RESEARCH ARTICLE OPEN ACCESS

Design and Estimation of delay, power and area for Parallel prefix adders

Avinash Shrivastava *, Shefali Churhe**, Hemlata Bhagat***, Rajni Wamankar****

**(Department of Computer Science, XYZ University, Coimbatore-46*
Email: abcdef@gmail.com)
*** (Department of Information Technology, ABC University, USA*
Email: abcdef@yahoo.co.uk)

ABSTRACT
In Very Large Scale Integration (VLSI) designs, Parallel prefix adders (PPA) have the better delay performance. This paper investigates four types of PPA's (Kogge Stone Adder (KSA), Spanning Tree Adder (STA), Brent Kung Adder (BKA) and Sparse Kogge Stone Adder (SKA)). Additionally Ripple Carry Adder (RCA), Carry Look ahead Adder (CLA) and Carry Skip Adder (CSA) are also investigated. These adders are implemented in verilog Hardware Description Language (HDL) using Xilinx Integrated Software Environment (ISE) 13.2 Design Suite. These designs are implemented in Xilinx Spartan 6 Field Programmable Gate Arrays (FPGA). Delay and area are measured using XPower analyzer and all these adder's delay, power and area are investigated and compared finally.

Keywords: *parallel prefix adders; carry tree adders; FPGA, delay; power.*

I. INTRODUCTION

The binary addition is the basic arithmetic operation in digital circuits and it became essential in most of the digital systems including Arithmetic and Logic Unit (ALU), microprocessors and Digital Signal Processing (DSP). At present, the research continues on increasing the adder's delay performance. In many practical applications like mobile and telecommunications, the speed and power performance improved in FPGAs is better than microprocessor and DSP's based solutions. Additionally, power is also an important aspect in growing trend of mobile electronics, which makes large-scale use of DSP functions. Because of the Programmability, structure of configurable logic blocks (CLB) and programming interconnects in FPGAs, Parallel prefix adders have better performance. The delays of the adders are discussed [1]. In this paper, above mentioned PPA's and RCA and CLA KSA & are implemented and characterized on a Xilinx Spartan 6 FPGA. Finally, delay, power and area for the designed adders are presented and compared.

II. DRAWBACKS OF RIPPLE CARRY AND CARRY LOOK AHEAD ADDER

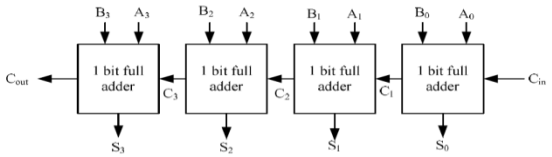

Fig.2.1 4 bit ripple carry adder

In order to reduce the delay in RCA (or) to propagate the carry in advance, we go for carry look ahead adder .Basically this adder works on two operations called propagate and generate The propagate and generate equations are given by

$$P_i = Ai \oplus Bi \quad (1)$$

$$G_i = A_i . B_i \quad (2)$$

For 4 bit CLA, the propagated carry equations are given as

$$C_1 = G_0 + P_0 C_0 \quad (3)$$

$$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0 \quad (4)$$

$$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0 \quad (5)$$

$$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0 \quad (6)$$

Equations (3), (4), (5) and (6) are observed that, the carry complexity increases by increasing the adder bit width. So designing higher bit CLA becomes complexity. In this way, for the higher bit of CLA's, the carry complexity increases by increasing the width of the adder. So results in bounded fan-in rather than unbounded fan-in, when designing wide width adders. In order to compute the carries in advance without delay and complexity, there is a concept called Parallel prefix approach.

KOGGE STONE ADDER

The Kogge–Stone adder concept was developed by Peter M. Kogge and Harold S.Stone

[1]. The Kogge–Stone adder is a parallel prefix form carry look-ahead adder. It generates the carry signals in O (log n) time, and is widely considered the fastest adder design possible. It is the common design for high-performance adders in industry. An example of a 4-bit Kogge–Stone adder is shown in Fig 3.Each vertical stage produces a "propagate" and a "generate" bit, as shown [1]. The final generate bits which are the carries are produced in the last stage (vertically), and these bits are xor'd with the initial propagate after the input to produce the sum bits [7]-[8]. An example can be analyzed with the first sum bit is calculated by xoring the propagate in the extreme bit (a "1") with the carry-in (a "0"), producing a "1". The second bit is calculated by xoring the propagate in second bit from the MSB (a "0") with first carry out (a "0"), producing a"0" [9]-[10].

