International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 10, October 2015
Available at http://internationaljournalofresearch.org

Design and Development of Reliable Multipliers using Adaptive Hold Logic

G.Sneha$^{\#1}$; N.Srikanth$^{\#2}$& M. Premchand$^{\#3}$
Department of Electronics and Communication Engineering, Swarnandhra College Of Engineering & Technology, Seetharampuram, Narsapur, A.P, INDIA
gsneha.999@gmail.com; srikanth648@yahoo.co.in; prem_stv@yahoo.co.in

ABSTRACT:
Multipliers, Adders and Registers are the basic building blocks of arithmetic units, in digital signal processing and Microprocessors. Performance of the circuit is very important and plays very crucial role in VLSI implementation. Multipliers are core component which can upgrade or degrade the performance. Aging Effect is one of the criteria to degrade the performance. Aging occurs due to timing waste in the circuits. The multiplication operation speed can be increased by designing different types of multipliers. So, a high performance reliable multiplier is designed to improve the performance of the system. The column bypassing multiplier and row bypassing multipliers which are improved from array multiplier are designed with two different types of adders and embedded into an proposed architecture which consists of Adaptive hold Logic module with Variable latency technique and Razor flip flop to improve the performance. To improve the performance of these two multipliers, we performed the final addition level with Carry look ahead adder. In this paper, we compared 64-bit multipliers designed with ripple carry adder and carry look ahead adder in final addition and the results shown that performance of multiplier with carry look ahead adder was improved.

Index Terms—Adaptive hold logic (AHL) reliable multiplier; variable latency

I. INTRODUCTION

The main aim of this paper is to design and develop a high performance reliable multiplier with Adaptive hold logic. A Digital multiplier is one of the most important arithmetic units in Digital Signal Processing (DSP's) Systems, Communication Systems, Microprocessor Designs and other Application Specific Integrated Circuits (ASIC). These multipliers play an important role to upgrade or degrade the performance of that specific design. Performance is a very important criterion in VLSI system design. The factors to upgrade or degrade the overall performance are delay, area, speed, and power consumption. The Overall performance of the system depends on the throughput of the multiplier. Speed is one of the major constraints; if the speed of multiplication operation is low then it reduces the performance of the whole circuit.

These are the two effects which reduce the performance by degrading the transistor speed. They are Negative Bias Temperature Instability (NBTI) effect and Positive Bias Temperature Instability (PBTI) effect, in for the long haul due to timing violations there will be a failure of the system. The Negative Bias Temperature Instability (NBTI) effect occurs when a pMOS transistor is under negative bias it increases the threshold voltage of this transistor and reduces multiplier speed. When the transistor is in this situation, the reciprocal action between hydrogen-passivity Si atoms and inversion layer breaks the Si-H bond which was generated during oxidation process, and generates H or H_2 molecules shown in figure 1. When these molecules dispersed away, the interface traps are remained. These acquired interface traps between the gate oxide interface and silicon results in increasing the threshold voltage and reduces the circuit speed. When this biased voltage is evacuated, the reverse reaction occurs and reduces the negative bias temperature instability effect but this reaction cannot get rid of the interface traps at the time of the stress phase and in long term the threshold voltage increases. So, it is necessary to design a high performance reliable multiplier.

Similarly, the Positive Bias Temperature Instability (PBTI) effect occurs when an nMOS transistor is under positive bias. Threshold voltage shifts which occurs under the negative gate bias are considered as negative bias temperature instability and that occur under positive bias are referred as positive bias temperature instability or charge trapping. In comparison with negative bias temperature instability effect, the positive bias temperature instability effect is less on the polygate/oxide transistors so this effect can be ignored.

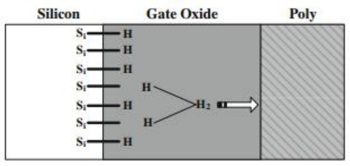

Figure-1: NBTI Mechanism

When the performance is gradually reducing then the life time of the transistors also reduces. The MOSFET’s are the basis of CMOS chips which functions as an electric switch. A MOSFET consists of four terminals, they are the gate, the body, the drain, and the source, where as the source and the body are generally connected. The voltage which is applied to gate regulates whether the current can flow amid of the source and the drain. Whereas the gate is electrically insulated with a thin layer of dielectric material, and the electric field which is applied across to it adjusts the conductivity of the hidden semiconductor channel connects with the source and the drain.

