An Efficient Design for a Reliable Multiplier to Avoid Aging Effect in MOS Transistor Using AHL Circuit

D.Varalakshmi
Department of Electronics & Communication Engineering
Indra Ganesan College Of Engineering
varalakshmichithra@gmail.com

J. Manokaran
Department of Electronics & Communication Engineering
Indra Ganesan College Of Engineering
manoraj3@gmail.com

***Abstract*-Digital multipliers are mainly used in many arithmetic functional units. The performance of the multiplier depend upon its throughput. Negative bias temperature instability and the Positive bias temperature instability effect degrade the performance of the transistor and this effect is reduced by using the adaptive hold logic technique. By using adaptive hold logic in the circuit is used to increase the performance of the transistor and is used to reduce the accurancy of the transisitor. For large multipliers the number of adders are high hence there is delay in getting output quickly inorder to overcome the delay time consumption the modified boothencoding technique is implemented which is used to reduce number of adder stages by reducing the partial product stages is implemented in radix 4 encoder.Moreover this technique can be applied to both column by passing and row by passing multiplier**

***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 mainly used in fourier transform, digital cosine transform and digital filtering. These application depend upon the throughput of the multiplier. if the multiplier is too slow it will degrade the speed of the circuit. Due to this degradation two effects occur they are negative Bias temperature Instability and positive bias Temperature instability. The negative bias temperature instability is ocuured during the interaction between inversion layer holes and hydrogen passivated atom breaks under the oxidation process by generating the hydrogen molecule. Diffusion process leads to interface traps between the molecule due to this threshold voltage is increased which will reduce the speed of the circuit. By removing the bias voltage the reverse reaction will occur and the NBTI effect is reduced but there is no possibility of removing all the interface traps which is generated during the stress phase, only the threshold voltage is increased throught the process.

The positive bias temperature is also a similar effect which is occured in the NMOS transistor. By comparing this with negative bias temperature instability it is very small effect and it is usually ignored. To avoid this effect a reliable high performance multiplier is designed. The reliable multiplier is designed 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 effect Traditional methods are used to mitigate the aging effect by using oversizing and guard banding but this technique is insufficient in power. several mapping technology is also used to reduce the aging effect. An aging aware reliable multiplier is designed with novel adaptive hold logic circuit. The AHL can be adjust to obtain the reliable and can overcome the NBTI &PBTI effects. The adaptive hold logic technique can be used for both column and row by passing multiplier. They have been mainly done with the variable latency design to overcome the aging effect .

Sizing algorithm is proposed to size the gates in a circuit, taking the temporal performance degradation into account. This algorithm estimates the threshold-voltage degradation of all individual pMOS transistors in a circuit based on their activities and accordingly sizes the circuit for a desired performance. The proposed sizing tool uses Lagrangian relaxation(LR) algorithm for global optimization of gate sizes. We estimate the performance degradation of several ISCAS benchmark circuits implemented in 70-nm Berkeley Predictive Technology model (BPTM) and show that the performance degradation of all the circuits has the same power-law dependence as the threshold voltage on time. We also size the circuits using the proposed algorithm taking the temporal performance degradation into account and compare the area overhead with that of nominal design.

Scaling of MOS device geometries poses hard limitations on the development of new generations of integrated circuits. In particular, reliability has been indicated as one of the most serious concerns Electromigration, hot-carrier injection, and time-dependent dielectric breakdown have been indicated as the main responsible of reliability decrease In CMOS circuits, NBTI effects occur in p-type transistors when a logic “0” is applied to the gate terminal (gate-to-source voltage $Vgs = -Vdd$, i.e., negative bias). Under this condition, called the *stress* state, the magnitude of the threshold voltage (Vth) increases over time, resulting in a degradation of the drive current. In contrast, when a logic “1” is applied to the gate terminal ($Vgs = 0$), NBTI stress is actually removed. The latter condition, called the *recovery* state, induces a progressive yet partial recovery of the Vth.

