International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 1 Issue 8 ǁ Dec 2013 ǁ PP.28-32

Design Of 64-Bit Parallel Prefix VLSI Adder For High Speed Arithmetic Circuits

Ashutosh Kumar[1], Rakesh Jain[2]

[1]*Department of Electronics & communication Engineering, Suresh Gyan Vihar University, Jaipur (Rajasthan), India*

[2]*Assistant professor, Department of Electronics & communication Engineering, Suresh Gyan Vihar University, Jaipur (Rajasthan), India*

ABSTRACT: *Parallel prefix adder is a kind of process for speeding up the addition of the system of writing and calculating with numbers which use only two digits. Parallel prefix adders are also known as carry-tree adders and they are known to have the best performance in VLSI designs. Due to constraints on logic blog configurations a routing overhead, this performance advantage does not translate directly into FPGA implementations. Identifying the absolutely accurate area-delay tradeoff curve of the parallel prefix is an interesting problem that has received more attention in research because parallel prefix adder on the other hand represents a type of general adder structure that displays publically in flexible area-time tradeoffs for the design of adder. Many different types of parallel prefix adders are made to increase for optimizing area, fan out, speed and performance. For high speed performance tree like structure is must which helps in greater way. There are many different method used for designing parallel prefix adder based on their speed, size and performance. For area optimization we use Brent-Kung method. If our main purpose is to get the least timing then we have to use Kogg-Stone adder method.*

Keywords: *carry-tree adder, kogg-stone, fan-out, operators, FPGA, operators.*

I. INTRODUCTION

Binary adders are the basic modules in computer arithmetic design, and therefore it has been examined for a period of ten years. Entirely a few standard fast adders, such as the carry-skip adder, the carry-look-ahead adder and the carry-select adder were proposed in the past. Each of the fast adders presents a unique area-time tradeoff in the design space. The structure of all fast adders is ad-hoc. Many different method used for designing parallel prefix adder based on their speed, size and performance. For area optimization we use Brent-Kung method. If our main purpose is to get the least timing then we have to use Kogg-Stone adder method. Lander & Fischer method is used where the fan out of the system is higher. If we have to take lesser area we have to use bitwise timing constraint. Using this, switching from one part to other takes lesser time hence optimizing the performance and time. In general day to day life the size of electronics materials are decreasing rapidly and hence are able to port from one place to other. For this purpose there are large requirement of such devices which takes low power and at the same time gives higher performance. Hence integrated circuits are mostly used in markets. Now to perform and optimize the device we have to use certain basic algorithm for simple addition of bits. These algorithms are designed keeping in mind about the size, performance, fan out, area and the complexity of the circuits. For this reason parallel prefix addition methods are widely accepted as the best method to calculate sum. And the algorithm used in this method is known as Kogg-Stone method fastest in addition because it gives minimum fan out. Here delay in terms of power and generation of carries is also very less as compared to others.

In parallel prefix addition, carry signal is calculated at every stage with the help of generate and propagate unit. The addition can be performed in three steps.

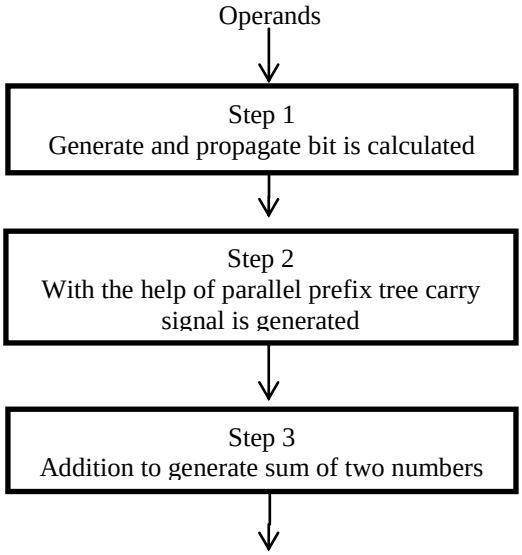
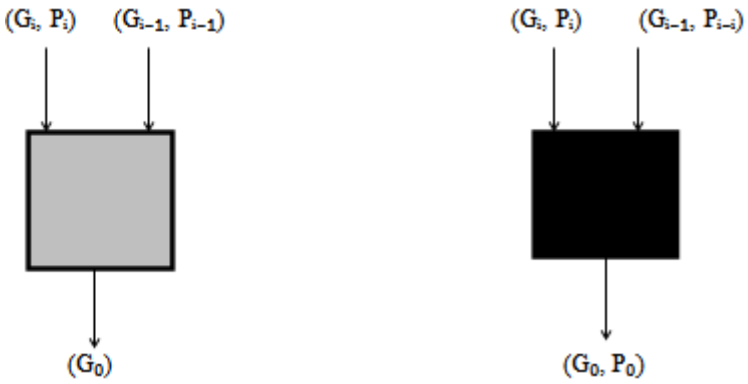