The mechanisms that break the Si H bonds at Si/oxide interface are Hot Carrier Injection (HCI), Negative Bias Temperature Instability (NBTI), and Oxide breakdown. The first degenerative mechanism is charge carriers (*p*-channel , or holes for positive, *n*-channel, or electrons for negative, MOSFETs), Over time there will be a little energy than the before will be in the conductive channel which is in between the source and the drain and these are trapped in the insulated dielectric. This process is called Hot-Carrier Injection, finally it builds up the electric charge in the dielectric layer, with increase in the voltage which will turn on the transistor. When this threshold voltage (V_{th}) increases, then the transistor switches very slowly.

The aging mechanism occurs when the voltage is applied to gate it generates electrical defects, those are the traps, and not beyond the dielectric. If they become more then these charge traps will join and forms an unequivocal short circuit amid of the current channel and the gate. This type of failure is known as Oxide breakdown, or time dependent dielectric breakdown. Compared to other aging mechanisms, this will cause a continuous decrease in performance.

A common method to alleviate the aging effect is gate over sizing and guard banding however, this path can be despondent and there will be inefficient power and area. Many negative bias temperature instability (NBTI) aware methodologies can avoid aging effect problems. A NBTI sleep transistor is designed to decline the aging effect on the pMOS sleep transistors. An NBTI optimization method considers path sensitization. To decline the power and extend life time of the circuit, bodybasing techniques and dynamic voltage scaling are introduced.

Addition is mostly performed arithmetic operation in digital systems. A high speed and low power consumed adder is used in many applications especially it is an important criteria for VLSI system design. The multiplication is done by using different types of adders only. The Multiplication operation follows shift and add algorithm by taking a two n-bit operands as a multiplicand and multiplicator and gives a 2n bit result. This can be done by using serial or parallel methods. A serial multiplier has less complex circuitry but the delay will be more due to ripple where as the parallel multiplier has less delay but has more complex circuitry due to this it consumes more power than a serial multiplier. There are three main steps followed by architecture:

1. Generation of partial products.
2. Partial products reduction.
3. Final addition

II ARRAY MULTIPLIER

Array multipliers are parallel and high speed multiplier. Figure 2 shows the 4x4 array multiplier. By using a combinational circuit, the multiplication of two unsigned binary numbers are obtained with one operation and that results the product of multiplying the two numbers. This multiplier is based on add and shift algorithm. The imminent addition is carried out by using carry save addition; the final product is obtained by using any adder.

The first phase of addition is done by using Carry Save Addition method, this phase is designed with either full adders or half adders. If full adders are considered then the third bit will be considered as ‘0’. The output signals of each 1-bit adder are sum & carry, the sum bit goes down and the carry bit goes to lower level left full adder as an input. The last level the sums and carries are merged, a ripple carry adder is implemented for carry propagation

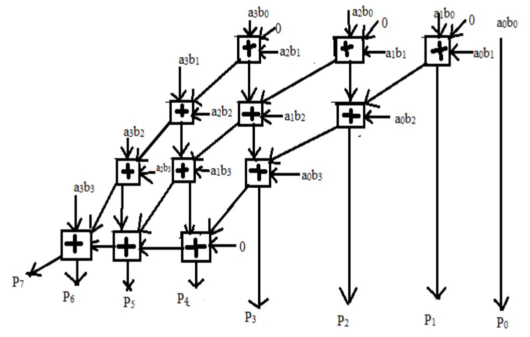

Figure-2: 4-bit Array multiplier

.

III PROPOSED RELIABLE MULTIPLIER

The architecture of high performance reliable multiplier is shown in figure 3. The architecture consists of two n-bit inputs to the parallel in parallel out register, column/row bypassing multiplier, Razor flip flop and Adaptive Hold Logic circuit. The two bypassing multipliers are implemented in this architecture and the difference between these two multipliers is the inputs given to Adaptive hold logic. The input data of the adaptive hold logic in the architecture with row bypassing multiplier is multiplicator and for column bypassing multiplier is multiplicand. The column bypassing multiplier and row bypassing multiplier operation are discussed in the above chapter.

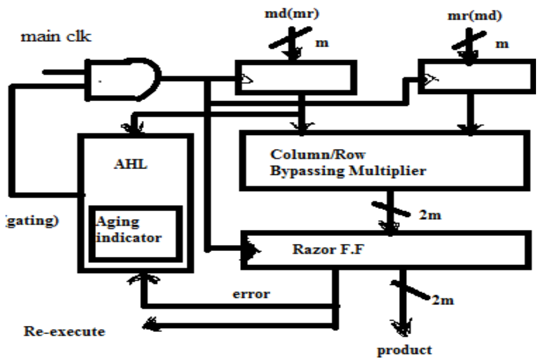

Figure-3: Proposed architecture (md means multiplicand; mr means multiplicator).

