International Journal of Reconfigurable and Embedded Systems
Vol. 7, No. 2, July 2018, pp. 115~123
ISSN: 2089-4864, DOI: 10.11591/ijres.v7.i2.pp115-123

Analysis of CMOS Logic and Transmission Gate for 64 Bit Parallel Prefix Adders

Nehru.K, Nagarjuna T., Somanaidu U.
Department of Electronics and Communication Engineering, Institute of Aeronautical Engineering, Hyderabad, India

Article Info

Article history:

Received Apr 16, 2018
Revised Jun 17, 2018
Accepted Jun 25, 2018

Keywords:

Brent kung adder
Carry look ahead adder
Kogge stone
Ladner fischer
Prefix adder

ABSTRACT

Parallel prefix adder network is a type of carry look ahead adder structure. It is widely considered as the fastest adder and used for high performance arithmetic circuits in the digital signal processors. In this article, an introduction to the design of 64 bit parallel prefix adder using transmission technique which acquires least no of nodes with the lowest transistor count and low power consumption is presented. The 64 bit parallel prefix adder is designed and comparison is made between other previously parallel prefix adders. The result shows that the proposed 64 bit parallel prefix adder is slightly better than existing parallel prefix adders and it considerably increases the computation speed.The spice tool is used for analysis with different supply voltages.

Corresponding Author:

Nehru K.,
Department of Electronics and Communication Engineering,
Institute of Aeronautical Engineering,
Hyderabad, 500043, India.
Email: nnehruk@gmail.com

1. INTRODUCTION

Simple overlaying of all output evaluation trees from the unoptimized prefix algorithm leads to the tree-prefix algorithm proposed by [1]. This leads to a high fan-out of some black nodes (O *(n),* unbounded fan-out*) but* results in the smallest possible number of node delays (minimal depth), a small number of signals and very few wiring tracks*(O(log n)).*The Kogge-Stone adder is a parallel prefix form of carry look-ahead adder. It generates the carry signals in *O (log n)* time, and is widely considered as the fastest adder design possible. It is the common design for high-performance adders in industry [2-4]. This is achieved by using a large number of independent tree structures in parallel.The generate signal indicates that the outgoing carry is 1, independent of the incoming carry, while the propagate signal indicates that the outgoing carry is equivalent to the incoming carry. A simpler Brent-Kung adders was been proposed to solve the disadvantages of Kogge-Stone adders [5], [6]. It has only $2N\text{-}2\text{-}\log_{2N}$ carry merge blocks, so the cost and wiring complexity is greatly reduced. But the logic depth of Brent-Kung adders increases to $2\log_{2N\text{-}1}$, so the speed is slower.Han and Carlson proposed an algorithm which combines the advantages of the Brent-Kung and the Kogge-Stone algorithms by mixing them [7], [8]. The first and last levels are of the Brent-Kung type while the Kogge-Stone graph is used in the middle. The number of parallel trees and thus the number of black nodes and interconnections is reduced at the cost of a slightly longer critical path, compared tothe Kogge-Stone adder [9], [10].

2. PARALLEL PREFIX ADDERS

2.1. New Prefix Cell Operators

In this PPA, the dot operator '●' and the semi-dot operator '●' are introduced. The dot operator '●' is defined by the equation (1) and the semi-dot operator '●' is defined by the equation (2).

$$(P_i,G_i) \bullet (P_{i-1},G_{i-1})=(P_i.P_{i-1}, G_i+P_i.G_{i-1}) \quad (1)$$

$$(P_i,G_i) \circ (P_{i-1},G_{i-1})=(G_i+P_i.G_{i-1}) \quad (2)$$

In the above equation, '•' operator is applied on two pairs of bits (P_i,G_i) and (P_{i-1},G_{i-1}) . These bits represent generate and propagate signals used in addition. The output of the operator is a new pair of bits which is again combined using a dot operator '●' or semi-dot operator '●' with another pairs of bits. This procedural use of dot operator '●' and semi-dot operator '●' creates a prefix tree network which ultimately ends in the generation of all carry signals.

$$\overline{G} = \overline{(a_i . b_i)} \quad (3)$$

$$\overline{P} = \overline{(a_i \oplus b_i)} \quad (4)$$

In the final step, the sum bits of the adder are generated with the propagate signals of the operand bits and the preceding stage carry bit using a xor gate. The semi-dot operator '●' will be present as last computation node in each column of the prefix graph structures, where it is essential to compute only generate term, whose value is the carry generated from that bit to the succeeding bit.

