AGING-AWARE RELIABLE MULTIPLIER DESIGN WITHADAPTIVE HOLD LOGIC

SIRISHA, RATHOD BHUDEVI
Assistant Professor, M.TECH STUDENT
Dept of ECE,Megha Institute of Engineering & Technology For womens,Edulabad,Ghatkesar mandal,RangaReddy Dist,Telangana ,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 undernegative 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 a 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. In addition we removed the tristate buffer from the coloumn by pass multiplier. So that we can reduced the gate count and improve the efficiency and speed and reduce the power consumption.

Index Terms—Adaptive hold logic (AHL), negative bias temperature instability (NBTI), positive bias temperature instability (PBTI), reliable multiplier, variable latency.

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.

Furthermore, negative bias temperature instability (NBTI) occurs when a pMOS transistor is under negative bias (Vgs= −Vdd). In this situation, the interaction between inversion layer holes and hydrogen-passivated Si atoms break 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 Vthis increased in the long term. Hence, it is important to design a reliable high-performance multiplier. 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 ismore 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 oversizing; 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 to guarantee the performance of the circuit during its lifetime. In], 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 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 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.

variable-latency design divides the circuit into two parts: 1) shorter paths and 2) longer paths. Shorter pathscan execute correctly in one cycle, whereas longer paths need two cycles 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. A short path activation function algorithm was proposed in to improve the accuracy of the hold logic and to optimize the performance of the variable-latency circuit. An instruction scheduling algorithm was proposed into schedule the operations on nonuniform latency functional units and improve the performance of Very Long Instruction Word processors. In, a variable-latency pipelined multiplier architecture with a Booth algorithm was proposed. In, 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 and. However, no variable-latency multiplier design that considers the aging effect and can adjust dynamically has been done

A. 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;

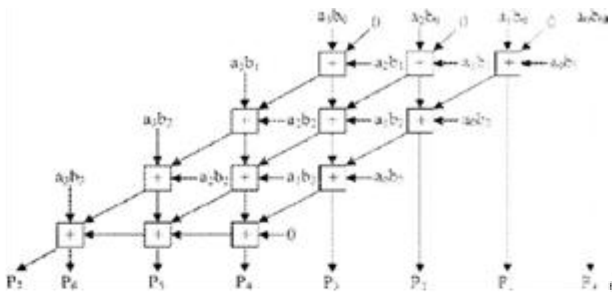


Fig. 1. 4 × 4 normal AM.

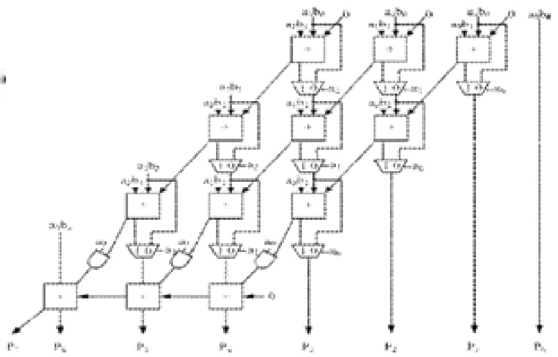


Fig. 2. 4 × 4 column-bypassing multiplier.

row-bypassing multiplier, variable-latency design, and NBTI/PBTI models. Section III details the aging-aware variable-latency multiplier based on the column- or rowbypassing multiplier. The experimental setup and results are presented in Section IV. Section V concludes this paper.

II. PRELIMINARIES

A. Column-Bypassing Multiplier

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 (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, 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 sumbit 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 aican be used as the

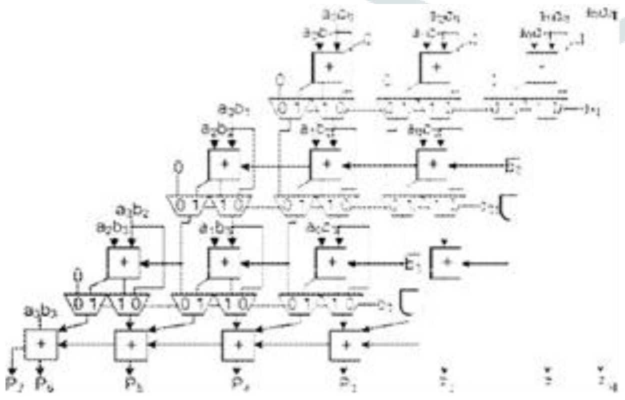


Fig. 3. 4 × 4 row-bypassing multiplier.

