Design and Characterization of Parallel Prefix Adders

Mr.M.Pavan Kumar Reddy,(M.Tech),
Dept of ECE,
Srinivasa Institute of Technology &
Science,Kadapa.

Guide: Mr K.BALA, M.Tech,
Associate Professor,
Srinivasa Institute of Technology &
Science,Kadapa.

ABSTRACT:

While designing Very Large Scale Integration (VLSI) circuits having thousands of transistors, Parallel prefix adders (PPA) play a crucial role by providing better delay performance. In this paper we have observed the characteristics of four types of Parallel prefix adders (PPA). They are:

1.Kogge Stone Adder (KSA),

2.Spanning Tree Adder (STA),

3.Sparse Kogge Stone Adder (SKA) and

4.Brent Kung Adder (BKA).

Apart from the above popular adders, we have observed the performance and characteristics of Ripple Carry Adder (RCA), Carry Lookahead Adder (CLA) and Carry Skip Adder (CSA). These adders arestudied by simulating them in verilog Hardware Description Language (HDL) using Xilinx Integrated Software Environment (ISE) 14.7 Design Suite. These designs are simulated and executed in Xilinx Virtex 5 FieldProgrammable Gate Arrays (FPGA). We have compared all the above adders in terms of their characteristics like adder's delay, power and area.

Keywords : VLSI, Delay, parallel prefix adders, FPGA, Performance

I. INTRODUCTION:

VLSI began in the 1970s when complexsemiconductor and communication technologies were being developed. The microprocessor is a VLSI device. Before the introduction of VLSI technology most ICs had a limited set of functions they could perform. An electronic circuit might consist of a CPU, ROM, RAM and other glue logic. VLSI lets IC makers add all of these into one chip.Binary arithmetic is integral part of all the digital computers and many other digital systems. 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 operations.

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, it is trivial to modify an adder into an adder–subtractor. Other signed number representations require a more complex 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 previous less significant stage. The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. bit binary numbers. The circuit produces a two-bit output, output carry and sum typically represented by the signals C_{out} and S, where

$$\mathrm{sum} = 2 \times C_{out} + S \, .$$

Great amount of research is being conducted around the world to design and manufacture adders with better delay performance. A full adder can be constructed from two half adders by connecting A and B to the input of one half adder, connecting the sum from that to an input to the second adder, connectingC_i to the other input and OR the two carry outputs. The critical path of a full adder runs through both XOR-gates and ends at the sum bit . Assumed that an XOR-gate takes 3 delays to complete, the delay imposed by the critical path of a full adder is equal to

$$T_{FA} = 2 \cdot T_{XOR} = 2 \cdot 3D = 6D$$

The carry-block subcomponent consists of 2 gates and therefore has a delay of

$$T_c = 2D$$

When compared to microprocessor and DSP's dependent solutions, Field Programmable Gate Arrays (FPGA) provide better performance in terms of speed and power for all major real world applications including but not limited to smart phones, tablets, PC and other communication and digital devices. Moreover, power performance is also a significantfeature that is being demanded by end users of mobile phones and smart phones. This in turn is putting a lot pressure on the manufacturers of electronic devices, which makes large-scale use of DSP functions.

Since we can Program, arrange the configurable logic blocks (CLB) and programming interconnects in Field Programmable Gate Arrays (FPGA)s, Parallel prefix adders delivers better performance. The delays of the adders are discussed in detail [1]. In this research paper, four major Parallel prefix adders (PPA)s, Ripple Carry Adder (RCA)and Carry Skip Adder (CSA) are executed and analyzed on a Xilinx virtex 5 FPGA. The basic characteristics like delay, power and area for the designed adders are calculated and evaluated against each other.