The impact of NBTI in random logic manifests itself as an increase of the propagation delay. Increased Vth reduces in fact the drive current of individual gates, which thus require more time to propagate the input signals, and might then no longer meet the timing constraints .To overcome this issue, a number of recent works proposed various design techniques to compensate and/or tolerate such NBTI-induced effects. The basic idea behind these approaches is to identify the gates which are responsible for the overall performance degradation (i.e., critical cells) and selectively apply *gate/transistor resizing* to guarantee larger design margin and/or *circuit transformation* Such solutions are effective because the total delay degradation of a circuit is usually significantly smaller than that of individual devices, on the order of a few percent per year of operation . This is mainly due to the fact that the propagation delay of a circuit consists of the sum of rising and falling transitions, and NBTI only affects rising transitions. Moreover, the stress time of some cell may be extremely low due to the logical structure of the circuit, thus masking the aging effect of “0” values.

Negative-bias temperature instability (NBTI) and positive-bias temperature instability (PBTI) weaken PFET and NFET over the lifetime of usage, leading to performance and reliability degradation of nanoscale CMOS SRAM. In addition, most of the state-of-the-art SRAM designs employ replica timing control circuit to mitigate the effects of leakage and process variation, optimize the performance, and reduce power consumption. NBTI and PBTI also degrade the timing control circuits and may render them ineffective. In this paper, we provide comprehensive analyses on the impacts of NBTI and PBTI on a two-port 8T SRAM design, including the stability and Write margin of the cell, Read/Write access paths, and replica timing control circuits. We show, for the first time, that because the Read/Write replica timing control circuits are activated in every Read/Write cycle, they exhibit distinctively different degradation behavior from the normal array access paths, resulting in degradation of timing control and performance.

Traditionally, timing optimization attempts to decrease the delay of a circuit by reducing the worst-case critical path delay; however, in many designs, typical input patterns do not activate such paths. In addition, due to the process, voltage, and temperature variations, minimizing the worst-case delay can be very pessimistic and inefficient. High-*k* gate dielectrics and metal gates are expected to replace conventional SiON/polysilicon gate stacks to meet aggressive gate leakage current specifications for advanced CMOS technologies. In particular for low-power (LP) applications, replacement of SiON gate dielectric by a high-*k* gate dielectric with larger physical thickness will reduce the gate current .

An aging-aware reliable multiplier is designed 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. estimate the performance degradation of several ISCAS benchmark circuits implemented in 70-nm Berkeley Predictive b the performance degradation of all the circuits has the same power-law dependence as the threshold voltage on time. We also size the circuits using the proposed algorithm taking the temporal performance degradation into account and compare the area overhead with that of nominal design. The aging aware reliable design is implemented to avoid the mitigation in the circuit this is applicable for column by passing and row bypassing Multiplier. The detailed explanation of column & row by passing Multiplier is given below.

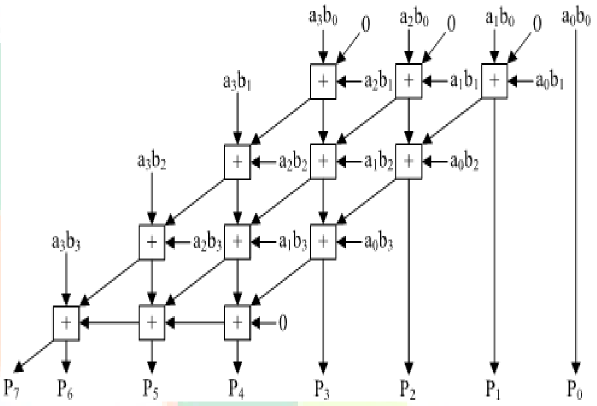


Fig. 1. 4 ×4 normal AM

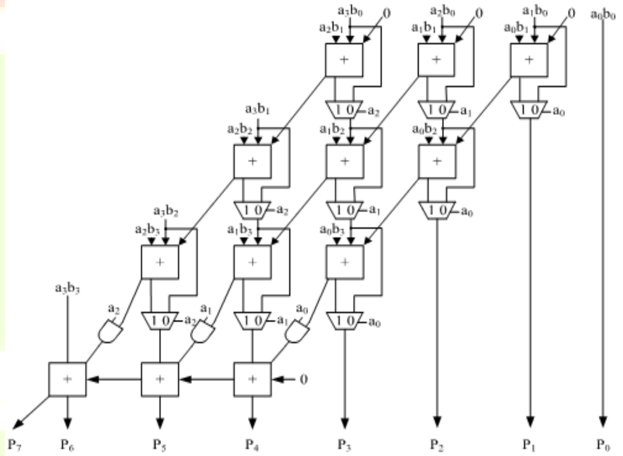


Fig. 2. 4 ×4 column-bypassing multiplier.