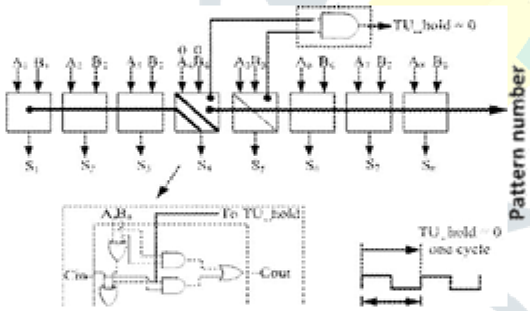
selector of the multiplexer to decide the output of the FA, and aican also be used as the selector of the tristate gate to turn off the input path of the FA. If aiis 0, the inputs of FA are disabled, and the sum bit of the current FA is equal to the sum bit from its upper FA, thus reducing the power consumption of the multiplier. If aiis 1, the normal sum result is selected. More details for the column-bypassing multiplier can be found in [22].B. 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 b1 is 0, the multiplexers in the first row select aib0 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 b2 is 0, no switching activities will occur in the second-row FAs. However, the FAs must be active in the third row because the b3 is not zero. More details for the row-bypassing multiplier can also be found in [23].

C. Variable-Latency Design

Section I mentioned that the variable-latency design was proposed to reduce the timing waste occurring in traditionalcircuits that use the critical path cycle as an execution cycle period. The basic concept is to execute a shorter path using a shorter cycle and longer path using two cycles. Since most paths execute in a cycle period that is much smaller than the critical path delay, the variable-latency design has smaller average latency.

For example, Fig. 4 is an 8-bit variable-latency ripple carry adder (RCA). A8–A1, B8–B1 are 8-bit inputs, and S8–S1 are



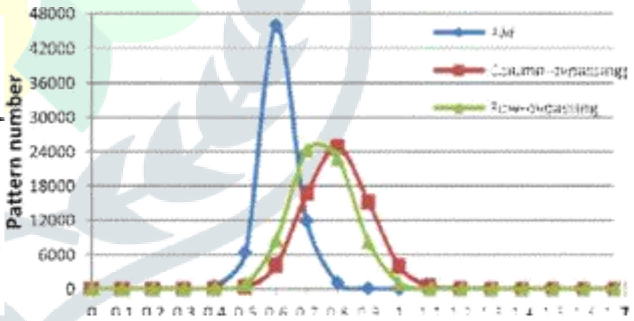


Fig. 4. 8-bit RCA with a hold logic circuit.Fig. 5. Path delay distribution of AM, column-, and row-bypassing multipliers for 65536 input patterns.

the outputs. Supposing the delay for each FA is one, and the maximum delay for the adder is 8.Through simulation, it can be determined that the possibility of the carry propagation delay being longer than 5 is low. Hence, the cycle period is set to 5, and hold logic is added to notify the system whether the adder can complete the operation within a cycle period.

Fig. 4 also shows the hold logic that is used in this circuit. The function of the hold logic is (A4 XOR B4)(A5 XOR B5). If the output of the hold logic is 0, i.e., A4 = B4 or A5 = B5, either the fourth or the fifth adder will not produce a carryout. Hence, the maximum delay will be less than one cycle period. When the hold logic output is 1, this means that the input can activate paths longer than 5, so the hold logic notifies the system that the current operation requires two cycles to complete. Two cycles are sufficient for the longest path to complete (5 * 2 is larger than 8).

The performance improvement of the variable-latency design can be calculated as follows: if the possibility of each input being 1 is 0.5, the possibility of (A4 XOR B4) (A5 XOR B5) being 1 is 0.25. The average latency for the variable-latency design is 0.75 5+0.25 10 = 6.25. Compared with the simple fixed-latency RCA, which has an average latency of 8, the variable-latency design can achieve a 28% performance improvement.

