International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 14
October2016

# Parallel-Prefix Adders Implementation Using Reverse Converter Design

Submitted by: M.SHASHIDHAR

Guide name: J.PUSHPARANI, M.TECH

## Department of ECE

**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. In VLSI implementations, parallel-prefix adders are known to have the best performance. Parallel prefix adder is the most flexible and widely used for binary addition. Parallel Prefix adders are best suited for VLSI implementation. Numbers of parallel prefix adder structures have been proposed over the past years intended to optimize area, fan-out, and logic depth and inter connect count. This paper presents a new approach to redesign the basic operators used in parallel prefix architectures. The number of multiplexers contained in each Slice of an FPGA is considered here for the redesign of the basic operators used in parallel prefix tree.

Parallel-prefix adders (also known as carry-tree adders) are known to have the best performance in VLSI designs. However, this performance advantage does not translate directly into FPGA implementations due to constraints on logic block configurations and routing overhead. This paper investigates three types of carry-tree adders (the Kogge-Stone, sparse Kogge-Stone, and spanning tree adder) and compares them to the simple Ripple Carry

Adder (RCA) and Carry Skip Adder (CSA). These designs of varied bit-widths were implemented on a Xilinx Spartan 3E FPGA and delay measurements were made with a high-performance logic analyzer. Due to the presence of a fast carry-chain, the RCA designs exhibit better delay performance up to 128 bits. This new design is implemented with 16-bit width operands of various parallel prefix adders on Xilinx Spartan FPGA. The experimental results indicate that the new approach of basic operators make some of the parallel prefix adders architectures faster and area efficient

In this project for simulation we use Modelsim for logical verification, and further synthesizing it on Xilinx-ISE tool using target technology and performing placing & routing operation for system verification on targeted FPGA.

**1. INTRODUCTION**

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the rapid advances in integration technologies, large-scale systems design - in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing,

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 14
October2016

telecommunications, and consumer electronics has been rising steadily, and at a very fast pace. The driving force for the fast development of this field.

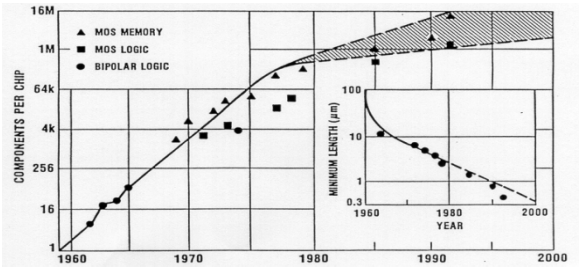


**Fig:** Evolution Of Integration Density And Minimum Feature Size

Therefore, the current trend of integration will also continue in the foreseeable future. Advances in device manufacturing technology, and especially the steady reduction of minimum feature size (minimum length of a transistor or an interconnect realizable on chip) support this trend.

### 1.1 VLSI Design Flow

The design process, at various levels, is usually evolutionary in nature. It starts with a given set of requirements. Initial design is developed and tested against the requirements. When requirements are not met, the design has to be improved. If such improvement is either not possible or too costly, then the revision of requirements and its impact analysis must be considered.

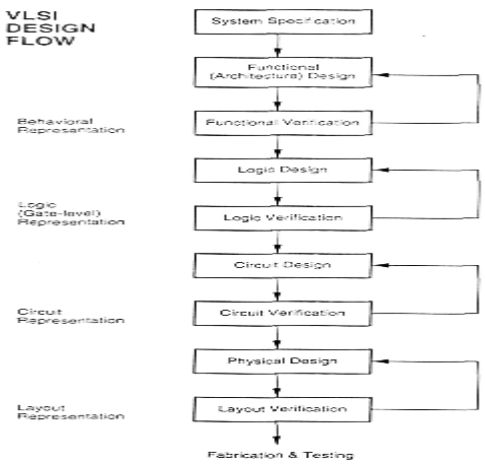


**Fig:** A more simplified view of VLSI design flow

### 1.2 Design Hierarchy

A hierarchy structure can be described in each domain separately. However, it is important for the simplicity of design that the hierarchies in different domains can be mapped into each other easily. In the physical domain, partitioning a complex system into its various functional blocks will provide a valuable guidance for the actual realization of these blocks on chip. Obviously, the approximate shape and size (area) of each sub-module should be estimated in order to provide a useful floor plan