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. 2. 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 $1010_2 * 1111_2$, 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.

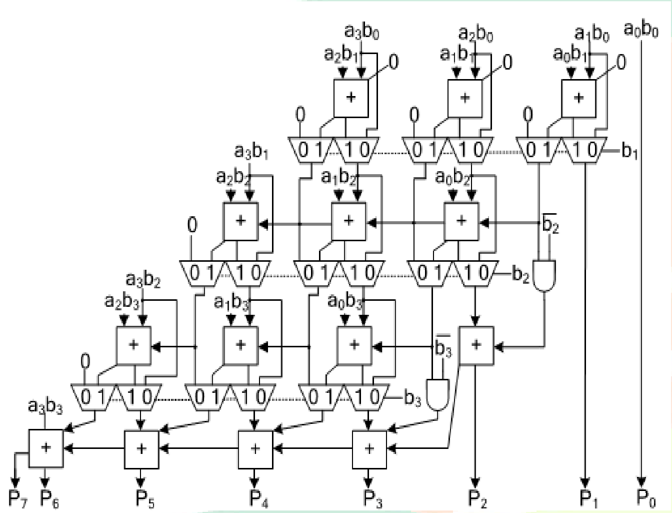


Fig. 3. 4 ×4 row-bypassing multiplier.

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 *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].

C.Variable-Latency Design

variable-latency design was proposed to reduce the timing waste occurring in traditional circuits 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.

D. Aging Model

The NBTI (PBTI) effect occurs when a pMOS (nMOS) transistor is under negative (positive) bias voltage, resulting in *V*th drift. When the bias voltage is removed, the recovery process occurs, reducing the *V*th drift If a pMOS (nMOS) transistor is under constant stress, this is referred to as static NBTI (PBTI). If both stress and recovery phases exist, it is referred to as dynamic NBTI (PBTI). The *V*th drift 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 model

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

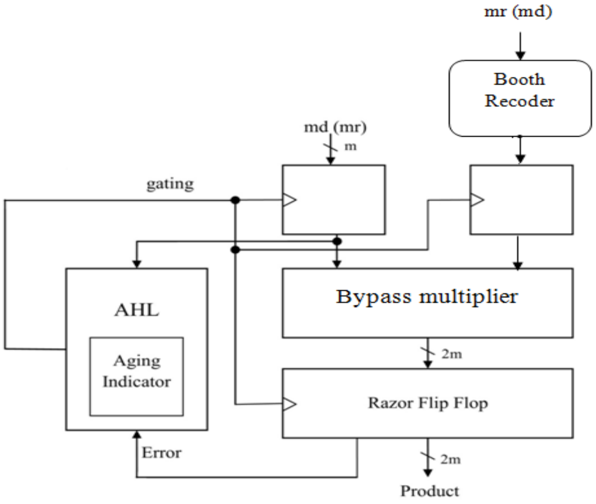
where *α* is a function of stress frequency (*f*) and signal probability (*S*). Since the impact of frequency is relatively insignificant, the effect of signal frequency is ignored. *K*DC is a technology-dependent constant

$$KDC= A \times TOX\times \sqrt{}CO\ X\ (VGS- Vth)\times 1 - VDS/\alpha(VGS-Vth)\times \exp\ (EOX/E0) \times \exp(- Ea/ET) \qquad (2)$$

where *A* is a constant, and *T*OX is the oxide thickness. *E*OX is the gate electric field, which is (*V*GS–*V*th*)/T*OX; *k* is the Boltzmann constant, and *T* is the temperature. *E*0 and *Ea* are technology-independent characteristics of the reaction that are equal to 1.9–2.0 MV/cm and 0.12 eV, respectively. 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. The BTI effect is not considered during circuit design, the increased delay may cause system failure in the long term.

II. PROPOSED AGING-AWARE MULTIPLIER

To minimize delay, Area and to reduce power consumption an aging aware circuit is designed with Booth encoding Technique it is shown with the digaramatic representation.