HAN CARLSON ADDER

The idea of Han-Carlson prefix tree is similar to Kogge-Stone's structure since it has a maximum fan-out of 2 or f = 0. The difference is that Han-Carlson prefix tree uses much less cells and wire tracks than Kogge-Stone. The cost is one extra logic level. Han-Carlson prefix tree can be viewed as a sparse version of Kogge-Stone prefix tree. In fact, the fan-out at all logic levels is the same (i.e. 2). The pseudo-code for

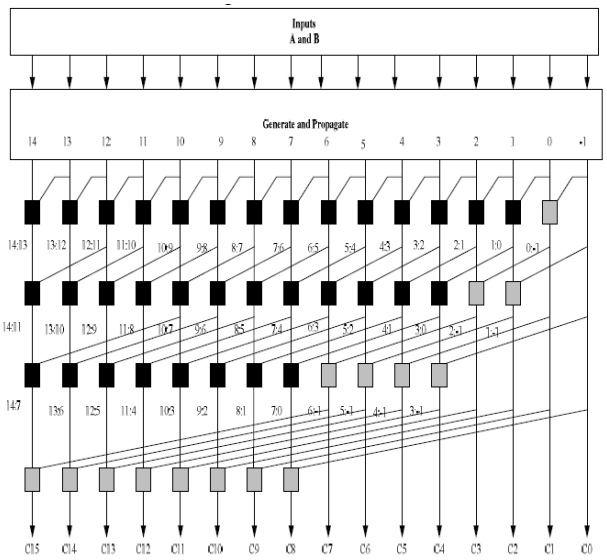

Fig 3.1 16 - bit Kogge Stone Adder

Kogge-Stone's structure can be easily modified to build a Han-Carlson prefix tree. The major difference is that in each logic level, Han-Carlson prefix tree places cells every other bit and the last logic level accounts for the missing carries. Figure 4.3.9 shows a 16-bit Han-Carlson prefix tree, ignoring the buffers. The critical path is shown with thick solid. This type of Han-Carlson prefix tree has $\log_2 n + 1$ logic levels. It happens to have the same number cells as Sklansky prefix tree since the cells in the extra logic level can be move up to make the each of the previous logic levels all have n=2 cells. The area is estimated as (n=2) $\log_2 n$. When n = 16, the number is 32.

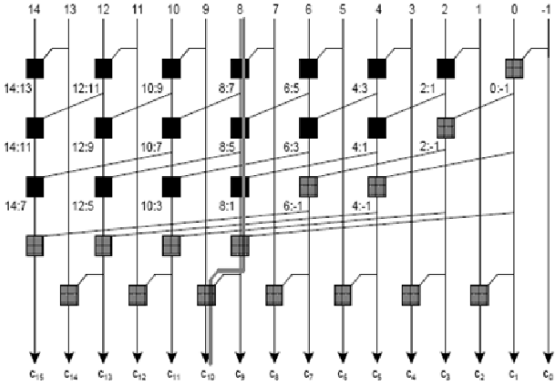

Fig 4.1:16 bit Han Carlson Prefix Tree

III. PROBLEM IDENTIFICATION

The major problem for binary addition is the carry chain [4]. As the width of the input operand increases, the length of the carry chain increases. Figure 1.1 demonstrates an example of an 8-bit binary adder operation and how the carry chain is affected. This example shows that the worst case occurs when the carry travels the longest possible path, from the least significant bit (LSB) to the most significant bit (MSB) [4]. In order to improve the performance of carry-propagate adders, it is possible to accelerate the carry chain, but not eliminate it. Consequently, most digital designers often resort to building faster adders when optimizing computer architecture, because they tend to set the critical path for most computations.

PROBLEM STATEMENT

1. Carry Adder; Carry look-Ahead Adder and Carry Select Adder. Ripple Carry Adder (RCA) shows the compact design but their computation time is longer.
2. Time critical applications make use of Carry Look-Ahead Adder (CLA) to derive fast results but it leads to increase in area. But the carry select adder provides a compromise between the small areas but longer delay of RCA and large area with small delay of Carry Look Ahead adder.
3. The carry-look ahead adder calculates one or more carry bits before the sum calculates, Due to this reduces the delay time to calculate the result of the larger number of value bits but Carry Look Adder is not suitable in constant delay for wider bit adder
4. Carry Look Adder has substantial loading capacitance and large delay and large power consumption.

