https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

Age-Acknowledging Reliable Multiplier Design with Adaptive Hold Logic

Dontabhaktuni Jayakumar[1], Ch Ananda Kumar[2], B. Rambhupal Reddy[3]

[1,2]Assistant professor Department of ECE, Holy Mary Institute of Technology and Science, Hyderabad, India
[3]Assistant Professor, AVN Institute of Engineering and Technology, Hyderabad, India

Abstract *— Digital multipliers are among the most critical arithmetic functional units. The overall performance of these systems depends on the throughput of the multiplier. Meanwhile, the negative bias temperature instability effect occurs when a pMOS transistor is under negative bias (Vgs = −Vdd), increasing the threshold voltage of the pMOS transistor, and reducing multiplier speed. A similar phenomenon, positive bias temperature instability, occurs when an nMOS transistor is under positive bias. Both effects degrade transistor speed, and in the long term, the system may fail due to timing violations. Therefore, it is important to design reliable high performance multipliers. In this paper, we propose an aging-aware multiplier design with novel adaptive hold logic (AHL) circuit. The multiplier is able to provide higher throughput through the variable latency and can adjust the AHL circuit to mitigate performance degradation that is due to the aging effect. Moreover, the proposed architecture can be applied to a column- or row-bypassing multiplier. The experimental results show that our proposed architecture with 16 ×16 and 32 ×32 column-bypassing multipliers can attain up to 62.88% and 76.28% performance improvement, respectively, compared with 16×16 and 32×32 fixed-latency column-bypassing multipliers. Furthermore, our proposed architecture with 16 × 16 and 32 × 32 row-bypassing multipliers can achieve up to 80.17% and 69.40% performance improvement as compared with 16×16 and 32 × 32 fixed-latency row-bypassing multipliers.*

Keywords— AHL, MOS, multipliers.

I. INTRODUCTION

Digital multipliers are among the most critical arithmetic functional units in many applications, such as the Fourier transform, discrete cosine transforms, and digital filtering. The throughput of these applications depends on multipliers, and if the multipliers are too slow, the performance of entire circuits will be reduced.

The corresponding effect on an nMOS transistor is positive bias temperature instability (PBTI), which occurs when an nMOS transistor is under positive bias. Compared with the NBTI effect, the PBTI effect is much smaller on oxide/polygate transistors, and therefore is usually ignored.

However, for high-k/metal-gate nMOS transistors with significant charge trapping, the PBTI effect can no longer be ignored. In fact, it has been shown that the PBTI effect is more significant than the NBTI effect on 32-nm high-k/metal-gate processes

A traditional method to mitigate the aging effect is overdesign [5], [6], including such things as guard-banding and gate over sizing. However, this approach can be very pessimistic and area and power inefficient. To avoid this problem, many NBTI-aware methodologies have been proposed. An NBTI-aware technology mapping technique was proposed in [7] to guarantee the performance of the circuit during its lifetime. In [8], an NBTI-aware sleep transistor was designed to reduce the aging effects on pMOS sleep-transistors, and the lifetime stability of the power-gated circuits under consideration was improved. Wu and Marculescu [9] proposed a joint logic restructuring and pin reordering method, which is based on detecting functional symmetries and transistor stacking effects. They also proposed an NBTI optimization method that considered path sensitization [12]. In [10] and [11], dynamic voltage scaling and body-basing techniques were proposed to reduce power or extend circuit life. These techniques, however, require circuit modification or do not provide optimization of specific circuits.

Traditional circuits use critical path delay as the overall circuit clock cycle in order to perform correctly. However, the probability that the critical paths are activated is low. In most cases, the path delay is shorter than the critical path. For these noncritical paths, using the critical path delay as the overall cycle period will result in significant timing waste. Hence, the variable-latency design was proposed to reduce the timing waste of traditional circuits. The variable-latency design divides the circuit into two parts: 1) shorter paths and 2) longer paths. Shorter paths can execute correctly in one cycle, whereas longer paths need two cycles

