Volume : 2 | Issue : 6 | June 2013 • ISSN No 2277 - 8179

# Design and Characterization of PPA and Comparing with RCA and CSA in EDA Tool

Engineering

**KEYWORDS :** Adders, Parallel Prefix Adders, Delay, Area.

| | |
|---|---|
| **Nitish Kumar** | National Institute of Technology, Patna-800006. |
| **Alok Ranjan** | National Institute of Technology, Patna-800006. |
| **(Dr.) Ramesh Kumar** | National Institute of Technology, Patna-800006. |

## ABSTRACT

*The binary adder is the critical element in most digital circuit designs including digital signal processors (DSP) and microprocessor data path units. As such, extensive research continues to be focused on improving the power, delay performance of the adder. Parallel-prefix adders (also known as carry-tree adders) are known to have the best performance in VLSI designs. This dissertation investigates four types of carry-tree adders (the Kogge-Stone, sparse Kogge-Stone, spanning tree adder and Brunt-Kung adders) and compares them to the simple Ripple Carry Adder (RCA) and Carry Skip Adder (CSA). These designs of varied bit-widths were implemented in a Xilinx and delay measurements were made with a high-performance logic analyzer. In this project for simulation we use Modelsim for logical verification, and further synthesizing it on Xilinx-ISE tool*

*Summary*

*Both measured and simulation results from the study have shown that parallel-prefix adders are not as effective as the simple ripple-carry adder at low to moderate bit widths. This is not unexpected as the Xilinx FPGA has a fast carry chain which optimizes the performance of the ripple carry adder. However, contrary to other studies, we have indications that the carry-tree adders eventually surpass the performance of the linear adder designs at high bit-widths. This is important for large adders used in precision arithmetic and cryptographic applications where the addition of numbers on the order of a thousand bits is not uncommon. Because the adder is often the critical element which determines to a large part the cycle time and power dissipation for many digital signal processing and cryptographical implementations, it would be worthwhile for future FPGA designs to include an optimized carry path to enable tree based adder designs to be optimized for place and routing.*

## I Introduction

The Binary adders are one of the most basic and widely used arithmetic operations in modern integrated circuits. They tend to play a critical role in determining the performance of the design. Arithmetic operations are the regular common operations in digital integrated circuits. The simplest circuit adds, subtracts, and multiplies or anything. The computation should be fast and the area consumed by the arithmetic units should be small. These are the two basic requirements for any adder. Area and Time consumed by the circuit are the basic and important requirements. Numbers can be represented in digital circuits in various ways. Hence, developing efficient adder architecture is crucial to improving the efficiency of the design. Generally ripple carry adder uses for binary addition. After the design of ripple carry adder several techniques are used for the computation of parallel adders. Carry look ahead adders are based on parallel prefix computation gives the better performance than ripple carry adder.As such, extensive research continues to be focused on improving the delay performance of the adder.

This paper investigates four types of carry-tree adders (the Kogge-Stone, sparse Kogge-Stone, spanning tree adder and Brunt-Kung adders) and compares them to the simple Ripple Carry Adder (RCA) and Carry Skip Adder (CSA) in EDA tool.

## II RIPPLE CARRY ADDER DESIGN

Ripple carry adder is a digital circuit it adds two binary numbers in parallel form. It contains of full adders connected in a chain. The output carry of each full adder is connected to the input carry of next full adder. But, computation of the carry-in signals at every bit is the most critical and time consuming process. Let a,b are the two input signals and ci is the corresponding carry-in signal to produce the sum(S) and carry(C) outputs.

$$s = a \text{ xor } b \text{ xor } ci \quad (1)$$

$$c = a \text{ and } b \text{ or } b \text{ and } ci \text{ or } a \text{ and } ci \quad (2)$$

Carry-look ahead Adder: In carry-look ahead adder, the focus is to design a circuit which can efficiently compute the (n-1) carry values. In every given bit positions generate (G) and propagate (P) signals are compute. The carry-Look ahead adder uses two special signals Generate (G) and Propagate (P) to predict the Propagation of a Carry signal without using a carry chain.

$$G = (a \text{ and } b) \quad (3)$$

$$P = (a \text{ xor } b) \quad (4)$$

## III CARRY-TREE ADDERS DESIGN

It is extends from the idea of carry-look ahead Computation's Classes of Parallel carry-look ahead schemes are formed targeting at high performance applications. These are also called recurrence solver adders. Different Parallel schemes are formed (a) kogge-stone adder (b) sparse Kogge-stone adder (c) spanning tree adder d) brent-kung adder. Parallel-prefix adders are the variation of well known carry-look ahead adders. The difference between CLA and PPA is for the generation of carry signals. In PPA prefix operator is introduced and it is used for the carry signal generation.