VI. DIFFERENCE BETWEEN PARALLEL- PREFIX ADDERS AND OTHERS

The PPA's pre-computes generate and propagate signals are presented in [2]. Using the fundamental carry operator (f_{co}), these computed signals are combined in [3].The fundamental carry operator is denoted by the symbol "***O***",

$$(g_L, p_L)\ O\ (g_R, p_R) = (g_L + p_L \cdot g_R, p_L, p_R) \quad (7)$$

For example, 4 bit CLA carry equation is given by

$$C_4 = (g_4, p_4) o\ [(g_3, p_3) o\ [(g_4, p_4) o\ (g_3, p_3)]] \quad (8)$$

For example, 4 bit PPA carry equation is given by

$$C_4 = [(g_4, p_4) o\ (g_3, p_3)] o\ [(g_4, p_4) o\ (g_3, p_3)] \quad (9)$$

Equations (8) and (9) are observed that, the carry look ahead adder takes 3 steps to generate the carry, but the bit PPA takes 2 steps to generate the carry.

VII. PARALLEL-PREFIX ADDER STRUCTURE

Parallel-prefix structures are found to be common in high performance adders because of the delay is logarithmically proportional to the adder width [2].
PPA's basically consists of 3 stages

- Pre computation
- Prefix stage
- Final computation

The Parallel-Prefix Structure is shown in figure 7.1.

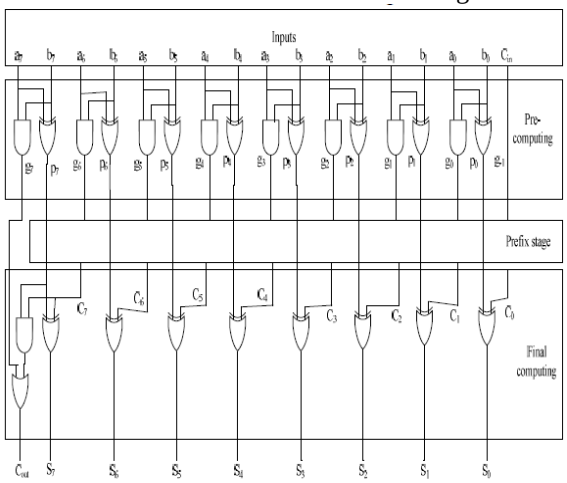

Fig 7.1 Parallel prefix structure with carry save notation

A. Pre computation

In pre computation stage, propagates and generates are Computed for the given inputs using the given equations (1) and (2).

B. Prefix stage

In the prefix stage, group generate/propagate signals are computed at each bit using the given equations. The black cell (BC) generates the ordered pair in equation (7), the gray cell (GC) generates only left signal, following [2].

$$G_{i:k} = G_{i:j} + P_{i:j} \cdot G_{j-1:k} \quad (10)$$

$$P_{i:k} = P_{i:j} \cdot P_{j-1:k} \quad (11)$$

More practically, the equations (10) and (11) can be expressed using a symbol "o "denoted by Brent and Kung. Its function is exactly the same as that of a black cell i.e.

$$G_{i:k} : P_{i:k} = (G_{i:j} : P_{i:j}) O\ (G_{j-1:k} : P_{j-1:k}) \quad (12)$$

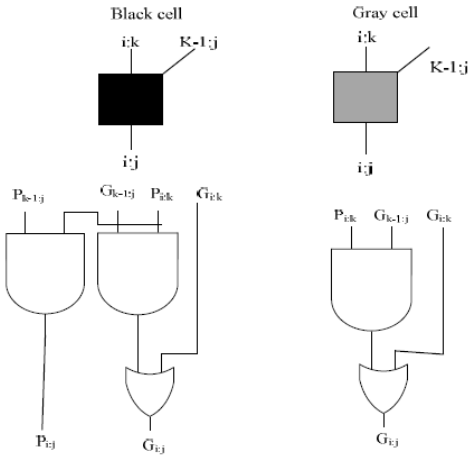

Fig.7.2 black and gray cells with logic definitions

The "o" operation will help make the rules of building prefix structures.

C. Final computation

In the final computation, the sum and carryout are the final output.

$$S_i = P_i \cdot G_{i-1:-1} \quad (13)$$

$$C_{out} = G_{n:-1} \quad (14)$$

Where "-1" is the position of carry-input. The generate/propagate signals can be grouped in different fashion to get the same correct carries. Based on different ways of grouping the generate/propagate signals, different prefix architectures can be created. Figure 3 shows the definitions of cells that are used in prefix structures, including BC and GC. For analysis of various parallel prefix structures, see [2], [3] & [4].