https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

to execute. When shorter paths are activated frequently, the average latency of variable-latency designs is better than that of traditional designs. For example, several variable-latency adders were proposed using the speculation technique with error detection and recovery [13]–[15]. A short path activation function algorithm was proposed in [16] to improve the accuracy of the hold logic and to optimize the performance of the variable-latency circuit. An instruction scheduling algorithm was proposed in [17] to schedule the operations on non-uniform latency functional units and improve the performance of Very Long Instruction Word processors. In [18], a variable latency pipelined multiplier architecture with a Booth algorithm was proposed. In [19], process-variation tolerant architecture for arithmetic units was proposed, where the effect of process-variation is considered to increase the circuit yield. In addition, the critical paths are divided into two shorter paths that could be unequal and the clock cycle is set to the delay of the longer one. These research designs were able to reduce the timing waste of traditional circuits to improve performance, but they did not consider the aging effect and could not adjust themselves during the runtime. A variable-latency adder design that considers the aging effect was proposed in [20] and [21]. However, no variable-latency multiplier design that considers the aging effect and can adjust dynamically has been done.

II. PAPER CONTRIBUTION

In this paper, we propose an aging-aware reliable multiplier design with a novel adaptive hold logic (AHL) circuit. The multiplier is based on the variable-latency technique and can adjust the AHL circuit to achieve reliable operation under the influence of NBTI and PBTI effects. To be specific, the contributions of this paper are summarized as follows: 1) novel variable-latency multiplier architecture with an AHL circuit. The AHL circuit can decide whether the input patterns require one or two cycles and can adjust the judging criteria to ensure that there is minimum performance degradation after considerable aging occurs. 2) comprehensive analysis and comparison of the multiplier's performance under different cycle periods to show the effectiveness of our proposed architecture; 3) an aging-aware reliable multiplier design method that is suitable for large multipliers. Although the experiment is performed in 16- and 32-bit multipliers, our proposed architecture can be easily extended to large designs; 4) the experimental results show that our proposed architecture with the 16×16 and 32×32 column-bypassing multipliers can attain up to 62.88% and 76.28% performance improvement compared with the 16 × 16 and 32 × 32 fixed-latency column-bypassing (FLCB) multipliers. In addition, our proposed architecture with 16 × 16 and 32 × 32 row-bypassing multipliers can achieve up to 80.17% and 69.40% performance improvement as compared with 16×16 and 32×32 fixed-latency row bypassing multipliers. The paper is organized as follows. Section II introduces the background of the column-bypassing multiplier, row-bypassing multiplier, variable-latency design, and NBTI/PBTI models. Section III details the aging-aware variable-latency multiplier based on the column- or row by passing multiplier. The experimental setup and results are presented in Section IV. Section V concludes this paper.

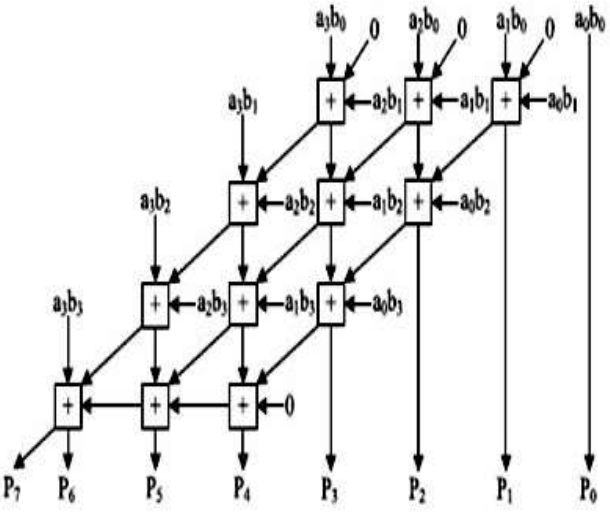


Fig. 1. 4 x 4 normal AM.

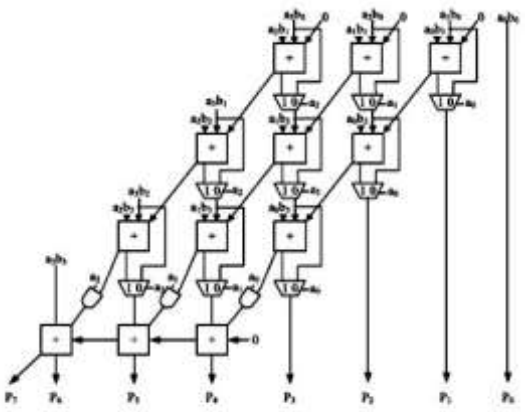


Fig. 2. 4 x 4 column-bypassing multiplier.

