Design and Comparative Analysis of CMOS Full Adder Cells Using Tanner EDA Tool

JATINDER KUMAR
Student M.TECH (E.C.E),
GGSCET, Bathinda,
Punjab, India.
jatinder.k.gupta@gmail.com

Abstract — The binary adders are the fundamental and key component in digital signal processors, general purpose microprocessors and data-processing application specific integrated circuits. Therefore, binary adders are crucial building blocks in very large scale integrated circuits. Their efficient implementation is highly important because a costly carry-propagation operation involving all operand bits has to be performed. With the increasing level of device integration and growth in complexity of microelectronic circuits, power dissipation, area and delay have become the predominant design goals for fast adder cells. In this paper, various 8-bit CMOS adder circuits are designed and implemented using TANNER EDA tool. The adder designs are simulated at different supply voltages and the results are compared to find an efficient adder structure. To minimize power-delay product a prefix adder has been proposed and compared with other adder structures. The results show that the proposed prefix adder has least power-delay product as compared to other adder designs. The proposed prefix adder has better results than the hybrid prefix adder in 180nm technology at 1.8V but in 90nm technology it has higher value of PDP at all voltages other than 2.5V.

Keywords — Carry Select Adder, Carry Increment Adder, Carry Skip Adder, Carry Look-Ahead Adder, Prefix Adder, 8-Bit Adder, CMOS, Power-Delay Product, TANNER EDA.

I INTRODUCTION

The fundamental and most commonly used arithmetic operation in many VLSI systems is the most speed limiting element and therefore its performance and power optimization is of utmost importance [10]. Main task of this operation is to add two binary numbers, and it is implemented by a full adder cell. Furthermore, an addition as an operation is used in many other useful and more complex operations, e.g., multiplication, division, and address calculation. All these operations are realized by complex structure of transistors. Most of these systems have the adder in their crucial path. The crucial path is made up of transistors that produce the maximal time-delay in the output signal. The behavior of the transistors in the critical path essentially determines performance of the entire system. Hence, performance of the adders can be considered as tremendously significant for VLSI systems.

With the technology scaling to deep sub-micron region, the speed of the operation increases rapidly. Similarly the power consumption of the semiconductor chips also increases significantly due to the increasing density of the components on these chips. Therefore, in realizing modern VLSI circuits, low-power consumption and high-speed are the dominant factors which need to be considered. The dereliction of high-power circuits relates to the growing popularity of portable electronic devices. Laptops, compact digital cameras, pagers and cell phones use small batteries as a power source, which provides a finite time of operation before recharging. In order to make battery life longer, low power operation is desirable in integrated circuits. At the same time, there is a necessity of fast adders to minimize delay. Therefore careful optimization of the adder is not trivial.

For the optimization of speed in adders, the most important factor is carry generation. For the implementation of a fast adder, the generated carry should convey to the output as fast as possible, so as to reduce the worst path delay which in turn determines the ultimate speed of the digital system. For timing optimization in design, a network can be improved either at circuit or at logic level. Circuit level optimization can be achieved by manipulating transistor sizes and circuit topologies whereas logic level optimization is carried out by manipulating boolean equations. In the optimization for area, care should be taken in the design of the building blocks of the structure, which determine the area occupied by the architecture and, finally, also affect the speed. This work is provoked with the objective of improving the performance parameters associated with CMOS adder circuits such as power, delay and area. The goal is to elaborate efficient and versatile synthesis algorithm for the best performing adder architecture found in the comparisons. The comparative analysis will help designers to have a better knowledge of design metric and to choose between trade-off of design.

II LITERATURE REVIEW

The density and operating speed of integrated circuit computing components has increased nearly exponentially for a few decades, following a trend defined by Moore's Law. The size of the technology is shrinking day by day and if this dissipate large amount of power then large amount of heat will be dissipated in short span of time and that heat has to be removed. Heat removal may become a limiting factor if the package cannot sufficiently dissipate this heat or if the required thermal components are too expensive for the application. Secondly high-power circuits fail due to the increasing popularity of portable electronic devices. Laptops, portable cameras and cell phones all use batteries as a power source [16]. These devices provide a finite time of operation before they need recharging. To increase battery life, low power operation is desirable in integrated circuits. Identification and modeling of different components is very important for estimation and reduction of power dissipation.