2.2. Four Operators Cells

In the first stage, generation and propagation signals generated by XORgates respectively. For deriving the carry signals in the second stage, this architecture introduces four different computation nodes for achieving improved performance.

2.3. 8 Bit Parallel Prefix Adder

The 8-bit parallel prefix having four stages for generating carries in the middle PPA network is shown in Figure 1. This 8-bit PPA contains three odd-dot cells, one even-dot cells, three odd-semi-dot cells and two even-semi-dot cells along with three inverters pairs.

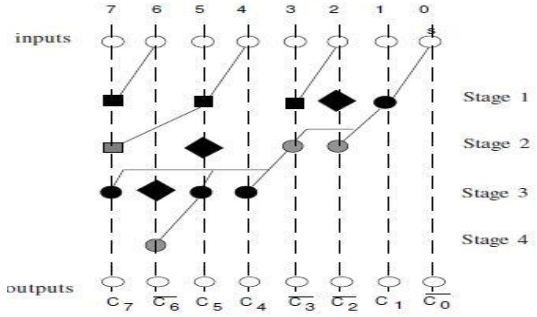

Figure 1. 8-Bit parallel prefix adder

2.4. 16 Bit Parallel Prefix Adder

The 16 bit parallel prefix adder is shown in Figure 2.The generation and propagation signals can be generated by using the equation (3) and (4). This stage is responsible for creations of group generate and group propagates signals. The 16-bit parallel prefix having five stages for generating carries in the middle PPA network. This 16-bit PPA contains seven odd-dot cells, four even-dot cells, nine odd-semi-dot cells and six even-semi-dot cells along with seven inverters pairs. The second stage in the prefix addition is termed as prefix computation.

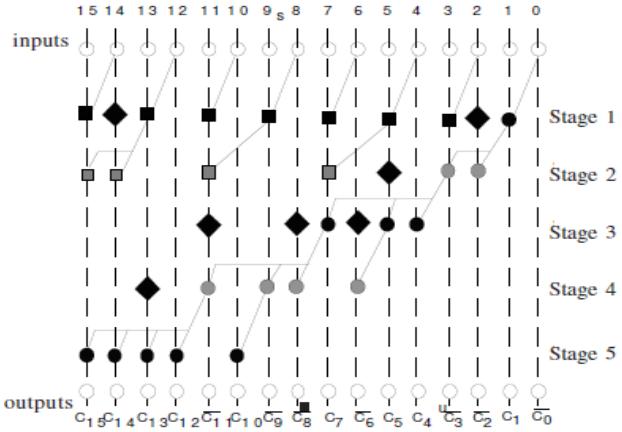

Figure 2. 16-Bit parallel prefix adder

2.5. 32-Bit Parallel Prefix Adders

Figure 3 shows the architecture of the proposed 32-bit parallel prefix adder. The objective is to eliminate the massive overlap between the prefix sub-terms being computed. Hence the associate property of the dot operator is employed to keep the number of computation nodes at a minimum. The first stage of the computation is called as pre-processing. The first stage in the architectures of the 32-bit prefix adder involve the creation of generate and propagate signals for individual operand bits in active low format. The equations (3) and (4) represent the functionality of the first stage.

From the equations (3) and (4), a_i, b_i represent input operand bits for the adder, where 'i' varies from 0 to 31. The second stage in the prefix addition is termed as prefix computation. This stage is responsible for creation of group generates and group propagate signals. The stages with odd indexes use odd-dot and odd-semi-dot cells where as the stages with even indexes use even-dot and even-semi-dot cells.