### 1.3 Low power Vlsi

As technology advanced, chips grew, and it was possible to integrate more functions into one chip. Just as for TTL, newer technology, called CMOS, threatened to replace NMOS in the 1980s because CMOS proved to consume even less power. With further advances InTechnology and fabrication, the integration densities and the rate at which chips operate have increased drastically, causing power consumption to be of primary concern. In addition, the new requirements set by device portability, reliability, and costs have helped in alleviating the power

consumption threat in CMOS circuits. Because the power problem is getting more concerning, very large-scale integrated circuit (VLSI) designers need to develop new efficient techniques to reduce the power dissipation in current and future technologies, a task that is full of challenges but yet exciting to explore. Ideally, in the steady state of CMOS circuits there is no static power consumption, which is the most attractive characteristic of CMOS technology.

## 2. INTRODUCTION TO ADDERS

In electronics, an adder or summer is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic unit(s), but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar. Although adders can be constructed for many numerical representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers.The **half adder** adds two one-bit binary numbers A and B. It has two outputs, S and C (the value theoretically carried on to the next addition); the final sum is 2C + S. The simplest half-adder design, pictured on the right, incorporates an XOR gate for S and an AND gate for C. With the addition of an OR gate to combine their carry outputs, two half adders can be combined to make a full adder

A **full adder** adds binary numbers and accounts for values carried in as well as out. A one-bit full adder adds three one-bit numbers, often written as $A$, $B$, and $C_{in}$; $A$ and $B$ are the operands, and $C_{in}$ is a bit carried in from the next less significant stage. The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. binary numbers

A **carry-look ahead adder** (CLA) is a type of adder used in digital logic. A carry-look ahead adder improves speed by reducing the amount of time required to determine carry bits. It can be contrasted with the simpler, but usually slower, *ripple carry adder* for which the carry bit is calculated alongside the sum bit, and each bit must wait until the previous carry has been calculated to begin calculating its own result and carry bits (see adder for detail on ripple carry adders). The carry-look ahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits. The Kogge-Stone adder and Brent-Kung adder are examples of this type of adder.

A **carry-save adder** is a type of digital adder, used in computer micro architecture to compute the sum of three or more $n$-bit numbers in binary. It differs from other digital adders in that it outputs two numbers of the same dimensions as the inputs, one which is a sequence of partial sum bits and another which is a sequence of carry bits.

In electronics, a **carry-select adder** is a particular way to implement an adder, which is a logic element that computes the $(n+1)$-bit sum of two $n$-bit numbers. The carry-select adder is simple but rather fast, having a gate level depth of $O(\sqrt{n})$.

A 16-bit carry-select adder with a uniform block size of 4 can be created with three of these blocks and a 4-bit ripple carry adder. Since carry-in is

known at the beginning of computation, a carry select block is not needed for the first four bits. The delay of this adder will be four full adder delays, plus three MUX delays.

A 16-bit carry-select adder with variable size can be similarly created. Here we show an adder with block sizes of 2-2-3-4-5. This break-up is ideal when the full-adder delay is equal to the MUX delay, the total delay is two full adder delays, and four mux delays.

## 3. PARALLEL PREFIX ADDERS

### 3.1 Introduction

Computer arithmetic is a wide topic. It spans from the selection of a suitable number system to the implementation of a complete arithmetic-logic unit (ALU). This chapter will, however, focus on only three topics: single-bit adders, wide-bit adders, and multipliers. Those circuits are crucial for the performance of many components such as digital signal processors (DSPs) and general purpose processors. Fused multiply add unit plays an important role in modern microprocessor. It performs floating point multiplication followed by an addition of the product with a third floating point operand. In 2007, a seven cycle fused multiply add pipeline unit was proposed as a part of the floating point unit in IBM's POWER6 microprocessor. In this fused multiply add dataflow, the product should be aligned before it is added with the addend. Because the magnitude of the product is unknown in the early stages prior to the combination with the addend it is difficult to determine a priori which operand is bigger.

**Parallel Prefix Adders**

A parallel prefix circuit is a combinational circuit with n inputs x1, x2,xn producing the outputs x1,x20x1, :::; xn xn□1:::x1 where is the associatively binary operation. The first stage of the adder generates individual P and G signals. The remaining stages constitute the parallel prefix circuit with the fundamental carry operation serving as the associative binary operation. This part of the adder can be designed in many different ways.

