Low Power Aging-Aware Register File Design by Duty Cycle Balancing

Shuai Wang, Tao Jin, Chuanlei Zheng, and Guangshan Duan
State Key Laboratory of Novel Software Technology
Department of Computer Science and Technology
Nanjing University, Nanjing, Jiang Su, China
Email: swang@nju.edu.cn, {taojin, zhengchl, guangshan_duan}@smail.nju.edu.cn

***Abstract*—The degradation of CMOS devices over the lifetime can cause the severe threat to the system performance and reliability at deep submicron semiconductor technologies. The negative bias temperature instability (NBTI) is among the most important sources of the aging mechanisms. Applying the traditional guardbanding technique to address the decreased speed of devices is too costly. Due to presence of the narrow-width values, integer register files in high-performance microprocessors suffer a very high NBTI stress. In this paper, we propose an aging-aware register file (AARF) design to combat the NBTI-induced aging in integer register files. The proposed AARF design can mitigate the negative aging effects by balancing the duty cycle ratio of the internal bits in register files. By gating the leading bits of the narrow-width values during the register accesses, our AARF can also achieve a significantly power reduction, which will further reduce the temperature and NBTI degradation of integer register files. Our experimental results show that AARF can effectively reduce the NBTI stress with a 36.9% power saving for integer register files.**

I. Introduction

With continuous scaling of the semiconductor technology, the degradation of the performance and reliability of the CMOS devices over the lifetime due to aging mechanisms has become a major concern [1]. The increased current density and temperature in future devices will further accelerate the degradation. Bias temperature instability (BTI), hot-carrier injection, and gate-oxide wearout are the primary aging mechanisms for CMOS devices [2][3][4]. The negative bias temperature instability (NBTI) for pMOS devices is one of the most prominent and persistent threats for future technologies. NBTI will cause an increase in the threshold voltage (V_{th}) of the pMOS devices when negative voltage is applied at the gate (logic '0'). The threshold voltage can be increased by 50mV, which may result in a degradation of the circuit speed by 20% or cause functional failure during the expected lifetime [5][6].

The conventional methodology to address the decreased speed of devices due to NBTI is guardbanding. The guardbanding is a technique where the operating frequency is reduced in order to overcome the degradation that may be incurred over the lifetime of the devices. For example, a large guardband of 20% in cycle time is required, given that the circuit speed may be reduced by 20% due to NBTI. The conventional guardbanding technique is too expensive because of the worst-case behavior caused by the uneven utilization of different devices on the chip. Moreover, in future technologies, the guardbanding technique may not be suitable to guarantee performance and reliability requirements for future devices [1].

In general, the aging of devices are proportional to the device stress time and the switching frequency of the internal nodes. Therefore, if a device has a highly biased duty cycle ratio, i.e., logic '0' for the pMOS device, it will have a heavy stress and the aging of the device will be accelerated. Since the register file is holding the current processor context, as well as intermediate computation results, the performance and reliability of the register file are very important for high-performance and reliable microprocessor design. However, due to the presence of the narrow-width values (the data with many leading 0s/1s can be represented by fewer bits than the full data width), integer register files suffer a very highly biased duty cycle ratio, thus a heavy NBTI stress, especially for these leading 0s nodes in the register entries. If we adopt the traditional guardbanding design based on the worst-case behavior, it will result in dramatic increases in the guardbands and estimated devices failures. Therefore, microarchitecture solutions to balance the utilization and the aging stress are needed.

In this paper, we propose an aging-aware register file (AARF) design to combat the lifetime degradation in the performance and reliability of the integer register file by exploiting the narrow-width values. Based on the fact that the leading bits (leading 30 bits for 64-bit data) of narrow-width register values are not needed during register accesses, we propose to bit-flip/complement these leading bits periodically. By carefully choosing the bit-flipping/complementing time interval, the average duty cycle ratio of the integer register file can be well balanced with negligible performance and power overheads. Further, to reduce the power consumption of the register file, the leading bits of the narrow-width values are gated during the register access for power saving. Previous research [7][8] shows that the increasing device operating temperature will accelerate the NBTI degradation. Therefore, our low power AARF design can further reduce the temperature (power density) and mitigate the NBTI degradation of the register file.

The rest of the paper is organized as follows. In the next section, we discuss related work in aging-aware/NBTI-aware designs. In Section III, we provide detailed designs of our proposed low power AARF. We present our experimental setup and results in Section IV. Section V draws the conclusion and discusses the future work.