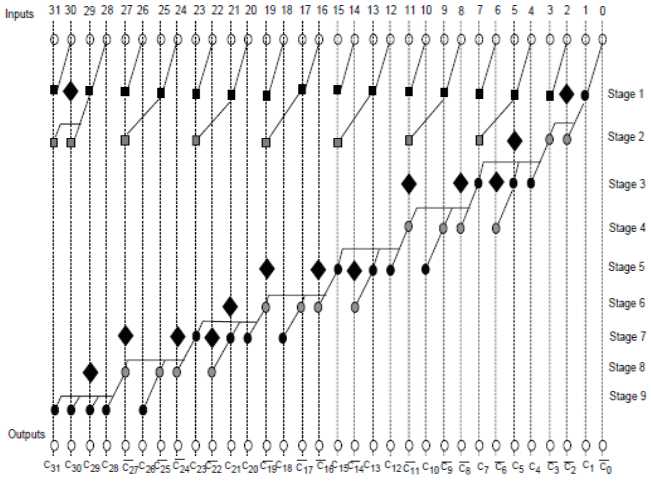

Figure 3. 32-Bit parallel prefix adder

3. PROPOSED 64-BIT PARALLEL PREFIX ADDER

Figure 4 shows the architecture of 64 bit parallel prefix adder. The proposed 64-bit parallel prefix adder has ten stages of implementation. CMOS logic family will implement only inverting functions. Thus cascading odd cells and even cells alternatively gives the benefit of elimination of two inverters between them, if a dot or a semi-dot computation node in an odd stage receives both of its input edges from any of the even stages and vice-versa. But it is essential to introduce two inverters in a path, if a dot or a semi-dot computation node in an even stage receives any of its edges from any of the even stages and vice-versa. From the prefix graph of the proposed structure shown in Figure 4, we assume that there are only few edges with a pair of inverters, to make (G, P) as $(\overline{G}, \overline{P})$ or to make $(\overline{G}, \overline{P})$ as (G, P) respectively

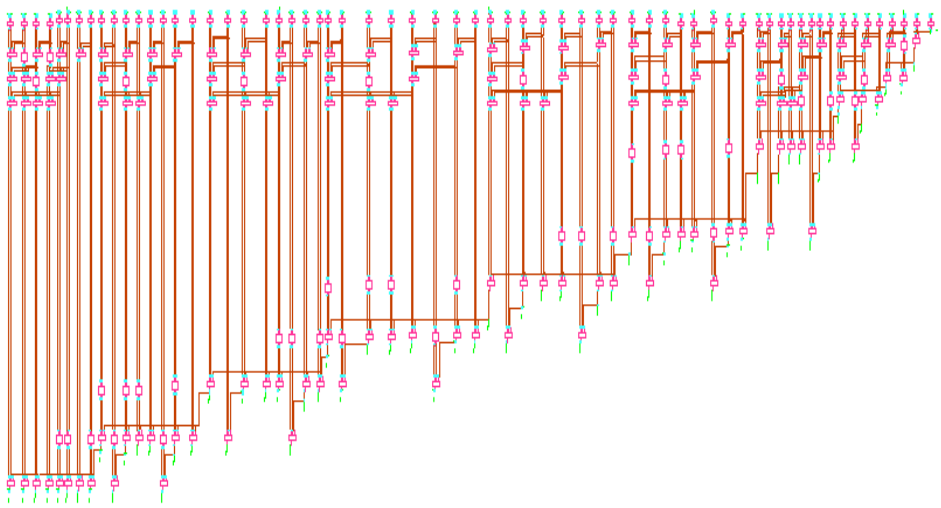

Figure 4. 64-Bit proposed parallel prefix adder

The pair of inverters in a path is represented by a ◆ in the prefix graph. By introducing two cells for dot operator and two cells for semi-dot operator, we have eliminated a large number of inverters. Due to inverter elimination in paths, the propagation delay in those paths would have reduced. Further we achieve a benefit in power reduction, since these inverters if not eliminated, would have contributed to significant amount of power dissipation due to switching. The output of the odd-semi-dot cells gives the value of the carry signal in that corresponding bit position. The output of the even-semi-dot cell gives the complemented value of carry signal in that corresponding bit position. The final stage in the prefix addition is termed as post-processing. The final stage involves generation of sum bits from the active low propagate signals of the individual operand bits and the carry bits generated in true form or complement form. The first stage and last stage are intrinsically fast because they involve only simple operations on signals local to each bit position. The intermediate stage embodies long distance propagation of carries, so the performance of the adder depends on the intermediate stage.

4. SIMULATION AND RESULT DISCUSSION

4.1. 8-Bit Parallel Prefix Adder

Schematic and simulation results for 8 bit parallel prefix adder are shown in Figure 5 and Figure 6. The inputs given are bit patterns to the T spice. The schematic is drawn by using library model files. The T spice is used to generate the netlist from the S edit.S-edit is a tool is used for schematic.