Fig. 5 shows the path delay distribution of a 16 × 16 AM and for both a traditional column-bypassing and traditional row-bypassing multiplier with 65536 randomly chosen input patterns. All multipliers execute operations on a fixed cycle period. The maximum path delay is 1.32 ns for the AM, 1.88 ns for the column-bypassing multiplier, and 1.82 ns for the row-bypassing multiplier. It can be seen that for the AM, more than 98% of the paths have a delay of <0.7 ns. Moreover, more than 93% and 98% of the paths in the FLCB and row-bypassing multipliers present a delay of <0.9 ns, respectively. Hence, using the maximum path delay for all paths will cause significant timing waste for shorter paths, and redesigning the multiplier with variable latency can improve their performance.

Another key observation is that the path delay for an operation is strongly tied to the number of zeros in the multiplicands in the column-bypassing multiplier. Fig. 6 shows the delay distribution of the 16×16 column-bypassing multiplier under three different numbers of zeros in the multiplicands: 1) 6; 2) 8; and 3) 10. Three thousand randomly selected patterns are used in each experiment. It can be seen as the number of zeros in the multiplicands increases, delay distribution is left shifted, and average delay is reduced. The reason for this is the multiplicand is used as the select line for column-bypassing multipliers, and if more zeros exist in the multiplicand, more FAs will be skipped, and the sum bit from the upper FA is passed to the lower FA, reducing the path delay. Note that similar experiments are also done for row-bypassing multipliers. However, because the results are similar, they are not shown to avoid duplications.

For a row-bypassing multiplier, the multiplicators are used to determine whether a pattern needs one cycle or two cycles to complete an operation because the multiplicator is used as the select line.

This makes the column-bypassing multiplicand and row-bypassing multiplier excellent candidates for the variablelatency design since we can simply examine the number of zeros in the multiplicand or multiplicator to predict whether the operation requires one cycle or two cycles to complete.

D. Aging Model

As mentioned in Section I, the NBTI (PBTI) effect occurs when a pMOS (nMOS) transistor is under negative (positive) bias voltage, resulting in Vthdrift. When the bias voltage is removed, the recovery process occurs, reducing the Vthdrift. phases exist, it is referred to as dynamic NBTI (PBTI). The Vthdrift of pMOS (nMOS) transistor due to the static NBTI (PBTI) effect can be described by dc reaction-diffusion (RD) framework. If transistors are under alternative stress and recovery phases, the dc RD model should be modified to an ac RD mod.

$$Vth(t) = KAC \times tn = \alpha(S, f) \times KDC \times tn \qquad (1)$$

whereα is a function of stress frequency (f) and signal probability (S). Since the impact of frequency is relatively insignificant, the effect of signal frequency is ignored. KDC is a technology-dependent constant

$$K_{DC} = A \times T_{OX} \times C_{OX}(V_{GS} - V_{th})$$

$$\times 1 - V_{DS}/\alpha(V_{GS} - V_{th})$$

$$\times \exp(E_{OX}/E_0) \times \exp - \frac{E_a}{kT} \quad (2)$$

whereA is a constant, and TOX is the oxide thickness. EOX is the gate electric field, which is (VGS–Vth)/TOX; k is the Boltzmann constant, and T is the temperature. E0 and Eaare technology-independent characteristics of the reaction that are equal to 1.9–2.0 MV/cm and 0.12 eV, respectively. More details about this model can be found in [26].

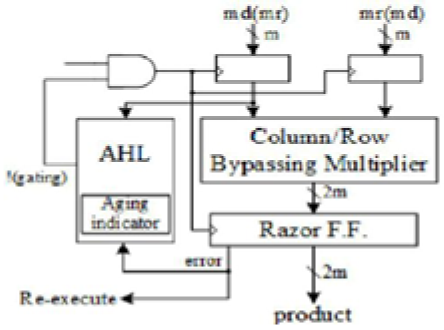
In this paper, we use 32-nm high-k metal gate models. We set the temperature at 125 °C in our simulation and use the above BTI model to predict the BTI effect on the circuits. Fig. 7 shows the simulated delays of the 16 × 16 columnand row-bypassing multipliers under a seven-year NBTI/PBTI effect. From this figure, it can be seen that the BTI effect increased the critical path circuit delay by 13%. Hence, if the BTI effect is not considered during circuit design, the increased delay may cause system failure in the long term.