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. 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. For large multipliers the number of adder required are high, which is not a desired quality. To overcome this drawback we propose a radix 4 booth encoder which reduces the number of adder stages by reducing the stages of partial products.

Modified Booth (MB) is a prevalent form used in multiplication. It is a redundant signed-digit radix-4 encoding technique. Its main advantage is that it reduces by half the number of partial products in multiplication comparing to any other radix-2 representation. with direct recoding of the sum of and in its MB representation. Let us consider the multiplication of 2's complement numbers X and Y with each number consisting of n=2k bits .The multiplicand y can be represented in MB form as:

$$Y = \langle y_{n-1} y_{n-2} \cdots y_1 y_0 \rangle_{2's} = -y_{2k-1} \cdot 2^{2k-1} + \sum_{i=0}^{2k\ \ 2} y_i \cdot 2^i$$

$$= \langle \mathbf{y}_{k\ 1}^{MB} \mathbf{y}_{k\ 2}^{MB} \cdots \mathbf{y}_1^{MB} \mathbf{y}_0^{MB} \rangle_{MB} = \sum_{j=0}^{k-1} \mathbf{y}_j^{MB} \cdot 2^{2j} \quad (1)$$

$$\mathbf{y}_j^{MB} = -2y_{2j+1} + y_{2j} + y_{2j-1}. \quad (2)$$

Digits y_j^{MB}□ {-2,-1,0,+1,+2}, 0 ≤ j ≤ k-1, correspond to the three consecutive bits y2j+1, y2j and y2j-1with one bit overlapped and considering that y_{-1} = 0. Table 3.1 shows how they are formed by summarizing the MB encoding technique. Each digit is represented by three bits named s, one and two. The sign bit shows if the digit is negative (s=1) or positive (s=0). Signal one shows if the absolute value of a digit is equal to 1 (one=1) or not (one=0). Signal two shows if the absolute value of a digit is equal to 2 (two=1) or not (two=0). Using these three bits we calculate the MB digits by the following relation:

$$\mathbf{y}_j^{MB} = (-1)^{s_j} \cdot [one_j + 2 \cdot two_j]. \quad (3)$$

Table 3.1

Modified booth encoding table

Binary			$\mathbf{y}_j^{MB}$	*MB Encoding*			*Input Carry*
y_{2j+1}	y_{2j}	y_{2j-1}		sign=s_j	×1=one_j	×2=two_j	$c_{in,j}$
0	0	0	**0**	0	0	0	0
0	0	1	**+1**	0	1	0	0
0	1	0	**+1**	0	1	0	0
0	1	1	**+2**	0	0	1	0
1	0	0	**-2**	1	0	1	1
1	0	1	**-1**	1	1	0	1
1	1	0	**-1**	1	1	0	1
1	1	1	**0**	1	0	0	0

The above tabulation is used to calculate the booth encoding by using the equation 1,2,3.

Both X and Y consist of n=2k bits and are in 2's complement form. Equation (4) describes the generation of the partial products:

$$PP_j = X \cdot \mathbf{y}_j^{MB} = \overline{p}_{j,n} 2^n + \sum_{i=0}^{n-1} p_{j,i} \cdot 2^i. \quad (4)$$

$$p_{j,i} = \left((x_i \ominus s_j) \wedge one_j \right) \vee \left((x_{i-1} \oplus s_j) \wedge two_j \right). \quad (5)$$

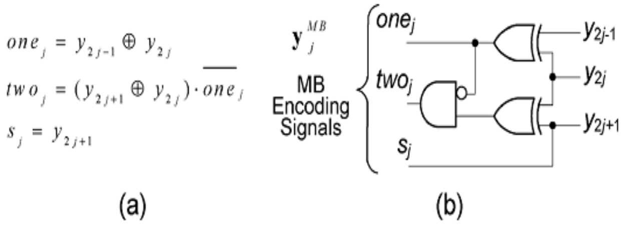


Fig 5. Boolean Equation

The generation of the i-th bit $p_{j,i}$ of the partial product PP_j is based on the next logical expression while Fig. 6illustrates its implementation at gate level.