$$(gin1,1) \; o \; (gin2,2) = (g1+p1.g2,p1.p2) \quad (5)$$

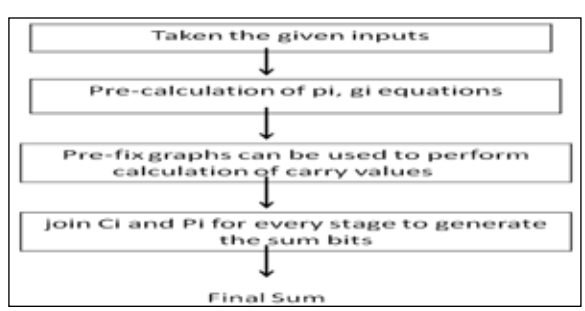


**Figure 1 Flow Chart of PPA**

In this equation is applied on two pairs of bits (gin1, pin1) and (gin2 and pin2).These pairs represent generate and propagate signals used in addition. The output of the operator is a new pair of bits generated in equation(5).With the use of this operator a prefix network is constructed and the second stage of the adder is implementation of this network. The structure of the Prefix network determines the type of prefix adder. These operators can be combined in different ways to form various adder structures. The key advantage of the tree structured adder is that the critical path due to the carry delay is on the order of log2N for an N-bit wide adder. The arrangement of the prefix network

gives rise to various families of adders. First the focus is on the Kogge-Stone adder has minimal logic depth and fan-out.

Kogge-stone adder design: Step1: First to generate propagation and generation signals for each bit. Step2: To generate black cell and gray cell equations. Step3: In each step gray cells are generated. Step4: By using gray cell equation to generate the carry bits directly. Step5: To combine propagation and carry bits for each step to generate sum. Here we designate BC as the black cell which generates the ordered pair in equation (5); the gray cell (GC) generates the left signal only. The interconnect area is known to be high.

Another carry-tree adder is known as the sparse kogge-stone adder. The sparse kogge-stone adder uses ripple carry and kogge-stone adder strutcture.This design terminates with 4-bit RCA and it uses less number of carry values in the method of KS structure. It allows a large adder to be composed of small adders by generating intermediate carries quickly. Power and area of the carry generation is improved and routing congestion is substantially reduced but delay is increase compare to kogge-stone adder. This adder gives intermediate results between ripple carry adder and kogge-stone adder. Third type of carry-tree adder is known as the spanning tree adder. This design also terminates with 4-bit RCA .It uses minimum no of multi input gates. It gives better results compared to Sparse Kogge-stone adder.

**A. Use of Simulation software**

These designs of varied bit-widths were implemented in a Xilinx 12.1 and Modelsim 6.4b and delay measurements were made with a high-performance logic analyzer.