II. DISADVANTAGES OF RIPPLE CARRY AND CARRY LOOKAHEAD ADDER :

The layout of a ripple-carry adder is simple, which allows for fast design time; however, the ripple-carry adder is relatively slow, since each full adder must wait for the carry bit to be calculated from the previous full adder. The gate delay can easily be calculated by inspection of the full adder circuit. Each full adder requires three levels of logic. In a 32-bit ripple-carry adder, there are 32 full adders, so the critical path (worst case) delay is 2 (from input to carry in first adder) + 31 * 3 (for carry propagation in later adders) = 95 gate delays. The general equation for the worst-case delay for a n-bit carry-ripple adder is

$$T_{CRA}(n) = T_{HA} + (n-1)\cdot T_c + T_s = T_{F}$$

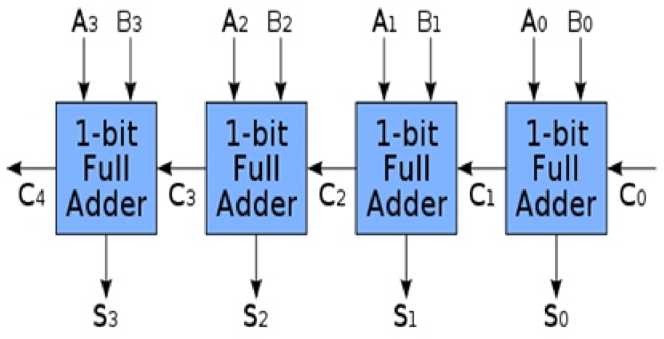
The delay from bit position 0 to the carry-out is a little different:

$$T_{CRA_{[0:c_{out}]}} = T_{HA} + n \cdot T_c = 3D + n$$

The carry-in must travel through n carry-generator blocks to have an effect on the carry-out

$$T_{CRA_{[c0:cn]}}(n) = n \cdot T_c = n \cdot 2D$$

A design with alternating carry polarities and optimized AND-OR-Invert gates can be about twice as fast.



III. COMPARISION BETWEEN PARALLEL-PREFIX ADDERS AND OTHERS:

The PPA's pre-computes generate and propagate signals are explained in detail in [2]. Using the fundamental carry operator (fco), these computed signals are combined in [3].The fundamental carry operator is denoted by the symbol "o ",

(g L , p L)|(g R , p R) =(g L +p L . g R , p L , p R)(1)

For example, 4 bit CLA carry equation is given by

C4=(g 4 , p 4) |[(g 3 , p 3) |[(g 4 , p 4) |(g 3 , p 3)]] (2)

For example, 4 bit PPA carry equation is given by

C4=[(g 4 , p 4) |(g 3 , p 3)] |[(g 4 , p 4) |(g 3 , p 3)] (3)

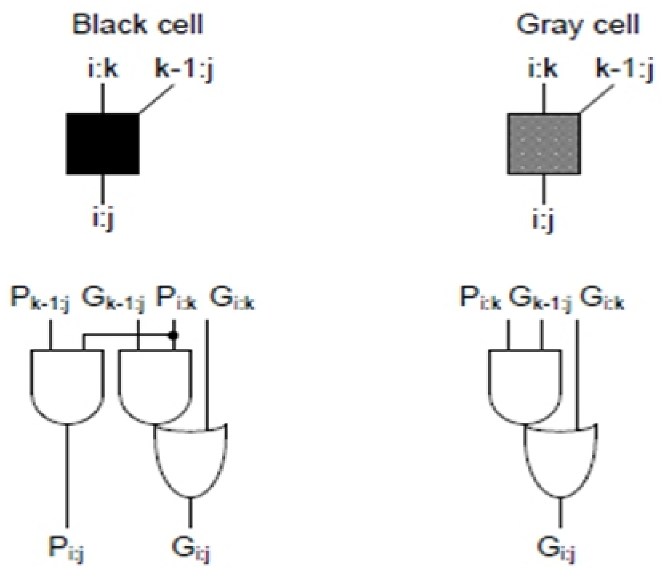

Black cell
i:k
k-1:j
i:j
Gray cell
i:k
k-1:j
i:j
$P_{k-1:j}$ $G_{k-1:j}$ $P_{i:k}$ $G_{i:k}$
$P_{i:j}$
$G_{i:j}$
$P_{i:k}$ $G_{k-1:j}$ $G_{i:k}$
$G_{i:j}$

Fig 3. Black and Gray Cell Logic Definitions Equations (2) and (3) 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.

