Aging-Aware Reliable Super Column Multiplier Design with Adaptive Hold Logic for Finite Impulse Response Filter

Shital Tatyaso Atole
UG Student
Department of Electronics Engineering
SVPM COE Malegaon, Baramati, Pune, India

Abstract

In VLSI, scaling methods are very important in reducing the power dissipation. The two major constraints for delay in any VLSI circuits are latency and throughput. In many digital systems we are using digital multipliers. The overall performance of digital systems depends on the throughput of the multiplier. But there are two effects that degrade the transistors throughput and so it degrades the speed of transistors and after long time, system may fail due to timing violations. These two effects are 1) Negative Bias Temperature Instability (NBTI), 2) Positive Bias Temperature Instability (PBTI) effect. Negative Bias Temperature Instability (NBTI) effect means when a pMOS transistor is under negative bias, increasing the threshold voltage (Vth) of the pMOS transistor, and reducing multiplier speed. A same phenomenon, Positive Bias Temperature Instability (PBTI), occurs when an nMOS transistor is under positive bias. Therefore it is important to design reliable high-performance multipliers. The new approach is an aging-aware super column multiplier design with Adaptive Hold Logic (AHL) circuit. The multiplier is able to provide higher throughput through the variable latency and can adjust the AHL circuit to reduce performance degradation that is due to the aging effect.

Keywords: Adaptive Hold Logic (AHL), Negative Bias Temperature Instability (NBTI), Positive Bias Temperature Instability (PBTI), Reliable Multiplier, Variable Latency

I. Introduction

In most of the digital applications, such as the Fourier transform, discrete cosine transforms, and digital filtering. The throughput of these applications depends on the multipliers, if the multipliers are too slow, the performance of the whole circuit will be reduced so speed of the circuit will be reduced. Also, Negative Bias Temperature Instability (NBTI) occurs when a pMOS transistor is under negative bias (Vgs = −Vdd). NBTI manifests as an increase in the threshold voltage and consequent decrease in drain current. In this situation, first, interface traps are generated. Those traps cannot be recovered over a reasonable time of operation. Some authors refer to them as permanent traps. Those traps are the same as the one created by channel hot carrier. In the case of NBTI, it is believed that the electric field is able to break Si-H bonds located at the silicon-oxide interface during the oxidation process. H is released in the substrate where it migrates interaction between inversion layer holes and hydrogen-passivated Si atoms breaks the Si–H bond generated during the oxidation process, generating H or H2 molecules. When these molecules diffuse away, interface traps are left. The accumulated interface traps between silicon and the gate oxide interface result in increased threshold voltage (Vth), reducing the circuit switching speed. When the biased voltage is removed, the reverse reaction occurs, reducing the NBTI effect. However, the reverse reaction does not eliminate all the interface traps generated during the stress phase, and Vth is increased in the long term. Hence, it is important to design a reliable high-performance multiplier.

The same effect on an nMOS transistor is positive bias temperature instability (PBTI), which occurs when an nMOS transistor is under positive bias. The PBTI effect is much smaller on oxide/polygate transistors compared with the NBTI effect, and therefore is usually ignored [1]-[3]. A traditional method to mitigate the aging effect is overdesign [4], [5], including such things as guard-banding and gate oversizing; however, this approach can be very pessimistic and area and power inefficient. Scaling methods are very important in reducing the power dissipation. The two major constraints for delay in any VLSI circuits are latency and throughput. Scaling means changing of channel width and channel length. With the scaling down of the CM OS technologies, Negative Bias Temperature Instability (NBTI) has become a major concern due to its impact on PMOS transistor aging process and the corresponding reduction in the long-term reliability of CMOS circuits. To avoid this problem, many NBTI-aware methodologies have been proposed.

In [6] an NBTI-aware technology mapping technique was proposed to guarantee the performance of the circuit during its lifetime. 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 in [7]. A set of efficient NBTI-aware circuit design solutions, including both static and dynamic strategies that aim at improving the lifetime stability of power-gated circuits by means of oversizing, body biasing, and stress-probability reduction while minimizing the design overheads.