A. Bypassing techniques

The main disadvantage of this multiplier is the potential susceptibility due to glitching problem because of ripple carry adder in the last stage. The bypassing method is designed to reduce the dynamic power consumption in multiplier depends on the number of zeros in the input. To achieve confinement, transmission gates are used as ideal switches with less power consumption, and less area. Two new designs were proposed from an array multiplier. They are column bypassing multiplier and row bypassing multiplier. In column bypassing multiplier the adders in the column are bypassed by using a multiplexer, so the operations can be disabled if the interrelated bit on multiplicand is '0' So, the extra correcting circuit is eliminated. This column bypass multiplier designed with one multiplexer and two tri-state gates in adder cell shown in Figure 4.

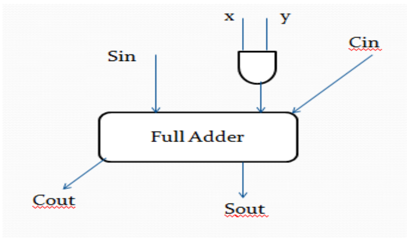

Figure-4: Adder cell

The full adder cell is replaced by the full adder bypassing cell which was shown figure 5. The transmission gates in this cell will hold the inputs applied to the full adder to stop transitions. If the selector bit of a multiplexer is '0' then the sum, Sin will propagate it to output sum, Sout and if the selector bit is '1' the output sum of the full adder is the output (Sout). The multiplexer select line is the multiplicand of the given input pattern. This full adder bypassing cell is for the column bypassing multiplier.

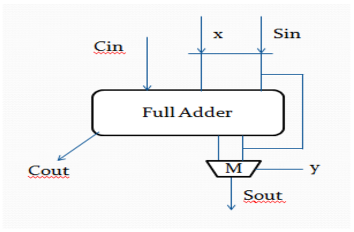

Figure-5: Full adder bypassing cell

B. Column Bypassing multiplier

The column bypassing multiplier is improved from an normal Array multiplier. The column bypassing multiplier consists of (n-1) rows of carry save adder with each (n-1) rows of full adder bypassing cells. Each full adder in carry save adder will have two outputs one is the sum and the other output is carry. The sum bit of the full adder bypassing cell will goes down where as the carry bit of the full adder bypassing cell will goes to the lower left full adder. In last row the final addition is done by using carry look ahead adder. 1-bit adder operations are disabled in the column bypassing multiplier if the analogous bit in the multiplicand is '0'. column bypassing multiplier architecture is shown in figure 6. Let us consider the inputs are 1100*0110, the full adders in the first and second diagonals, two of the three bits are 0, they are the partial product a_ib_i and the carry bit from its upper right full adder. The output of the full adders in the diagonals is '0' and the sum bit is equal to the third bit, which is the output of its upper full adder.

The full adders in this architecture are modified to add one multiplexer and two tristate gates. The multiplicand a_i bit is used as a selector for the multiplexers and it can be used as a selector to the tristate gate to turn off the full adder input data path. When the selector bit is '0' then the full adders will be disabled, thus the power consumption is reduced in the multiplier.

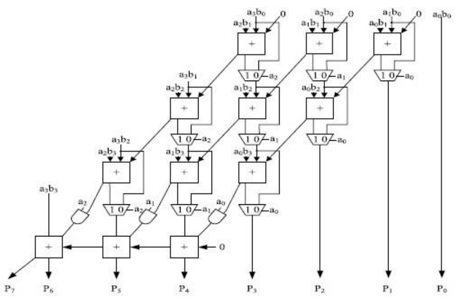
Figure-6: 4-bit Column Bypassing multiplier

C. Row Bypassing multiplier

A low power row bypassing multiplier is designed from an array multiplier to reduce the power. The low power row bypassing multiplier operation is same as the low power column bypassing multiplier, but the input bit to selector of the multiplexer is multiplicator. The figure 8 represents the 4-bit row bypassing multiplier architecture.

Each multiplicator input bit is connected to an full adder through an tristate gate. Let us consider the inputs are 1011*1001, the two inputs for the first and second row multiplexer selector is ‘0’. When b_1 is ‘0’ the multiplexer in the first row select a_ib_0 is sum bit and it selects the ‘0’ as the carry bit. Hence the inputs in this architecture are bypassed to full adders in the second row and the full adders input paths turn off by the tristate gates. So there will be no switching activity in the first row and second row full adders, hence the power consumption also reduced. The switching activity will be there in third row because the multiplexer selector bit b_3 is ‘0’.