VIII. METHODOLOGY

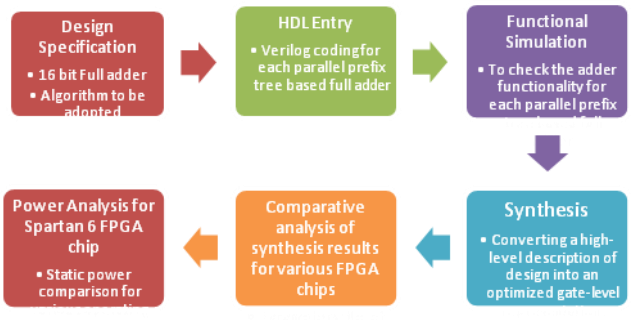

Fig. 8.1 Design Methodology

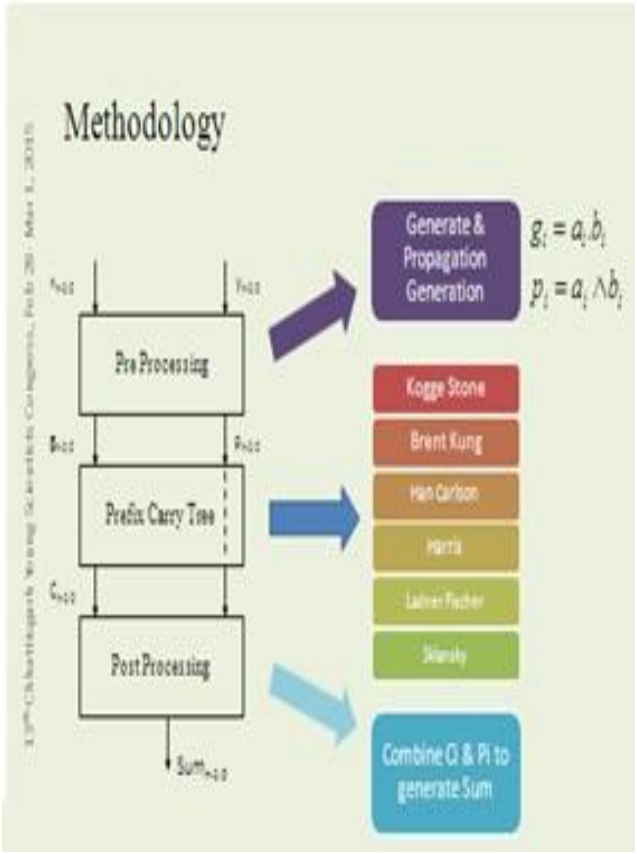

Fig.8.2 Addition Procedure using Parallel Prefix Structure

IX. RESULT

RIPPLE CARRY ADDER

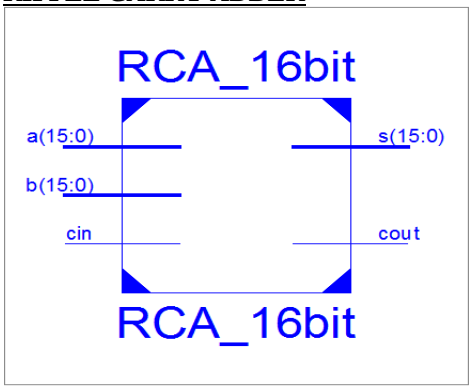

Fig.9.1 RTL View of Ripple Carry Adder

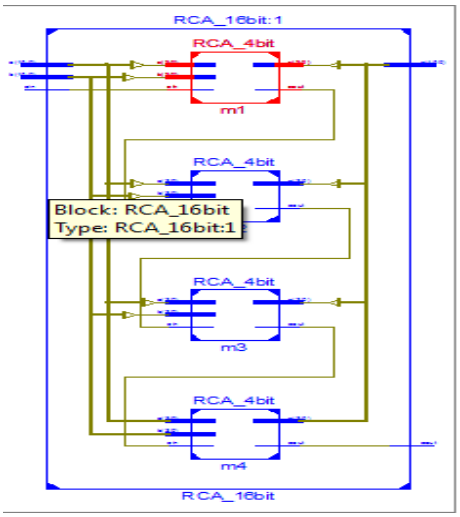

Fig 9.2 Logic Diagram of Ripple Carry Adder

KOGGE STONE ADDER

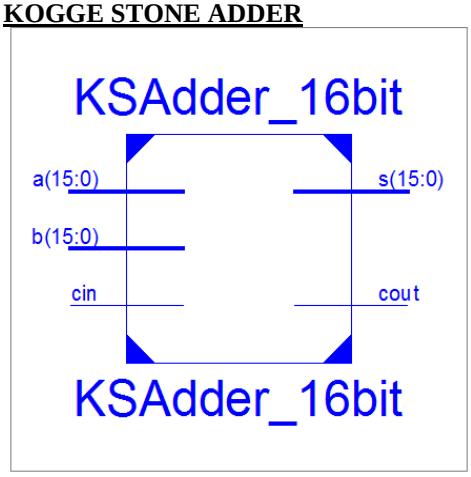

Fig.9.3 RTL View of Kogge Stone Adder

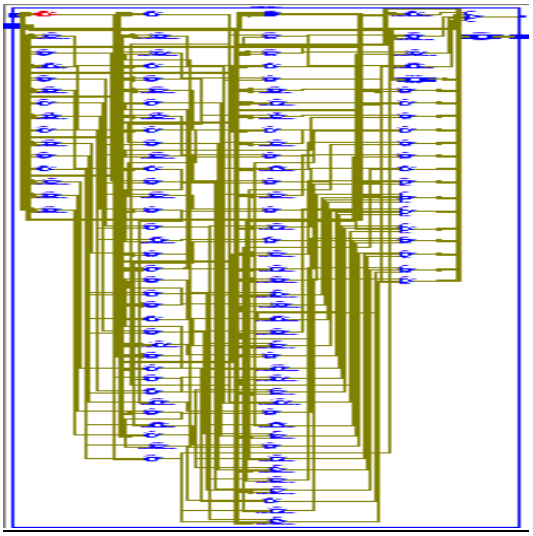

Fig 9.4 Logic Diagram of Kogge Stone Adder

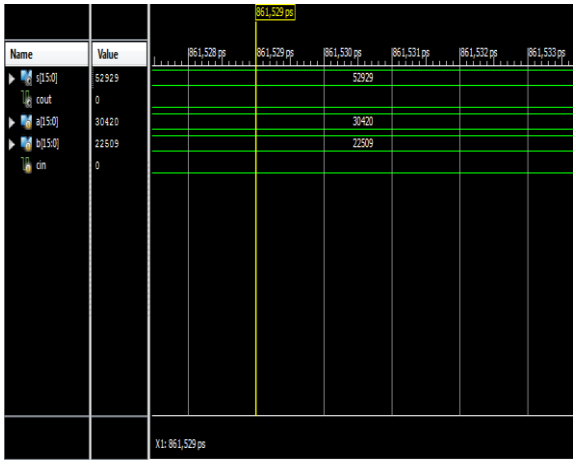

Fig. 9.5 Simulation Result of Ripple Carry Adder

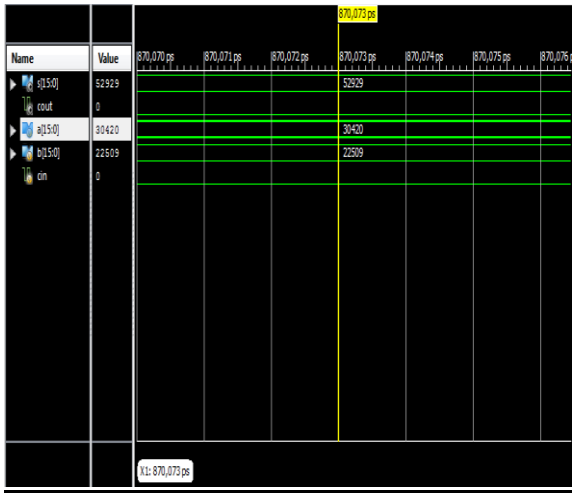

Fig.9.8 Simulation Result of Kogge Stone Adder

HAN CARLSON ADDER

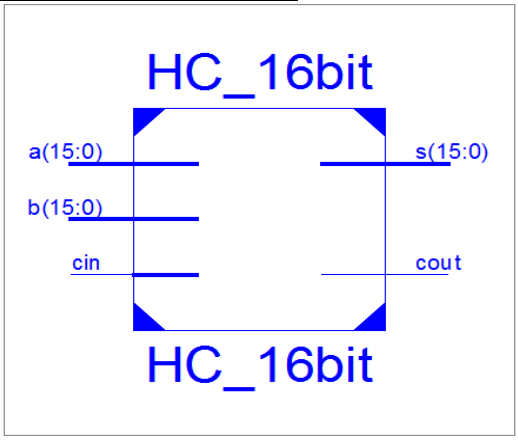

Fig 9.6 RTL View of Han Carlson Adder

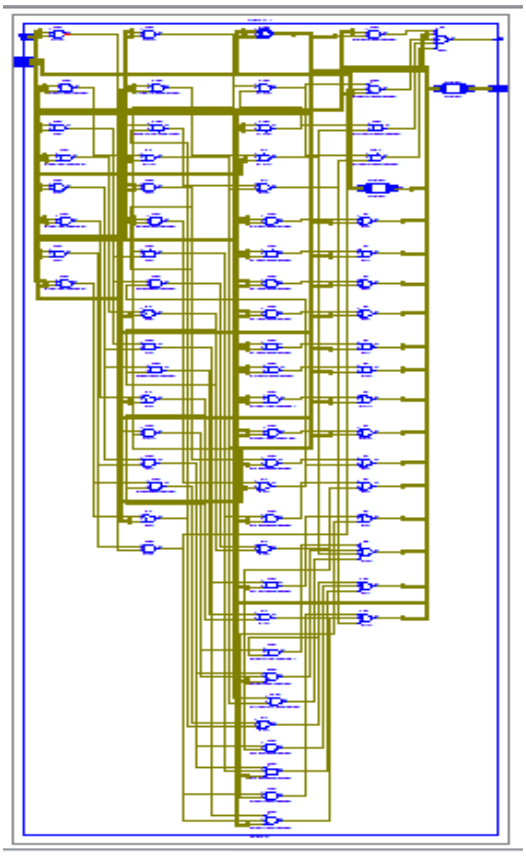