Several algorithms have been presented for high speed parallel addition, and there is generally a tradeoff between speed and area. For the optimization of speed in adders, the most important factor is carry generation. For the implementation of a fast adder, the generated carry should convey to the output as fast as possible, so as to reduce the worst path delay which in turn determines the ultimate speed of the digital system. The most basic adder circuit is carry ripple adder. The carry ripple adder is built by cascading single-bit full-adder cells [4]. In this adder circuit, each full-adder begins its counting only after carry-out signal from preceding stage is available. Hence the vital path delay in carry ripple adder is governed by its carry-out propagation route. As shown in Fig 1 for an N-bit adder, the critical path is the N-bit carry propagation path. So as number of bit N grows, the delay time of carry ripple adder increases consequently in a linear way.

The carry ripple adder is good and area efficient when computing for fewer bits or when speed is not a prime issue, but with the increase in the number of bits, the delay associated with this adder will be high. The carry select adder improves the shortcomings of carry ripple adder by removing the linear dependency between calculation delay time and input word length [7].

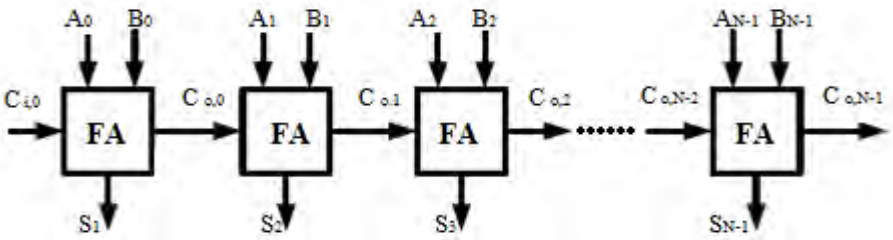

Fig. 1 The N-bit carry ripple adder Constructed by N set single bit full adder

The carry select adder has some advantages like low power consumption and simple layout. But carry select adder has larger area and slower speed because each full adder must wait for the carry bit to be calculated from the previous full adder. The carry skip adder [15] and carry look-ahead [8] improves the speed of operation but their power consumption is high which degrades the overall PDP. For large addition, the delay of carry look-ahead adders is influenced by the delay of passing the carry across the look-ahead stages. This delay could be improved by looking ahead across the look-ahead stages. In general, a multilevel tree of look-ahead structures can be constructed to achieve delay that grows with (log N) where 'N' represents the number of bits. Such adders are variously referred to as tree adders or parallel prefix adders [13]. Further improvement in speed can be achieved from carry increment adder. The carry increment adder is derived from carry select adder. It combines the carry increment approach with the processing of carry bits on multiple levels of hierarchy [2].

Several circuit design techniques are compressed in order to find their efficiency in terms of speed and power dissipation. In these days MOSFETs (Metal Oxide Semiconductor Field Effect Transistor), namely n-channel (NMOS) and p-channel (PMOS) transistors are used for designing logic gate in digital circuitry and other blocks such as flip-flops or memories. A transistor ideally behaves like a switch. When a circuit consists of both NMOS and PMOS transistors, it is said to be implemented in complementary metal oxide semiconductor (CMOS). The two main advantages of CMOS devices are high noise immunity and low static power consumption [1]. Today CMOS is the predominant design technology of the microelectronics industry.

III ADDER ARCHITECTURES

1. Carry Select Adder

In carry select adder, sum and carry are calculated by assuming input carry as 1 and 0 prior the input carry arrives. When actual carry input generated, the actual calculated values of sum and carry are determined using a multiplexer. Hence the critical path of N-bit in carry ripple adder circuit is greatly reduced and therefore the computation delay of the carry select adder is much smaller than the carry ripple adder. However, calculating sum and carry by assuming input carry as 1 and 0 needs duplicate addition circuit. To remove the duplicated addition circuit in the conventional carry select adder, the carry select adder can be realized by sharing the

common boolean logic term [7]. Sharing common boolean logic helps in reducing the number of transistors and achieves a lower PDP. Through examining the truth table 1 of a single-bit full-adder, we can find out that the output of summation signal as carry-in signal is logic "0" is the inverse signal of itself as carry-in signal is logic "1".

Table 1: The truth table of single-bit full-adder with Cin=0 & Cin=1

C_{in}	A	B	S0	C0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

To share the common boolean logic term, we only need to implement one XOR gate with one inverter gate to generate the summation signal pair. For carry generation, one OR gate and one AND gate is required. The correct summation output and carry out can be selected according to the logic state of carry input signal.