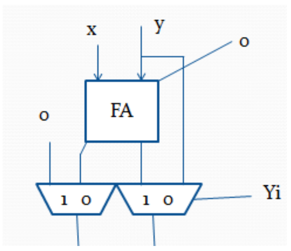

Figure-7: Full adder Bypassing cell in RBM

The full adder cell is replaced by the full adder bypassing cell which was shown figure 7. The transmission gates in this cell will hold the inputs applied to the full adder to stop transitions. If the selector bit of a multiplexer is ‘0’ then the sum, Sin will propagate it to output sum, Sout and if the selector bit is ‘1’ the output sum of the full adder is given to the multiplexer, based on the select input the output is obtained. The multiplexer select line is the multiplicator of the given input pattern. This full adder bypassing cell is for the row bypassing multiplier.

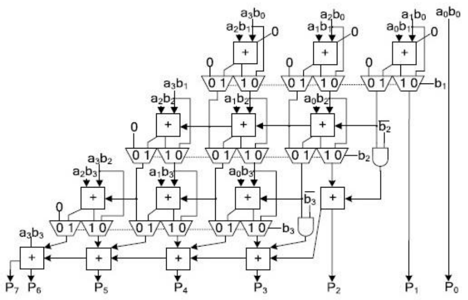
Figure-8: 4-bit Row Bypassing multiplier

D. Razor flip flop

The Razor flip flops are used to disclose, whether the timing violations arises before the arrival of next input pattern. The razor flip flop is shown in figure 9. The 1-bit razor flip flop consists of a shadow latch, multiplexer, main flip flop and xor gate. The main flip flop in the module catches the executed result with a regular clock signal. Shadow latch will catch the executed result with a delayed clock signal, which is very slower than the regular clock signals. Suppose the bit of the shadow latch is not matched with the main flip flop execution bit, means the path delay of present operation exceeds the cycle period, and the main flip flop will holds an incorrect result. If any error occurs, the razor flip flop will set to 1 to declare the system to re-execute the operation and it notice the adaptive hold logic circuit that the error was aroused.

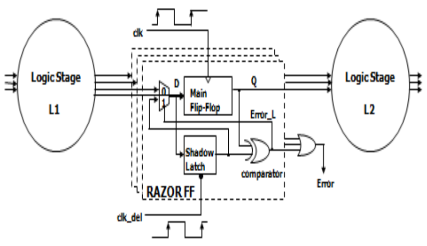

Figure-9: Razor flip flops.

The razor flip flops detects that the operation is done in one cycle pattern or if in two cycle pattern then the operation is re-executed.

E. Adaptive Hold Logic

The Adaptive hold logic circuit is one of the key components in the high performance reliable multiplier. The figure 10 shows the adaptive hold logic circuit. The adaptive hold logic circuit contains two judging blocks, an aging indicator, one D-flip flop and a multiplexer. The aging indicator is one of import component in this module to indicate even if the circuit is suffered from a eloquent performance degradation which occurred due to aging effect. This aging indicator is realized with a simple counter that counts the no. of errors in a certain set of operations and at the end of these operations it will set to zero. If it failed to complete the operation in one clock cycle then it causes timing violations. These timing violations will be captured by the razor flip flop, which will generates error signal. If the error occurs frequently and exceeds a threshold predefined value then the circuit is suffered from eloquent performance degradation and this is due to the aging effect and the output signal of the aging indicator will be '1'. If the output signal of an aging indicator is '0' then it indicates that the circuit is not eloquent.

In Adaptive hold logic circuit, there are two judging blocks the first block will give the output as '1' when the no. of zero's in the input (multiplicand -column bypassing multiplier or multiplicator for row bypassing multiplier) is larger than n where n is a positive number. The second judging block will give the output as '1' if the number of zeros in the input data (multiplicand for column bypassing multiplier or multiplicator for row bypassing multiplier) is larger than n+1. These two decision blocks will elect the input patterns requires one pattern or two cycle patterns but at a time only one of them is considered. Initially consider the aging effect is not eloquent, and the aging indicator will give the output '0' hence the first decision block will be used. Then, after a some of period of time if the aging effect is eloquent the second decision block is considered.

