International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 17
December 2017

Design of a High Speed Multiplier by Using Ahl

Kasimbee Shaik
M.Tech (DSCE)

ABSTRACT: Digital multipliers are among the most critical arithmetic functional units. Such as the addition, subtraction, multiplication, division. The overall performance of these systems depends on the throughput of the multiplier, and to design the multiplier speed. In the long term, the system may fail due to timing violations. Therefore, it is important to design reliable high-performance multipliers. In this project VLSI design of a high speed AHL based multiplier has been proposed. The multiplier is able to provide higher throughput through the variable latency and can adjust the AHL circuit to mitigate performance degradation that is due to the aging effect. The proposed architecture can be applied to a column-bypassing multiplier. The experimental results show that our proposed architecture with 64 × 64 column-bypassing multipliers.

Keywords: Adaptive Hold Logic (AHL), Reliable Multiplier.

I. INTRODUCTION

Digital multipliers are among the most critical arithmetic functional units in many applications, such as the digital signal processing, cryptography. The through put of these applications depends on multipliers, and if hence, it is important to design a reliable high -performance multiplier. A traditional method to mitigate the aging effect is overdesign, multiplier speed is decreased. Then the delay is very high. To avoid this problem, Including such as adaptive hold logic circuit; however, this approach can be very important of the and area and delay, speed. In techniques were proposed to reduce delay or extend circuit life. These techniques, however, require circuit modification Traditional circuits use delay as the overall circuit clock cycle in order to perform correctly. Hence, the variable-latency designs to reduce traditional circuits. The variable-latency design divides the circuit into two parts shorter paths and longer paths. Shorter paths can execute correctly in one cycle, whereas longer paths need two cycles to execute. When shorter paths are activated frequently, the average latency of variable latency designs is better than that of traditional designs. For example, several variable-latency adders were proposed using a short path activation function algorithm was proposed in to improve the accuracy of the hold logic and to optimize the performance of the variable -latency circuit [1]-[2].

An instruction scheduling algorithm was proposed in to schedule the operations on latency and improve the speed. In a variable-latency pipelined multiplier architecture with a Booth algorithm future work proposed. In process-variation tolerant architecture for arithmetic units was proposed, where the effect of process-variation is considered to increase the circuit. The designs were able to reduce the delay of the traditional circuits to improve performance, but consider the aging effect and could not adjust themselves during the runtime. A variable-latency adder design that considers the aging effect was proposed. However, no variable-latency multiplier design that considers the aging effect and can adjust dynamically has been done [11]. Recently, Multipliers play an important role in many applications such as microprocessor, digital signal processing and most often used in arithmetic operations. The throughput of these applications mainly depends on multipliers. When the multiplier operations are too slow in the circuit, then the performance of the entire circuits will be reduced.

II.LITERATURE SURVEY

A column-bypassing multiplier is an improvement on the normal array multiplier. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial products are shifted according to their bit orders and then added [9]. Column-bypassing multiplier design is proposed in which the FA operations are disabled if the corresponding bit in the multiplicand is 0. Procedure for the column by passing multiplier: 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: the sum bit goes down .the carry bit goes to the lower left FA. The last row is a ripple adder for carry propagation. Column to column data is passing, there 3 inputs that are given to the full adder [13]. When selection line a0=0 then that value is given as the mux 0 condition is taken as the output. When selection line a0=1 then that value is given as the mux 1 condition is taken as the output of the full adder is sum. For example a1b0 and a0b1, 0 these 3 inputs that are given to the full adder. When a1b0 is given to the mux 0 condition. When the a0b1, a1b0, 0 are 3 inputs then sum is (0+1+1) =0, carry is 1.Disadvantages of existing method. Delay is

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 17
December 2017

very high: when we are doing normal multiplication that is taken more time so than the delay is very high. Due to the aging effect the speed of column by passing multiplier is reduced to overcome the speed problem; delay problem then adaptive hold logic is used. In the proposed project, VLSI design of a high speed of the column bypassing multipliers adaptive hold logic. Column bypassing multipliers is used can be examined by the number of zeros in either the multiplicand or multiplier to predict whether the operation requires one cycle or two cycles to complete [10]. When input patterns are random, the number of zeros and ones in the multiplier and multiplicand follows a normal distribution. Hence, the two aging-aware multipliers are using similar architecture, and the difference between the bypassing multipliers lies in the input signals of the AHL.According to the bypassing selection in the column bypassing multiplier.