Fig1: steps of parallel prefix addition

As given in flow chart, parallel prefix addition can be done using three steps. In the very first step, we used to calculate two bit which are named as generate bit and propagate bit. In the next step we generate carry signal with the help of prefix tree. And in last step sum of two numbers can be done using some equations as described earlier.

II. OPERATIONS

In parallel prefix addition, there are mainly two operators used in addition. These are called as black and grey operators. Now if we look at the working of both the operators then we will come to the conclusion that black operator takes two bits called generate bit and propagate bit (G_i, P_i) and (G_{i-1}, P_{i-1}) and gives results as (G_0, P_0) which a single set of generate and propagate bit.



Black operator Grey operator

Fig2: operators

The equation it takes to perform the given task can be written as

$G_0 = G_i$ OR (P_i AND G_{i-1})

$P_0 = P_i$ AND P_{i-1}

At the same time the inputs to the grey operator are (G_i, P_i) and (G_{i-1}, P_{i-1}) which gives only one output (G_0). A black operator consists of two AND gates and a single OR gate and gives output as (G_0, P_0) but a grey operator consists of a single or and a single and gates and gives only one output G_0. Parallel prefix adder's main idea is drawn from carry look ahead adder. In large numbers of bit if we have to perform the addition then parallel prefix adder is most effective way of doing this. In three steps, it uses tree like structure. This differs from the carry look ahead adder in generating carries. In parallel prefix addition propagation of

carry is the most important part. Here we use basic carry operator which we have discussed earlier namely black and grey operators. We can design black operator with the help of multiplexer also.

III. ALGORITHM AND EQUATIONS

There are many different types of parallel prefix adders are made to increase for optimizing area, fan out, speed and performance. For high speed performance tree like structure is must which helps in greater way. The algorithm used in this method is known as Kogg-Stone method fastest in addition because it gives minimum fan out. Here delay in terms of power and generation of carries is also very less as compared to others. It is a type of carry look ahead adder which is in the form of prefix adder. Here carry is generated in log n times which are the minimum time taken among any others. Because of its performance, it is very useful in industries.