III. PRELIMINARIES

A column-bypassing multiplier is an improvement on the normal array multiplier (AM). The AM is a fast parallel AM and is shown in Fig. 1. The multiplier array consists of (n−1) rows of carry save adder (CSA), in which each row contains

https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

$(n-1)$ full adder (FA) cells. Each FA in the CSA array has two outputs: 1) the sum bit goes down and 2) the carry bit goes to the lower left FA. The last row is a ripple adder for carry propagation. The FAs in the AM are always active regardless of input states. In [22], a low-power column-bypassing multiplier design is proposed in which the FA operations are disabled if the corresponding bit in the multiplicand is 0. Fig. 2 shows a 4×4 column-bypassing multiplier. Supposing the inputs are 10102 * 11112, it can be seen that for the FAs in the first and third diagonals, two of the three input bits are 0: the carry bit from its upper right FA and the partial product *aibi* . Therefore, the output of the adders in both diagonals is 0, and the output sum bit is simply equal to the third bit, which is the sum output of its upper FA. Hence, the FA is modified to add two tristate gates and one multiplexer. The multiplicand bit *ai* can be used as the selector of the multiplexer to decide the output of the FA, and *ai* can also be used as the selector of the tristate gate to turn off the input path of the FA. If *ai* is 0, the inputs of FA are disabled, and the sum bit of the current FA is equal to the sum bit from its upper FA, thus reducing the power consumption of the multiplier. If *ai* is 1, the normal sum result is selected. More details for the column-bypassing multiplier can be found in [22].

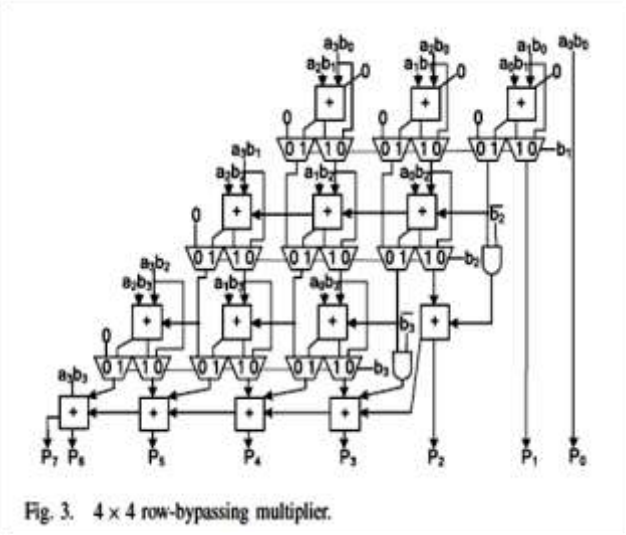


Fig. 3. 4 × 4 row-bypassing multiplier.

A low-power row-bypassing multiplier [23] is also proposed to reduce the activity power of the AM. The operation of the low-power row-bypassing multiplier is similar to that of the low-power column-bypassing multiplier, but the selector of the multiplexers and the tristate gates use the multiplicator. Fig. 3 is a 4 × 4 row-bypassing multiplier. Each input is connected to an FA through a tristate gate. When the inputs are 11112 * 10012, the two inputs in the first and second rows are 0 for FAs. Because *b*1 is 0, the multiplexers in the first row select *aib*0 as the sum bit and select 0 as the carry bit. The inputs are bypassed to FAs in the second rows, and the tristate gates turn off the input paths to the FAs. Therefore, no switching activities occur in the first-row FAs; in return, power consumption is reduced. Similarly, because *b*2 is 0, no switching activities will occur in the second-row FAs. However, the FAs must be active in the third row because the *b*3 is not zero. More details for the row-bypassing multiplier can also be found in [23].