978-3-9810801-8-6/DATE12/©2012 EDAA

II. Related Work

As the technology is continuously scaling down, the exacerbated performance and reliability concerns caused by the lifetime degradation of CMOS devices have drawn a wealth of research. In [9], the impact of the NBTI on the SRAM cells was studied and an NBTI-aware SRAM structure operating in the inverted mode during half of the time was proposed. Albert et al. proposed and evaluated the design of Penelope, an NBTI-aware processor [10]. Penelope consists of generic strategies to mitigate the degradation in both combinational and storage blocks. It has global strategies as well as specific mechanisms to protect all types of structures, such as memory-like blocks, in the processor. A microarchitecture redundancy scheme was proposed by Shin et al. for combating NBTI-induced wearout failure in on-chip cache SRAM [11]. In [12], a holistic approach (Colt) to equalize the duty cycle ratio and the usage frequency of the devices in modern microprocessor was proposed. Colt employed the complement mode execution, cache set rotation, and operand identifier swapping schemes to mitigate the detrimental effects of aging.

Compared to the data flipping technique proposed for SRAM cells in [9], which requires extra XOR gates to invert the data back to the normal mode during the SRAM cell access, our low power AARF does not need to do the bit flipping during the register access, thus it has no impact in cycle time. Colt in [12] uses the flipped register content without the need to flip them back. However, the complement mode applied to the whole data path, control path, and storage hierarchy (including register files) is too complicated for management. Moreover, extra XOR gates are still needed in Colt to do the bit complementing. In our AARF, no extra XOR gates are needed for bit-flipping/complementing since the bit-flipping/complementing operation in our AARF is just writing ones or zeros to the leading bits according to their current status. Penelope [10] relies on the idle time of the processor resources, such as pipelines, cache blocks, registers, etc., to balance the duty cycle ratios of the devices. The power consumption will increase due to their value sampling and the updates to the RINV registers. Our AARF does not relies on the availability of the idle times. Even under the situation that the register file is heavily in use, our AARF can still mitigate the NBTI stress effectively. Moreover, compared to all previous work, our AARF can also reduce the power consumption, thus the temperature of the register file. Therefore, it will further mitigate the aging effect.

III. Low Power Aging-Aware Register File (AARF)

A. Narrow-Width Values

The narrow-width values in high-performance microprocessors have been well studied and exploited for performance and power optimizations [13][14]. In a 64-bit microprocessor, values that can be represented by less than 64 bits are generally referred to as narrow-width values. In our simulated microprocessor, the experimental results show that on average 96% of the produced integer register values can be represented by no more than 34 bits and most of them have 30 bits leading zeros.

The presence of narrow-width values is a significant contributor to the NBTI stress of integer register files. For example, in a 64-bit register file, the higher (leading) 30 bits of the register entry stay ‘0’ most of the time, which will accelerate the aging effect. Our experimental results shows that on average the leading 30 bits of the register entry will stay logic ‘0’ in 97.5% of the cycle time during the execution, which produces an extremely heavy NBTI stress on the device.

B. Low Power Value-Aware Register File (VARF)

In [15], a thermal-aware register file (TARF) was proposed by exploiting the narrow-width values. For the low power design, we propose a simplified value-aware register file (VARF) with low hardware overheads. Our value-aware register file (VARF) can significantly reduce the power consumption of the register file by avoiding/disabling accesses to the leading zero bits. For register read, the original 64-bit value can be restored by using the exiting sign extension logic at the inputs of the ALUs. Instead of controlling/activating the bitlines according to the bit width of the narrow-width values, we divides the integer register values into two categories: 34-bit narrow-width values and 64-bit regular values.

For the narrow-width detection, we utilize the existing leading-0/1 detection logic within the functional units to overlap the timing overhead in deeply pipelined designs. Figure 1 shows the schematic diagram of the proposed low power VARF. In the our low power VARF, the register file is partitioned into two halves: a lower 34-bit half and an upper 30-bit half. One narrow-width flag bit is added to each register entry for bit control. In most cases, the narrow-width values are stored in the lower 34 bits and the upper 30 bits are gated for power saving. For instance, during the register file read, after precharging the bitlines, the wordline of the upper half is gated by the narrow-width flag bit, which means that the power consumption is reduced by only accessing the lower half of the entire register file for narrow-width values and the upper half is rarely accessed. The multiplex is placed in Execute stage to minimize the performance overhead. Note that compared to the TARF proposed in [15], our low power VARF has a much simpler design, which does not need the support of the value swapping/interleaving between two halves. The narrow-width values are always stored in the lower half of our VARF. The TARF needs more complicated control logics to maintain their values. In addition, the upper half of our VARF is only 30-bit compared to the 34-bit in TARF. Therefore, the space overhead of our VARF is 1 bit (narrow-width flag bit) out of the 64-bit register entry (1/64 = 1.6%), while TARF needs 6 additional bits (6/64 = 9.4%).