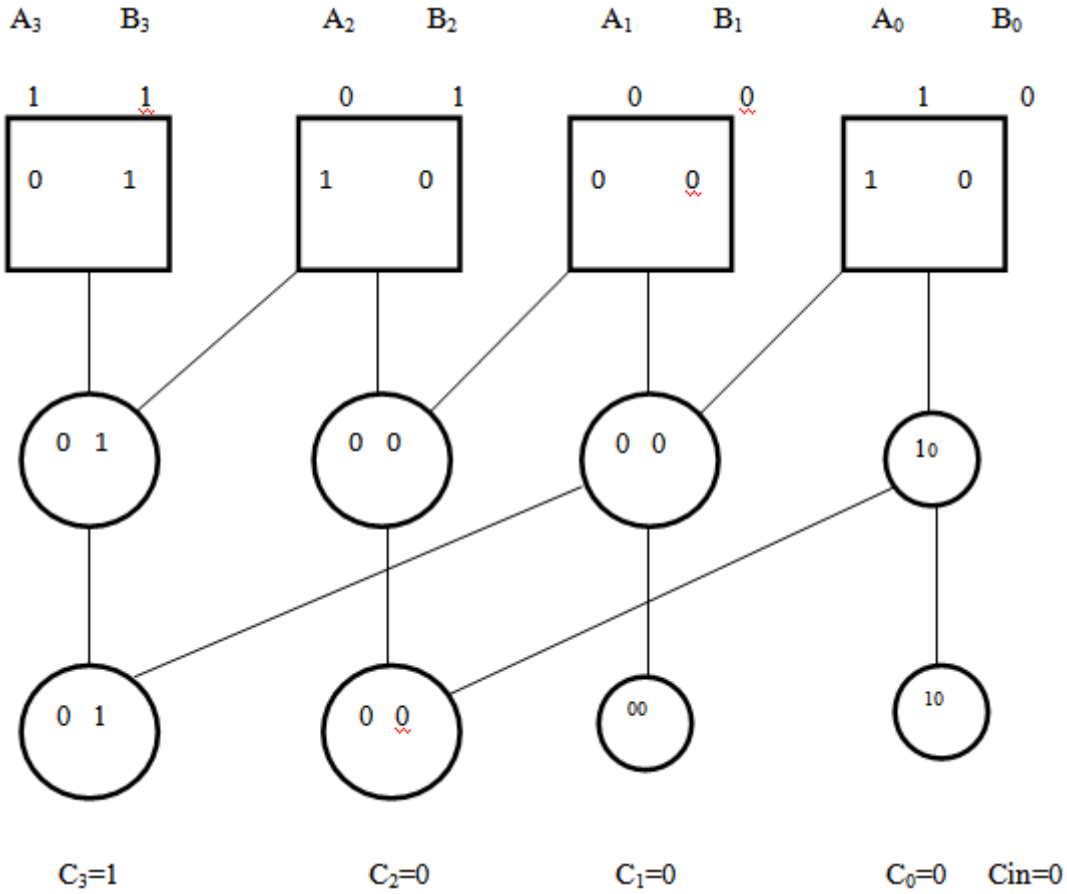


Fig3: Example of 4-bit Kogg-Stone adder

In this figure how kogg stone adder works is shown. There is a propagate bit and another is generate bit produced by every vertical stages. Carry bits are generated in last step. Sum bits are produced by xoring carry bit with bits which are initially propagated. Here carry is generated in n times and is the fastest available method for addition. we have to calculate propagate bit denoted by P as well as generate bit denoted by G. These bits are calculated by using some formula which is described as follows.

$P = Ai \text{ xor } Bi.$ $P = Pi \text{ and } Pi\ prev$

$G = Ai \text{ and } Bi.$ $G = (Pi \text{ and } Gi\ prev\ ) \text{or } Gi$

$Ci = Gi$

S

$Ki = Pi \text{ xor } Ci - 1$

The output provided by kogg stone method can be written as:

$KO = (A0 \text{ XOR } B0) \text{XOR } Cin$

$K1 = (A1 \text{ XOR } B1) \text{XOR } (A0 \text{ AND } B0)$

$K2 = (A2 \text{ XOR } B2) \text{XOR } \big((A1 \text{ XOR } B1) \text{AND } (A0 \text{ AND } B0) \text{OR} (A1 \text{ AND } B1)\big)$

K3= (A3 XOR B3) XOR (((A2 XOR B2) AND (A1 XOR B1) AND (A0 AND B0) OR ((((A2 XOR B2) AND (A1 AND B1) OR (A2 AND B2)))

K4= (((A3 XOR B3) AND (A2 XOR B2) AND (A1 XOR B1)) OR (((A3 XOR B3) AND (A2 AND B2) OR (A3 AND B3)))