Wu and Marculescu [8] proposed a joint logic restructuring and pin reordering method, which is based on detecting functional symmetries and transistor stacking effects. In this the proposed methodology involves only wire perturbation and introduces no gate area overhead at all. Experimental results reveal that, by using this approach, on average 56% of performance loss due to NBTI can be recovered. Moreover, our methodology reduces the number of critical transistors remaining under severe NBTI and thus, transistor resizing can be applied to further mitigate NBTI effects with low area overhead. They also proposed an NBTI optimization method that considered path sensitization [11].

In [9] and [10], 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 in one cycle correctly, whereas longer paths require two cycles to execute. When shorter paths are activated frequently, the average latency of variable-latency designs is better than that of traditional designs. A short path activation function algorithm was proposed in [15] 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 [16] to schedule the operations on nonuniform latency functional units and improve the performance of Very Long Instruction Word processors.

In [17], variable-latency pipelined multiplier architecture with a Booth algorithm was proposed. In [18], 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 [19]. However, no variable-latency multiplier design that considers the aging effect and can adjust dynamically has been done.

A. Column-Bypassing Multiplier

A column-bypassing multiplier is an improvement on the normal array multiplier (AM). The AM is a fast parallel AM. The multiplier array consists of (n−1) rows of carry save adder (CSA), in which each row contains (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 [21], 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. 1 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. So 0’s were directly bypassed.

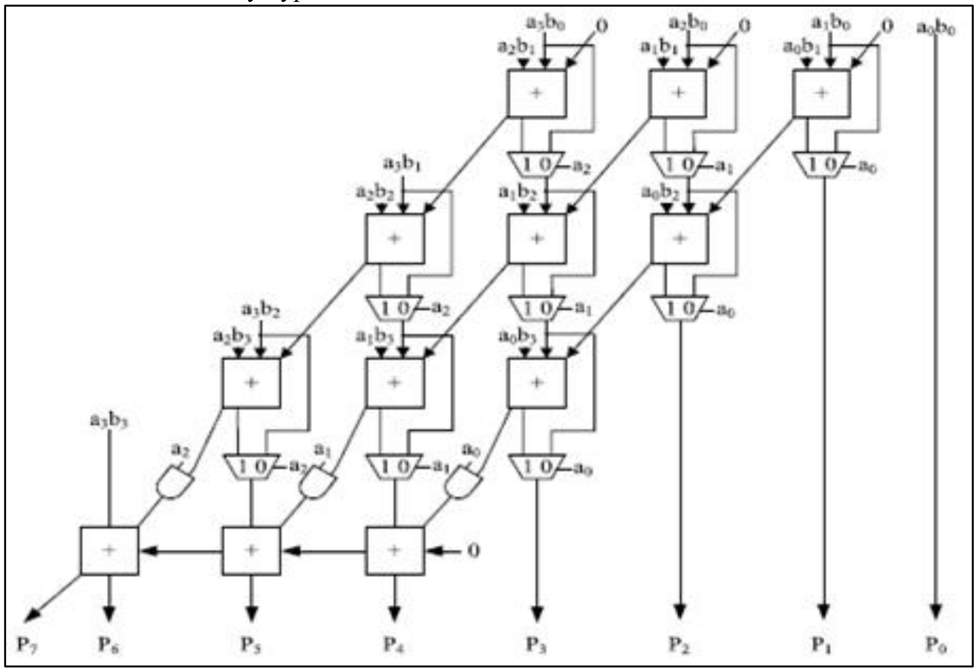


Fig. 1: 4×4 Column-bypassing multiplier [2]

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.

II. 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. It consists of proposed architecture, razor flip flop, and Adaptive Hold Logic (AHL) circuit.

A. Proposed Architecture

Fig. 2 shows our proposed aging-aware multiplier architecture, which includes two m-bit inputs (m is a positive number), one 2m-bit output, one column- or row-bypassing multiplier, 2m 1-bit Razor flip-flops, and an AHL circuit.