C. Duty Cycle Balancing in VARF

For the original register file design, as we discussed above, the leading 30 bits of the register entries are all zeros most of the time due to the dominant narrow-width values. The unbalanced duty cycles for these bits will significantly increase the NBTI degradation. In our low power VARF design, the leading 30 bits (upper half) of the narrow-width values are gated during the register read, which means that these upper bits are not used and can be treated as ‘idle’. To balance the duty cycles of these ‘idle’ bits, we propose an aging-aware register file (AARF) design based on the low power VARF. In our AARF, we propose to periodically flip/complement these upper idle bits at a predefined time interval. For example, at

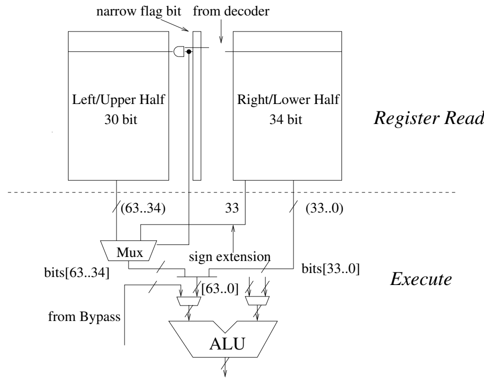

Fig. 1. The schematic diagram of the low power VARF.

the very beginning, these upper 30 bits are all zeros. After a certain time interval, we flip them to all ones. Then after another time interval, we do the complementing again to bring them back to all zeros. Therefore, the duty cycle ratio of these upper idle bits can be perfectly balanced to ∼50%. Note that the bit-flipping/complementing in our AARF is just writing all zeros or ones to the upper 30 bits. Therefore, no extra XOR gates are needs to do the flipping, which means that our AARF design has much lower overheads compared to previous schemes in [10][12].

Since the upper idle bits are not accessed during the register read, the bit-flipping/complementing operation is not in the critical path and has no impact on the performance. However, in order to reduce the power overhead due to the bit-flipping/complementing operation, we can choose a large time interval. The narrow-width flag bit is utilized to control whether the bit flipping/complementing should be performed to upper half or not. Therefore, no additional hardware overheads are needed for each register entry. Overall, our AARF design not only can significantly reduce the NBTI stress to register file (upper half) by effectively balancing its duty cycles, but also can achieve the power saving of the register accesses compared to the original register file design. In addition, the reduced power density in the register file will also result in the reduction in device temperature, which will further mitigate the negative aging effect.

IV. Experimental Evaluation

A. Experimental Setup

We derive our simulators from SimpleScalar V3.0 [16] to model a microprocessor similar to Alpha 21364. Table I gives the detailed configuration of the simulated microprocessor. To evaluate the power efficiency of our AARF design, a modified version of the Wattch power model [17] is used for power profiling (at 32nm technology) during the simulation.

For experimental evaluation, we use the SPEC CPU2000 benchmark suite compiled for the Alpha Instruction Set Architecture (ISA). We use the reference input sets for this study. Each benchmark is first fast-forwarded to its early single simulation point specified by SimPoint [18]. We use the last 100 million instructions during the fast-forwarding phase to warm-up if the number of skipped instructions is more than 100 million. Then, we simulate the next 100 million instructions in detail.

TABLE I
Parameters of the simulated processor.

Processor Core	
Datapath Width	4 inst. per cycle
Int Issue Queue	20 entries
FP Issue Queue	15 entries
Load/Store Queue	64 entries
Active list (ACL)	80 entries
Int Register File	80 registers
FP Register File	72 registers
Function Units	4 IALU, 2 IMULT/IDIV 2 FALU, 1 FMULT/FDIV/FSQRT 2 MemPorts
Branch Predictor	
Branch Predictor	Alpha 21264 tournament predictor 32-entry RAS
BTB	2048-entry 2-way
Memory Hierarchy	
L1 I/DCache	64KB, 2 ways, 64B blocks, 2 cycles
L2 UCache	4MB, 8 ways, 128B blocks, 12 cycles
Memory	225 cycles first chunk, 12 cycles rest
TLB	Fully-assoc., 128 entries

