pre-trained model from the Caffe library for GoogLeNet [13] as the baseline for analyzing the accuracy effect of approximate memory. The simulation results for other networks such as AlexNet and VGGNet are expected to be similar to GoogleNet as shown in [7].

A set of 10,000 test images from ImageNet 2012 dataset [23] is used to measure the accuracy of the CNNs using the proposed approximate memory. The inference accuracy of each test is the Top-1 accuracy normalized to the accuracy of the corresponding test without using memory approximation. The experiments are conducted with various (*offset, incr*) values under different working temperatures.

### 3. CNN Tests on Real Approximate DRAM on FPGA

To verify the practicality of the error injection model, this study presents a hardware platform as shown in Fig. 3. An FPGA board is connected to the host computer via the Peripheral Component Interconnect Express (PCIe) port. The Direct Memory Access (DMA) module transfers data between the host computer and the DDR3 DRAM. The memory controller of the DRAM is customized to study the behavior of the approximate DRAM through long refresh periods. Because a device for temperature control is not available, these experiments are conducted at room temperatures. At low temperatures, the refresh period can be set even up to 100 s. A pre-trained model of VGGNet is used in this test. It should be noted that the CNN algorithms run on the host computer, not on the FPGA board.

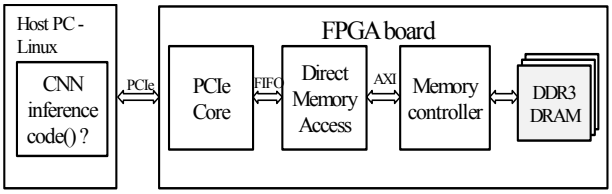
The role of the approximate DRAM on the FPGA board is to inject error into the CNN model at runtime. For each test, the CNN model is temporarily saved in the approximate DRAM for a refresh period, then copied back to the host PC to run the classification.

## IV. Simulation Results

### 1. Performance and Refresh Power Reduction

For the proposed approximate architecture, the reduction of refresh power consumption is derived mathematically from the following equations:

$$\begin{aligned} \text{Psave}\,2 &= 1 - \frac{10 + 64\sum_{n=0}^{10} 2/(n*incr + offset)}{32} \\ &= 0.6875 - \sum_{n=0}^{10} \frac{4}{(n*incr + offset)} \end{aligned} \quad (2)$$




(a)

(b)

**Fig. 3.** Hardware platform for CNN testing on real approximate DRAM (a) Block diagram, (b) Hardware setup for test.

where (*offset, incr*) are the aforementioned parameters. The refresh power reduction of the proposed architecture with respect to their parameters is shown in Fig. 4. When the *offset* is 4,096 ms, the refresh power reduction is almost saturated. Thus, for the approximate devices, the refresh period needs not be longer than 4,096 ms.

The system performance of the three deep networks is listed in Table 3. The *baseline* represents the results of the inference programs running on conventional memory, whereas the others represent the results of programs running on the proposed approximate memory. The proposed architecture has a similar instruction per cycle (IPC) to *baseline*. This is because the approximate architecture requires no change in the DRAM's number of IOs and cache line size.

The breakdown of the power consumption is depicted in Fig. 5. The simulation assumes that one-fourth of the memory space is precise and three-fourths is approximate. This assumption makes sense as the memory footprint profiling results listed in Table 1 indicate that most of the memory accessed pages can be approximated. The approximate part is refreshed during a long period so that the refresh power is almost zero. In addition, the power consumption of the proposed scheme is normalized with the *baseline*, which uses conventional memory. *APPROX2* consumes more background power than the *baseline* because it uses twice more DRAM chips (×4

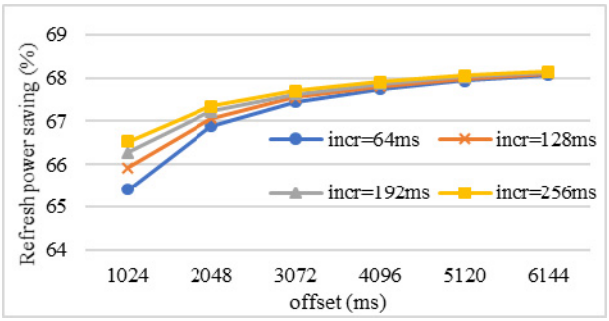



**Fig. 4**. Refresh power reduction of *APPROX2* w.r.t. (*offset, incr*).

**Table 3.** System performance (IPC)

| Networks | Test scenario | IPC |
|---|---|---|
| AlexNet | *Baseline*<br>*APPROX2* | 1.561<br>1.563 |
| VGGNet | *Baseline*<br>*APPROX2* | 1.212<br>1.211 |
| GoogLeNet | *Baseline*<br>*APPROX2* | 1.798<br>1.804 |

