International Journal of Emerging Engineering Research and Technology
Volume 5, Issue 1, January 2017, PP 12-19
ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)
DOI: http://dx.doi.org/10.22259/ijeert.0501003

Aging –Aware Multiplier with AHL using FPGA

Zade Jyothi

M.Tech (VLSI), Sri Devi Womens Engineering College

Dr.BL.Malleshwari, M.S.,Ph.d

Principal, Sri Devi Womens Engineering College

ABSTRACT

In this we propose an aging-aware multiplier design with a novel adaptive hold logic (AHL) circuit and razor flip flop is used. The multiplier is able to provide higher though put through the variable latency and can adjust the AHL circuit to mitigate performance degradation that is due to aging effect. Digital multipliers are among the most vital arithmetic functional units. The overall performance of these systems depends on the through put of the multiplier. Due to occurrence of negative bias temperature instability effect and positive bias temperature instability effect these both effects degrade the speed of the transistor and in the long term, the system may fail due to timing violations. Therefore it is required to design reliable high performance multipliers.

The experimental results show that our proposed architecture with 8x8 and row bypassing and 16x16 column and row bypassing. Furthermore we can also develop 32x32 and 64x64 multiplier also an extension to this project. By this we can also decrease the delay when compared to 8x8 and 16x16 multiplier

Keywords: NBTI(Negative bias temperature instability), PBTI(Positive bias temperature instability), AHL(Adaptive hold logic),DSP(digital signal processing), BTI(bias temperature instability)

INTRODUCTION

When input patterns arrive, the column bypassing multiplier and the AHL circuit execute simultaneously. According to the number of 0's in the md, the AHL circuit decides if the input patterns require 1 or 2 cycles. If the input pattern requires two cycles to complete, AHL will output 0 to disable the clock signal of the flip-flops. Otherwise, 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-flop. 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 in correct.

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 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 summary, our proposed multiplier design has some key features. First, it is a variable latency design that minimizes the timing waste of the non critical 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. When the circuit is aged, and many errors occur, the AHL circuit uses these judging block to decide if an input is one cycle or two cycles.

PROPOSED AGING-AWARE MULTIPLIER

The proposed scheme is aging-aware reliable multiplier design as shown in Fig1. 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.

Proposed Architecture

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

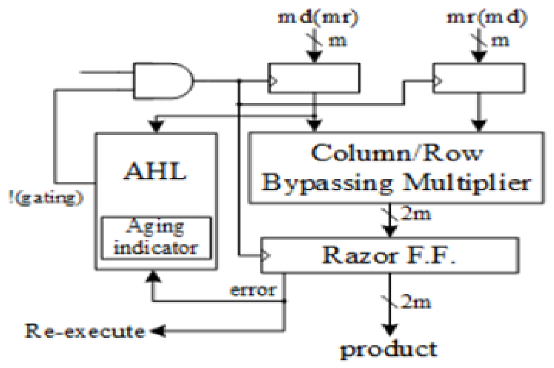

Figure1. *Proposed architecture (md means multiplicand; mr means multiplicator).*

Hence, the two aging-aware multipliers can be implemented using similar architecture, and the difference between the two bypassing multipliers lies in the input signals of the AHL. According to the bypassing selection in the column or row-bypassing multiplier, the input signal of the AHL in the architecture with the column-bypassing multiplier is the multiplicand, whereas 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. 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 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. If not, the operation is re-executed with two cycles. Although the re-execution may seem costly, the overall cost is low because the re-execution frequency is low. More details for the Razor flip-flop can be found.

The AHL circuit is the key component in the aging-ware variable-latency multiplier. Fig3. 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 agingindicator 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 as shown in Fig2.