IV. STRUCTURE OF PARALLEL-PREFIX ADDER:

Parallel-prefix structures are found to be widespread in high performance adders for the reason that of the delay is logarithmically proportional to the adder width [2].
Parallel prefix adders's(PPA) basically consists of 3 stages

- Pre computation
- Prefix stage
- Final computation

The Parallel-Prefix Structure is shown in figure 2.

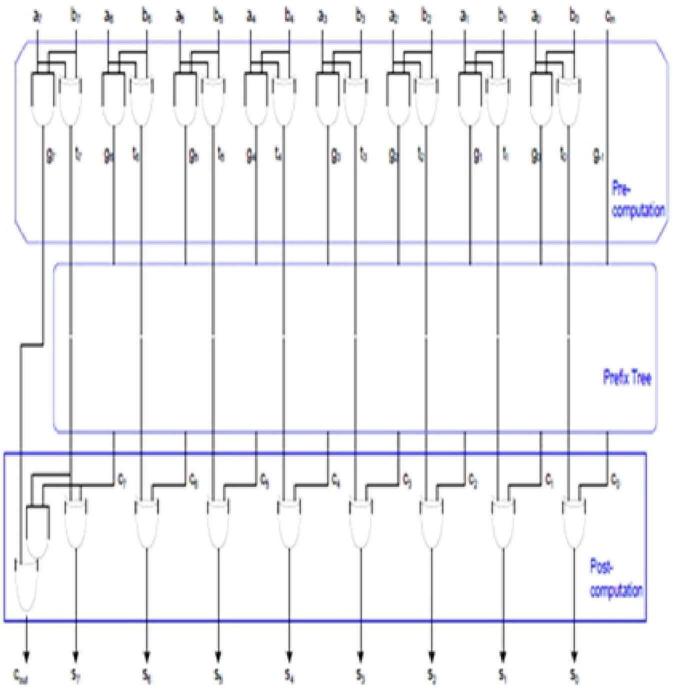


Fig 2. Parallel-Prefix Structure with carry save notation

A. Pre computation
In pre computation stage, propagates and generates are computed for the given inputs.
B. Prefix stage

In the prefix stage, group generate/propagate signals arecomputed at each bit. The black cell (BC) generates the ordered pair, the gray cell (GC) generates only left signal.

$G_i: k=G_{i: j}+P_{i : j} .G_{j-i}: k$

$P_{i:k}=P_{i: j} . P_{j -1:k}$

More practically, the above equations 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.

$G_{i : k}: P_{i : k} =(G\, i : j :P\, i : j)o(G\, j -1: k: P\, j -1: k)$

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.
$S_i =P_i .G_{I -1:-1}$
$C_{out} =G_{n :- 1}$

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.

The 16 bit SKA uses black cells and gray cells as well as full adder blocks too. This adder computes the carries using the BC's and GC's and terminates with 4 bit RCA's. Totally ituses 16 full adders.

Another PPA is known as STA. Like the SKA, this adder also terminates with a RCA. It also uses the BC's and GC's and full adder blocks like SKA's but the difference is the interconnection between them [7].The 16 bit STA is shown in the below figure 5.