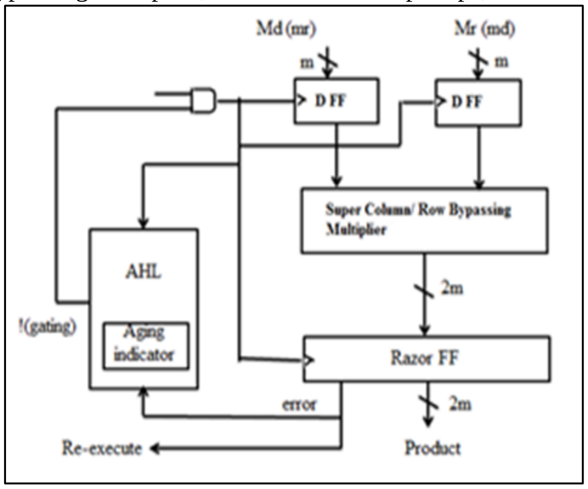


Fig. 2: Proposed architecture (md means multiplicand; mr means multiplicator).

In the proposed architecture, the super column- or 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. Therefore, using the number of zeros or ones as the judging criteria results in similar outcomes. 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 column or 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 whether timing violations occur before the next input pattern arrives. Fig. 3 shows the details of Razor flip-flops. A 1-bit Razor flip-flop contains a main flip-flop we are using D FF, shadow latch, XOR gate, and mux. D FF works on the edge triggered clock signal. The D 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 version clock signal, which is slower than the normal clock signal. Latch is a level triggered 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. If error occurred then re-execute the same input signal 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 reexecuted with two cycles. Although the reexecution may seem costly, the overall cost is low because the reexecution frequency is low.

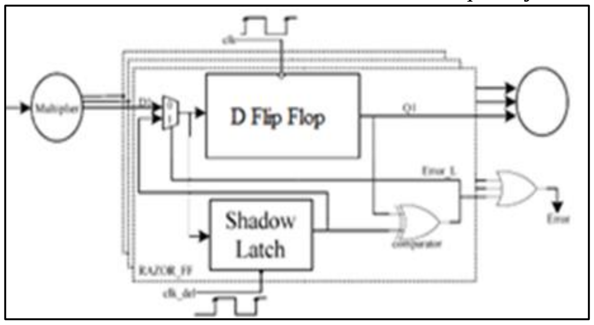


Fig. 3: Razor Flip Flop

The AHL circuit is the key component in the aging-ware variable latency multiplier. Fig. 4 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. Error occurred in razor FF is given to the AHL circuit. 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. This counter output is a select line for the multiplexer. 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 occurred continuously 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 operation of the AHL circuit are as follows: when an input pattern given, both judging blocks will decide whether the input 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 means counter.

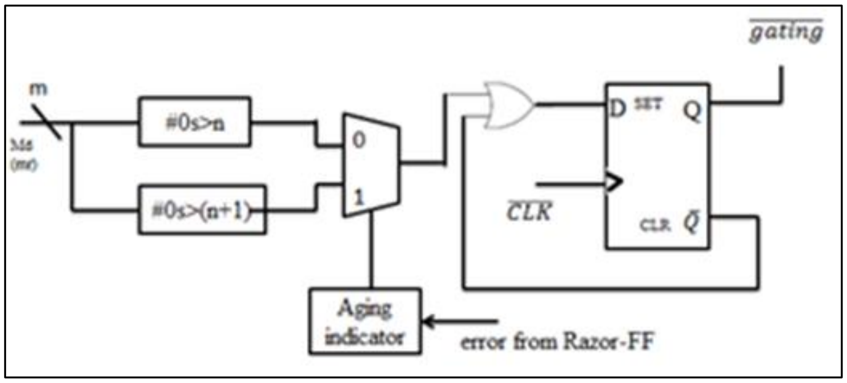


Fig. 4: Adaptive Hold Logic (AHL)

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 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 reexecuted using two cycles to ensure the operation is correct. In this situation, the extra reexecution 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 reexecution 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.