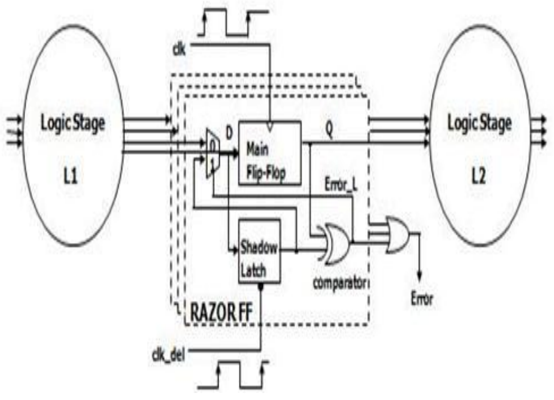

Figure2. *Razor flip flops.*

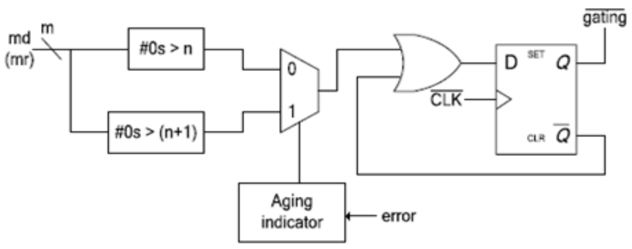

Figure3. *Diagram of AHL (md means multiplicand; mr means multiplicator).*

Column-Bypassing Multiplier

The column bypassing multiplier is constructed as follows. First, the modified HA cell is shown in Figure4. Note that we only need two three-state gates and one multiplexor in this design. If aj = 0, the HA will be disabled. For a Braun multiplier, there are only two inputs for each FA in the first row (i.e., row 0). Therefore, when aj = 0, the two input of FA0,j are disabled, and thus it output carry bit will not be changed. Therefore, all three inputs of FA1,j are fixed, which prohibit its output from changing.

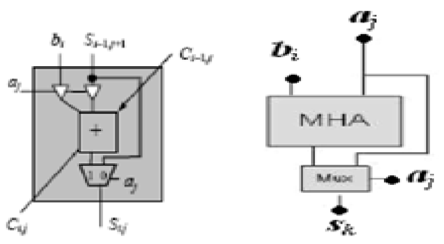

Figure4. *Number of 2 TO 1 Multiplexers required to design column bypass multiplier are (n-1)*(n-1).*

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. 4. 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.5 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 ai bi . 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.

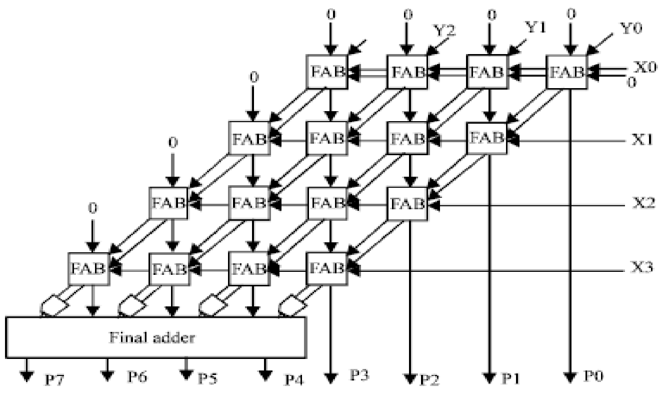

Figure5. 4 × 4 column-bypassing multiplier

Row-Bypassing Multiplier

A low-power row-bypassing multiplier is also proposed to reduce the activity power of the Array Multiplier. 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. 6 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 ai b0 as the sum bit and select 0 as the carry bit. The inputs are by passed 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

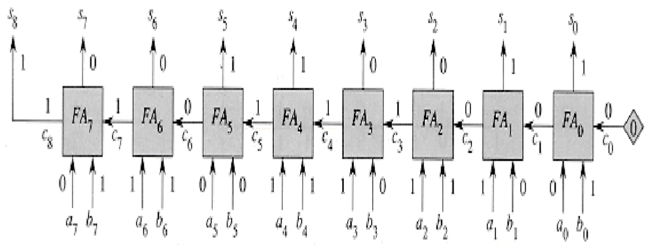
s_8 s_7 s_6 s_5 s_4 s_3 s_2 s_1 s_0
FA_7 FA_6 FA_5 FA_4 FA_3 FA_2 FA_1 FA_0
c_8 c_7 c_6 c_5 c_4 c_3 c_2 c_1 c_0
a_7 b_7 a_6 b_6 a_5 b_5 a_4 b_4 a_3 b_3 a_2 b_2 a_1 b_1 a_0 b_0

Figure6. *Row by Passing Multiplier*

For a low power row bypassing multiplier, the addition in the jth row can be disabled to reduce the power dissipation if the bit bj in the multiplier is 0, i.e, all partial products ai bj, $0 \leq i \leq n-1$, are zero. As a result, the addition operations in the jth row of CSA is in the Fig. 6 is bypassed and the outputs from the (j-1)th row of CSAs is directly fed to the (j+1)th row of CSAs without affecting the multiplication result. In the design, each modified FA in the CSA array is attached by three tristate buffers and two 2-to-1 multiplexers. The tri-state buffer shown in decides whether to disable the full adders or not according to the multiplier bits bj. And then utilizes two multiplexers to select the correct outputs. The extra correcting circuits must be added to correct the multiplication result.