III. 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 2m-bit output, one column- or row-bypassing multiplier, 2m 1-bit Razor flip-flops [27], and an AHL circuit.



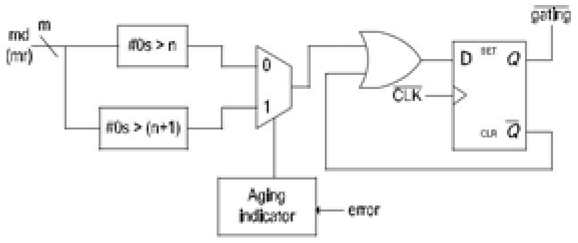


Fig. 6. Proposed architecture Fig. 9.Diagram of AHL (md means multiplicand; mr means (md means multiplicand; mr means multiplicator). multiplicator).

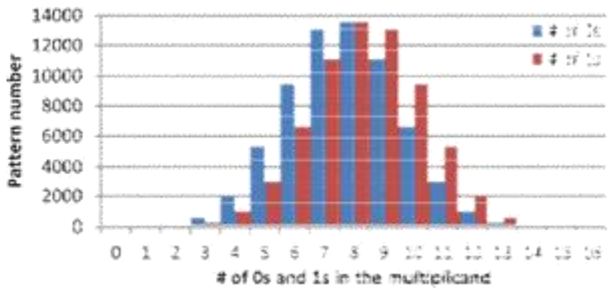


Fig. 7.Pattern number distribution based on the number of 0 s and 1 s in the multiplicand.

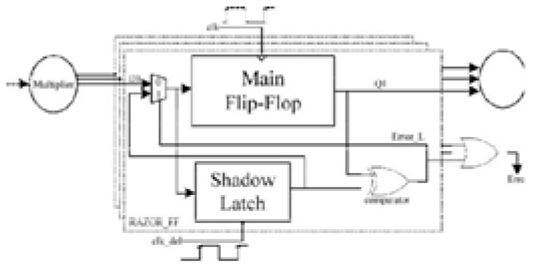


Fig. 8. Razor flip flops.

The inputs of the row-bypassing multiplier are the symbols in the parentheses.

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.

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 detectwhether timing violations occur before the next input pattern arrives.

Fig. 9 shows the details of 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 reexecute 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 reexecuted with two cycles. Although the reexecution may seem costly, the overall cost is low because the reexecution frequency is low. More details for the Razor flip-flop can be found.

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 positivenumber, 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

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 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 reexecute 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.

IV. EXPERIMENTAL RESULT

Our experiments are conducted in a Linux operating system. We adopt a 32-nm high-k predictive technology model [1] to estimate the BTI degradation for seven years. The proposed multiplier is designed in Verilog and converted to SPICE files using SpringSoft Laker. Then SynposysNanosim is used to analyze the delay and power of the circuit. The Vthdrift caused by BTI is estimated using the BTI model proposed in Section II-D and is added into the SPICE files during simulation.

In the variable-latency design, the average latency is affected by both the percentage of one-cycle patterns and the cycle period. If more patterns only require one cycle, the average latency is reduced. Similarly, if the cycle period is reduced, the average latency is also reduced. However, the cycle period cannot be too small. If the cycle period is too small, large amounts of timing violations will be detected by the Razor flip-flops, and the average latency will increase. Hence, it is important to analyze the tradeoff between the percentage of one-cycle patterns and the cycle period. To achieve this, we analyze three scenarios for both 16×16 and 32×32 variablelatency column-bypassing (VLCB) and variable-latency rowbypassing (VLRB) multipliers. We also compare the results with the AM, a FLCB multiplier, and a fixed-latency rowbypassing (FLRB) multiplier.