Kogge-stone prefix tree adder

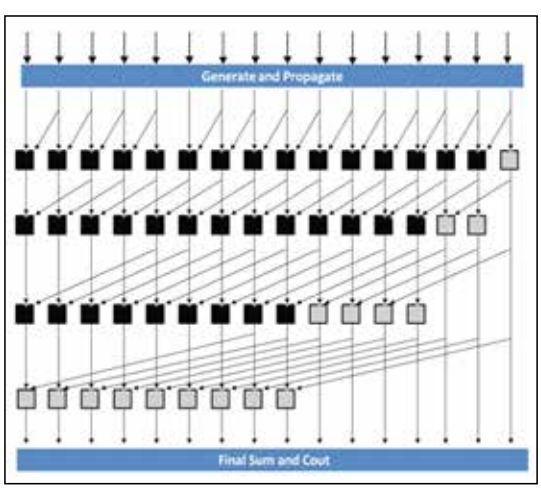


**Figure 2 Kogge-stone prefix tree adder**

Sparse Kogge-Stone adder

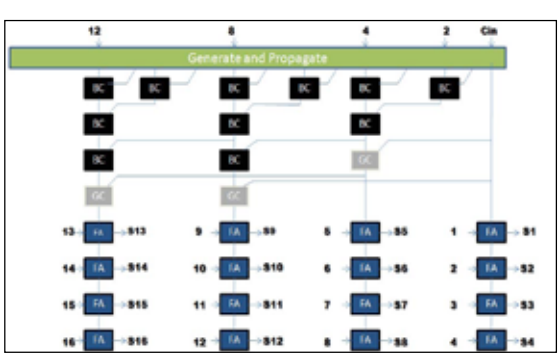


**Figure 3 Sparse Kogge-Stone adder**

Spanning Tree Adder Carry Look ahead Adder

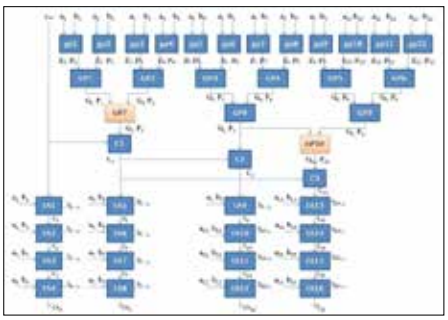

**Figure 4 Spanning Tree Adder Carry Look ahead Adder**

Brent-Kung Adder

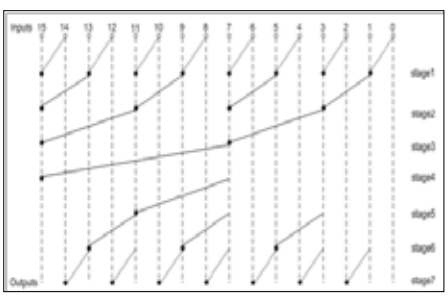

**Figure 5 Brent-Kung Adder**

**IV ADDER SELECTIONS**

Ripple carry adder Carry-look ahead adder Kogge-stone adder Sparse kogge-stone adder Spanning tree adder Brent-kung adder These adders were selected and these are implemented with bit sizes of 8, 16, 32 bits. This variety of sizes gives the performance of adders with area and delay values. The performance of these adders are compared with ,delay, and no of look up tables.

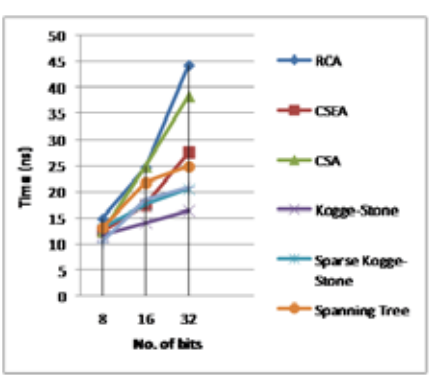


**Figure 6 Time delay vs Bit-Width**

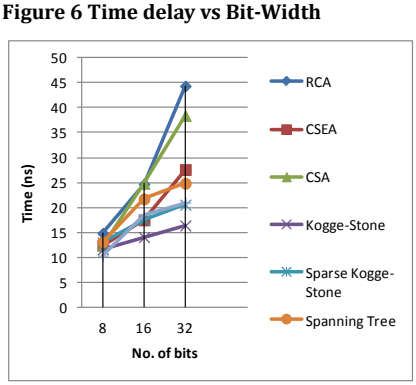


**Figure 7 Number of LUT vs Bit-Width**

**V. CONCLUSION**

Both measured and simulation results from this study have shown that parallel-prefix adders are not as effective as the simple ripple-carry adder at low to moderate bit widths. This is not unexpected as the Xilinx FPGA has a fast carry chain which optimizes the performance of the ripple carry adder. However, contrary to other studies, we have indications that the carry-tree adders eventually surpass the performance of the linear adder designs at high bit-widths. This is important for large adders used in precision arithmetic and cryptographic applications where the addition of numbers on the order of a thousand bits is not uncommon. Because the adder is often the critical element which determines to a large part the cycle time and power dissipation for many digital signal processing and cryptographical implementations, it would be worthwhile for future FPGA designs to include an optimized carry path to enable tree based adder designs to be optimized for place and routing.

## REFERENCE

[1] N.H.E.Weste and D.Harris, CMOS VLSI Design, 4thedition, Pearson–Addison-Wesley, 2011. | [2] K. Vitoroulis and A. J. Al-Khalili, "Performance of Parallel Prefix Adders Implemented with Technology," IEEE Northeast Workshop on Circuits and Systems, pp. 498-501, Aug. 2007. | [3] P. Ndai, S. Lu, D. Somesekhar, and K. Roy, "Fine-Grained Redundancy in Adders," Int. Symp. On Quality Electronic Design, pp. 317-321, March 2007. | [4]M.Bečvář and P.Štukjunger, "Fixed-Point Arithmetic in FPGA," Acta Polytechnic, vol. 45, no. 2, pp. 67-72, 2005 | [5] D. Gizopoulos, M. Psarakis, A. Paschalis, and Y.Zorian, "Easily Testable Cellular Carry Look ahead Adders," Journal of Electronic Testing: Theory and Applications 19, 285-298, 2003. [6] D. Harris, "A Taxonomy of Parallel Prefix Networks," in Proc. 37th Asilomar Conf. Signals Systems and Computers, pp. 2213–2217, 2003. | [7] S. Xing and W. W. H. Yu, "FPGA Adders: Performance Evaluation and Optimal Design," IEEE Design & Test of Computers, vol. 15, no. 1, pp. 24-29,Jan. 1998. | [8] T. Lynch and E. E. Swartzlander, "A Spanning Tree Carry Look ahead Adder," IEEE Trans. on Computers,vol. 41, no. 8, pp. 931-939, Aug. 1992. | [9] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," IEEE Trans. Comput., vol. C-31, pp.260-264, 1982. | [10] P. M. Kogge and H. S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations," IEEE Trans. on Computers, Vol. C-22, No 8, August 1973. |