IV. RESULTS AND COMPARISIONS

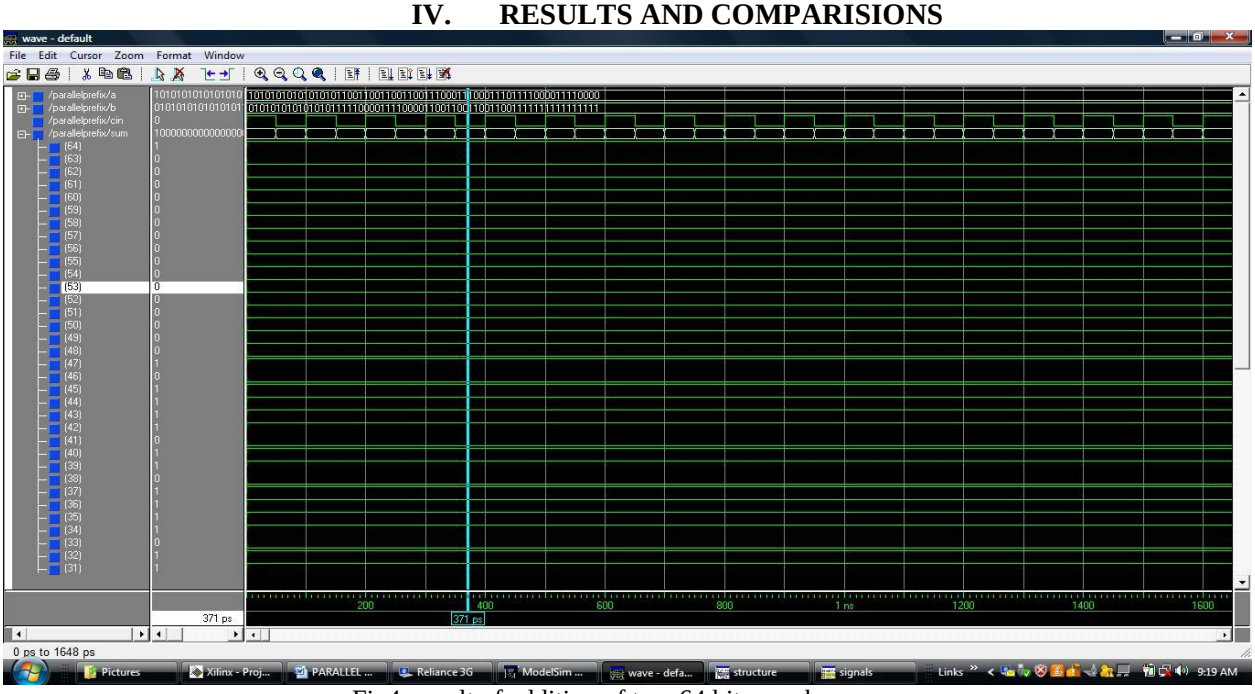


Fig4: result of addition of two 64 bits numbers

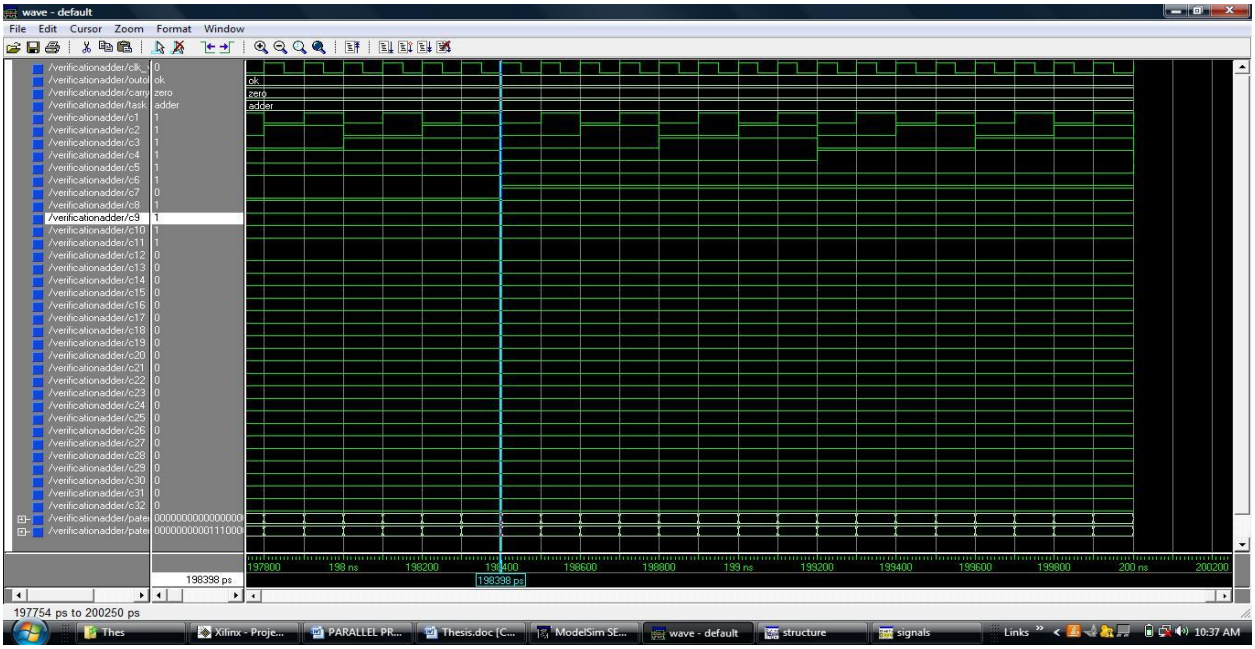


Fig5: results of automatic test form generation

This waveform is generated during the course of verifying the process. This method is used for auto generation of test pattern means here test pattern is generated itself which can be seen if we compare these two figures. In this method carry is generated and propagated itself so no need to generate carries as compared to others which improve in overall performances.

Transistor level simulator is used to determine depth and fan-out of different kind of mechanism used in parallel prefix adder.

NAME	DEPTH	FAN-OUT
RCA-64	63.3	3
SK-64	5.9	32
KS-64	5.6	2

Table1: Comparison between different adders

Adder	BK	SK	KS	HC	LF	KA
DELAY	15.34	13.43	12.32	16.76	13.54	12.37

Table2: Delay comparison

This is the delay comparison between different kind of method or algorithm used in parallel prefix addition. As we see from the table KS adder provides minimum delay hence is the fastest among all. The above delay results are compared by using Xilinx 9.1 software.

V. CONCLUSION AND FUTURE SCOPE

64 bit parallel prefix adder for high speed arithmetic circuit is designed successfully with the help of kogg stone algorithm. Xilinx ISE tool is used in synthesizing codes and for simulation purpose Model Sim is used. Binary adders are the basic modules in computer arithmetic design, and therefore it has been examined for a period of ten years. Entirely a few standard fast adders, such as the carry-skip adder, the carry-look-ahead adder and the carry-select adder were proposed in the past. Each of the fast adders presents a unique area-time tradeoff in the design space. But after comparing all these methods we can see that the parallel prefix adder used in addition