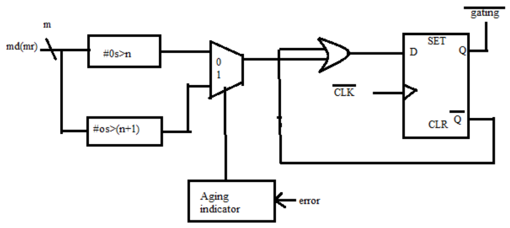

Figure-10: Adaptive Hold Logic circuit

The multiplexer in this adaptive hold logic circuit selects the result of an aging indicator. OR operation is performed for the result of an multiplexer and gating signal to resolve the input signal of an D flip flop. If !gating signal is '1' , then the flip flops will latch the current data in to the later cycle. Then the output of the mux is '0' then it shows that the input signal pattern requires 2 cycles to outright the operation. So the !gating signal which is an input to AND gate will disable the clock signal to flip flops in next cycle. Here the very important point is that the input flip flop will disable for only at one clock cycle ,in the next clock cycle the D- flip flop will take '1'.

Conventional circuits are based on the fixed latency design circuit. In this design, critical path delay is calculated for the entire circuit within overall clock cycle to check whether it performs correctly. The fixed latency design has significant timing waste because this circuit will calculate the overall critical path delay. To reduce this timing waste in traditional circuits, a variable-latency design was proposed. In this variable- latency design, two types of paths are determined. They are the shortest paths and longest paths. Shortest paths are allowed to be executing within one cycle. The longest paths are allowed to be executing within two or more cycles. If the shorter paths are activated more often, then compared to fixed latency designs, the average latency of variable latency designs are superior.

So, to improve the performance a reliable multiplier is designed with an Adaptive Hold Logic (AHL) circuit. This AHL circuit is designed to achieve reliable operation under the influence of Negative Bias Temperature Instability and Positive Bias Temperature Instability effects with Variable latency design technique. A column-bypassing multipliers and row-bypassing multipliers are included in this architecture which is an improvement from an array multiplier. A Razor flip flop is used to hold the logic and the output of the flip flop is given as an input to the aging indicator.

F. Variable-Latency Design

In order to perform the overall circuit clock cycle correctly the conventional circuits use longest path delay. This path delay is shorter than the longer path delay. The timing waste results in the circuit because we used the critical path delay as the overall cycle for non longer paths. To decrease the timing wastage in conventional circuits a variable latency technique was designed. The circuit was divided by the variable latency design into two parts. They are
1) Shorter paths
2) Longer paths.

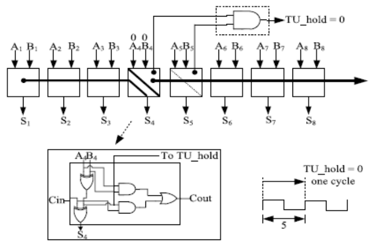

Figure-11: Variable latency design for RCA with hold circuit

Shorter paths can execute in one cycle and the longest path can execute in two cycles. If the shorter paths are used frequently, then the average latency of this design is better than that of conventional designs. Several variable latency adders are designed using a speculation technique with error recovery and detection. To improve the accuracy of the hold logic and to optimize the performance of variable latency circuit, they proposed a short path activation function algorithm. The critical paths which are divided in to two shorter paths can be unequal and clock cycle is set to delay of the longer one. All these designs will improve the performance and degrades the timing wastage of conventional circuits.

The basic concept of this variable latency design is to decrease the timing wastage occurred in the conventional circuits which use the longest path as an executed cycle period. It executes a shortest path takes one cycle and longer path using two cycles. The variable latency design has less average latency, since these paths executed in a cycle which is very less than the longest path delay. For an instance, figure 11 shows an variable latency ripple carry adder. Let us consider A and B are the 8-bit inputs, the outputs are the sum S8-S1. Suppose the delay for each 1-bit adder is one then the maxima delay for ripple carry adder is eight. It can be determine that the carry propagated delay is longer than five is less, then the one cycle period is set to five and then the hold logic is combined to the system to check even if this adder can outright the operation within a cycle period. The Figure 3.8 represents the hold logic which showed in this circuit. The hold logic function is (A_4 XOR B_4) (A_5 XOR B_5). If $A_4 = B_4$ or $A_5 = B_5$, then the output of this circuit will be 0 so the 4th or 5th adder will not give carry out. So the maxima delay is lower than a cycle period. If the output of the hold logic is 1, then the system requires 2 cycles to complete the operation which is the longest path.

The same method can be applicable for carry look ahead adder to minimize the timing waste in conventional circuits. By using ripple carry adder the propagation delay is more and to reduce the delay a variable latency design technique is used in carry look ahead adder.

To overcome the disadvantages in Ripple carry adder, we selected the carry look ahead adder. In ripple carry adder, the carry propagation delay is more. It is advisable for small bit binary adders like 4x4 and 8x8, but for higher bit adders like 16x16, 32x32, 64x64 the complexity increases and speed of operation decreases because every time each full adder should wait for the carry which is propagated from the previous full adder.