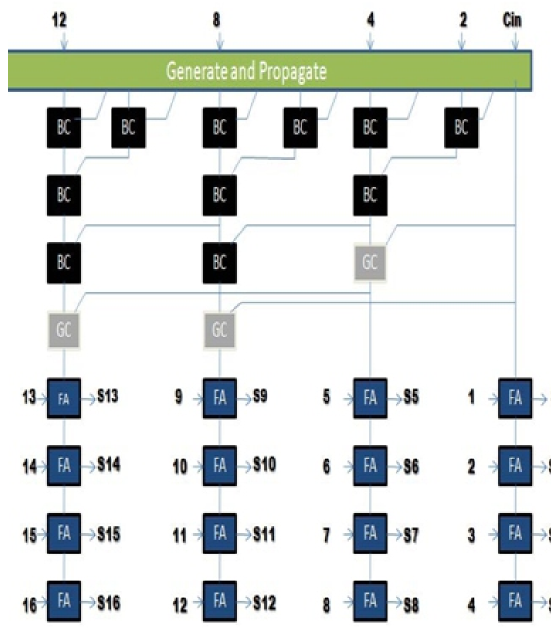


Fig 4. 16-bit Sparse Kogge-Stone Adder

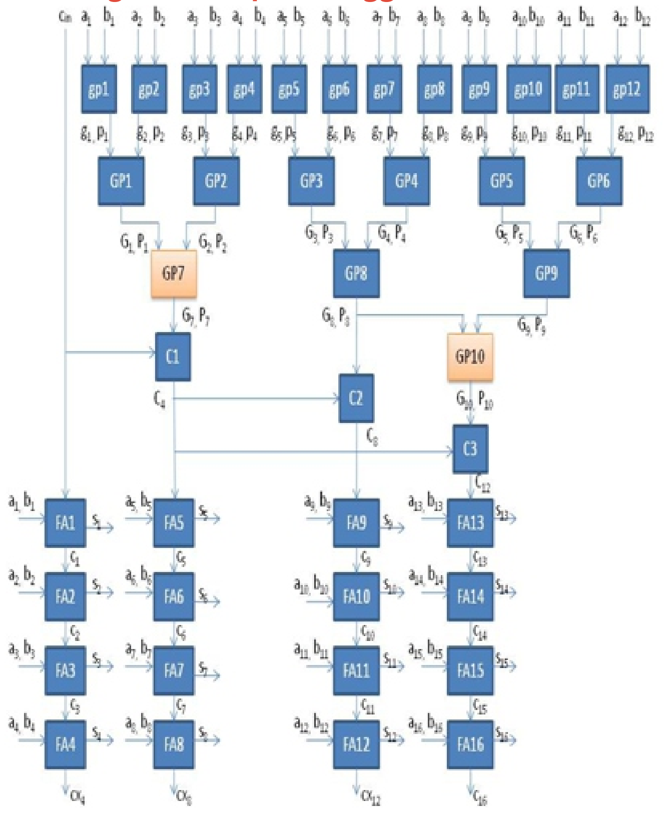


Fig 5. 16 Bit Spanning Tree Adder

KSA is a different of prefix trees that utilise the least logic levels. A 16-bit KSA is shown in Figure 6. The 16 bit kogge stone adder employs BC's and GC's and it won't utilise full adders. The 16 bit KSA uses 36 BC's and 15 GC's. And this adder completely functions on generate and propagate blocks. So the delay is less when weigh against to the previous SKA and STA.

The 16 bit BKA uses 14 BC's and 11 GC's but kogge stone uses 36 BC's and 15 GC's. So BKA has less architecture and occupies less area than KSA. The 16 bit BKA is shown in the below figure 7.