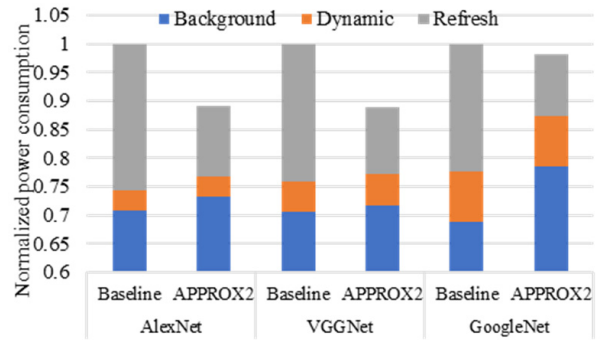



**Fig. 5**. Power breakdown of some deep networks.

devices) in a rank and the background power consumed by the ×4 chip is larger than the power half-consumed by the ×8 device. However, because refresh power is forcefully reduced, the total power consumption of *APPROX2* is less than that of the *baseline*. It skips 68.1% of the refresh operations for the approximate part, while preserving system performance. Fig. 5 shows that *APPROX2* saves up to 11.1 % of the total power consumption. More saving in the future is expected when the computing system uses larger memory modules composed of higher density devices such as 16, 32, and 64 Gb memory. As predicted in [2], the refresh power accounts for 50% of the total power consumption when using 64 Gb devices. Therefore, the refresh saving can achieve 25.5% (=0.75×50%×68.1%) of the total energy for the hybrid memory system assumed in this experiment.

## 2. Simulation of Classification Tasks with Bit Error Injection

At temperatures lower than 60 °C, the accuracy loss is negligible with all (*offset, incr*) pairs. However, the accuracy drop is significant for temperatures higher than 60 °C. It can be seen in Fig. 6 that with *offset* = 3072, 4096 ms, an accuracy degrades significantly. On contradictory, with *offset* = 1024 ms, the accuracy loss is negligible (within 1%) at both 80 and 90 °C. The refresh saving achieved with the (1024, 256) option is as high as 66.5%.

## 3. CNN Tests on Real Approximate DRAM on FPGA

Fig. 7 shows that the error injection model simulates well the error pattern of real approximate DRAM. Fig. 7(a) shows the result of the FPGA test and Fig. 7(b) is constructed based on the test results of the bit error model. The experimental results show that the performance of the real approximate DRAM is slightly better than that of the error injection. This is because the error probabilities used in the bit injection are the worst-case values measured in the real approximate DRAM [22]. At room temperature, although there is no critical bit protection, the refresh period can be prolonged to tens of seconds with very small performance loss.

## 4. Comparison with the Previous Works

This study attempts another approach that greatly reduces the power consumption by allowing the error occurrence in the DRAM cells. Previous studies on approximate memory through refresh control are presented in [5] and [6]. In [5], the experiments show a slight reduction of 20-25% self-refresh power in the idle mode only. Research in [6] does not indicate any power measurements. The transposed architecture in [7] saves 70% of refresh power consumption at cost of additional buffers and control logics. On the other hand, the proposed approximate memory design requires only separate chip select signals for DRAM devices in the same rank and minor change in the memory controller. Meanwhile, the proposed architecture reduces 66.5% of the refresh power with negligible performance loss at any operating temperatures. It should be noted that the

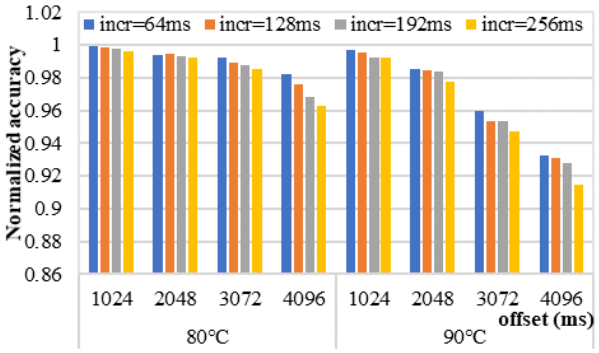
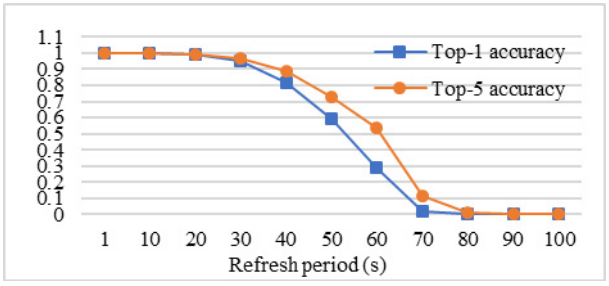
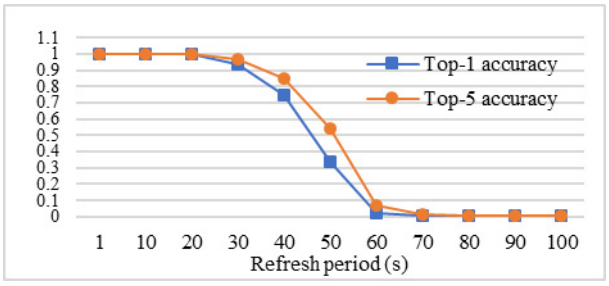