Fig 9.7 Logic Diagram of Han Carlson Adder

COMPARISON TABLE

Table 9.1.1: Comparison of Slice utilization, No. of logic levels required & Delay

Target device - Spartan6
xc6slx45-3csg324

Adder Name	No. of Slices LUT	No. of Logic Level	Delay (in ns)	
			Spartan 6	Vertex 6
Ripple Carry	24	10	12.244	4.599
Kogge Stone	43	10	11.935	4.629
Han Carlson	26	10	12.145	4.469

Table 9.1.2: Static Power Comparison at various temperatures
Temp Grade: C-Grade, V_{ccint}= 1.2V, V_{ccaux}= 2.5v

Adder Name	Process Amb. Temp. 0°C		Process Amb. Temp. 25°C		Process Amb. Temp. 50°C		Process Amb. Temp. 75°C	
	Typical (Watt)	Max. (Watt)	Typical (Watt)	Max. (Watt)	Typical (Watt)	Max. (Watt)	Typical (Watt)	Max. (Watt)
RCA	0.022	0.049	0.037	0.088	0.068	0.175	0.137	0.404
KSA	0.022	0.049	0.037	0.088	0.068	0.175	0.137	0.405
Han Carlson	0.022	0.049	0.037	0.121	0.068	0.175	0.137	0.405

IX. RESULT DISCUSSION

The no. of slices LUT and no of logic level observed for adder designs from synthesis reports in Xilinx ISE13.2 are compared and shown in figure. The area of the adder designs is measured in terms of look up tables (LUT) and input output blocks (IOB) taken for Xilinx ISE 13.2 in Spartan 6, and vertex 6 FPGA chip is plotted in the figure. As per reference [1] ISE software doesn't give exact delay of the adders because it is not able to analyze the critical path over the adder. From the comparison table it is clear that out of all adders Kogge Stone Adder has less delay.

According to the synthesis report out of three parallel prefix adders, Ripple Carry Adder has better delay because of taking least logic level for compaction where as on the basis of delay and area (slice utilization of the no of LUT required) Han Carlson Adder is the best on an average.