IV. PROPOSED AGING-AWARE MULTIPLIER

This section details the proposed aging-aware reliable multiplier design. It introduces the overall architecture and the functions of each component and also describes how to design AHL that adjusts the circuit when significant aging occurs. *A. Proposed Architecture* Fig. 8 shows our proposed aging-aware multiplier architecture, which includes two *m*-bit inputs (*m* is a positive number), one 2*m*-bit output, one column- or row-bypassing multiplier, 2*m* 1-bit Razor flip-flops [27], and an AHL circuit. In the proposed architecture, the column- and row-bypassing multipliers can be examined by the number of zeros in either the multiplicand or multiplicator to predict whether the operation requires one cycle or two cycles to complete. When input patterns are random, the number of zeros and ones in the multiplicator and multiplicand follows a normal distribution, as shown in Figs. 9 and 10. Therefore, using the number of zeros or ones as the judging criteria results in similar outcomes.

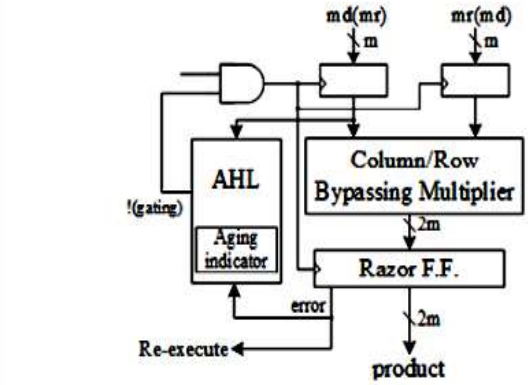


Fig. 8. Proposed architecture (md means multiplicand; mr means multiplicator).

Hence, the two aging-aware multipliers can be implemented using similar architecture, and the difference between the two bypassing multipliers lies in the input signals of the AHL. According to the bypassing selection in the columnor row-bypassing multiplier, the input signal of the AHL in the architecture with the column-bypassing multiplier is the multiplicand, whereas that of the row-bypassing multiplier is the multiplicator. Razor flip-flops can be used to detect

https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

whether timing violations occur before the next input pattern arrives. Fig. 11 shows the details of Razor flip-flops.

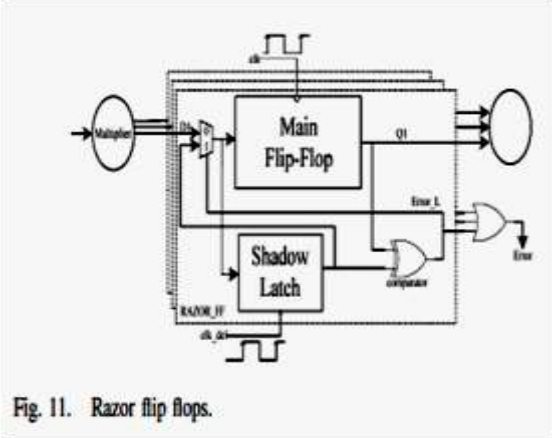


Fig. 11. Razor flip flops.

A 1-bit Razor flip-flop contains a main flip-flop, shadow latch, XOR gate, and mux. The main flip-flop catches the execution result for the combination circuit using a normal clock signal, and the shadow latch catches the execution result using a delayed clock signal, which is slower than the normal clock signal. If the latched bit of the shadow latch is different from that of the main flip-flop, this means the path delay of the current operation exceeds the cycle period, and the main flip-flop catches an incorrect result. If errors occur, the Razor flip-flop will set the error signal to 1 to notify the system to re-execute the operation and notify the AHL circuit that an error has occurred. We use Razor flip-flops to detect whether an operation that is considered to be a one-cycle pattern can really finish in a cycle. If not, the operation is re-executed with two cycles. Although the re-execution may seem costly, the overall cost is low because the re-execution frequency is low. More details for the Razor flip-flop can be found in [27].