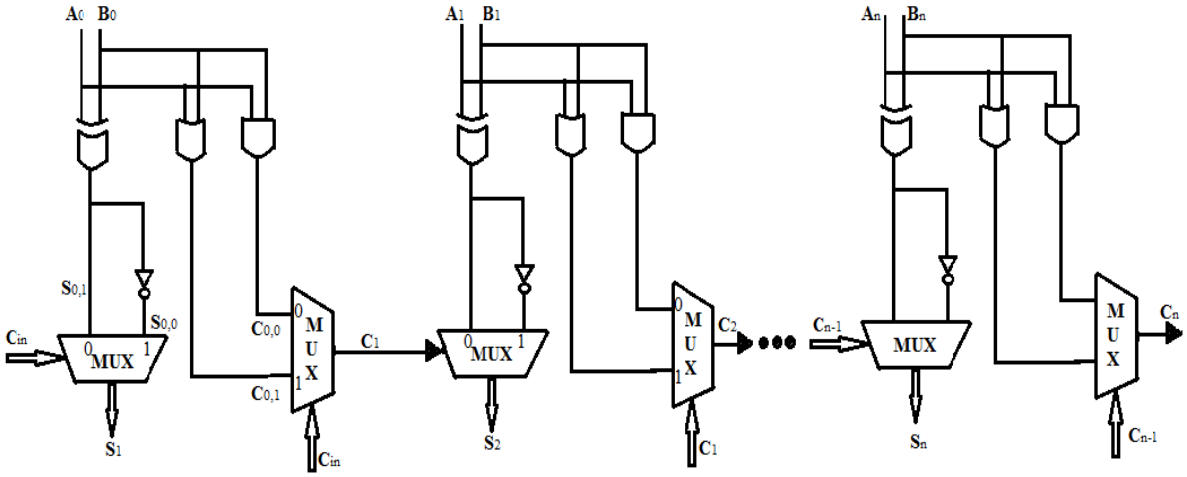

Fig. 2 Carry Select Adder Circuit

2. Carry Look-Ahead Adder

In carry look-ahead architecture instead of rippling the carry through all stages (bits) of the adder, it calculates all carries in parallel based on equation:

$$C_i = \overline{\overline{G_i} \cdot \overline{P_i \cdot C_{i-1}}} \quad (1)$$

In equation (1) the $\overline{G_i}$ and P_i terms are defined as complementary generate and propagate for the ith bit. If carry generate G_i is true then a carry is generated at the ith bit. If carry propagate is true then the carry-in to the ith bit is propagated to the carry-in of ith+1 bit. They are defined by equations (2) and (3) where Ai and Bi are the binary inputs of ith stage.

$$\overline{G_i} = \overline{A_i \cdot B_i} \quad (2)$$

$$P_i = A_i \oplus B_i \quad (3)$$

The sum generator XOR's the carry-in calculated from the previous two bits and the propagate signal of the current two bits; hence, the name carry look-ahead adder. The sum is given by the relation

$$S_i = P_i \oplus C_{i-1} \quad (4)$$

The circuit diagram of Carry Look Ahead adder is shown in Fig. 3.

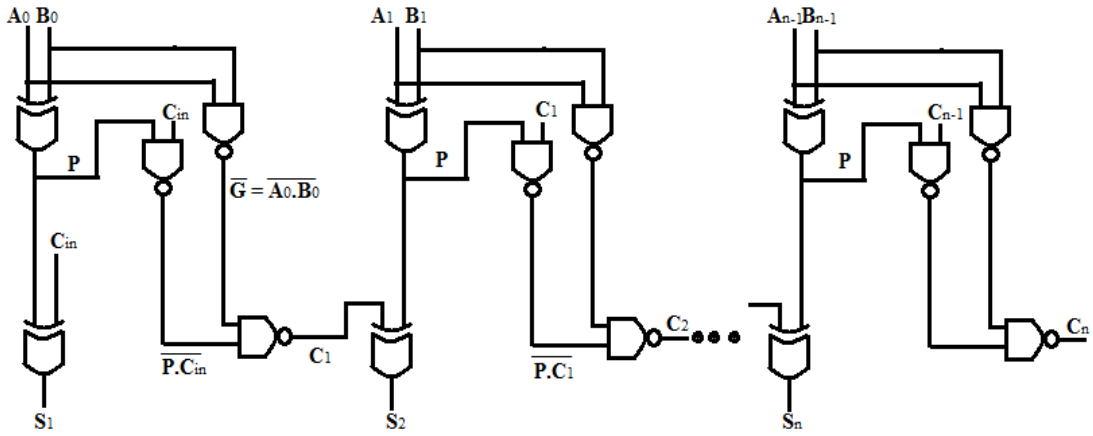

Fig. 3 Carry Look-Ahead Adder Circuit

3. Carry Skip Adder

The carry-skip or carry-bypass adder is much like the ripple carry adder, except that it has a carry bypass path. This structure divides the bits of the adder into an even number of stages M where each stage has a carry bypass path that forwards the carry-in of the M_i stage to the first carry-in of the M_{i+1} stage. If the binary inputs are such that the carry would normally ripple (or propagate) from the input of the M_i stage to the input of the M_{i+1} stage, then the carry takes the bypass path. The carry output of ith stage is calculated as:

$$C_i = \overline{\overline{G_i} \cdot \overline{P_i.P_{i-1}....P_0.C_i}} \tag{5}$$