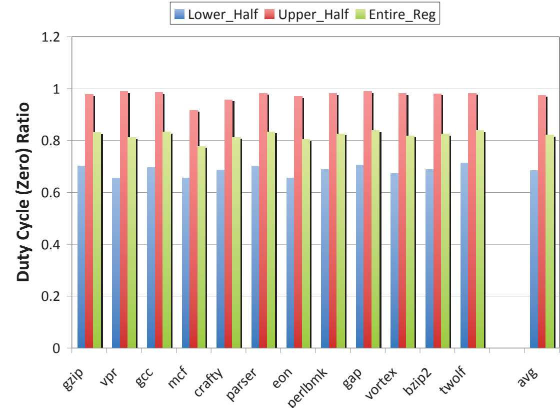

Fig. 2. The average stress duty cycle (zero) ratio for original integer register files.

B. Experimental Results and Analysis

To study the NBTI stress of the original register file design, especially the upper 30 bits for the narrow-width values, we profile the stress duty cycle ratio for the original register file by dividing it into two halves: the lower 34-bit half and the upper 30-bit half. According to our profiling results, around 96% of the integer register values can be presented by no more than 34 bits. Therefore, the NBTI stress of the leading (upper) 30 bits should be very high. Figure 2 shows that the lower 34-bit half (`Lower_Half`) has a stress duty cycle ratio of 68.5%, while the upper 30-bit half has much higher (`Upper_Half`) stress duty cycle ratio of 97.5%. If we consider the NBTI stress for the entire register file (Entire_Reg), the average stress duty cycle ratio is 82.1%. The results confirm us that we need aging-aware design to reduce the NBTI stress of the register file, especially for the upper 30 bits.

To implement our low power AARF, first we need to decide time interval for bit-flipping/complementing. If we use a small interval, the power overhead will increase, but the duty cycle will be more perfectly balanced. If a large interval is adopted, the power consumption will be reduced, but the effectiveness of the duty cycle balancing will be hurt. Based on our experimental results, a 40K-cycle bit-flipping/complementing

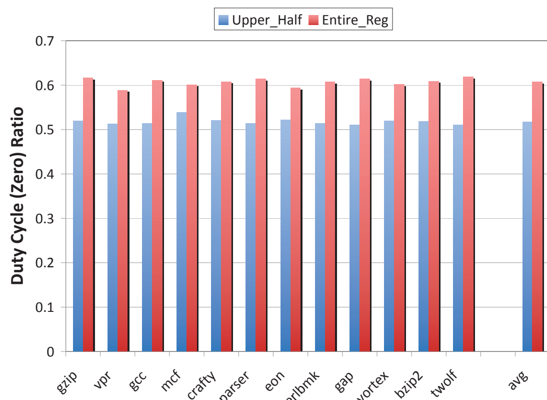

Fig. 3. The average stress duty cycle (zero) ratio for the low power AARF.

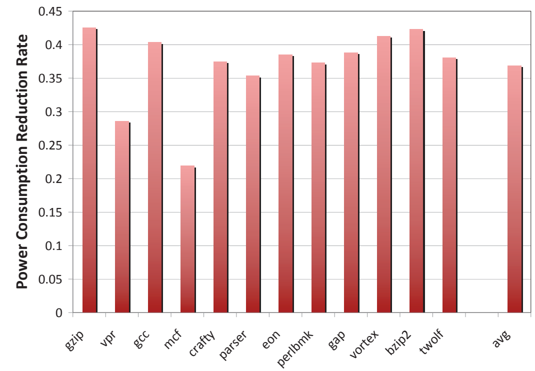

Fig. 4. The power consumption reduction rate for the low power AARF

interval has negligible power and performance overheads with nearly perfect duty cycle balancing capability. Therefore, we choose the 40K-cycle bit-flipping/complementing interval for our low power AARF design.

Figure 3 shows the average stress duty cycle ratio for the low power AARF design. For the upper 30-bit half (`Upper_Half`) in the AARF, the stress duty cycle ratio is reduced to 51.8%, which is very close to the ideal stress duty cycle ratio of 50%. If we consider the entire register file, the average stress duty cycle ratio is also reduced to 60.7%. Previous study has shown that the gate-oxide failure probability is proportional to the device stress time [4]. Therefore, we can expect a similar MTTF (mean time to failure) improvement for the register file. Note that our AARF design has no impact on the stress duty cycle ratio of the lower 34-bit half, which remains 68.5%. For the future work, we will target at reducing the stress duty cycle of the entire register file to $\sim$50%, i.e., mainly targeting at the lower 34 bits.