In Fig.9 the average latency of the three different VLCBs in the 16 × 16 multiplier is smaller than that of the FLCBs. Compared with the AM, the average latency can be either larger or smaller. For example, the average latency is smaller than the AM when the cycle period is larger than 0.85 ns in the 16 × 16 multiplier. When the cycle period in 16 × 16 VLCBs is larger than 0.85 ns, the average latency of Skip-7 is the smallest of the three scenarios. However, the average latency of Skip-7 is the largest of the three scenarios when the cycle period is <0.8 ns.

Similarly, in Fig.9 , the average latency of these VLRB multipliers is smaller than that of the FLRBs. Compared with the AM, the average latency can be either larger or smaller. For example, the average latency is smaller than that of the AM when the cycle period is larger than 0.74 ns in the

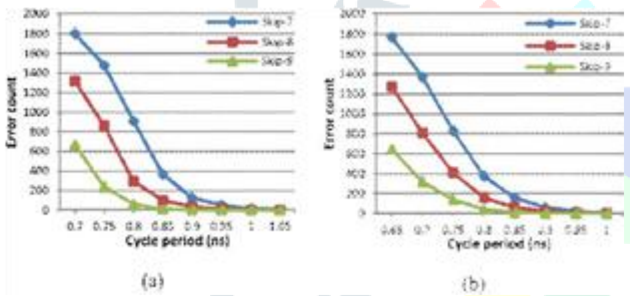
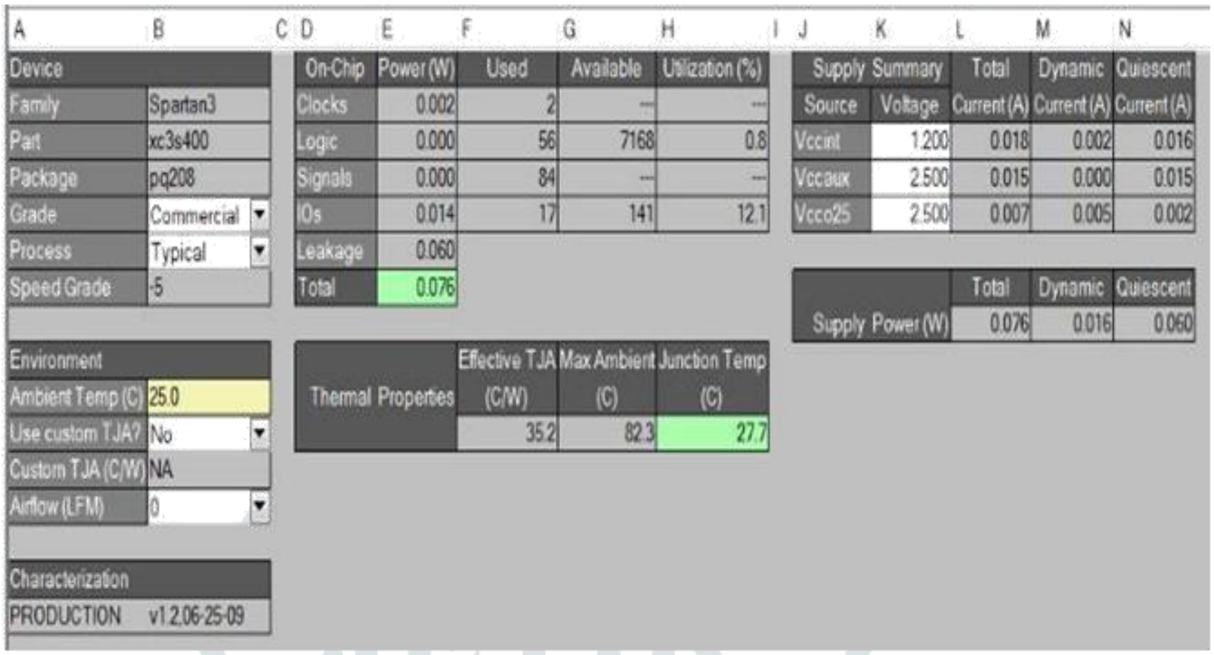


Fig. 10 .A-VLRB. Error count in 10000 cycles. (a) 16 × 16 A-VLCB. (b) 16 × 16

16 × 16 VLRBs. The Skip-7 in the 16 × 16 VLRBs has the smallest average latency when the cycle periods are >0.75 ns. However, Skip-7 has the largest average latency when the cycle periods are <0.7 ns. This situation is similar to that of the VLCBs. The VLRB with a smaller skip number has more errors in smaller cycle periods, so the average latency is increased due to the penalties incurred. In contrast, it has fewer errors in larger cycle periods, so the VLRB with a smaller skip number, i.e., Skip-7, has the smallest average latency in large cycle periods.