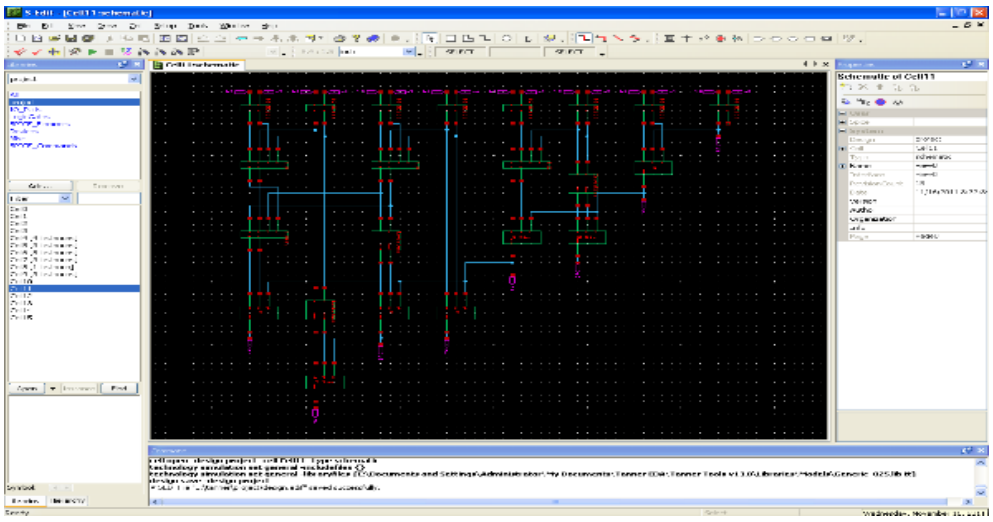

Figure 5. Schematic diagram for 8-Bit parallel prefix adder

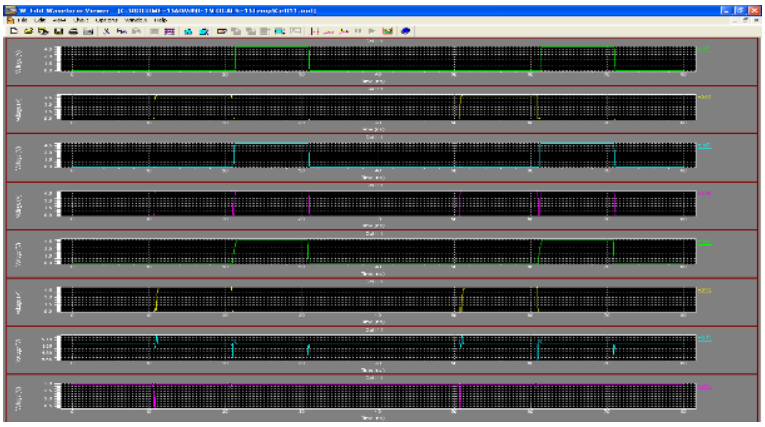

Figure 6. Transient result for 8-Bit parallel prefix adder

4.2. 16-Bit Parallel Prefix Adders

Schematic and simulation results for 16 bit parallel prefix adder are shown in Figure 7 and Figure 8. The inputs given are bit patterns to the T spice. The schematic is drawn by using library model files.The T spice is used to generate the netlist from the S edit.S-edit is a tool is used for schematic.