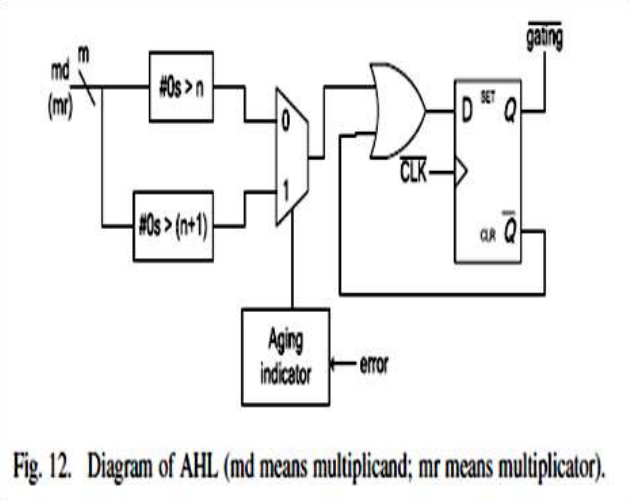
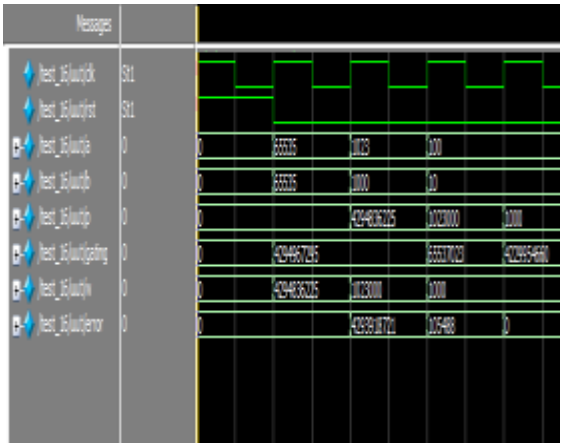


Fig. 12. Diagram of AHL (md means multiplicand; mr means multiplicator).

The AHL circuit is the key component in the aging-ware variable-latency multiplier. Fig. 12 shows the details of the AHL circuit. The AHL circuit contains an aging indicator, two judging blocks, one mux, and one D flip-flop. The aging indicator indicates whether the circuit has suffered significant performance degradation due to the aging effect. The aging indicator is implemented in a simple counter that counts the number of errors over a certain amount of operations and is reset to zero at the end of those operations. If the cycle period is too short, the column- or row-bypassing multiplier is not able to complete these operations successfully, causing timing violations. These timing violations will be caught by the Razor flip-flops, which generate error signals. If errors happen frequently and exceed a predefined threshold, it means the circuit has suffered significant timing degradation due to
the aging effect, and the aging indicator will output signal 1. Otherwise, it will output 0 to indicate the aging effect is still not significant, and no actions are needed. The first judging block in the AHL circuit will output 1 if the number of zeros in the multiplicand (multiplicator for the row-bypassing multiplier) is larger than n (n is a positive number, which will be discussed in Section IV), and the second judging block in the AHL circuit will output 1 if the number of zeros in the multiplicand (multiplicator) is larger than $n + 1$. They are both employed to decide whether an input pattern requires one or two cycles, but only one of them will be chosen at a time. In the beginning, the aging effect is not significant, and the aging indicator produces 0, so the first judging block is used. After a period of time when the aging effect becomes significant, the second judging block is chosen. Compared with the first judging block, the second judging block allows a smaller number of patterns to become one-cycle patterns because it requires more zeros in the multiplicand (multiplicator). The details of the operation of the AHL circuit are as follows: when an input pattern arrives, both judging blocks will decide whether the pattern requires one cycle or two cycles to complete and pass both results to the multiplexer. The multiplexer selects one of either result based on the output of the aging indicator. Then an OR operation is performed between the result of the multiplexer, and the .Q signal is used to determine the input of the D flip-flop. When the pattern requires one cycle, the output of the multiplexer is 1. The !(gating) signal will become 1, and the input flip flops will latch new data in the next cycle. On the other hand, when the output of the multiplexer is 0, which means the input pattern requires two cycles to complete, the OR gate will output 0 to the D flip-flop. Therefore, the !(gating) signal will be 0 to disable the clock signal of the input flip-flops in the next cycle. Note that only a cycle of the input flip-flop will be disabled because the D flip-flop will latch 1 in the next cycle. The