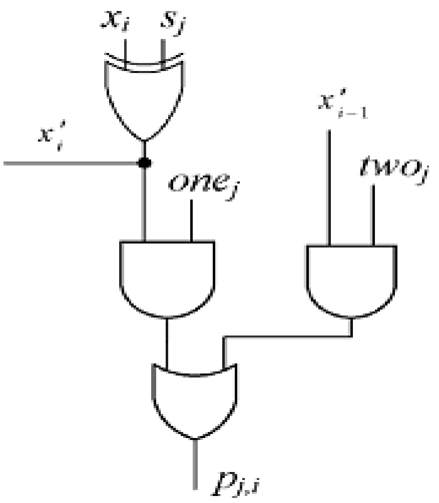


Fig 6 Generation of the i-th bit $p_{j,i}$ of the partial product PP_j for the conventional MB multiplier.

Where $c_{in,j}$=(one_j □ two_j) □ s_j (see Table 3.1).

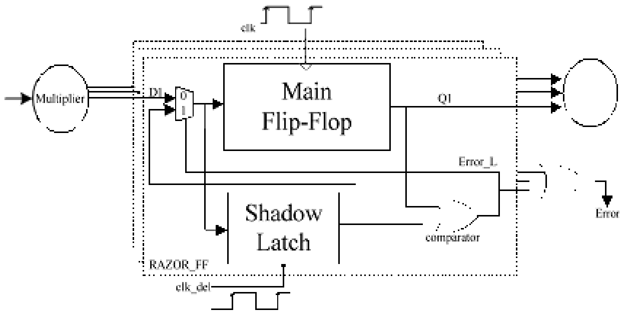


Fig 7. Razor Flip Flop

Fig. 7 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 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.

The AHL circuit is the key component in the aging-aware variable-latency multiplier. Below figure 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 meansthe circuit has suffered significant timing degradation due to the aging effect, and the aging indicator will output signal 1.

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), 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 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.

TABLE 2
Comparison Between Existing And Proposed Method

Parameter	**Existing method**	**Proposed method**
No of slice LUTs	20	26
Total memory usage	186088 kilobytes	185704 kilobytes
Total CPU time	6.45 sec	4.02 sec

By seeing the above tabulation it is clear that the proposed method occupies less area ultilize low power than the exisiting method.

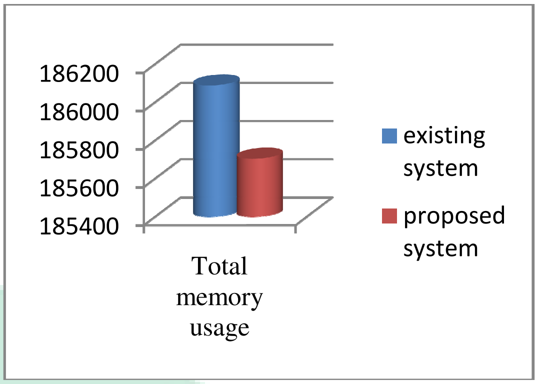


Fig 8.Total Memory Usage
Total Memory Usage by using the booth encoding technique.

By comparing with existing system the proposed system uses only less memory usage.

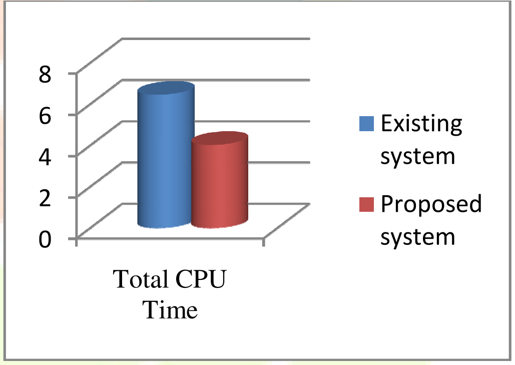


Fig 9.Total CPU Times
The total usage of CPU is less compared to the existing system.

III. CONCLUSION

This paper proposed an aging-aware variable-latency multiplier design with theBooth encoding Technique . which is used to reduce the area by reducing its partial stages.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 becomes narrower, the resistance and delay of the wire will being 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.

IV.REFERENCE

[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

[7] 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.

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

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

[10] 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.

[11] 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.

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

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

[14] 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.

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

[16] 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.

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

[24] 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.

[25] 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.

[26] R. Vattikonda, W. Wang, and Y. Cao, "Modeling and minimization of pMOS NBTI effect for robust nanometer design," in *Proc. 43rd.*