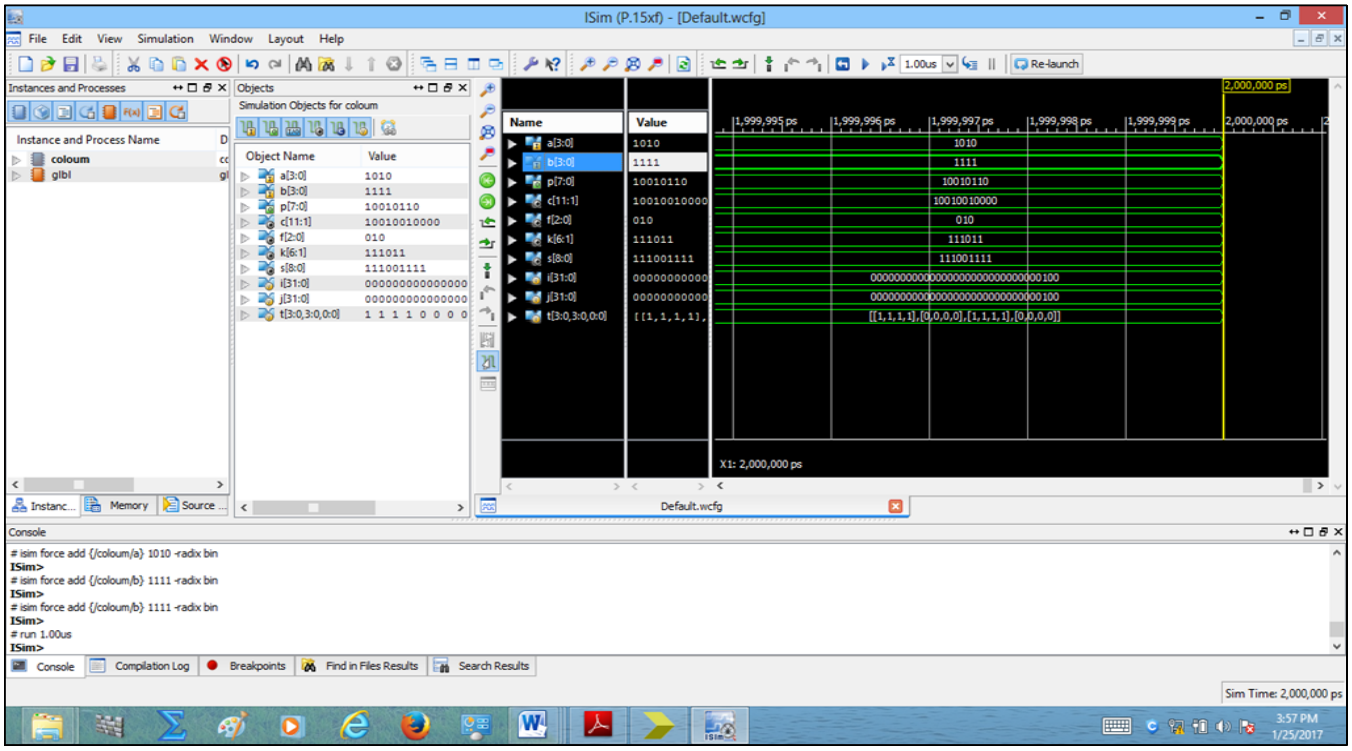
III. APPLICATION: SIGNAL PROCESSING

In this section the application of the proposed multipliers to signal processing is illustrated. The application was Finite Impulse Response (FIR). The FIR convolution is a cross-correlation between the input signal and a time- reversed copy of the impulse-response.

IV. 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. Also it requires less area for the super multiplier.

V. RESULT AND DISCUSSION



```
a[3:0]          1010
b[3:0]          1111
p[7:0]          10010110
c[11:1]         10010010000
f[2:0]          010
k[6:1]          111011
s[8:0]          111001111
i[31:0]         00000000000000000000000000000100
j[31:0]         00000000000000000000000000000100
t[3:0,3:0,0:0]  [[1,1,1,1],[0,0,0,0],[1,1,1,1],[0,0,0,0]]
```

```
# isim force add {/coloum/a} 1010 -radix bin
ISim>
# isim force add {/coloum/b} 1111 -radix bin
ISim>
# isim force add {/coloum/b} 1111 -radix bin
ISim>
# run 1.00us
ISim>
```

Fig. 4: Simulation of 4*4 Column Bypassing Multiplier

ACKNOWLEDGMENT

The author would like to thank anonymous reviewers for their constructive comments and valuable suggestions that helped in the improvement of this paper.

REFERENCES

[1] 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.
[2] S. Zafar, A. Kumar, E. Gusev, and E. Cartier, "Threshold voltage instabilities in high-k gate dielectric stacks," IEEE Trans. Device Mater. Rel., vol. 5, no. 1, pp. 45–64, Mar. 2005.
[3] 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.
[4] 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.
[5] 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 techniqures 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] [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 process variation 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.