International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 3, Issue 5, September-October 2014 **ISSN 2278-6856**

Implementation of 32-Bit Wave Pipelining Sparse Tree Adders

A.Padma Priya[1] , M.Prema kumar[2]

[1]M.Tech.,Student, [2] Associate professor

[1,2] Department of ECE, Shri Vishnu Engineering College for Women, Vishnupur, Bhimavaram, A.P, India.

Abstract

In this paper, we propose 32-bit parallel prefix sparse tree adder. In general N-bit adders like Ripple carry adders(slow adders compare to other adders), and carry lookahead adders(area consuming adders) are used in earlier days. But now the most of industries are using parallel prefix adders because of their advantages compare to other adder, The prefix sparse tree adders are faster and area efficient. Parallel prefix adder is a technique for increasing the speed in DSP processor while performing addition. We simulate and synthesis different types of 32-bit sparse tree adders using Xilinx ISE tool, By using these synthesis results, We noted the performanceparameters like number of LUT's and delay. We compare these three adders interms of LUT's represents area) and delay values..

Index terms: - digital arithmetic, carry skip adder, kogge-stone adder ,carry operator, prefix adder.

1 .INTRODUCTION

Digital addition is a fundamental operation of processorsand digital computer systems, not only to provide basic addition functions but also to provide many other logical operations. Addition and other arithmetic operations are generally performed by an arithmetic logic unit (ALU) contained with the computer's processor unit. The requirements of the adder are that it is primarily fast and secondarily efficient in terms of power consumption. The binary adder is the one type of element in most digital circuit designs including digital signal processors(DSP) and microprocessor data path units. Therefore fast and accurate operation of digital system depends on the performance of adders. Hence improving the performance of adder is the main area of research in VLSI system design. Till now we are using adders like carry look ahead adders, carry skip adders, and kogge stone adders to get fast operation. But this type of adders require much time and area to get operation. To overcome these drawbacks a new structure is implemented that is parallel prefix sparse tree adder. In VLSI implementations, parallel-prefix adders are known to have the best performance. Parallel prefix (or tree prefix) adders provide a good theoretical basis to make a wide range of design trade-offs in terms of delay, area and power. Parallel Prefix Adders (PPA) is designed by considering carry look adder as a base. Here, designing and implementing the 32-bit sparse tree adders on FPGAs are described. This paper is organized as follows; Section II explains the Parallel prefix adders. SectionIII explains existed 16 bit sparse tree adder. Section IV explains 32-bit Sparse-tree adder with detail structure of CSA, CLA and KSA adders respectively. A section V deals with simulation results and comparisons of area and delay.

2.The Parallel prefix adders

Parallel-prefix adders, also known as carry-tree adders[8], pre-compute the propagate and generate signals. These signals are variously combined using the fundamental carry operator (fco).

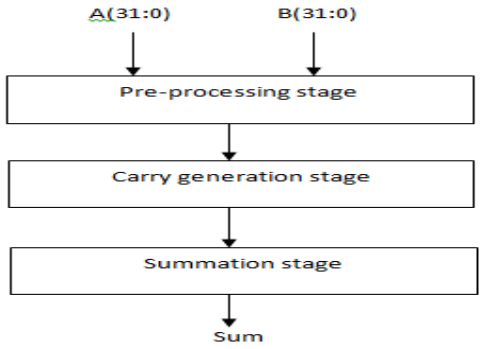

Fig1:block diagram of parallel prefix adder

The parallel prefix adders are more flexible and are used to speed up the binary additions. Parallel prefix adders are obtained from Carry Look Ahead (CLA) structure. We use tree structure form to increase the speed of arithmetic operation. Parallel prefix adders are fastest adders and these are used for high performance arithmetic circuits in industries. The construction of parallel prefix adder involves three stages

1. Pre- processing or initialization stage
2. Carry generation network
3. Post processing or summation stage

Pre-possessing stage

In this stage we compute, generate and propagate signals to each pair of inputs A and B. These signals are given by the logic equations 1&2:

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

$$Gi=Ai \,\&\, Bi \quad (2)$$

Carry generation network

In this stage we compute carries corresponding to each bit. Execution of these operations is carried out in parallel [9]. After the computation of carries in parallel they are segmented into smaller pieces. It uses carry propagate and generate as intermediate signals which are given by the logic equations 3&4:

$$CPi:j=Pi:k+1 \,\&Pk:j \quad (3)$$

$$CGi:j=Gi:k+1 + (Pi:k+1 \,\&Gk:j) \quad (4)$$

summation

This is the final step to compute the summation of input bits. It is common for all adders and the sum bits are computed by logic equation 5&6.

$$Ci-1=(Pi \,\&Cin) +Gi \quad (5)$$

$$Si = Pi \oplus Ci-1 \quad (6)$$

32-bitKogge stone adder

KSA[12] is a parallel prefix form carry look ahead adder. It generates carry in O (logn) time and is widely considered as the fastest adder .it is widely used in the industry for high performance arithmetic circuits. In KSA, carries are computed fast by computing them in parallel at the cost of increased area.

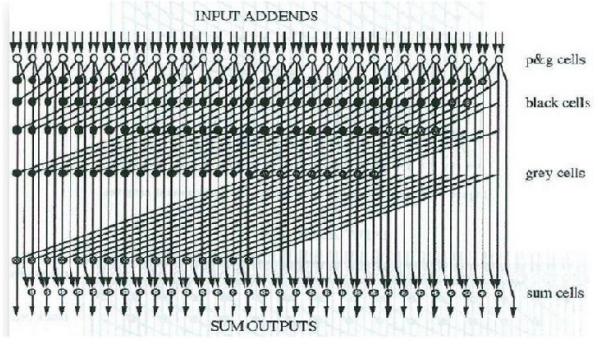

Fig2: structural diagram of 32-bit kogge stone adder.

The 32-bit Kogge–Stone adder is shown in fig2. The structural diagram of 32 bit sparse tree consists of p&g cells, black cells, gray cells and sum cells. The p&g cells produce the propagate and generate bits. So we called this stage as pre processing stage, as said above. In the carry generation stage we have this gray and black cells. The operation of gray cell is same as in eqn(3), whereas the operation of black is same as shown in equation(4) . The sum cells i.e, Xor cells are there in summation stage, to add the computation bits as said in logic equations 5 & 6.

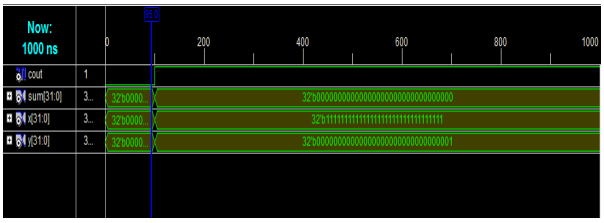

Fig3: simulation result of 32 bit kogge stone adder.

The synthesis results of 32 bit carry skip adder, carry look ahead adder and kogge stone adder with respect to delay and no. of LUTs is shown in below table.

Table1: Comparison of adders

Topology	Delay	No of LUTs	No of slices
32 bit CLA	45.502ns	160	89
32 bit CSA	37.026ns	184	102
32 bit KSA	22.509ns	197	113

The above result shows that these adders occupy large area and the performance is also low. To overcome this drawback a new structure is developed which is sparse tree adder[1].

3.Existed 16-bit sparse tree adder

In the 16-bit adder design, we chose the sparse-tree structure,[1] to reduce the number of wiring junctions needed for its implementation without any significant effect on its processing rate. As a side effect, this will also lead to a more energy-efficient design by reducing the total bias current and power consumption. Fig. 4 illustrates the structural diagram of our sparse-tree adder. It consists of the following three stages: Initialization, Prefix-Tree and Summation.

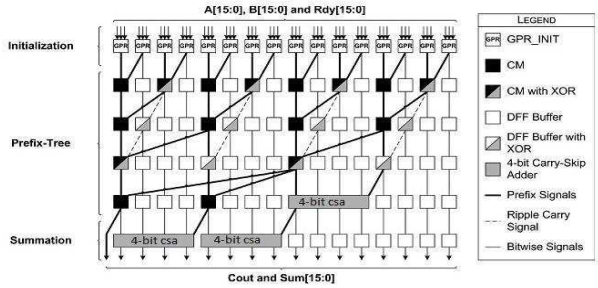

Fig4.Structural diagram of the 16-bit sparse-tree adder.