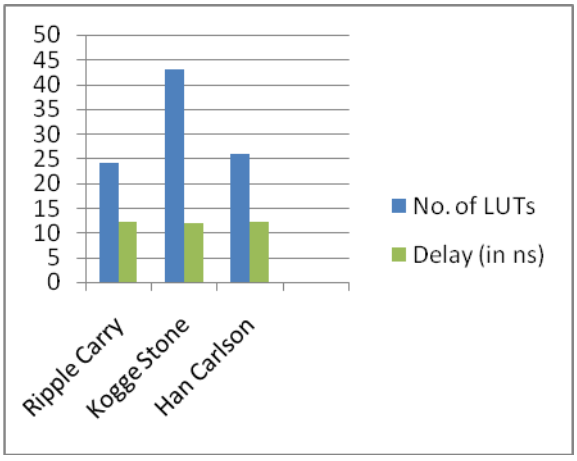

Fig 9.1.3: Comparison of Slice utilization & Delay

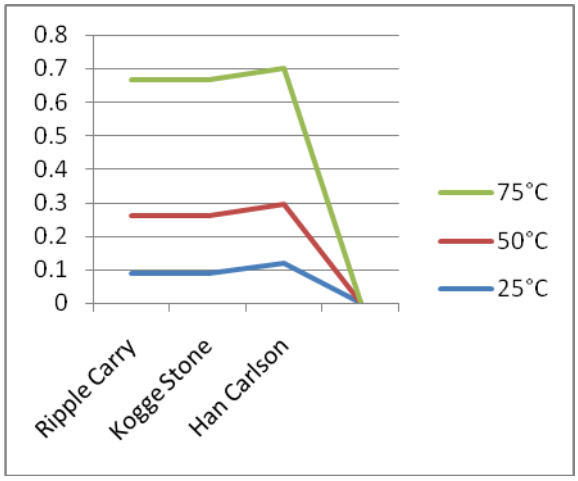

Fig 9.1.4: Static Power Comparison at various Temperatures

X. CONCLUSION

From the study of analysis done on delay, area and power we have concluded that the efficiency is improved by 6.5% in ours delay for RCA, when compared to Kogge Stone adder and it is improved by 2.53% for Han Carlson adder in Spartan 6 FPGA chip. so we can say that RCA is the best because of taking least logic level for compaction where as on the basis of delay and area (slice utilization or the no of LUT's required) Han Carlson is the best on an average where as power analysis report shows that all most all of the parallel prefix adders takes more or less the same power.

REFERENCES

[1]. R. K. Richards, Arithmetic Operations in Digital Computers. D. Van Nostrand Co., Princeton, N.J., 1955.
[2]. A. Weinberger and J. Smith, .A logic for high-speed addition,. National Bureau of Standards, no. Circulation 591, pp. 3.12, 1958.
[3]. A. Tyagi, .A reduced area scheme for carry-select adders,. IEEE Trans. Computers, vol. 42, no. 10, pp. 1163.1170, Oct. 1993.
[4]. H. Ling, .High speed binary adder,. IBM Journal of Research and Development, vol. 25, no. 3, pp. 156.166, 1981.
[5]. R. P. Brent and H. T. Kung, .A regular layout for parallel adders,. IEEE Trans. Computers, vol. C-31, no. 3, pp. 260.264, Mar. 1982.119
[6]. P. Kogge and H. Stone, .A parallel algorithm for the efficient solution of a general class of recurrence relations,. IEEE Trans. Computers, vol. C-22, no. 8, pp. 786.793, Aug. 1973.
[7]. S. Knowles, .A family of adders,. in Proc. 15th IEEE Symp. Comp. Arith., June 2001, pp. 277.281.