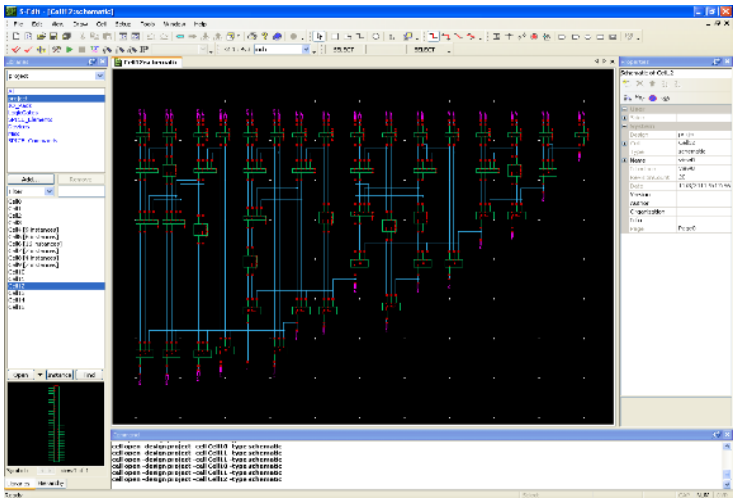

Figure 7. Transient result for 16-Bit parallel prefix adder

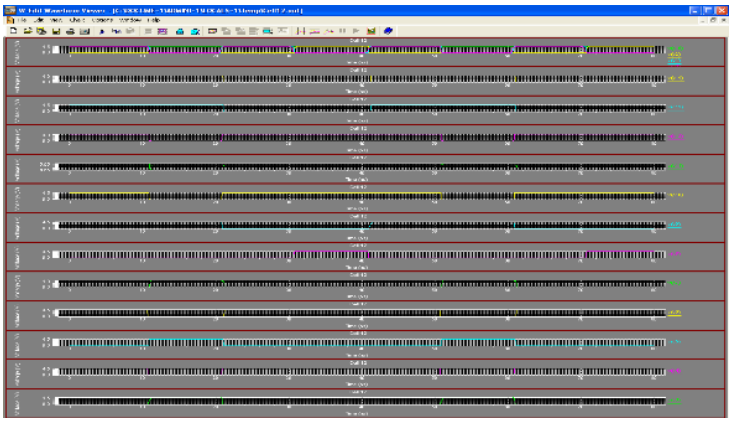

Figure 8. Transient result for 16-Bit parallel prefix adder

4.3. 32-bit Parallel Prefix Adders

A schematic and simulation result for 32 bit parallel prefix adder is shown in Figure 9 and Figures 10 & 11. The inputs given are bit patterns to the T spice. The schematic is drawn by using library model files.The T spice is used to generate the netlist from the S edit.S-edit is a tool is used for schematic.

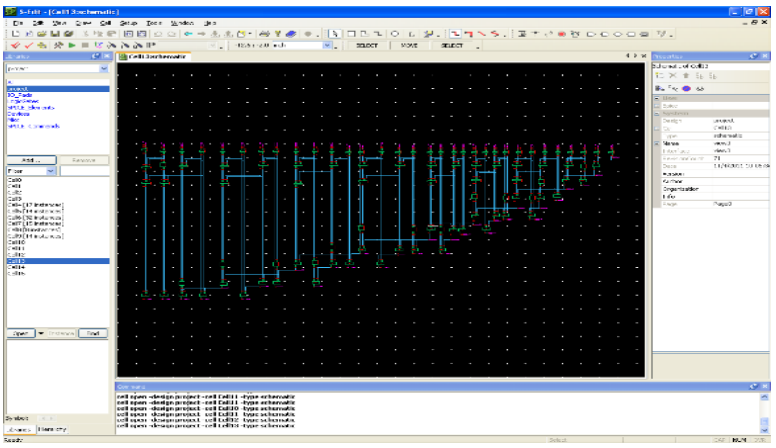

Figure 9. Block diagram for 32-Bit parallel prefix adder

Figure 10. Transient result for 32-Bit parallel prefix adder

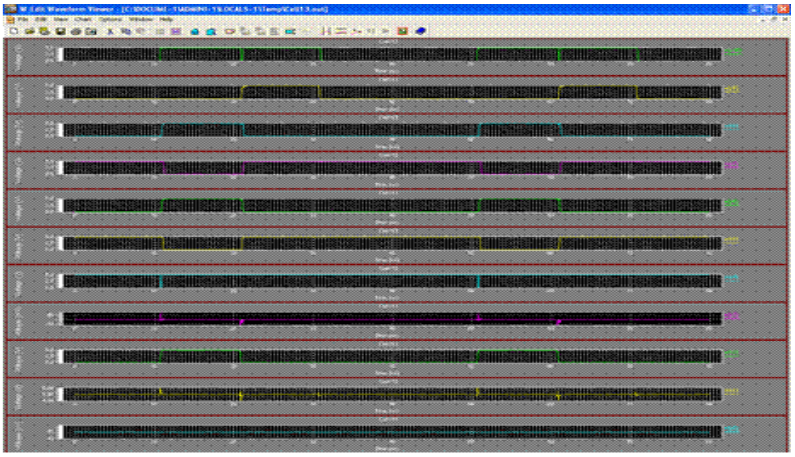

Figure 11. Transient result for 32-Bit parallel prefix adder

4.5. Proposed 64-Bit Parallel Prefix Adder

Schematic and simulation results for 64 bit parallel prefix adder is shown in Figure 12 and Figures 13, 14 & 15. The inputs given are bit patterns to the T spice. The schematic is drawn by using library model files. The T spice is used to generate the netlist from the S edit. S-edit is a tool is used for drawing schematic.