The Initialization stage receives two 16-bit data operands A and B to create bitwise Generate (G) and Propagate (P) signals. Those will be merged in alogarithmic manner in the Prefix-Tree stage. The initialization stage consists of GPR logic blocks, one for each bit. The GPR creates the bitwise prefix functions described as Gi = Ai • Bi and Pi = Ai ⊕ Bi where i is the bit index column ranging from 16 down to 0 in the 16-bit adder. The clock is the R_{dy}signal provided to all bits additionally, it is necessary to create the trailing reset signal R which will be used to reset the asynchronous elements in the Prefix-Tree. The Prefix-Tree stage consists of Carry-Merge (CM) blocks to merge the prefix signals and provide a group carry to each 4-bit summation block. DFF (D flipflop) buffers appropriately delay prefix and bitwise P signals until they are ready to be merged or processed at the Summation stage, respectively. The first three levels of the Prefix-Tree also perform the ripple-carry addition within each 4-bit group before data arrive at the Summation stage. The Summation stage computes the final sum with 4-bit carry-skip adders . The lower-half of the adder (bits 7:0) can start the Summation stage early because all appropriate signals are ready. The upper-half of the adder (bits 8:15) must wait until carries for this upper half are calculated by the very last level of the *Prefix-Tree* stage. Fig 7 shown below illustrates the carry skip adder block diagram.

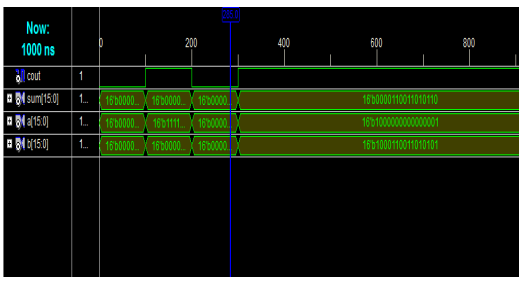

Fig5: Simulated output of 16-bit Sparse-treewith CSA.

4.Proposed 32-bit Sparse Tree adder

The microarchitecture of the 32 bit sparse tree adder[10] has two main features: asynchronous hybrid wave-pipelined processing and a prefix sparse-tree carry generate-propagate structure for arithmetic.

Wave-pipelining:

In data-driven wave-pipelining, data Waves are "self-propagate" through combinational (non-clocked) logic gates without any need for clock signals. The data waves are followed by reset waves that "clean up" the residual logic states of the gates before the next data wave arrival.

A.Sparse Tree adder with CSA

The sparse tree adder[4] is designed to add two 32-bit[3] numbers. The design of this parallel prefix adder is same as 16-bit sparse tree adder. The structural diagram is shown below.

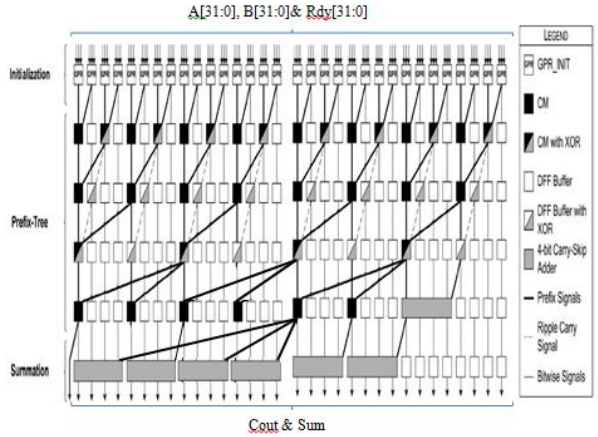

Fig 6.Structural diagram of the 32-bit sparse-tree adder. The carry-out (Cout) is the left-most bit and to the right of it is the most significant bit of the Sumresult (bit 31). The right-most bit is the least significant bit (bit 0).

The carry skip adder structure is shown in fig7. As the name indicates, Carry Skip Adder (CSA)[4] uses skip logic in the propagation of carry . It is designed to speed up the addition operation by adding a propagation of carry bit around a portion of entire adder. The carry-in bit designated as C0. The output of RCA (the last stage) is C4. The Carry Skip circuitry consists of two logic gates. AND gate accepts the carry-in bit and compares it with the group of propagated signals.