When the corresponding partial product is zero, the RBAC disables unnecessary transitions and bypasses the inputs to outputs. Two multiplexers augmented to the outputs of the adder transmit the input-carry bit and the input-sum bit of the previous addition to the outputs.

The tri state buffers placed at the input of the adder cells disable signal transitions in the adders which are bypassed, and the input carry bit and input sum bit are passed to downwards.

Razor Flip-Flop

The Razor technique was developed at the Electrical Engineering and Computer Science Department of the University of Michigan. The key idea of Razor is to purposely operate the circuit at sub-critical voltage and tune the operating voltage by monitoring the error rate. This eliminates the need for conservative voltage margins. If the non-zero error rate is maintained sufficiently low, then the power overhead from correcting these errors is minimal, while the power savings from the reduced operating voltage can be substantial. The key idea of the Razor was to purposely operate the circuit at the sub-critical voltage and reduce the operating voltage by analyzing the error rate. This eliminates the need for conservative voltage margins. The trade-off would be between the power penalties incurred from error correction against the additional power attained from working at a lower supply voltage

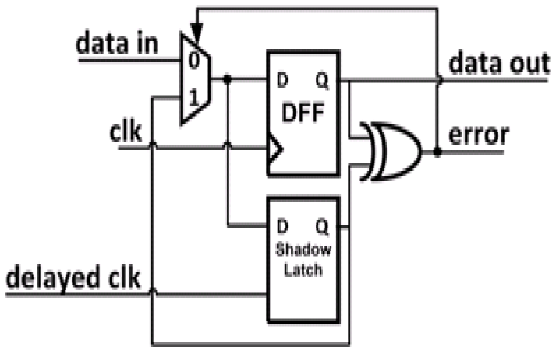

Figure7. *Razor Flipflop*

Fig7. 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. 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 there execution frequency is low. More details for the Razor flip-flop can be found.

If the combinational logic doesn't complete its computation in time, the main flip-flop will have latch an incorrect value, while the shadow latch will have latch the late-arriving correct value. The error signal would then go high, prompting restoration of the value from the shadow flip-flop latch into the main flip-flop. The Razor Flip-flop response is given in Fig8

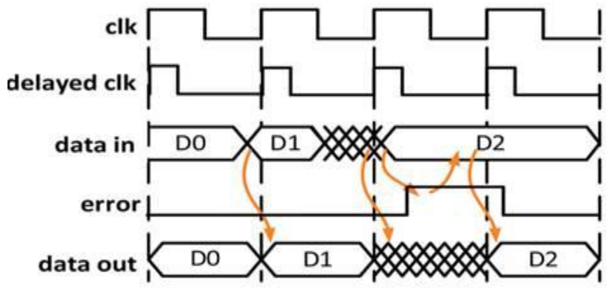

Figure8. *Razor Flipflop Waveform*

ADAPTIVE HOLD LOGIC

AHL Architecture

A novel architecture of an Adaptive Hold Logic (AHL) circuit is proposed which will reduce the aging effects. The Adaptive Hold Logic (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.

The Adaptive Hold Logic (AHL) circuit is as shown in fig.10 Assume the AHL circuit has a m bit input. The Adaptive Hold Logic (AHL) circuit consists of the following blocks.

a. Aging Indicator b. Judging Blocks

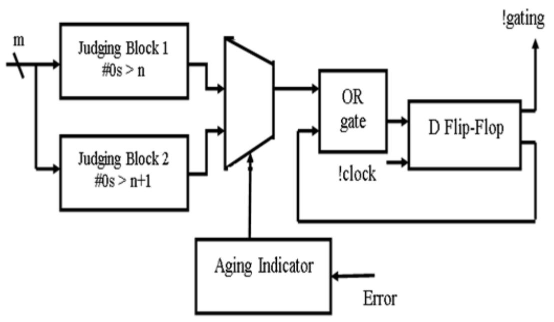

Figure9. *Proposed design of Adaptive Hold Logic (AHL) circuit*

Aging Indicator

The Aging Indicator indicates that whether the circuit has suffered significant performance degradation due to aging effects. The aging effect is not significant in the beginning, so the aging indicator produces output as 0. The Aging Indicator is implemented in a counter that counts the number of errors over a certain amount of operations. These operations may be multiplication or addition. It resets to zero at the end of those operations.

Judging Blocks

There are two judging blocks in Adaptive Hold Logic (AHL) circuits. The 1st judging block will generate an output as 1 if the number of zeros in the input sequence is larger than n. If the number of zeros in the input sequence is larger than n+1 then the output of the 2nd judging block is 1. The value of n is defined by the user. The operation of Adaptive Hold Logic (AHL) circuit are as follows:

The AHL circuit is the key component in the aging-ware variable-latency multiplier. Fig. 9 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. 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.