Where P_i is the propagate signal and $\overline{G_i}$ is the complementary generate signal and is given as:

$$P_i = A_i \oplus B_i \tag{6}$$

$$\overline{G_i} = \overline{A_i \cdot B_i} \tag{7}$$

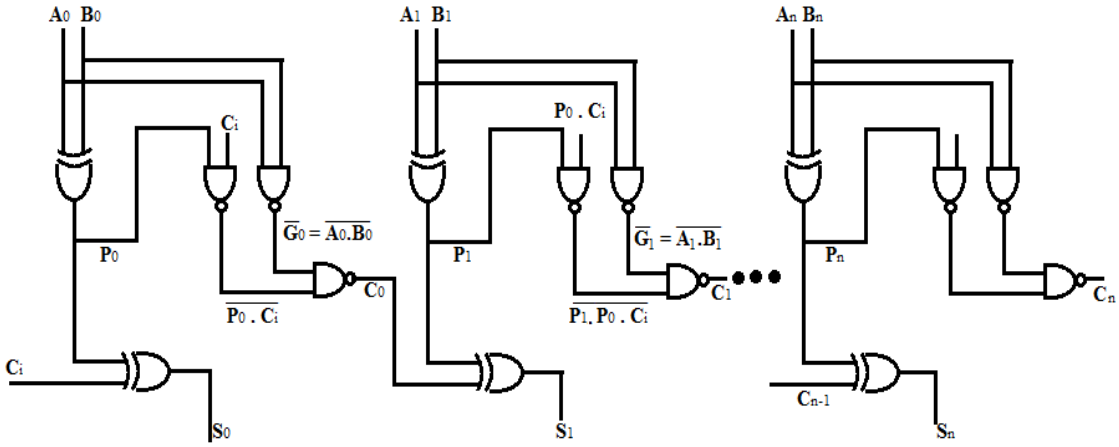

Fig. 4 Carry Skip Adder Circuit

4. Carry Increment Adder

The carry increment adder computes the carry and sum values for a block carry in value of 0 i.e. by assuming carry input value 0 and then increment these values depending on the final block carry in. First of all complementary generate ($\overline{G_i}$) and propagate signals (p_i) are derived using inputs A_i and B_i at each stage. Afterwards carry of each stage is calculated. The equation for the carry output of the ith stage is given as:

$$C_i = C_i^0 + \overline{P_i} + \overline{C_{ib}} \tag{8}$$

where

$$C_i^0 = \overline{\overline{G_i} \cdot \overline{p_i \cdot C_{i-1}^0}} \tag{9}$$

$$P_i = \overline{p_i \cdot P_{i-1}} \tag{10}$$

$$p_i = A_i \oplus B_i \tag{11}$$

and

$$\overline{G_i} = \overline{A_i \,.\, B_i} \tag{12}$$

The sum of a stage is derived using propagate signal (p_i) of that particular stage and carry output of previous stage according to the following equation:

$$S_i = p_i \oplus Ci\text{-}1 \tag{13}$$

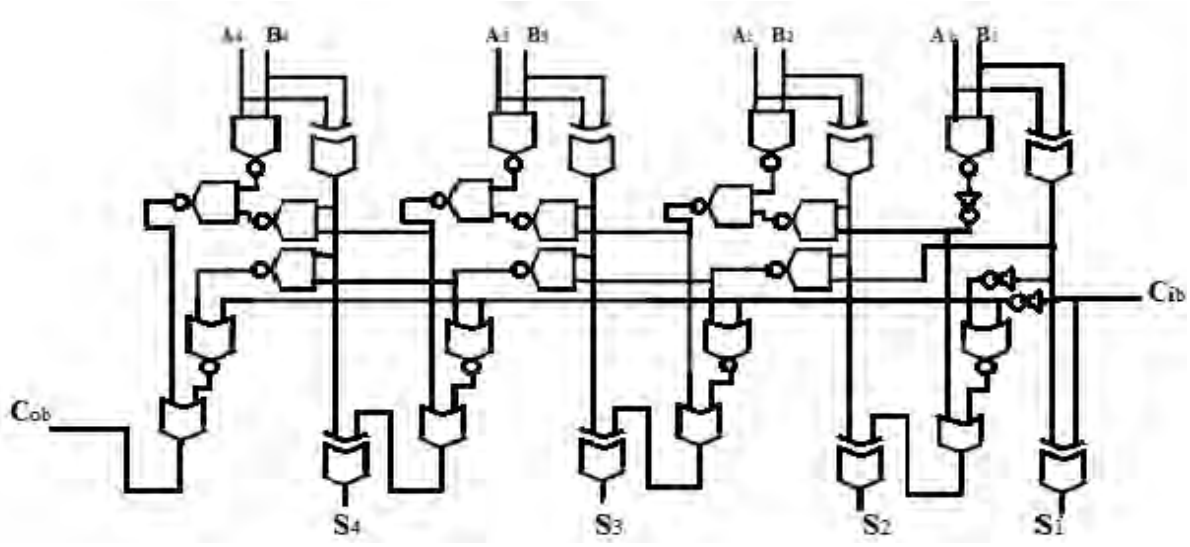

Fig. 5 Carry Increment Adder Circuit

5. Hybrid Prefix Adder

Prefix adders (PPA) are family of adders derived from the commonly known carry look ahead adders. These adders are best suited for adders with wider word lengths. PPA circuits use a tree network to reduce the latency to O(log2 N) where 'N' represents the number of bits. A three step process is involved in the construction of hybrid prefix adder. The first step involves the creation of kill (Ki) and complementary generate ($\overline{G_i}$) using the equations:

$$K_i = \overline{A_i + B_i} = \overline{A_i} \,.\, \overline{B_i} \tag{14}$$

$$\overline{G_i} = \overline{A_i \,.\, B_i} \tag{15}$$

In the above equations, A_i , B_i represent input operand bits for the adder, where 'i' varies from 0 to 7. The Propagate signal (Pi) is derived using the generate and kill signals and is given by

$$P_i = \overline{G_i + K_i} \tag{16}$$

For deriving carry signals in the second stage, this architecture has four different computation nodes. There are two cells for dot operator. First cell for the dot operator named odd-dot represented by a ' ■', is defined by the equation:

$$(g,\bar{k}) = (\bar{g_i},k_i) \blacksquare (\bar{g_{i-1}},k_{i-1}) = (\overline{\bar{g_i} \cdot (k_i + \bar{g_{i-1}})},\ \overline{k_i + k_{i-1}}) \tag{17}$$

The second cell for the dot operator named even-dot represented by a ' □ ', is defined by the equation:

$$(\bar{g},k) = (g_i,\bar{k_i}) \square (g_{i-1},\bar{k_{i-1}}) = (\overline{g_i + \bar{k_i}\ g_{i-1}},\ \overline{\bar{k_i} \cdot \bar{k_{i-1}}}) \tag{18}$$

There are two cells for the semi-dot operator. First cell for the semi-dot operator named odd-semi dot represented by a '•' and the second cell for the semi-dot operator named even-semi dot represented by a ' ⬭', are defined using following equations:

$$(g) = (\bar{g_i},k_i) \bullet (\bar{g_{i-1}},k_{i-1}) = (\overline{\bar{g_i} \cdot (k_i + \bar{g_{i-1}})}) \tag{19}$$

$$(\bar{g}) = (g_i,\bar{k_i}) \,\square\, (g_{i-1},\bar{k_{i-1}}) = (\overline{g_i + \bar{k_i}\ g_{i-1}}) \tag{20}$$

The odd-semi-dot cells gives the value of the carry signal in that corresponding bit position and even-semi-dot cell gives the complemented value of carry signal in that corresponding bit position. The final stage involves generation of sum bits from the derived Propagate signal (Pi) of the individual operand bits and the carry bits

generated in true form or complement form. The architecture of the proposed prefix adder is shown in Fig. 6. The pair of inverters are represented by a '◆' in the architecture.