https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

overall flow of our proposed architecture is as follows: when input patterns arrive, the column- or row-bypassing multiplier, and the AHL circuit execute simultaneously. According to the number of zeros in the multiplicand (multiplicator), the AHL circuit decides if the input patterns require one or two cycles. If the input pattern requires two cycles to complete, the AHL will output 0 to disable the clock signal of the flip-flops. Otherwise, the AHL will output 1 for normal operations. When the column- or row-bypassing multiplier finishes the operation, the result will be passed to the Razor flip-flops. The Razor flip-flops check whether there is the path delay timing violation. If timing violations occur, it means the cycle period is not long enough for the current operation to complete and that the execution result of the multiplier is incorrect. Thus, the Razor flip-flops will output an error to inform the system that the current operation needs to be re-executed using two cycles to ensure the operation is correct. In this situation, the extra re-execution cycles caused by timing violation incurs a penalty to overall average latency. However, our proposed AHL circuit can accurately predict whether the input patterns require one or two cycles in most cases. Only a few input patterns may cause a timing variation when the AHL circuit judges incorrectly. In this case, the extra re-execution cycles did not produce significant timing degradation. In summary, our proposed multiplier design has three key features. First, it is a variable-latency design that minimizes the timing waste of the noncritical paths. Second, it can provide reliable operations even after the aging effect occurs. The Razor flip-flops detect the timing violations and re-execute the operations using two cycles. Finally, our architecture can adjust the percentage of one-cycle patterns to minimize performance degradation due to the aging effect. When the circuit is aged, and many errors occur, the AHL circuit uses the second judging block to decide if an input is one cycle or two cycles. The **simulation result** for the proposed model is shown in the figure:



V. CONCLUSION

This paper proposed an aging-aware variable-latency multiplier design with the AHL. The multiplier is able to adjust the AHL to mitigate performance degradation due to increased delay. The experimental results show that our proposed architecture with 16×16 and 32×32 column-bypassing multipliers can attain up to 62.88% and 76.28% performance improvement compared with the 16 × 16 and 32 × 32 FLCB multipliers, respectively. Furthermore, our proposed architecture with the 16×16 and 32×32 row-bypassing multipliers can achieve up to 80.17% and 69.40% performance improvement compared with the 16 × 16 and 32 × 32 FLRB multipliers. In addition, the variable-latency bypassing multipliers exhibited the lowest average EDP and achieved up to 10.45% EDP reduction in 32 × 32 VLCB multipliers. Note that in addition to the BTI effect that increases transistor delay, interconnect also has its aging issue, which is called electro migration. Electro migration occurs when the current density is high enough to cause the drift of metal ions along the direction of electron flow. The metal atoms will be gradually displaced after a period of time, and the geometry of the wires will change. If a wire becomes narrower, the resistance and delay of the wire will be increased, and in the end, electro migration may lead to open circuits. This issue is also more serious in advanced process technology because metal wires are narrower, and changes in the wire width will cause larger resistance differences. If the aging effects caused by the BTI effect and electro migration are considered together, the delay and performance degradation will be more significant. Fortunately, our proposed variable latency multipliers can be used under the influence of both the BTI effect and electromigration. In addition, our proposed variable latency multipliers have less performance degradation because variable latency multipliers have less timing waste, but traditional multipliers need to consider the degradation caused by both the BTI effect and electromigration and use the worst case delay as the cycle period.

REFERENCES

[1] Y. Cao. (2013). *Predictive Technology Model (PTM) and NBTI Model* [Online]. Available: http://www.eas.asu.edu/~ptm

[2] S. Zafar *et al.*, "A comparative study of NBTI and PBTI (charge trapping) in SiO2/HfO2 stacks with FUSI, TiN, Re gates," in *Proc. IEEE Symp. VLSI Technol. Dig. Tech. Papers*, 2006, pp. 23–25.

[3] S. Zafar, A. Kumar, E. Gusev, and E. Cartier, "Threshold voltage instabilities in high-k gate

https://dx.doi.org/10.24001/ijaems.3.2.2 *ISSN : 2454-1311*

dielectric stacks," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 1, pp. 45–64, Mar. 2005.

[4] H.-I. Yang, S.-C. Yang, W. Hwang, and C.-T. Chuang, "Impacts of NBTI/PBTI on timing control circuits and degradation tolerant design in nanoscale CMOS SRAM," *IEEE Trans. Circuit Syst.*, vol. 58, no. 6, pp. 1239–1251, Jun. 2011.

[5] R. Vattikonda, W. Wang, and Y. Cao, "Modeling and miimization of pMOS NBTI effect for robust naometer design," in *Proc. ACM/IEEE DAC*, Jun. 2004, pp. 1047–1052. [6] H. Abrishami, S. Hatami, B. Amelifard, and M. Pedram, "NBTI-aware flip-flop characterization and design," in *Proc. 44th ACM GLSVLSI*, 2008, pp. 29–34

[6] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar, "NBTI-aware synthesis of digital circuits," in *Proc. ACM/IEEE DAC*, Jun. 2007, pp. 370–375.