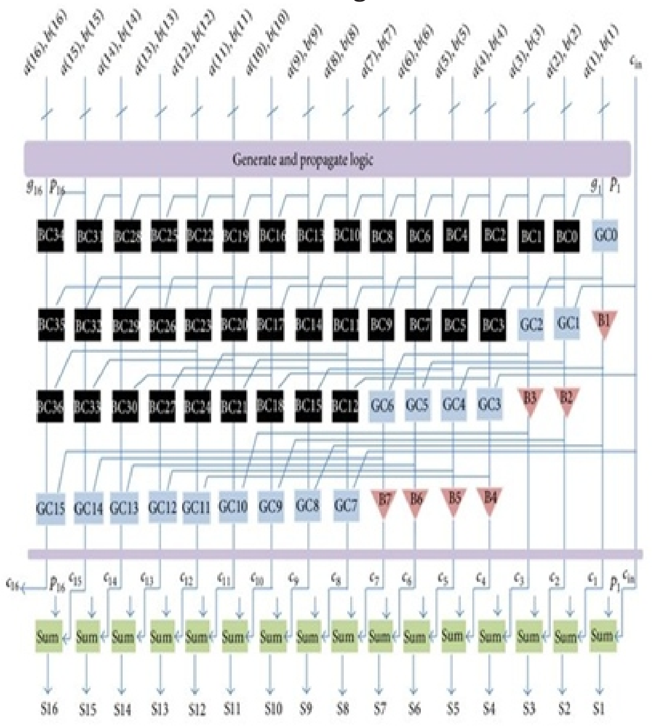


Fig. 6. 16 bit kogge stone adder

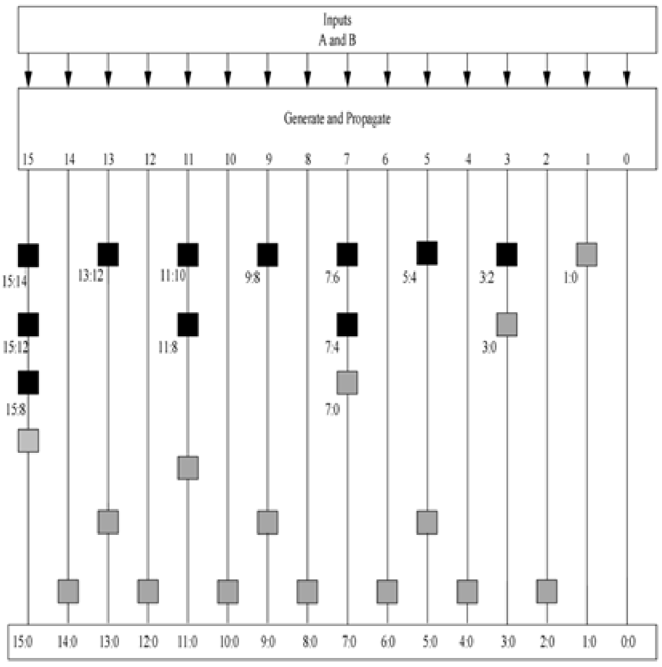


Fig. 7. 16 bit brent kung adder

BKA requires less area than the other 3 adders called SKA,KSA, STA. This adder utilizes limited number of propagate and generate cells than the other 3 adders. It needs less area to implement than the KSA and has less wiring congestion.

V. ANALYSIS AND INTERPRETATION OF RESULTS:

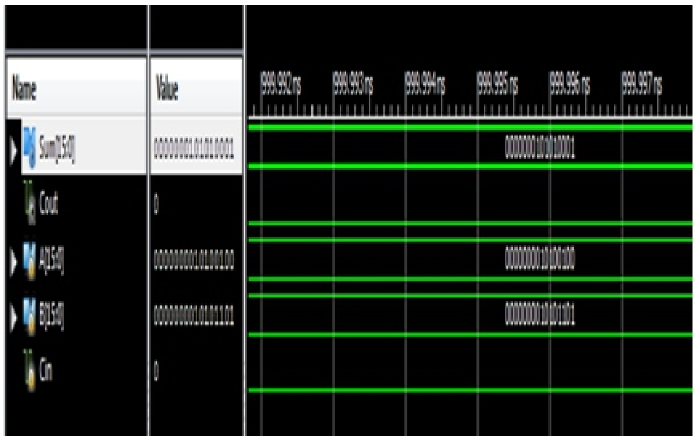


Fig 8.Simulation Result of Adder design in Xilinx ISE 14.7

The delays observed for adder designs from synthesis reports in Xilinx ISE 14.7 synthesis reports are shown in Figure9.