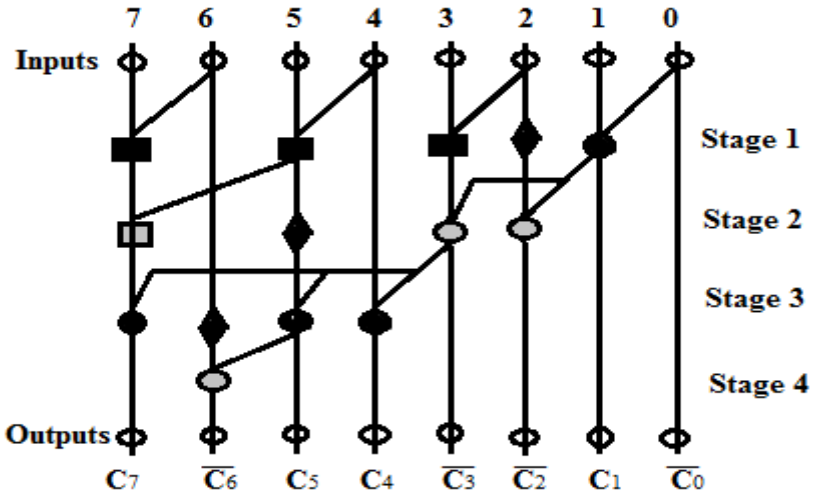

Fig. 6 Hybrid Prefix Adder Structure

6. Proposed Prefix Adder

The proposed prefix adder involves a three step process. At the first step of the proposed prefix circuit, two operations are performed for the creation of complementary generate ($\overline{G_i}$) and propagate signal (P_i). The complementary generate and propagate signal is defined using the following equations:

$$\overline{G_i} = \overline{A_i \,.\, B_i} \tag{21}$$

$$P_i = A_i \oplus B_i \tag{22}$$

At the second stage, the carry signals are derived using complementary generate and propagate signal. For carry generation this circuit uses four computation nodes. The equations for first cell for the dot operator named odd-dot represented by ' ■' and second cell named even-dot operator represented by ' □' are as follows:

$$(g,P) = (\overline{g_i}\,,\overline{P_i}) \blacksquare (\overline{g_{i-1}},\overline{P_{i-1}}) = \overline{\overline{g_i}(\overline{P_i}+\overline{g_{i-1}})},\ \overline{P_i+P_{i-1}} \tag{23}$$

$$(\overline{g},\overline{P}) = (g_i\,,P_i) \square (g_{i-1},P_{i-1}) = \overline{g_i+P_i.g_{i-1}}\ ,\ P_i+P_{i-1} \tag{24}$$

There are two cells for semi dot operator. First cell for semi dot operator named odd semi dot operator represented by '•' is defined as:

$$g = (\overline{g_i}\,,\overline{P_i}) \bullet (\overline{g_{i-1}},\overline{P_{i-1}}) = \overline{\overline{g_i}(\overline{P_i}+\overline{g_{i-1}})} \tag{25}$$

And the second cell for semi dot operator named even semi dot operator represented by ' ○ ' is defined using equation:

$$\overline{g} = (g_i\,,P_i) \circ (g_{i-1},P_{i-1}) = \overline{g_i+P_i.g_{i-1}} \tag{26}$$

The structure of the proposed prefix adder is shown in Fig. 7. The output of the odd semi dot cells gives the value of the carry signal in that corresponding bit position. The output of even semi dot cells gives the complemented value of the carry signal in that corresponding bit position. The pair of inverters are represented by a '◆' in the structure. At the final stage, sum is calculated from the derived propagate signal (P_i) and carry bits generated at each stage using XOR gate.

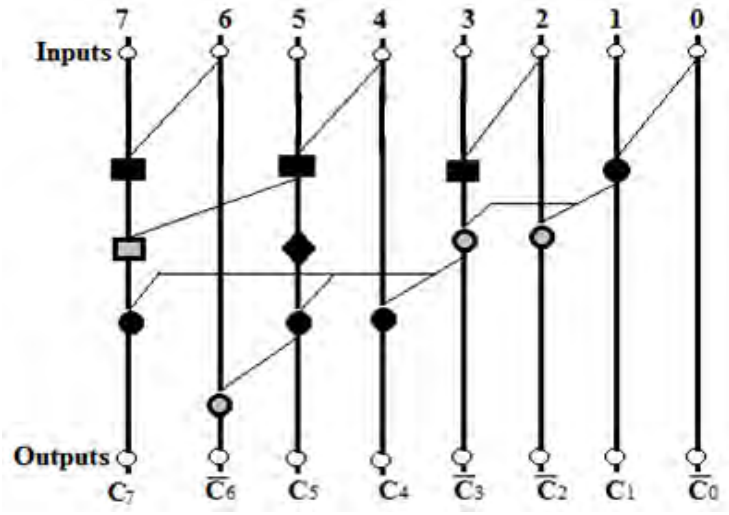

Fig. 7 Proposed Prefix Adder Structure

IV SIMULATION RESULTS

The circuits of 8-bit adder designs are realized using Tanner EDA tool. The average power and delay of each circuit is obtained by applying different set of voltages. The product of power and delay is computed to present power-delay product (PDP). The comparison of the adder circuits based on different parameters is presented.