This is because the path delay of one-cycle patterns in the 16 × 16 variable-latency bypassing multipliers with Skip-7 is larger than that of one-cycle patterns in the 16 × 16 variablelatency bypassing multipliers with Skip-8 and Skip-9.



Device	
Family	Spartan3
Part	xc3s400
Package	pq208
Grade	Commercial
Process	Typical
Speed Grade	-5

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.002	2	---	---
Logic	0.000	56	7168	0.8
Signals	0.000	84	---	---
IOs	0.014	17	141	12.1
Leakage	0.060			
Total	0.076			

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.018	0.002	0.016
Vccaux	2.500	0.015	0.000	0.015
Vcco25	2.500	0.007	0.005	0.002

	Total	Dynamic	Quiescent
Supply Power (W)	0.076	0.016	0.060

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	35.2	82.3	27.7

Characterization	
PRODUCTION	v1.2,06-25-09

When the cycle period is <0.8 ns, the 16 × 16 variable-latency bypassing multiplier with Skip-7 has more errors, as shown in Fig. 16(a) and (b). When an error occurs, the operation needs to be executed again using three extra cycles (one cycle for Razor flip-flops and two cycles for reexecution). More errorscause more penalties, and thus the average latency is increased. Although the percentage of one-cycle patterns in the 16 × 16 variable-latency bypassing multiplier with Skip-7 is the largest, it also has more errors in smaller cycle periods, and, as a result, performance is degraded.

However, when the cycle period is >0.85 ns, the error counts of the Skip-7, Skip-8, and Skip-9 variablelatency bypassing multipliers are similar, as shown in Fig. 16(a) and

(b) as well. Since the percentage of one-cycle patterns in the 16 × 16 variable-latency bypassing multiplier with Skip-7 is higher than

Fig. 11 power analysis.

that of the other two, the average latency becomes the lowest with fewer error counts. On the contrary, when thecycle period increases, the input patterns with small delays will have more timing waste, leading increased average latency, as shown in Fig. 10(a) and (b).

Fig. 10(a) and (b) compares the average latency of a 32 × 32 variable-latency bypassing multiplier under three different numbers. Fig. 10(a) and (b) displays the error count for three different skip numbers over different cycle periods.

In 32 × 32 multipliers, there are also three scenarios: 1) Skip-15; 2) Skip-16; and 3) Skip-17. The percentages of one-cycle patterns are shown in Table II. The latencies of the 32 × 32 AM, FLCB, and FLRB are 2.74, 3.88, and 3.95 ns, respectively.

Similar to the results for the 16×16 multipliers, the average latency of the 32 × 32 variable latency bypassing multipliers is lower than that of the AM and much lower than that of the fixe latency bypassing multipliers if proper cycle periods are used.

In addition, the average latency of Skip-15 is the smallest of the three scenarios when the cycle period is large and the largest of the three scenarios when the cycle period is