**Fig. 6.** Accuracy drop of *APPROX2* w.r.t (*offset, incr)* and high temperatures.




(a)




(b)

**Fig. 7.** Comparison of VGGNet test results on real approximate DRAM and bit injection simulation (a) FPGA test, (b) Bit injection simulation.

proposed architecture compensates for the problems of previous studies by reducing a significant amount of power consumption at various temperatures.

There are various compression methods such as pruning [24] and quantization [25-27], which aggressively reduce the memory access, thereby reducing DRAM power consumption. Different from low-bit quantization, the proposed scheme does not require additional effort of fine-tuning or retraining to preserve the accuracy of the networks. Therefore, a floating-point data in original CNN model trained by GPU can be used directly in the proposed architecture.

Compared to relevant research in [7, 18], which is able to reduce 26.0 % of the total energy for the hybrid memory, the proposed design achieves a competitive performance. Moreover, it does not require additional hardware overhead for multiple transposed buffers and address translation.

## V. Conclusions

This paper presents an efficient approximate DRAM design which can save 66.5 % of refresh power consumption and up to 25.5% of the total memory power consumption for future 64Gb device. The error robustness of the proposed scheme is confirmed by testing at extremely high operation temperature. It is noteworthy that the proposed approximate DRAM requires a minor change in the memory controller without changing the DRAM device or incurring any significant additional hardware resources.

## Acknowledgments

This study was supported by the Research Program funded by the SeoulTech(Seoul National University of Science and Technology).

## References

[1] Z. Deng, C. Xu, Q. Ci, and P. Faraboschi, “Reduced-precision memory value approximation for deep learning,” [Online]. Available: http://www.labs.hpe.com/techreports/2015/HPL-2015-100.html.

[2] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu, “RAIDR: Retention-aware intelligent DRAM refresh,” in *Proc. 39th Annu. Int. Symp. Comput. Archit.*, 2012, pp. 1-12.

[3] I. Bhati, Z. Chishti, S. Lu, B. Jacob, “Flexible auto-refresh: Enabling scalable and energy-efficient DRAM refresh reductions” in *Proc. 42nd Annu. Int. Symp. Comput. Archit*., 2015, pp. 235-246.

[4] A. Raha, S. Sutar, H. Jayakumar, V. Raghunathan, “Quality configurable approximate DRAM”, *IEEE Trans. Comput*., vol. 66, no. 7, pp. 1172-1187, July 2017.

[5] S. Liu, K. Pattabiraman, T. Moscibroda, and B. G. Zorn, “Flikker: Saving DRAM Refresh-power

through Critical Data Partitioning," in *Proc. 16th Int. Conf. Archit. Support Program. Languages Operating Syst.*, 2011, pp. 213-224.

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

[7] D. T. Nguyen, H. Kim, H.-J. Lee, I.-J. Chang, "An approximate memory architecture for a reduction of refresh power consumption in deep learning applications," in *Proc. IEEE Int. Symp. Circuit Syst.,* 2018, pp. 1-5.

[8] 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," in *Proc. 40th Int. Symp. Comput. Archit*., 2013, pp. 60-71.

[9] *1Gb Mobile LPDDR*, Micron Technology. [Online]. Available: www.micron.com/resource-details/1b6bc910-f3a6-4aa4-9fe7-0265026cf1d3.

[10] M. Courbariaux, J. David, Y. Bengio, "Training deep neural networks with low precision multiplications," [Online]. Available: arXiv: 1412. 7024.

[11] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1737-1746.

[12] N. Whitehead and A. Fit-florea, "Precision & Performance: Floating Point and IEEE 754 Compliance for NVIDIA GPUs," [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi= 10.1.1.231.215.