For faster carry propagation, we extend this into larger adders. The Carry Look ahead Logic which was shown in figure 11 calculates Ci in small number of logic levels. Sum and carry can be expressed in terms of generate and propagate.

The Carry generate $G_{i,}$ is only for specific input combinations x_i and y_i, the i is an adder stage to generate a carry if it gives a carry out as ‘1’. $G_i = A_iB_i$ must generate a carry when A=B=1.The Carry propagate $P_{i,}$ $P_i =$ Ai or Bi, when carry in must be equal to carry out here. The equations for sum and carry are shown below.

Sum,Si = Ai xor Bi xor Ci

Carry out, C_{i+1} =AiBi + AiCi + CiBi

= AiBi + Ci (Ai + Bi)

= Gi + CiPi

In this stage it produces carry when the carry is generated or else carry input should be 1 if it propagates carry. To avoid carry ripple we expanded Ci for each

$$C_1 = G_0 + P_0 \cdot C_0$$
$$C_2 = G_1 + P_1 \cdot C_1$$
$$C_2 = G_1 + G_0 \cdot P_1 + C_0 \cdot P_0 \cdot P_1$$
$$C_3 = G_2 + P_2 \cdot C_2$$
$$C_3 = G_2 + G_1 \cdot P_2 + G_0 \cdot P_1 \cdot P_2 + C_0 \cdot P_0 \cdot P_1 \cdot P_2$$
$$C_4 = G_3 + P_3 \cdot C_3$$
$$C_4 = G_3 + G_2 \cdot P_3 + G_1 \cdot P_2 \cdot P_3 + G_0 \cdot P_1 \cdot P_2 \cdot P_3 + C_0 \cdot P_0 \cdot P_1 \cdot P_2 \cdot P_3$$

The above equations represents the generate and propagate logic

In higher level circuits also we can use Carry look ahead 4-bit adders with each CLA logic circuit to produce the generate signal and propagate signal to Carry look ahead logic circuit. The group generate and group propagate are

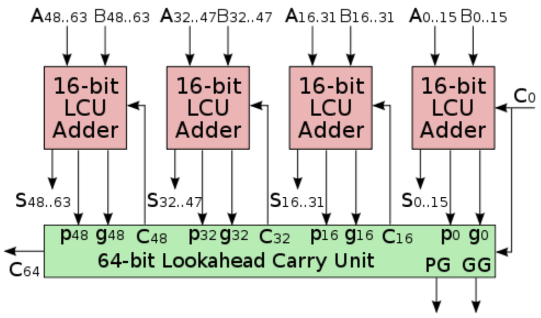

Figure-12: 64- bit Carry look ahead adder

In higher level circuits also we can use Carry look ahead 4-bit adders with each CLA logic circuit to produce the generate signal and propagate signal to Carry look ahead logic circuit. The group generate and group propagate are

$$GG = G_3 + G_2 \cdot P_3 + G_1 \cdot P_3 \cdot P_2 + G_0 \cdot P_3 \cdot P_2 \cdot P_1$$
$$PG = P_0 \cdot P_1 \cdot P_2 \cdot P_3$$

Each of the carry equationscan be implemented ina two level logic network. By using the carry look ahead adder, the computation time will decrease and the speed of operation will increase compared to ripple carry adder.

IV SIMULATION RESULTS

IV.I 64x64 Column bypassing Multiplier with Carry Look Ahead Adder

The results shown below are for 64x64 Column bypassing multiplier with carry look ahead adder. The RTL schematic, comparison tables and charts of performance parameters are shown below.