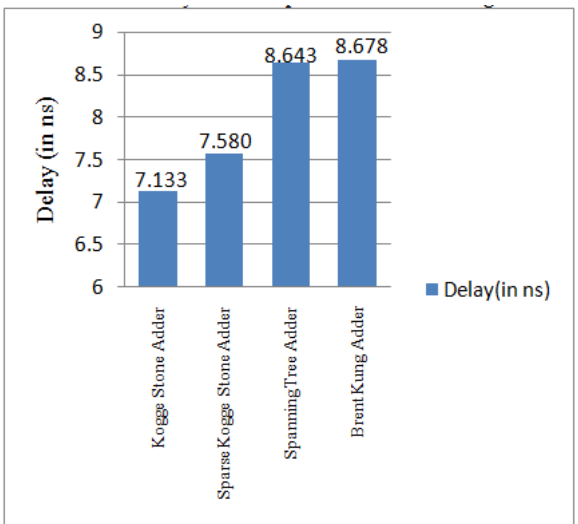


Fig 9. simulation results for the adder designs

The delays observed for adder designs from synthesis reports in Xilinx ISE 14.7 synthesis reports and delays and shown in figure10.

The area of the adder designs is measured in terms of look up tables (LUT) and input output blocks (IOB) taken for Xilinx virtex 5 FPGA is plotted in the figure 10,11. 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 [1]. From the figure 9, the CSA has more delay when compared to other adders. Out of all adders, RCA has less delay. SKA adder and BKA has about the same delay, where as KSA and STA has same delay. According to the synthesis reports, out of four parallel prefix adders, STA has better delay.

The results for area in terms of Look up tables (LUT's) and Input-output blocks (IOB's) are plotted in figure 10,11.For virtex 5 FPGA, available LUT's are 12480 and IOB's are 172. Out of the available LUT's and IOB's, the used LUT's and IOB's are given in the figure 10,11.