Although computing carry-propagate addition can use generate and propagate signals, its implementation in VLSI can be quite inefficient due to the number of wires that have to be connected together. Parallel-prefix adders solve this problem by making the wires shorter with simple gate structures to aid in the passing of groups of carries to the next weight

### 3.2 Tree or Parallel-Prefix Adders

To describe parallel-prefix adders, first of all the parallel-prefix or dot operator needs to be defined:

(GI, PI'). (GX,PX') = (GI + PI'GX,PI'PX')

Here the Ps and Gs can be either single or group propagate and generate signals. The Ps and Gs with index X are from a lower significance level than those with index I. In state-of-the-art designs conventional domino and static CMOS logic are used to implement the dot-operator cells. Alternative and more efficient implementations of the dot-operator cells are presented in Paper 6 where a dynamic pass-transistor logic style is employed. The new cells dissipate less power and use fewer transistors than their corresponding domino implementations and the speed penalty is small to none.

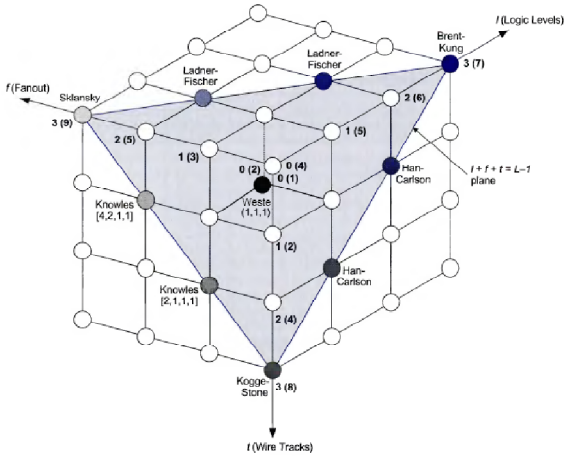


Fig: Taxonomy of prefix networks

## 4. INTRODUCTION TO VERILOG

Inthe semiconductor and electronic design industry, **Verilog** is a hardware description language(HDL) used to model electronic systems. Verilog HDL, not to be confused with VHDL (a competing language), is most commonly used in the design, verification, and implementation of digital logic chips at the register-transfer level of abstraction. It is also used in the verification of analog and mixed-signal circuits.

A Verilog design consists of a hierarchy of modules. Modules encapsulate *design hierarchy*, and communicate with other modules through a set of declared input, output, and bidirectional ports. Internally, a module can contain any combination of the following: net/variable declarations, concurrent and sequential statement blocks, and instances of other modules (sub-hierarchies). Sequential statements are placed inside a begin/end block and executed in sequential order within the block. But the blocks themselves are executed concurrently, qualifying Verilog as a dataflow language. A subset of statements in the Verilog language is synthesizable. Verilog modules that conform to a synthesizable coding style, known as RTL (register-transfer level), can be physically realized by synthesis software. Synthesis software algorithmically transforms the (abstract) Verilog source into a net list, a logically equivalent description consisting only of elementary logic primitives (AND, OR, NOT, flip-flops, etc.) that are available in a specific FPGA or VLSI technology. Further manipulations to the net list ultimately lead to a circuit fabrication blueprint

### Simulation Results

The below figures shows the simulation results of test cases applied to the DUT . figure 6.1shows the response of the device for the control test case at the usb interface.

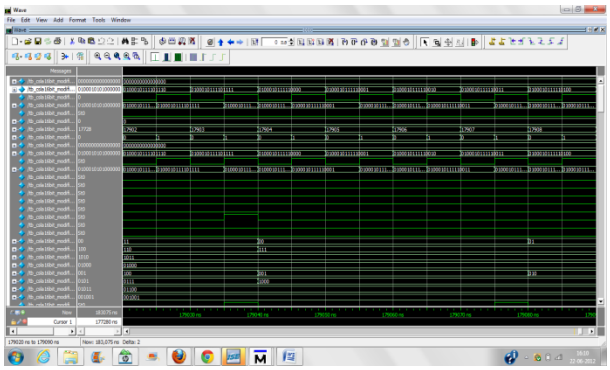

Fig: shows the master transmitter sending random data to the external slave device