A. Simulation result

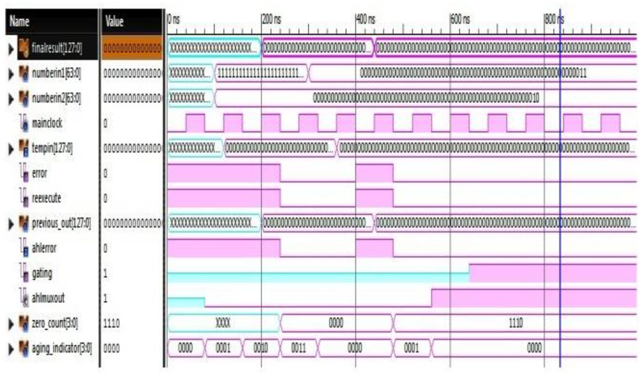

Figure-13: 64- bit CBM with CLA Simulation report

B. RTL Schematic

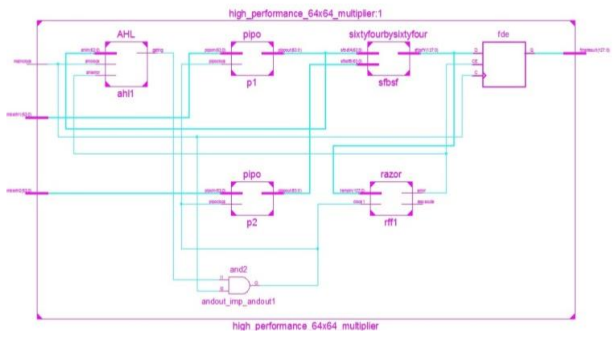

Figure-14: 64- bit CBM with CLA RTL schematic view

C. Comparison Table

The below shown table shows the delay and area delay product. It shows the difference between Ripple carry adder and Carry look ahead adder for 16,32 and 64 bit multipliers.

MULTIPLIERS	16X16		32X32		64X64	
	RCA	CLA	RCA	CLA	RCA	CLA
DELAY	29.629	28.21	43.301	43.226	71.889	70.914
AREA-DELAY PRODUCT	10663.56	9309.3	60429.94	55122.173	396324.057	356342.8

Table –I: Comparison of CBM with CLA, RCA for 16, 32 and 64bit multipliers

D. Comparison chart

The below figure shows the comparison between the 16x16, 32x32 and 64x64 bit multipliers area delay product. We can observe that the carry look ahead adder area delay product is less compared to ripple carry adder.

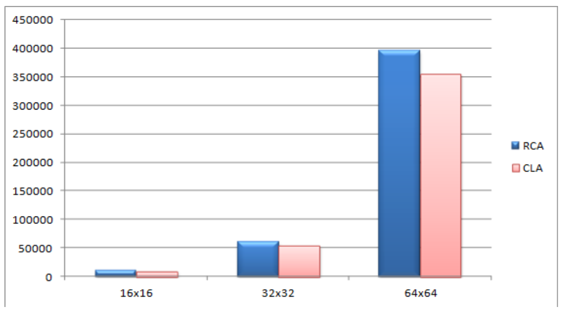

Figure-15: ADP comparison chart for CBM with CLA and RCA

IV.II Row bypassing Multipliers with Carry Look Ahead Adder

The results shown below are for 64x64 Row bypassing multiplier with carry look ahead adder.

A. Simulation result

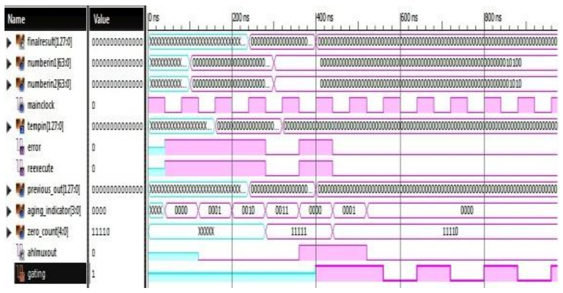

Figure-16: 64- bit RBM with CLA Simulation report

B. RTL Schematic

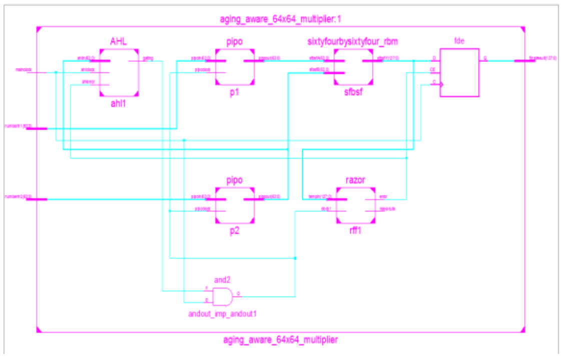

Figure-17: 64 -bit RBM with CLA RTL schematic view

C. Comparison Table

The below shown table shows the delay and area delay product. It shows the difference between Ripple carry adder and Carry look ahead adder for 16,32 and 64 bit multipliers.

MULTIPLIERS	16X16		32X32		64X64	
	RCA	CLA	RCA	CLA	RCA	CLA
DELAY	30.949	29.937	45.261	44.715	73.761	73.722
AREA-DELAY PRODUCT	14638.87	12154.42	84094.93	67296.07	527317.38	436360.5

Table- II: Comparison of RBM with CLA and RCA for 16, 32 and 64-bit multipliers

D. Comparison chart

The below figure shows the comparison between the 16x16, 32x32 and 64x64 bit multipliers area delay product. We can observe that the carry look ahead adder area delay product is less compared to ripple carry adder.