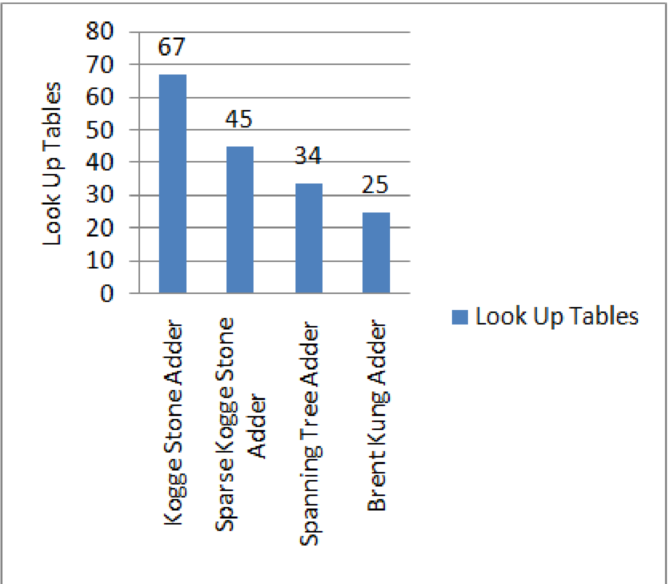


Fig 10 .LUT's observed in Xilinx ISE

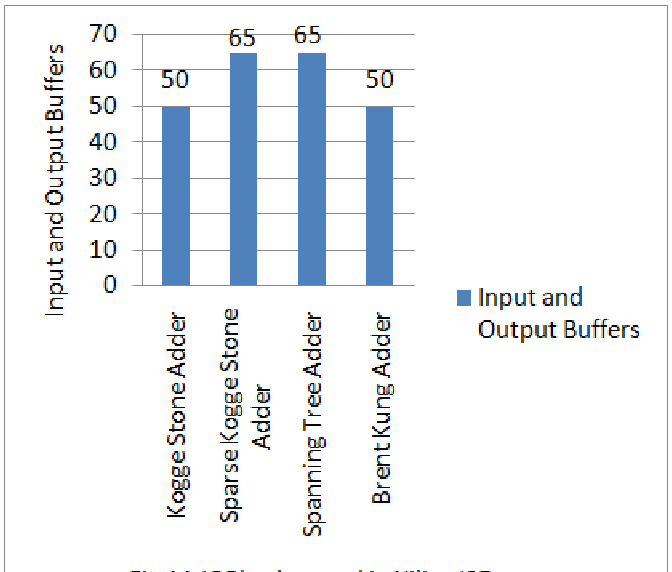


Fig 11.IOB's observed in Xilinx ISE

Out of four PPA's, BKA has better delay and has taken less LUT's and IOB's. Out of four PPA's, BKA has taken less area in terms of LUT's and IOB's. Out of all 7 adders (mentioned), RCA has taken less area. The power for the all mentioned adders is approximately 320.63(mw).

TABLE 1. Comparison of Delay,LUT's and IOB's for Adders

S.no	Adder Name (16 bit)	Xilinx ISE 14.7 Tool Delay (in ns)	Ref [1] Delay (in ns)	Device Utilization (LUT's) (12480)	Device Utilization (IOBs) (172)
1	Ripple carry Adder	4.211	2.578	24	50
2	Carry Lookahead Adder	5.172	-	24	50
3	Carry Skip Adder	4.895	-	29	50
4	Kogge Stone Adder	7.133	6.286	67	50
5	Sparse Kogge Stone Adder	7.580	-	45	65
6	Spanning Tree Adder	8.643	-	34	65
7	Brent Kung Adder	8.678	-	25	50

VI. CONCLUSION and Further Scope:

From the study of analysis done on Delay,area,LUT's& IOB's and power, we have concluded that Out of four PPA's, (Parallel prefix Adders) BKA has taken less area in terms of LUT's and IOB'sandhave the better delay performancewhen compared to all 7 adders (mentioned).

REFERENCES:

[1] David H.K.Hoe, Chris Martinez and Sri JyothsnaVundavalli", Design andCharacterizationof Parallel Prefix Adders using FPGAs", 2011 IEEE 43rd Southeastern Symposium in pp. 168-172, 2011.

[2] N. H. E. Weste and D. Harris, CMOS VLSI Design, 4th edition,Pearson–Addison-Wesley, 2011.

[3]R. P. Brent and H. T. Kung, "A regular layout for parallel adders," IEEETrans. Comput., vol. C-31, pp. 260-264, 1982.

[4] D. Harris, "A Taxonomy of Parallel Prefix Networks," in Proc. 37thAsilomar Conf. Signals Systems and Computers, pp. 2213–7, 2003.

[5]P. M. Kogge and H. S. Stone, "A Parallel Algorithm for the EfficientSolution of a General Class of Recurrence Equations," IEEE Trans. OnComputers, Vol. C-22, No 8, August 1973.

[6]D. Gizopoulos, M. Psarakis, A. Paschalis, and Y. Zorian, "EasilyTestable Cellular Carry Lookahead Adders," Journal of ElectronicTesting: Theory and Applications 19, 285-298, 2003.

[7]T. Lynch and E. E. Swartzlander, "A Spanning Tree Carry LookaheadAdder," IEEE Trans. on Computers, vol. 41, no. 8, pp. 931-939, Aug.1992.

[8]Beaumont-Smith, A, Cheng-Chew Lim ,"Parallel prefix adder design",Computer Arithmetic, 2001. Proceedings. 15th IEEE Symposium,pp.218 – 225,2001.M. Young, The Technical Writer's Handbook. MillValley, CA: University Science, 1989.

[9] K. Vitoroulis and A. J. Al-Khalili, "Performance of Parallel PrefixAdders Implemented with FPGA technology," IEEE NortheastWorkshop on Circuits and Systems, pp. 498-501, Aug. 2007. 172.

[10] S. Xing and W. W. H. Yu, "FPGA Adders: Performance Evaluation andOptimal Design," IEEE Design & Test of Computers, vol. 15, no. 1, pp.24-29, Jan. 1998