III.EXPERIMENTAL DESIGN

Multipliers play an important role in today's digital signal processing and various other applications. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, less delay and hence less area or even combination of them in one multiplier thus making them suitable for various high speed. Faster way to implement multiplication is to resort to an approach similar to manually computing a multiplication [7]. The entire partial products are applied to compute the final product. The approach is illustrated.This set of operation can be mapped directly into hardware. The resulting structure is called an array multiplier and combines the following three functions: partial product Generation, partial-product and final addition [8].

a3 a2 a1 a0 =A (multiplicand)
b3 b2 b1 b0 =B (multiplier)

---

a3b0 a2b0 a1b0 a0b0
a3b1 a2b1 a1b1 a0b1
a3b2 a2b2 a1b2 a0b2
a3b3 a2b3 a1b3 a0b3

P7 p6 p5 p4 p3 p2 p1 p0 (p=A*B)

Example of Manual Multiplication

(a) NORMAL ARRAY MULTIPLIER

Array multiplier is well known due to its regular structure. Multiplier circuit is based on add and shift algorithm. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial products are shifted according to their bit orders and then added [6]. The multiplier array consists of (n−1) rows of carry save adder (CSA), in which each row contains (n−1) full adder (FA) cells [3]. Each FA in the CSA array has two outputs: 1) the sum bit goes down2) the carry bit goes to the lower left FA 3) the last row is a ripple adder for carry propagation

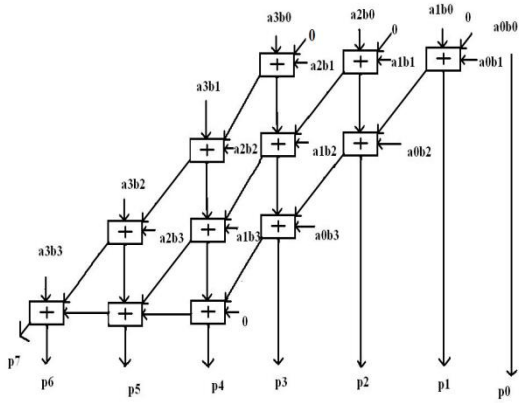

Fig. 1: 4 × 4 normal AM

For example of the a0=1 then the p0=1 and then a1b0=1, a0b1=0, 0 these are inputs of the full adder. And we get the sum is equal 1, carry is equal 0.then output p1=1.it is continuous the process. It depends on the multiplier that is the array multiplier. Here a, b are inputs of the full adder. That is P0, p1, p2, p3.p4, p5, p5, and p6, p7 of the result or product.

(b) COLUMN-BYPASSING MULTIPLIER

A column-bypassing multiplier is an improvement on the normal array multiplier (AM). The AM is a fast parallel AM. The FAs in the AM are always active regardless of input states. In [5], 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. Supposing the inputs are 10102* 11112, it can be seen that for the FAs in the first and third diagonals, two of the three input bits are 0: The carry bit from its upper right FA and the partial product (aibi) [3].

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 17
December 2017

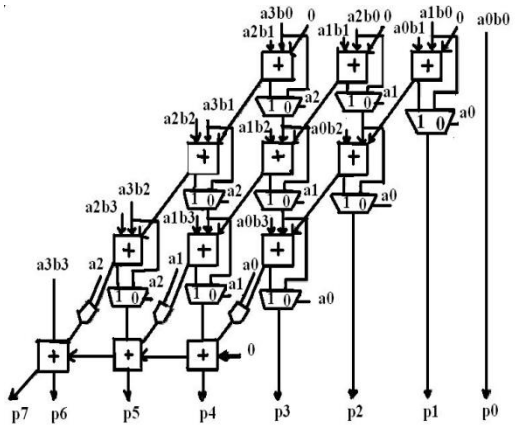

Fig.2: Column-Bypassing Multiplier

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[6].