Compared to other aging-ware designs, our AARF can also reduce the power consumption of the register file significantly. As we discussed in Section III, the power consumption of the register file will be reduced because only lower half of the entire register file is accessed for narrow-width values. The bit flipping/complementing operation has negligible power overhead due to the large (40K-cycle) time interval. Figure 4 shows that our AARF design can achieve a 36.9% power reduction for the integer register files. These power reduction in AARF can result in on average 5-degree temperature reduction in the register file, which can further mitigate the aging effect.

V. Conclusion and Future Work

The performance and reliability degradation due to the aging effect are becoming substantial for CMOS devices in future technologies. Because of the presence of the narrow-width values in the high-performance microprocessors, the integer register file suffers an extremely high NBTI stress, which will accelerate its lifetime degradation. In this paper, we propose an aging-aware register file (AARF) design to combat the aging effect in the integer register file. By periodically bit-flipping/complementing the leading bits of the narrow-width values in registers, the duty cycle ratio can be well balanced and the device stress will be significantly reduced. In addition, our AARF design can also achieve a 36.9% power reduction in the register file and reduce the average temperature by 5 degrees, which will further mitigate the aging effect in the register file. In future work, we will continue working on the duty cycle balancing for the lower 34 bits of the register file and reducing duty cycle ratio for the entire register file to $\sim$50%. We will also examine the narrow-with values in caches and their impact on aging.

Acknowledgment

This work was supported in part by a grant from Chinese NSF Award 61100035.

References

[1] S. Borkar, "Designing reliable systems from unreliable components: the challenges of transistor variability and degradation," *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Nov. 2005.
[2] W. Wang *et al.*, "The impact of nbti on the performance of combinational and sequential circuits," in *Proceedings of the Design Automation Conf.*, 2007.
[3] E. Rosenbaum *et al.*, "Effect of hot-carrier injection on n- and pmosfet gate oxide integrity," *IEEE Electron Device Letters*, vol. 12, no. 11, Nov 1991.
[4] E. Minami *et al.*, "Circuit-level simulation of tddb failure in digital cmos circuit," *IEEE Trans. on Semiconductor Manufacturing*, vol. 8, no. 3, Aug 1995.
[5] S. Borkar, "Electronics beyond nano-scale cmos," in *Proceedings of the Design Automation Conf.*, 2006.
[6] D. Schroder and J. Babcock, "Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing," *Journal of Applied Physics*, vol. 9, no. 1, July 2003.
[7] P. Hulbert, "Throughput gate dielectric reliability testing: Digging out from the backlog," *Keithley Instruments Inc. technical note*, 2004.
[8] S. Mahapatra *et al.*, "Investigation and modeling of interface and bulk trap generation during negative bias temperature instability of p-mosfets," *IEEE Trans. on Electron Devices*, vol. 51, no. 9, Sept 2005.
[9] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar, "Impact of nbti on sram read stability and design for reliability," in *Proceedings of Int. Sym. on Quality Electronic Design*, 2006.
[10] J. Abella, X. Vera, and A. Gonzalez, "Penelope: The nbti-aware processor," in *Proceedings of IEEE/ACM International Symposium on Microarchitecture*, 2007, pp. 85–96.
[11] J. Shin *et al.*, "A proactivewearout recovery approach for exploiting microarchitectural redundancy to extend cache sram lifetime," in *Proceedings of International Symposium on Computer Architecture*, 2008, pp. 353–362.
[12] E. Gunadi, A. Sinkar, N. Kim, and M. Lipasti, "Combating aging with the colt duty cycle equalizer," in *Proceedings of the IEEE/ACM Int. Symp. on Microarchitecture*, 2010, pp. 103–114.
[13] D. Brooks and M. Martonosi, "Dynamically exploiting narrow width operands to improve processor power and performance," in *Proc. of HPCA-5*, 1999.
[14] A. Aggarwal and M. Franklin, "Energy efficient asymmetrically ported register files," in *Proc. of ICCD'03*, 2003, pp. 2–7.
[15] S. Wang *et al.*, "Exploiting narrow-width values for thermal-aware register file designs," in *Proceedings of the Conference on Design, Automation and Test in Europe*, April 2009, pp. 1422–1427.
[16] D. Burger and T. M. Austin, "The simplescalar tool set, version 2.0," Computer Sciences Department, University of Wisconsin, Tech. Rep. 1342, 1997.
[17] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *Proc. International Symposium on Computer Architecture*, 2000.
[18] T. Sherwood *et al.*, "Automatically characterizing large scale program behavior," in *Proc. of ASPLOS X*, October 2002.