[8]. J. Sklansky, .Conditional-sum addition logic,. IRE Trans. Electronic Computers, vol. EC-9, pp. 226.231, June 1960.
[9]. R. Ladner and M. Fischer, .Parallel prefix Computation,. J. ACM, vol. 27, no. 4, pp. 831.838, Oct. 1980.
[10]. T. Han and D. Carlson, .Fast area-efficient VLS Adders, in Proc. 8th Symp. Comp. Arith., Sept. 1987, pp. 49.56.
[11]. A. Naini, D. Bearden, and W. Anderson, .A 4.5ns 96b CMOS adder design,. in Proc. IEEE Custom Integrate Circuits Conference, vol. 38, no. 8, Apr. 1965, pp. 114.117.
[12]. T. Kilburn, D. B. G. Edwards, and D. Aspinall, .Parallel addition in digital computers: a new fast carry circuit,. in Proc. IEE, vol. 106, pt. B, Sept. 1959, p. 464.
[13]. N. Szabo and R. Tanaka, Residue Arithmetic and Its Applications to Computer Technology. McGraw-Hill, 1967.
[14]. W. K. Jenkins and B. J. Leon, .The use of residue number systems in the design of finite impulse response digital filters,. IEEE Trans. Circuits and Systems, vol. 24, no. 4, pp. 171.201, Apr. 1977.
[15]. X. Lai and J. L. Massey, .A proposal for a new block encryption standard,. in Advances in Cryptology - EUROCRYPT'90, Berlin, Germany: Springer-Verilog, 1990, pp. 389.404.
[16]. S. S.-S. Yau and Y.-C. Liu, .Error correction in redundant residue number systems,. IEEE Trans. Computers, vol. C-22, no. 1, pp. 5.11, Jan. 1973.
[17]. F. Halsall, Data Communications, Computer Networks and Open Systems. Addison Wesley, 1996.
[18]. C. Efstathiou, D. Nikolos, , and J. Kalamatianos, .Area-time efficient modulo 2n+1 adder design,. IEEE Trans. Circuits and System-II, vol. 41, no. 7, pp. 463.467, 1994.
[19]. L. Kalamboukas, D. Nikolos, C. Efstathiou, H. T. Vergos, , and J. kalamatianos, .High-speed parallel-prefix modulo 2n + 1 adder, IEEE Trans. Computers, vol. 49, no. 7, special issure on computer arithmetic, pp. 673.680, July 2000.
[20]. C. Efstathiou, H. T. Vergos, and D. Nikolos, .Fast parallel-prefix modulo 2n + 1 adders,. IEEE Trans. Computers, vol. 53, no. 9, pp. 1211.1216, Sept. 2004.
[21]. H. T. Vergos, C. Efstathiou, and D. Nikolos, .Modulo 2n _ 1 adder design using Select-prefix blocks, IEEE Trans. Computers, vol. 52, no. 11, pp. 1399.1406, Nov. 2003.
[22]. V. Paliouras and T. Stouraitis, .Novel high-radix residue number system multipliers and adders, in Proc. 1999 IEEE Int'l Symp. Circuits and Systems VLSI (ISCAS '99), 1999, pp. 451.454.
[23]. S. Bi, W. J. Gross, W. Wang, A. Al-khalili, and M. N. S. Swamy, .An area-reduced scheme for modulo 2n + 1 addition/subtraction,. In Proc. 9th International Database Engineering & Application Symp, 2005, pp. 396.399.
[24]. R. Zimmermann, Efficient VLSI implementation of modulo (2n-1) addition and multiplication, in Proc. 14th IEEE Symp. Computer Arithmetic, 1999, pp. p.158.167.
[25]. J. Chen and J. E. Stine, .Enhancing parallel-prefix structures using carry-save notation, 51st Midwest Symp. Circuits and Systems, pp. 354.357, 2008.
[26]. G. E. Moore, .Cramming more components onto integrated circuits,. In Electronics, May 1965, pp. 25.5.1.25.5.4.
[27]. M. Lehman and N. Burla, .Skip techniques for high-speed carry propagation in binary arithmetic units,. IRE Trans. Electron. Comput., pp. 691.698, Dec. 1961.
[28]. O. J. Bedrij, Carry-select adder, IRE Trans. Electron. Comput. pp. 340.346, June 1962.
[29]. J. Sklansky, .Conditional sum addition logic, IRE Trans. Electron. Comput., pp. 226. 231, June 1960.
[30]. V. G. Oklobdzija, B. Zeydel, H. Dao, S. Mathew, and R. Krishnamurthy, .Energy delay estimation technique for high-performance microprocessor VSLI adders,. Proc. 16th IEEE Symp. Computer Arithmetic (ARITH-16'03), p. 272, June 2003.
[31]. R. Zimmermann, .Binary adder architectures for cell-based vlsi and their synthesis,.Ph.D. dissertation, ETH Dissertation 12480, Swiss Federal Institute of Technology, 1997.
[32]. R. Zimmermann and H. Kaeslin, .Cell-based multilevel carry-increment adders with minimal AT- and PT-products.
[33]. S. Majerski, .On determination of optimal distributions of carry skips in adders, IEEE Trans. Electron. Comput., pp. 45.58, Feb. 1967.
[34]. J. E. Stine, Digital Computer Arithmetic Data path Design Using Verilog HDL. Kluwer Academic, 2004.
[35]. R. W. Doran, .Variants of an improved carry-look-ahead-sum adder,. IEEE Trans. Computers, vol. 37, no. 9, pp. 1110.1113, 1988.
[36]. J. Grad, .Analysis and implementation of binary addition in nanometer cmos technology, Ph.D. dissertation, Department of Electrical Engineering: Illinois Institute of Technology, May 2006.

[37]. N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective. Addison Wesley, 2004.
[38]. D. Patil, O. Azizi, M. Horowitz, R. Ho, and R. Ananthraman, .Robust energy-efficient adder topologies,. in 18th IEEE International Symposium on Computer Arithmetic, June 2007, pp. 16.28.
[39]. N.Weste and K. Eshraghian, Principles of CMOS VLSI Design: A System Perspective. Addison-Wesley, 1985.
[40]. A. Avizienis, .Signed-digit number representations for fast parallel arithmetic,. IRE Trans. Electron. Comput. pp. 389.400, Sep. 1961.