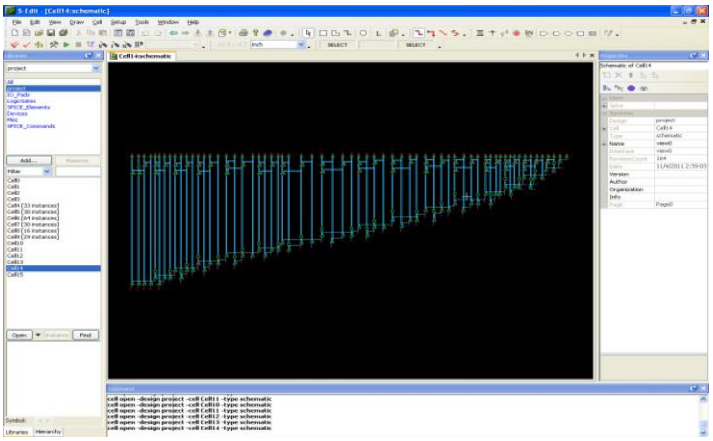

Figure 12. Schematic diagram for Proposed 64-Bit parallel prefix adder

Figure 13. Transient result for Proposed 64-Bit parallel prefix adder

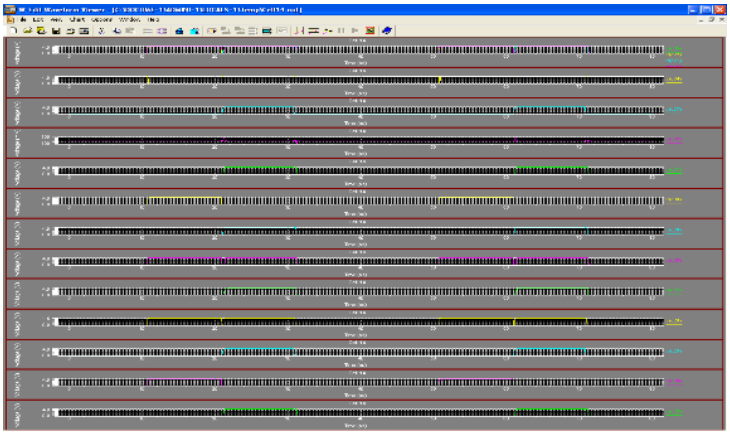

Figure 14. Transient result for Proposed 64-Bit parallel prefix adder

Figure 15. Transient result for proposed 64-Bit parallel prefix adder

4.6. Results & Discussions

The power comparison shows that when number of input bits increased, the power consumed by the adder will also increase as shown in Tables 1 and 2.The transistor count for different size of adder as shown in Tables 3 and 4.

Table 1. CMOS Based Prefix Adders-Power Consumptiion (mW)

Prefix Adders	8 BIT	16 Bit	32 Bit	64 Bit
Brent Kung	84.41	148.89	463.8	706.34
Ladner Fischer	84.62	139.40	374.65	778.21
Kogge Stone	82.32	159.87	385.44	733.44
Proposed Adder	76.41	141.45	348.33	684.23

Table 2. TG Based Prefix Adders–Power Consumption (mW)

Prefix Adders	8 BIT	16 B	32 Bit	64 Bit
Brent Kung	57.48	134.23	254.56	577.87
Ladner Fischer	59.57	136.10	258.43	584.23
Kogge Stone	69.34	139.25	259.66	621.24
Proposed Adder	51.33	123.45	221.60	522.56

Table 3. CMOS Based Prefix Adders-Mosfet Counts

Prefix Adders	8 BIT	16 Bit	32 Bit	64 Bit
Brent Kung	304	704	1598	3442
Ladner Fischer	318	782	1930	4170
Kogge Stone	318	782	1930	4170
Proposed Adder	282	658	1334	2676

Table 4. TG Based Prefix Adders–Mosfet Counts

Prefix Adders	8 BIT	16 Bit	32 Bit	64 Bit
Brent Kung	284	636	1404	3014
Ladner Fischer	292	684	1580	3488
Kogge Stone	292	684	1640	3496
Proposed Adder	278	634	1294	2600