Variable Latency Design

The 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. For example, Fig.10 is an 8-bit variable-latency ripple carry adder (RCA). *A*8–*A*1, *B*8–*B*1 are 8-bit inputs, and *S*8–*S*1 are

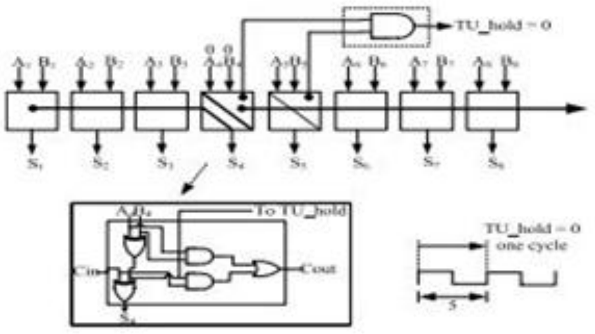

Figure10. *10-bit RCA with a hold logic circuit.*

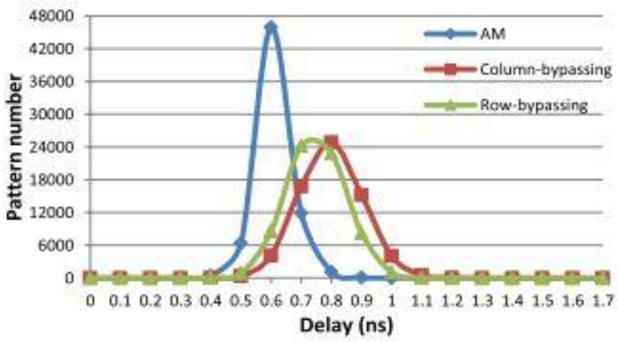

Figure11. *Path delay distribution of AM, column-, and row-bypassing multipliers for 65 536 input patterns.*

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 11 also shows the hold logic that is used in this circuit. The function of the hold logic is (*A*4 XOR *B*4)(*A*5 XOR *B*5). If the output of the hold logic is 0, i.e., *A*4 = *B*4 or *A*5 = *B*5, 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 (*A*4 XOR *B*4) (*A*5 XOR *B*5) 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.12 shows the path delay distribution of a 16 × 16 AM and for both a traditional column-bypassing and traditional row-bypassing multiplier with 65 536 randomly chosen input patterns. All multipliers execute operations on a fixed cycle period.

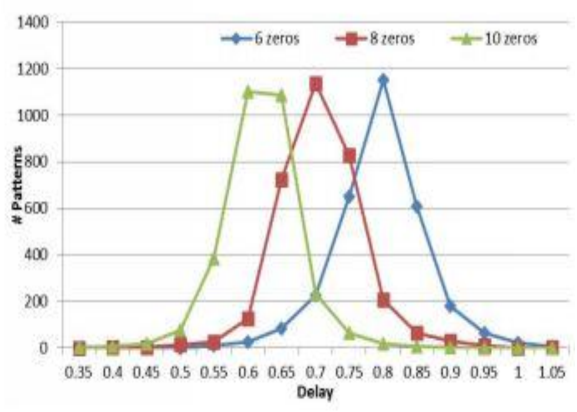

Figure12. *Delay distribution of a 16 column-bypassing multiplier under three different numbers of zeros in the multiplicands.*

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. 12 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.