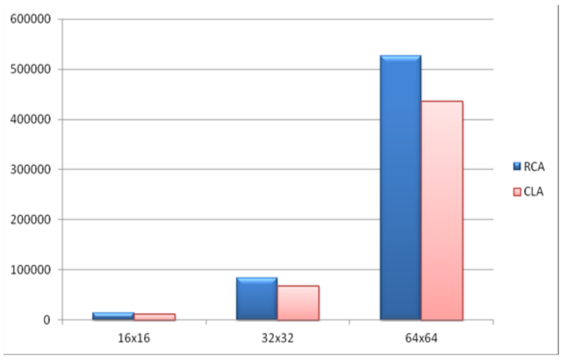

Figure-18: ADP comparison chart for RBM with CLA and RCA

V. CONCLUSION

This paper is based on a High performance Reliable multiplier design with the Adaptive Hold Logic. The Adaptive Hold Logic circuit is passed down to acquire decisive operation under the consequence of Negative Bias Temperature Instability and Positive Bias Temperature Instability effects. The proposed architecture implemented in this thesis are 64x64, 32x32, 16x16 bit column bypass and row bypass multipliers are designed to get less performance degradation. We are using Razor flip flop to hold the logic. And the output of this is given as an input to the aging indicator.

The area and delay of the column bypassing multiplier and row bypassing multiplier with carry look ahead adder is improved compared with the column bypassing multiplier and row bypassing multiplier with ripple carry adder. We observed 16x16, 32x32 and 64x64 bit multipliers. Hence the performance of a multiplier is increased.

VI. FUTURE SCOPE

For any design the intention is to increase the performance and reliability of that circuit. To reduce the count of LUT's, no. of slices, delay and to increase the speed of operation we designed a full adder bypass cell and replaced in the place of full adder. In this thesis we designed two bypassing multipliers from an array multiplier they are column bypassing multiplier and row bypass multiplier. The column bypass multiplier is designed with 2 tri-state buffers and a multiplexer to reduce the area and delay where as the row bypassing multiplier is designed using 2 tri-state buffers and 2 multiplexers. In future we can implement a multiplier which can perform both row and column bypassing technique in a single architecture by designing another full adder cell with three tri-state buffers, two multiplexers and an AND gate. By replacing this adder cell we can estimate that the speed of operation can be increased and comparative to the methods used in this thesis it can give the better results. To get rid of aging effect and to get less performance degradation we can use this multiplier in the proposed architecture which we have designed in this thesis. Finally, we can design a less performance degradation reliable multiplier with Adaptive Hold Logic.

REFERENCES

[1] Cho, Yu-Hung, Ing-Chao Lin, and Yi-Ming Yang. "Aging-aware reliable multiplier design", 2012 IEEE International SOC Conference, 2012.

[2] Keane, John, and Chris Kim. "An odomoeter for CPUs", IEEE Spectrum, 2011.

[3] Su, Yu-Shih, Da-Chung Wang, Shih-Chieh Chang, and Malgorzata Marek-Sadowska. "Performance Optimization Using Variable-Latency”.

[4] Jin-Tai Yan. "Low-cost low-power bypassing-based multiplier design", Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 05/2010

[5] Ching-Te Chuang. "Ring oscillator circuit structures for measurementof isolated NBTI/PBTI effects", 2008 IEEE International Conference on Integrated Circuit Design and Technology and Tutorial, 06/2008

[6] "Celestry's Reliability Analysis Products For Maximizing High-Performance Designs Chosen By Texas Ins", Business Wire, Oct 10

[7]R. Anitha. "FPGA Implementation of Braun’s Multiplier Using Spartan-3E, Virtex – 4, Virtex-5 and Virtex-6", Communications in Computer and Information Science, 2011.

[8] Leeser, Miriam, David Kaeli, Douglas Burger, James Goodman,Gurindar Sohi, Windsor Hsu, Jih-Kwon Peir, Ken Hinckley, and Ken Hinckley. "Fault Tolerance", Computer Science Handbook Second Edition CD-ROM, 2004.

[9] G. Burns. "Design and implementation of a 16 by 16 low-power two's complement multiplier", 2000 IEEE International Symposium on Circuits and Systems Emerging Technologies for the 21st Century Proceedings (IEEE Cat No 00CH36353) ISCAS-00, 2000

[10] R. Anitha. "FPGA Implementation of Braun’s Multiplier Using Spartan-3E, Virtex – 4, Virtex-5 and Virtex-6", CNCand Information Science, 2011