5. CONCLUSION

We have compared proposed low power 64 bit parallel prefix adder results with existing parallel prefix adders such as 8 bit PPA, 16-bit PPA and also with 32bit PPA. The proposed low power 64-bit parallel prefix adder was designed by using the four different prefix cell operators. The performances in terms of power, number of nodes, number of transistor tradeoffs for various input ranges were analyzed and discussed with the results .For 64-bit low power Parallel Prefix Adder seventeen stages were used for generating the sixty three carries outputs and 1339 nodes were formed in the 64-bit low power Parallel Prefix Adder. This makes our proposed low power adder was most suitable for complex digital systems.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the support of Institute of Aeronautical Engineering management for providing the lab facilities.

REFERENCES

[1] Yang S, Lau KT, Zhang Y. Design of Low Power CMOS Parallel Prefix Adder Cell. Journal of Electrical Engineering and Electronic Technology. 2017 Jan 25;2016.

[2] Gaur N, Tyagi D, Mehra A. Performance comparison of adder architectures on 28nm FPGA. InAdvances in Computing, Communication, & Automation (ICACCA)(Fall), International Conference on 2016 Sep 30 (pp. 1-5). IEEE.

[3] Zarandi AA, Molahosseini AS, Hosseinzadeh M, Sorouri S, Antao S, Sousa L. Reverse converter design via parallel-prefix adders: Novel components, methodology, and implementations. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2015 Feb;23(2):374-8.

[4] Rani G, Kumar S. Delay analysis of parallel-prefix adders. International Journal of Science and Research (IJSR). 2014 Jun;3(6):2339-42.

[5] Ramanathan P, Vanathi PT. High Performance Parallel Prefix Adder For Wider Word Lengths. Global Journal of Pure and Applied Mathematics. 2015;11(2):733-43.

[6] Perri S, Lanuzza M, Corsonello P. Design of high-speed low-power parallel-prefix adder trees in nanometer technologies. International Journal of Circuit Theory and Applications. 2014 Jul 1;42(7):731-43.

[7] Sasidharan S, Sandeep PM, Varatharaj M, Scholar UG, Departmentof EC. Design of Hybrid Parallel Prefix Adders for the Reverse Converters. International Journal of Engineering Science. 2016 Mar;2964.

[8] Roy S, Choudhury M, Puri R, Pan DZ. Towards optimal performance-area trade-off in adders by synthesis of parallel prefix structures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2014 Oct;33(10):1517-30.

[9] Banerjee S, Rao W. A general approach for highly defect tolerant parallel prefix adder design. InDesign, Automation & Test in Europe Conference & Exhibition (DATE), 2016 2016 Mar 14 (pp. 666-671). IEEE.

[10] Hassanzadeh A, Shabani A. Low Power Parallel Prefix Adder Design Using Two Phase Adiabatic Logic. Journal of Electrical and Electronic Engineering. 2015;3(6):181-6.

BIOGRAPHIES AUTHORS

Nehru K received his Bachelor Degree in Erode Sengunthar Engineering College, Anna University in 2005. And he obtained his Master Degree in R.M.K Engineering, Anna University in 2007.And he also obtained outstanding master student at that year. He is obtained his Ph.D in 2014 at Faculty of Information and Communication Engineering, Anna University, India. His main research interest is in the area of Low Power VLSI, Testing of VLSI Circuits, FPGA Design, CAD for VLSI, Signal processing. He has published papers on these topics in various international journals.

Nagarjuna Telagam is with the Electronics and Communication Engineering Department, Institute of Aeronautical Engineering, Hyderabad, India. He is a Research Scholar in Sathyabama University and is interested in the following topics: Wireless Communications, MIMO, OFDM, GFDM, Embedded Systems, VLSI. Currently, He is working as Assistant Professor. He Received his B.Tech degree from JNTU University/ Narayana Engineering College. He received his master degree from Anna University/ Loyola Institute of Technology. He is Anna University rank holder (20) for M.E degree in 2013. He published papers in different Referred Journals (nagarjuna473@gmail.com).

U.SOMANAIDU received B.TECH. degree in Electronics and Communications Engineering from JNTU University, Kakinada, in 2011,Currently doing M. Tech. in PEC Engineering college, from JNTUK University, Kakinada. His research interests include VLSI and Embedded systems.