is the fastest among all. Future scope includes in improving power dissipation since adder being the main element which presents in large part of the circuits so power is the major factor. It would be sufficient for future FPGA circuits in which optimized carry path is included which enable tree based designs for optimizing place and routing.

REFERENCES

[1] Jianhua LiuZhu, Haikun, Chung-Kuan Cheng, John Lillis, "Optimum prefix Adders in a Comprehensive Area, Timing and power Design Space"., Proceeding of the 2007 Asia and South pacific Design Automation conference. Washington.

[2] Kogge P, Stone H, "A parallel algorithm for the efficient solution of a general class Recurrence relations", IEEE Trans. Computers, vol.C-22, No.8, pp 786-793, Aug.1973.

[3] Giorgos Dimitrakopoulos and Dimitric Nikolos, "High Speed Parallel –Prefix VLSI Ling Adders", IEEE Trans on computers, Vol.54, No.2, Feb 2005.

[4] V.Choi and E.E.Swartz lander, Ir, "Parallel Prefix adder design with matrix representation",, in Proc.17th IEEE symp, comput.Arithmatic (ARITH).

[5] FPGAs David H. K. Hoe, Chris Martinez and Sri Jyothsna Vundavalli, "Design and characterization of Parallel Prefix Adders", IEEE 43rd Southeastern Symposium on system theory, March 2011.

[6] K. Vitoroulis and A. J. Al-Khalili, "Performance of Parallel Prefix Adders Implemented with FPGA technology," IEEE Northeast Workshop on Circuits and Systems, pp. 498-501, Aug. 2007.

[7] D. Harris, "A Taxonomy of Parallel Prefix Networks," in Proc. 37th Asilomar Conf. Signals Systems and Computers, pp. 2213–7, 2003.

[8] Skalansky "conditional sum additions logic" IRE Transactions, Electronic Computers.

[9] T. Lynch and E. E. Swartzlander, "A Spanning Tree Carry Lookahead Adder," IEEE Trans. on Computers, vol. 41, no. 8, pp. 931-939, Aug. 1992.

[10] R.Brent and H.Kung, ―A regular layout for parallel adders, IEEE.