1. Transistor Count

The area of a CMOS circuit depends on the number of transistors used in the circuit. The transistor count of different adder circuits is given below:

- Carry Select Adder 256
- Carry Look-ahead Adder 192
- Carry Skip Adder 206
- Carry Increment Adder 290
- Hybrid Prefix Adder 296
- Proposed Prefix Adder 276

2. Average Power

Table 2: Average Power Comparison of Adders using 90nm

Voltage / Adder	1V	1.5V	2V	2.5V
Carry Select Adder	25.101 uW	57.562 uW	117.821 uW	216.31 uW
Carry Look-ahead Adder	17.239 uW	38.844 uW	74.541 uW	133.99 uW
Carry Skip Adder	18.889 uW	42.35 uW	81.145 uW	144.182 uW
Carry Increment Adder	21.807 uW	48.512 uw	94.089 uW	165.226
Hybrid Prefix Adder	5.225 uW	15.673 uW	54.93 uW	134.13 uW
Proposed Prefix Adder	10.299 uW	23.719 Uw	46.713 uW	83.535 uW

3. Delay

Table 3: Delay Comparison of Adders using 90nm

Adder \ Voltage	1V	1.5V	2V	2.5V
Carry Select Adder	1.3103 ns	0.5037 ns	0.349 ns	0.2878 ns
Carry Look-ahead Adder	1.6343 ns	0.6572 ns	0.4556 ns	0.374 ns
Carry Skip Adder	1.9483 ns	0.8046 ns	0.5707 ns	0.4719 ns
Carry Increment Adder	0.5096 ns	0.1653 ns	0.1177 ns	0.1014 ns
Hybrid Prefix Adder	0.2932 ns	0.2167 ns	0.1109 ns	0.1632 ns
Proposed Prefix Adder	0.605 ns	0.2601 ns	0.1727 ns	0.1347 ns

4. Power-Delay Product

Table 4: Power-Delay Product Comparison of Adders using 90nm

Adder \ Voltage	1V	1.5V	2V	2.5V
Carry Select Adder	$32.889 * 10^{-15}$	$28.993 * 10^{-15}$	$41.119 * 10^{-15}$	$62.254 * 10^{-15}$
Carry Look-ahead Adder	$28.173 * 10^{-15}$	$25.528 * 10^{-15}$	$33.96 * 10^{-15}$	$50.112 * 10^{-15}$
Carry Skip Adder	$36.801 * 10^{-15}$	$34.074 * 10^{-15}$	$46.309 * 10^{-15}$	$68.039 * 10^{-15}$
Carry Increment Adder	$11.112 * 10^{-15}$	$8.019 * 10^{-15}$	$11.074 * 10^{-15}$	$16.753 * 10^{-15}$
Hybrid Prefix Adder	$1.5312 * 10^{-15}$	$3.396 * 10^{-15}$	$6.0917 * 10^{-15}$	$21.890 * 10^{-15}$
Proposed Prefix Adder	$6.23 * 10^{-15}$	$6.169 * 10^{-15}$	$8.067 * 10^{-15}$	$11.252 * 10^{-15}$

5. Comparison of Hybrid Prefix Adder and Proposed Prefix Adder using 180nm at 1.8V

Table 5: Performance Comparison using 180nm

Adder	Average Power	Delay	PDP
Hybrid Prefix Adder	48.835 uW	0.76 ns	$37.1146 * 10^{-15}$
Proposed Prefix Adder	32.742 uW	0.696 ns	$22.788 * 10^{-15}$

V CONCLUSION

From the simulation results it is clear that for all the adder designs, the carry increment adder emerges as the fastest adder design at all voltages except 1V but at the cost of transistor count which is quite high i.e. 290. The carry select adder is faster than carry look-ahead and carry skip adder but because of high average power the overall PDP increases. At 90nm the proposed prefix adder shows approximate 28% improvement in average power at 2.5V then the hybrid prefix adder and 15% improvement at 2V. At 1.5V and 1V it has higher average power than hybrid prefix adder. The proposed prefix adder has 17% less delay at 2.5V. It does not show improvement in delay at lower voltages and the delay increases as we decrease the voltage. At 2V and 1.5V it is 55% and 20% slower than hybrid prefix adder. At 1V the delay of proposed prefix adder is approximately double than that of hybrid prefix adder. Hence the proposed prefix adder shows a significant improvement in PDP at 2.5V but it has higher value of PDP at lesser voltages than hybrid prefix adder. At 180nm the proposed prefix adder shows 33% improvement in average power using 1.8V and 8% improvement in delay then the hybrid prefix adder. The power-delay product of proposed prefix adder at 180nm technology is 39% less than the prefix adder. Looking at each of the results for the various parameters, an adder suitable to particular criteria may be chosen.