RESULTS

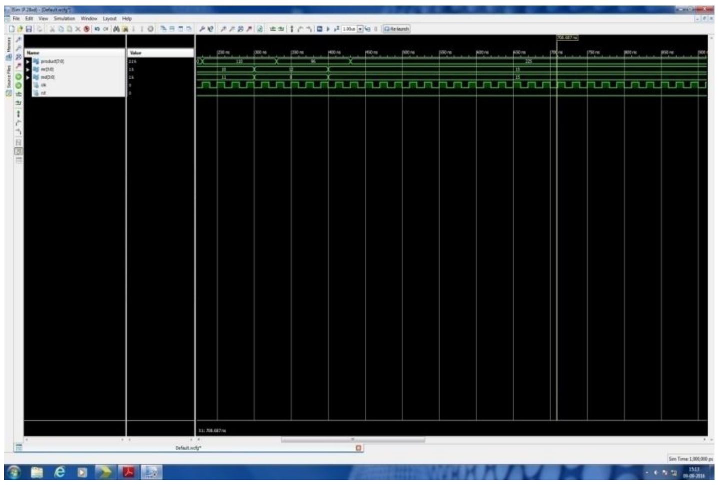

Figure13. *Output waveform of 4-bit aging-aware reliable multiplier*

The design entry is modelled using VHDL in Xilinx ISE Design Suite 8.1 Structural model of fixed latency multiplier and low power variable latency multiplier with AH logic in 4x4, 16x16 is developed. The low power variable latency multiplier with AH logic contains modules such as a column bypassing multiplier, the razor flip-flop and an adaptive hold logic. Compared with the fixed-latency multiplier, the variable-latency multiplier has higher power due to more complicated circuits. However, with changing the framework of normal variable-latency multiplier, it still has less power than that of the fixed-latency multiplier, because it uses both the clocking gating and a bypassing power reduction technique. Similarly in the case of delay, variable latency based multipliers has less delay when compared with fixed latency ones. Based on the circuit area, power and delay comparison, the 16x16 variable latency based multiplier with AH logic is the most efficient one and the 4x4 variable latency based multiplier with AH logic is the least efficient one. Hence, 16x16 variable latency multiplier with AH logic can be applied to digital FIR filter design so as to enhance its performance.

APPLICATIONS

The application of the proposed multipliers to image processing is illustrated. A multiplier is used to multiply two images on a pixel by pixel basis, thus blending the two images into a single output image. Image Processing is processing of images using mathematical operations by using any form of signal processing for which the input is an image, a series of images, or a video, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image.

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional signals while applying already set signal processing methods to them. It is among rapidly growing technologies today, with its applications in various aspects of a business.

CONCLUSION

The variable latency technique has been shown to be very useful in reducing the power consumption and time delay. The experimental results show that our proposed architecture with the 16x16 row- and column-bypassing multipliers has better performance compared with the 16x16 fixed-latency row- and column-bypassing multiplier. Variable-latency design minimizes the timing waste of the noncritical paths. This multiplier is able to adjust the adaptive hold logic to mitigate performance degradation due to delay problems. Variable-latency design can adjust clock cycle required by input patterns. And hence variable latency multipliers have less performance degradation when compared with traditional fixed latency multipliers, which needs to consider the degradation by both the NBTI effect and use the worst case delay as the cycle period. This can be used as an application of image multiplication. We can multiply two images using low power variable latency multiplier with AH logic, can be enhanced by reduced delay, area and power.

REFERENCES

[1] Ing-Chao Lin, Member, IEEE, Yu-Hung Cho, and YiMing Yang, "Aging-Aware Reliable Multiplier Design With Adaptive Hold Logic" IEEE Transactions On Very Large Scale Integration (VLSI) Systems.

[2] Prabhu E, Mangalam H, Saranya K, " Design of Low Power Digital FIR Filter based on Bypassing Multiplier", International Journal of Computer Applications (0975 – 8887) Volume 70– No.9, May 2013

[3] J.Sudha Rani, D.Ramadevi, B.Santhosh Kmar,Jeevan Reddy K, "Design of Low Power Col-umn bypass Multiplier using FPGA", IOSR Journal of VLSI and Signal Processing (IOSR-JVSP),Vol.1, Issue 3 (Nov. - Dec. 2012).

[4] Yu-Shih Su, Da-Chung Wang, Shih-Chieh Chang, Member, IEEE, and Malgorzata Marek-Sadowska, "Performance Optimization Using Variable-Latency Design Style", IEEE Trans-actions On Very Large Scale Integration (VLSI) Systems, Vol. 19, No. 10, October 2011.

[5] N.Ravi, Dr.T.S.Rao, Dr.T.J.Prasad," Performance Evaluation of Bypassing Array Multiplier with Optimized Design," International Journal of Computer Applications, Vol. 28, No.5, August 2011

[6] K.-C. Wu and D. Marculescu, " Aging-aware timing analysis and optimization considering path sensitization," in Proc. Date, 2011.

[7] Yiran Chen, Hai Li, Cheng-Kok Koh, Guangyu Sun, Jing Li,Yuan Xie, and Kaushik Roy, "Variable-Latency Adder (VL-Adder) Designs for Low Power and NBTI Tolerance", IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol.18, No.11, November 2010.

[8] Bipul C. Paul, Member, Kunhyuk Kang, Haldun Kufluoglu, "Impact of NBTI on the Tem-poral Performance Degradation of Digital Circuits", IEEE Electron Device Letters., Vol. 26, No. 8, August 2005.

[9] J. Ohban, V. G. Moshnyaga, and K. Inoue,"Multiplier energy reduction through bypassing of partial products," in Proc. APCCAS, 2002.