[7] A. Calimera, E. Macii, and M. Poncino, "Design techniques for NBTI tolerant power-gating architecture," *IEEE Trans. Circuits Syst., Exp. Briefs*, vol. 59, no. 4, pp. 249–253, Apr. 2012.

[8] K.-C. Wu and D. Marculescu, "Joint logic restructuring and pin reordering against NBTI-induced performance degradation," in *Proc. DATE*, 2009, pp. 75–80.

[9] Y. Lee and T. Kim, "A fine-grained technique of NBTI-aware voltage scaling and body biasing for standard cell based designs," in *Proc. ASPDAC*, 2011, pp. 603–608.

[10] M. Basoglu, M. Orshansky, and M. Erez, "NBTI-aware DVFS: A new approach to saving energy and increasing processor lifetime," in *Proc. ACM/IEEE ISLPED*, Aug. 2010, pp. 253–258.

[11] K.-C. Wu and D. Marculescu, "Aging-aware timing analysis and optimization considering path sensitization," in *Proc. DATE*, 2011, pp. 1–6.

[12] K. Du, P. Varman, and K. Mohanram, "High performance reliable variable latency carry select addition," in *Proc. DATE*, 2012, pp. 1257–1262.

[13] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in *Proc. DATE*, 2008, pp. 1250–1255.

[14] D. Baneres, J. Cortadella, and M. Kishinevsky, "Variable-latency design by function speculation," in *Proc. DATE*, 2009, pp. 1704–1709.

[15] Y.-S. Su, D.-C. Wang, S.-C. Chang, and M. Marek-Sadowska, "Performance" optimization using variable-latency design style," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 10, pp. 1874–1883, Oct. 2011.

[16] N. V. Mujadiya, "Instruction scheduling on variable latency functional units of VLIW processors," in *Proc. ACM/IEEE ISED*, Dec. 2011, pp. 307–312.

[17] M. Olivieri, "Design of synchronous and asynchronous variable-latency pipelined multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 4, pp. 365–376, Aug. 2001.

[18] D. Mohapatra, G. Karakonstantis, and K. Roy, "Low-power processvariation tolerant arithmetic units using input-based elastic clocking," in *Proc. ACM/IEEE ISLPED*, Aug. 2007, pp. 74–79.

[19] Y. Chen, H. Li, J. Li, and C.-K. Koh, "Variable-latency adder (VL-Adder): New arithmetic circuit design practice to overcome NBTI," in *Proc. ACM/IEEE ISLPED*, Aug. 2007, pp. 195–200.

[20] Y. Chen *et al.*, "Variable-latency adder (VL-Adder) designs for low power and NBTI tolerance," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 11, pp. 1621–1624, Nov. 2010.

[21] M.-C. Wen, S.-J. Wang, and Y.-N. Lin, "Low power parallel multiplier with column bypassing," in *Proc. IEEE ISCAS*, May 2005, pp. 1638–1641.

[22] J. Ohban, V. G. Moshnyaga, and K. Inoue, "Multiplier energy reduction through bypassing of partial products," in *Proc. APCCAS*, 2002, pp. 13–17.

[23] B. C. Paul, K. Kang, H. Kufluoglu, M. A. Alam, and K. Roy, "Impact of NBTI on the temporal performance degradation of digital circuits," *IEEE Electron Device Lett.*, vol. 26, no. 8, pp. 560–562, Aug. 2005.

[24] B. C. Paul, K. Kang, H. Kufluoglu, M. A. Alam, and K. Roy, "Negative bias temperature instability: Estimation and design for improved reliability of nanoscale circuit," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 4, pp. 743–751, Apr. 2007.

[25] R. Vattikonda, W. Wang, and Y. Cao, "Modeling and minimization of pMOS NBTI effect for robust nanometer design," in *Proc. 43rd ACM/IEEE DAC*, Aug. 2006, pp. 1047–1052.

[26] D. Ernst *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th Annu. IEEE/ACM MICRO*, Dec. 2003, pp. 7–18.