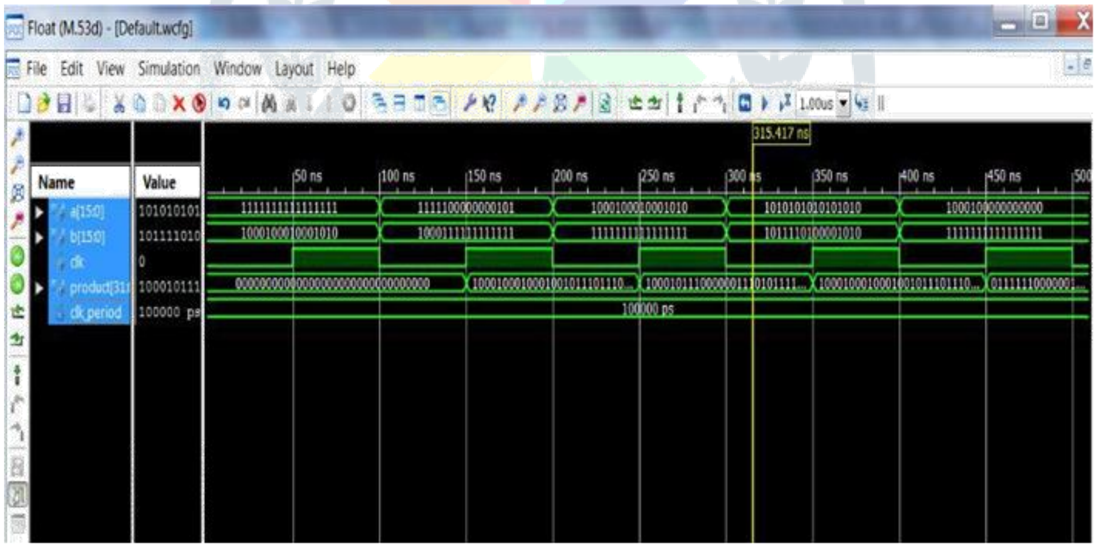
The reason for this is shown in Fig. 18(a) and (b). The 32 × 32 variable latency bypassing multipliers with Skip-15 exhibit more errors when the cycle period is short. Therefore, their average latency is the largest of the three scenarios. However, the average latency of the 32 × 32 variable latency bypassing multipliers with Skip-15 is the smallest when the cycle period is long because they have less timing waste.

B. Area Comparison

Fig. 18 compares the normalized area of the AM, FLCB, A-VLCB, FLRB, and A-VLRB in 16 × 16 and 32 × 32 multipliers. The data are normalized to the area of the AM. Inmultiplier, the area of the A-VLCB and A-VLRB is 22.9% and 23.5% higher than FLCB and FLRB. In the 32×32 multiplier, the area of the A-VLCB and A-VLRB is 12.3% and 5.7% higher than that of the FLCB and FLRB, respectively. This is because when a fixed-latency bypassing multiplier is changed to a variable-latency bypassingmultiplier, additional circuits are needed for AHL and Razor flip-flops to ensure the correct operations of the multiplier after degradation. Note that the increased area overhead ratio of the 32×32 A-VLCB and A-VLRB is much smaller than that of the 16 × 16 A-VLCB and A-VLRB. This is because AHL and Razor flip-flops both occupy a smaller area ratio in larger multipliers.

C.. Latency, Power, and Energy-Delay Product (EDP) Comparison of FLCB, A-VLCB, FLRB, and A-VLRB Over Seven Years

Fig.10 compares the latency, power, and EDP of the AM, FLCB, FLRB, A-VLCB, and A-VLRB in 16×16 multipliers from year 0 to year 7. The average latency is normalized to the latency of AM at year 0. To make comparison simpler, the cycle period of the A-VLRB and A-VLCB is set to 1.2 ns, and the skip number is 7, and therefore, no timing violations occur, and the average latency of the A-VLCB and A-VLRB are similar.



```
Name           Value
a[15:0]        101010101   1111111111111111 | 1111100000000101 | 1000100010001010 | 1010101010101010 | 1000100000000000
b[15:0]        101111010   1000100010001010 | 1000111111111111 | 1111111111111111 | 1011110100001010 | 1111111111111111
clk            0
product[31:0]  100010111   00000000000000000000000000000000 | 10001000100010010111101110... | 10001011100000001110101111... | 10001000100010010111101110... | 01111110000001...
clk_period     100000 ps   100000 ps
```

In addition, it can be seen the average latency of the AM is larger than the adaptive variable-latency multiplier after two years, and the average latency of the 16 × 16 A-VLCB and A-VLRB is 32.6%–50.1% and 28.8%–45.4% lower than that of the FLCB and FLRB, respectively. Therefore, significant latency reduction can be achieved using adaptive variablelatency multipliers.