IV.SIMULATION RESULTS

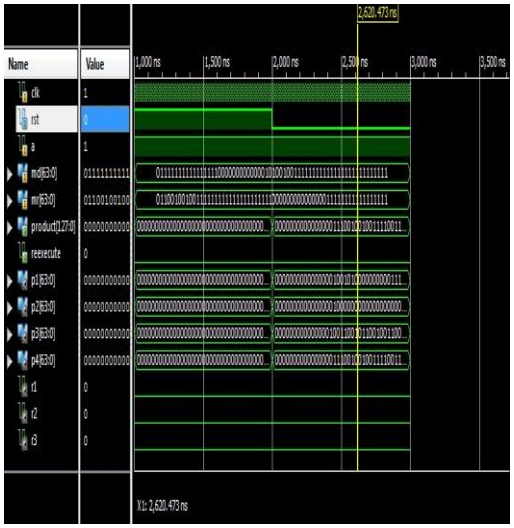

Fig.3: Output of proposed method 64*64 column bypassing multiplier with AHL

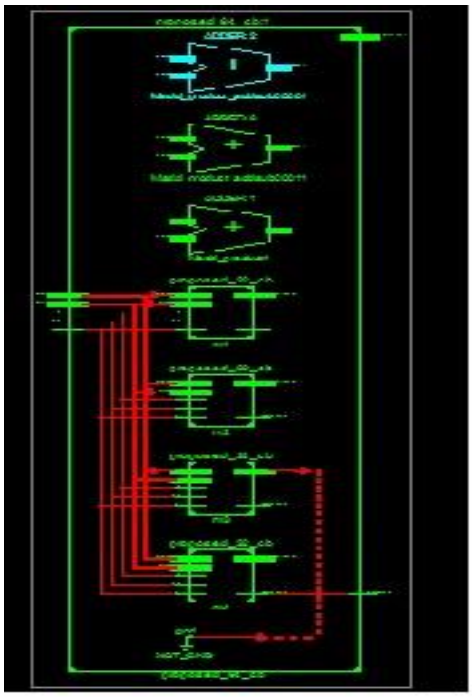

Fig.4: RTL diagram of proposed method 64*64 column bypassing multiplier with AHL

V.CONCLUSION

In this project we proposed has been, column/row by passing multiplier with AHL. The multiplier is able to adjust the AHL to mitigate performance degradation due to increased delay. The experimental results show that our proposed architecture with 64x64 multiplication.

REFERENCES

[1] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. Software Engineering Journal, 8(5):284–292, 1993.
[2] E. Bini and G. Buttazzo. Biasing effects in schedulability measures. In Euromicro Conference on Real-Time Systems, June 2004.
[3] E. Bini and G. Buttazzo. Measuring the performance of schedulability tests. Real-Time Systems, 30(1-2):129–154, 2005.
[4] K. Bowman, J. Tschanz, C. Wilkerson, S.-L. Lu, T. Karnik, V. De, and S. Borkar. Circuit techniques for dynamic variation tolerance. In Design Automation Conference (DAC), July 2009.
[5] A. Drake, R. Senger, H. Deogun, G. Carpenter, S. Ghiasi, T. Nguyen, N. James, M. Floyd, and V. Pokala. A distributed critical-path timing monitor for a 65nm

high-performance microprocessor. In International Solid-State Circuits Conference (ISSCC), Feb. 2007.
[6] M. Floyd, M. Allen-Ware, K. Rajamani, B. Brock, C. Lefurgy, A. Drake, L. Pesantez, T. Gloekler, J. Tierno, P. Bose, and A. Buyuktosunoglu. Introducing the adaptive energy management features of the POWER7 chip. IEEE Micro, 31(2):67–75, 2011.
[7] L. Huang, F. Yuan, and Q. Xu. Lifetime reliability-aware task allocation and scheduling for MPSoC platforms. In Design, Automation, and Test in Europe (DATE), Apr. 2009.
[8] J. Keane, T.-H. Kim, X. Wang, and C. Kim. On-chip reliability monitors for measuring circuit degradation. Microelectronics Reliability,
50(8):1039–1053, 2010.
[9] C. Lefurgy, A. Drake, M. Floyd, M. Allen-Ware, B. Brock, J. Tierno, and J. Carter. Active management of timing guardband to save energy in POWER7. In International Symposium on Microarchitecture (MICRO), Dec. 2011.
[10] J. Lehoczky. Fixed priority scheduling of periodic task sets with arbitrary deadlines. In Real-Time Systems Symposium (RTSS), Dec. 1990.
[11] J. Lehoczky, L. Sha, and Y. Ding. The rate monotonic scheduling algorithm: exact characterization and average case behavior. In RealTime Systems Symposium (RTSS), Dec. 1989.
[12] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. Performance Evaluation, 2(4):237–250, 1982.
[13] S. Rehman, M. Shafique, F. Kriebel, and J. Henkel. Reliable software for unreliable hardware: embedded code generation aiming at reliability. In International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), Oct. 2011.