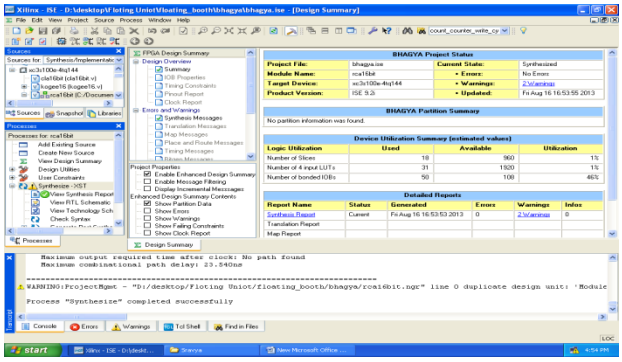

Fig: 16 -Bit Ripple Carry Adder

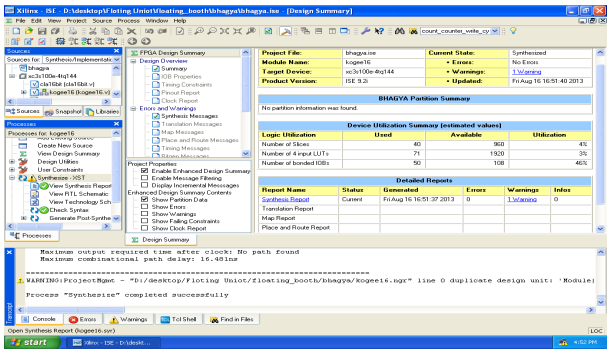

**Fig: 16 -Bit Kogge_Stone Adder**

## CONCLUSION

In this project I designed the ripple carry adder,Kogge stone adder, brent kung adder for 8,16,32,64,128 bits.And I found out the delay and area of the designed adders. As per the results I conclude that for higher order applications parallel pefix adders are best choice. In brent kung adder has more than 200% speed as compared to the RCA,but we required very less amount of delay. If we go for the Kogge stone it has a more than 600% speed as compared to the RCA. That's why for high speed applications we choose kooge stone adders. The 128 bit kooge stone adder delay is almost equal to the 16 bit RCA. The above all designs are done by the Verilog HDl , the simulation is done by model sim, and for synthesis we go for the Xilinx ISE.

## Future Work:

In Kogge stone it has high speed but it requires more amount of area. To overcome that problem we go for the sparse Kogge stone. In practically there is no difference between the both adders. For testing of the design we use the Verilog Hdl, If we go for the advanced verification methods we can reduce the verification of designs.

## REFERENCES

[1] A. Omondi and B. Premkumar, Residue Number Systems: Theory and Implementations. London, U.K.: Imperial College Press, 2007.

[2] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed., New York, NY, USA: Oxford Univ. Press, 2010.

[3] J. Chen and J. Hu, "Energy-efficient digital signal processing via voltage over scaling-based residue number system," IEEE Trans. Very Large Scale Integer. (VLSI) Syst., vol. 21, no. 7,

[4] C. H. Vun, A. B. Premkumar, and W. Zhang, "A new RNS based DA approach for inner product computation," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 60, no. 8, pp. 2139–2152, Aug. 2013.

[5] S. Antão and L. Sousa, "The CRNS framework and its application to programmable and reconfigurable cryptography," ACM Trans. Archit. Code Optim., vol. 9, no. 4, p. 33, Jan. 2013.

[6] A. S. Molahosseini, S. Sorouri, and A. A. E. Zarandi, "Research challenges in next-generation residue number system architectures," in Proc. IEEE Int. Conf. Comput. Sci. Educ., Jul. 2012, pp. 1658–1661.

[7] K. Navi, A. S. Molahosseini, and M. Esmaeildoust, "How to teach residue number system

to computer scientists and engineers," IEEE Trans. Educ., vol. 54, no. 1, pp. 156–163, Feb. 2011.

[8] Y. Wang, X. Song, M. Aboulhamid, and H. Shen, "Adder based residue to binary numbers converters for (2n − 1, 2n, 2n + 1)," IEEE Trans. Signal Process., vol. 50, no. 7, pp. 1772–1779, Jul. 2002.

[9] B. Cao, C. H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set {2n − 1, 2n, 2n + 1, 22n + 1} based on the new Chinese remainder theorem," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 50, no. 10, pp. 1296–13