Fig. 10compares the power of 16 × 16 AM, FLCB, FLRB, A-VLCB, and A-VLRB over seven years. To make comparison fair, the power of AM, FLCB, and FLRB includes the power ofFig.12 output wave analysis.

flip-flops at the input and output, and the power of A-VLCB and A-VLRB includes the power of flip-flops at the input and the power of Razor flip-flops at the output. It can be seen that the power consumption decreases progressively because the transistor threshold voltage increases due to the aging effect. It can also be seen that the AM has the largest average power and that the average power of the fixed-latency multiplier is less than its corresponding variable-latency multiplier (the power of FLCB is 12.1%–12.6% less than that of A-VLCB, and the power of FLRB is 7.1%–12.3% less than that of A-VLRB, on average). This is because the fixedlatency multiplier uses the bypassing technique (discussed in Section II) to reduce power consumption. Compared with the fixed-latency multiplier, the variable-latency multiplier has higher power due to more complicated circuts. However, the variable-latency multiplier still has less power than that of the AM because it uses both the clocking gating and a bypassing power reduction technique.

Moreover, the power of the 16 × 16 A-VLRB is larger than that of the 16 × 16 A-VLCB. This is because the row-bypassing multiplier is more complicated than thecolumn-bypassing multiplier and because the area overhead of the row-bypassing multipliers is larger than that of the column-bypassing multipliers, which results in more power consumption.

Fig. 10 compares the EDP of AM, FLCB, A-VLCB, FLRB, and A-VLRB in 16 × 16 multipliers over seven years. The EDP also decreases progressively because the transistor threshold voltage increases due to the aging effect. It can be seen that the EDP of the A-VLRB is higher than that of the AM in year 0 and lower than the AM after year 2. The average reduction is 3.6%. The EDP of the A-VLCB is a little higher than that of the AM

in year 0 and lower than that of the AM after year 1. The average reduction is 10.1%. Hence, variablelatency multipliers can achieve the lowest average EDP mainly because variable-latency multipliers have an average latency that is similar to that of the AM but characterized by lower power than the AM.The final result will be in

Similar to the average latency of the variablelatency bypassing multiplier is significantly less than that of the fixed-latency multiplier (the average latency of the A-VLCB is 31%–50.7% less than that of the FLCB, and the average latency of the A-VLRB is 33.2%–53.6% less than that of the average latency of the FLRB). The average latency of the variable-latency bypassing multipliers is higher than that of the AM in year 0 and lower than that of the AM after year 2.

Figcompares the power of the 32 × 32 AM, FLCB, FLRB, A-VLCB, and A-VLRB over seven years. To make comparison fair, the power of the AM, FLCB, and FLRB includes the power of flip-flops at the input and output, and the power of A-VLCB and A-VLRB includes the power of flip-flops at the input and the power of Razor flip-flops at the output. It can be seen that the power consumption decreases progressively each year due to the aging effect and increased transistor threshold voltage. Similarly, it can be seen that the AM has the largest average power and that the average power of the fixed-latency multiplier is less than its corresponding variable-latency multiplier (the power of the FLCB is 14.8%–15.2% less than that of A-VLCB, and the power of the FLRB is 9.1%–14.5% less than that of the A-VLRB). Moreover, the power of the 32 × 32 A-VLRB is larger than that of the 32×32 A-VLCB. This is because the area overhead of the row-bypassing multipliers is larger than that of the column-bypassing multipliers. Greater area overhead incurs more power consumption.

The EDP decreases progressively because the transistor threshold voltage increases due to the aging effect. It can be seen that the EDP of the A-VLRB is higher than that of the AM, and lower than that of the AM after year 2. The average reduction is 1.1%. The EDP of the A-VLCB is a little higherthan that of the AM in year 0 and lower than that of the AM after year 1. The average reduction is 10.45%. In summary, the variable-latency multiplier can achieve the lowest average EDP compared to the AM and fixed-latency bypassing multipliers.

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 electromigration. Electromigration 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 becomesnarrower, the resistance and delay of the wire will be increased, and in the end,electromigration 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 electromigration 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 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