REFERENCES

[1] A. Kumar and A.K. Goyal, "Study of various full adders using tanner EDA tool", International Journal of Computer Science and Technology, Vol. 3, pp. 581-585, 2012.

[2] A.N. Nagamani and B.K. Shivanand, "Design and performance evaluation of hybrid prefix adder and carry increment adder in 90nm regime", IEEE International Conference on Nanoscience, Engineering and Technology, pp. 198-201, 2011.

[3] B. Ramkumar and H.M. Kittur, "Low-power and area-efficient carry select adder", IEEE Transactions on Very Large Scale Integration Systems, Vol. 20, No. 2, pp. 371-375, February 2012.

[4] B. Vijayarani, "Design of synchronous full adder", International Conference on Computing and Control Engineering, April 2012.

[5] D. Garg and M.K. Rai, "CMOS based 1-bit full adder cell for low-power delay product", International Journal of Electronics Communication and Computer Technology, Vol. 2, No. 4, pp. 18-23, 2012.

[6] H. Lee and G.E. Sobelman, "A new low-voltage full adder circuit", Proceedings. Seventh Great Lakes Symposium on VLSI, pp. 88-92, 1997.

[7] I-Chyn Wey, Cheng-Chen Ho, Yi-Sheng Lin and Chien-Chang Peng, "An area-efficient carry select adder design by sharing the common Boolean logic term", Proceedings of the International Multiconference of Engineers and Computer Scientists, Vol. II, March 2012.

[8] J. Samanta, M. Halder and B. Prasad, "Performance analysis of high speed low power carry look-ahead adder using different logic styles", International Journal of Soft Computing and Engineering, Vol. 2, No. 2, pp. 330-336, Jan 2013.

[9] Km. Deepmala, T. Sharma, K.G. Sharma, K.G. and Prof.B.P. Singh, "New efficient 2T AND gate design", Proceedings of the International Conference on VLSI, Communication & Instrumentation, 2011.

[10] M. Alioto and G. Palumbo, "Analysis and comparison on full adder block in submicron technology", IEEE Transactions on VLSI System, Vol. 10, No. 6, pp. 806-823, December 2002.

[11] N. Agarwal and S. Anand, "Logical effort to study the performance of 32-bit heterogeneous adder", International Journal of Computer Applications, Vol. 65, No. 16, pp. 36-38, March 2013.

[12] P. Kumar and M. Vashishath, "Analytical comparison of different 1-bit full adder's scheme for 250nm CMOS technology", International Journal of Engineering Sciences, pp. 94-98, 2012.

[13] P. Ramanathan and P.T. Vanathi, "Hybrid prefix adder architecture for minimizing the power delay product", International Journal of Electrical and Computer Engineering, 4:9, 2009.

[14] R. Zimmermann and W. Fichtner, "Low power logic styles: CMOS versus Pass-Transistor logic", IEEE Journal of Solid-State Circuits, Vol. 32, No. 7, pp. 1-12, July 1997.

[15] S. Maity, B. Prasad and A.K. Singh, "Design and implementation of low-power high performance carry skip adder", International Journal of Engineering and Advanced Technology, Vol. 1, pp. 212-218, April 2012.

[16] S. Panda, A. Banerjee, B. Maji and Dr. A. K. Mukhopadhyay, "Power and delay comparison in between different types of full adder circuits", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 1, pp. 168-172, September 2012.

[17] V. Gowrishankar, D. Manoranjitham and P. Jagadeesh, "Efficient FIR filter design using modified carry select adder & wallace tree multiplier", International Journal of Science, Engineering and Technology Research, Vol. 2, pp. 703-711, March 2013.

[18] V. Kamalakannnan, Dr. P.V. Rao and V. Patil, "Low power and reduced area carry select adder", International Journal of Advances in Electrical and Electronics Engineering, Vol. 1, No. 2, pp. 128-133, 2012.

[19] Y.S. Reddy and V.V.G.S. Rajendra Prasad, "Comparison of CMOS and adiabatic full adder circuits", International Journal of Scientific & Engineering Research, Vol. 2., pp. 1-5, September 2011.