[13] Samson et. al., "EnerJ: Approximate data types for safe and general low-power computation", in *Proc. 32nd ACM SIGPLAN Conf. Program. Language Des. Implementation*, 2011, pp. 164-174.

[14] C. K. Luk *et al.*, "Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation," in *Proc. ACM SIGPLAN Conf. Prog. Lang. Des. Imp.*, 2005, pp. 190-200.

[15] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition", [Online]. Available: arXiv:1409.1556.

[16] Szegedy et al., "Going deeper with convolutions", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2015, pp. 1-9.

[17] K. Bick, D. T. Nguyen, H. -J. Lee, and H. Kim, "Fast and Accurate Memory Simulation by Integrating DRAMSim2 into McSimA+," *MDPI Electronics*, vol. 7, no. 8, p. 152, 2018.

[18] D. T. Nguyen, H. H. Nguyen, H. Kim, and H.-J. Lee, "An Approximate Memory Architecture for Energy Saving in Deep Learning Applications," *IEEE Trans. Circuits Syst. I, Reg. Papers,* vol. 67, no. 5, pp. 1588-1601, May, 2020.

[19] *Calculating Memory System Power for DDR3*, Micron Technology, 2007.

[20] Y. Jia et al., "Caffe: Convolutional architecture for fast feature embedding", in *Proc. 22nd ACM Int. Conf. Multimed.*, 2014*,* pp. 675-678.

[21] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet classification with deep convolutional neural networks", in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, vol. 1, pp. 1097-1105.

[22] M. Jung et al., "A Platform to analyze DDR3 DRAM's power and retention time", *IEEE Des. Test, ,* pp. 52-59, 2017.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg and L. Fei-Fei. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.*, pp. 211-252, 2015.

[24] S. Han, J. Pool, J. Tran, W. J. Dally, "Learning both weights and connections for efficient neural network,". [Online]. Available: arxiv.org/abs/1506. 02626.

[25] D. T. Nguyen, T. N. Nguyen, H. Kim, and H.-J Lee. "A High-Throughput and Power-Efficient FPGA Implementation of YOLO CNN for Object Detection," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.,* vol. 27, no. 8, pp. 1861-1873, 2019.

[26] D. D. Lin, S. S. Talathi, V. S. Annapureddy, "Fixed Point Quantization of Deep Convolutional Networks," in *Proc. Int. Conf. Mach. Learn. (ICML),* 2016, pp. 2849-2858.

[27] M. Restegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," [Online]. Available: arxiv.org/abs/1603.05279.

[28] D. T. Nguyen, H.-J. Lee, H. Kim, and I.-J. Chang, "An approximate DRAM design with a flexible refresh scheme for low power deep learning applications," in *Int. Conf. on Electronics, Information, and Communication, 2020.*

**Duy Thanh Nguyen** received the B.S. degree in electrical engineering from Hanoi University of Science and Technology, Hanoi, Vietnam, M.S. degree in Electrical and Computer Engineering from Seoul National University, Seoul, Korea, in 2011 and 2014, respectively. He is currently working toward the Ph.D. degree in Electrical and Computer Engineering at Seoul National University. His research interests include computer architecture, memory system, SoC design for computer vision applications.

**Hyun Kim** received the B.S., M.S. and Ph.D. degrees in Electrical Engineering and Computer Science from Seoul National University, Seoul, Korea, in 2009, 2011 and 2015, respectively. From 2015 to 2018, he was with the BK21 Creative Research Engineer Development for IT, Seoul National University, Seoul, Korea, as a BK Assistant Professor. In 2018, he joined the Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul, Korea, where he is currently working as an Assistant Professor. His research interests are the areas of algorithm, computer architecture, memory, and SoC design for low-complexity multimedia applications and deep neural networks.

**Hyuk-Jae Lee** received the B.S. and M.S. degrees in electronics engineering from Seoul National University, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in Electrical and Computer Engineering from Purdue University, West Lafayette, IN, in 1996. From 1996 to 1998, he was with the Faculty of the Department of Computer Science, Louisiana Tech University, Ruston, LS. From 1998 to 2001, he was with the Server and Workstation Chipset Division, Intel Corporation, Hillsboro, OR, as a Senior Component Design Engineer. In 2001, he joined the School of Electrical Engineering and Computer Science, Seoul National University, South Korea, where he is currently a Professor. He is a Founder of Mamurian Design, Inc., a fabless SoC design house for multimedia applications. His research interests are in the areas of computer architecture and SoC design for multimedia applications.