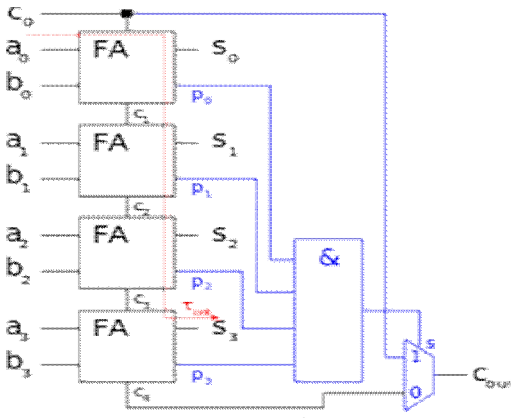

Fig7: structural diagram of carry skip adder.

B.Sparse tree adder with CLA

A Carry Look Ahead adder (CLA) is a type of adder used in digital circuits. A carry-look-ahead adder improves speed by reducing the amount of time required to determine carry bits..

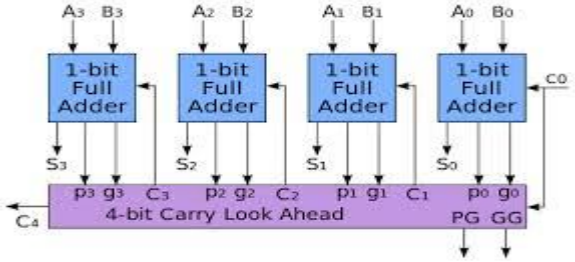

Fig8: structural diagram of carry look ahead adder

It [11] 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. 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. To reduce the computation time, engineers devised faster ways to add two binary numbers by using carry-look ahead adders. They work by creating two signals (P and G) for each bit position, based on if a carry is propagated through from a less significant bit position (at least one input is a '1'), a carry is generated in that bit position (both inputs are '1'), or if a carry is killed in that bit position (both inputs are '0'). In most cases, P is simply the sum output of a half-adder and G is the carry output of the same adder. After P and G are generated the carries for every bit position are created. Some advanced carry-look ahead architectures the Kogge-Stone adder. The modified 32-bit wave pipeline sparse-tree adder by using CLA figure

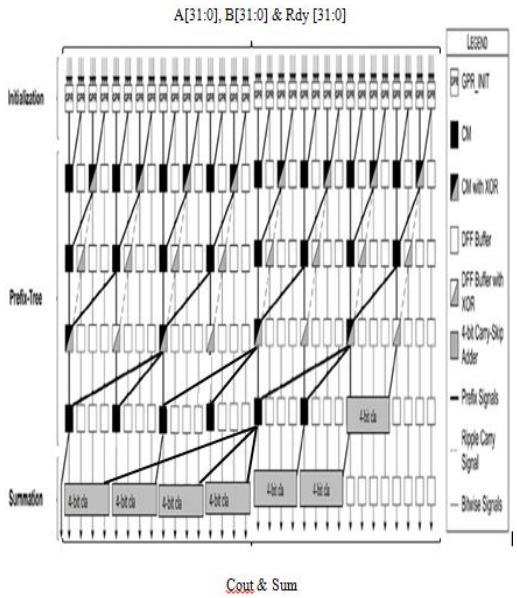

Fig 9: sparse tree adder with carry look ahead adder

C. Kogge-Stone (KS) adder

Kogge-Stone adder [2] is a parallel prefix form carry look ahead adder. The Kogge-Stone adder was developed by peter M. Kogge and Harold S. Stone which they published in 1973. Kogge-Stone prefix adder is a fast adder design. KS adder has best performance in VLSI implementations. Kogge-Stone adder has large area with minimum fan-out. The Kogge-Stone adder is widely known as a parallel prefix adder that performs fast logical addition. Kogge-Stone adder is used for wide adders because of it shows the less delay among other architectures. In fig10 each vertical stage produce Propagate and Generate bits. Generate bits are produced in the last stage and these bits are XORed with the initial propagate after the input to produce the sum bits

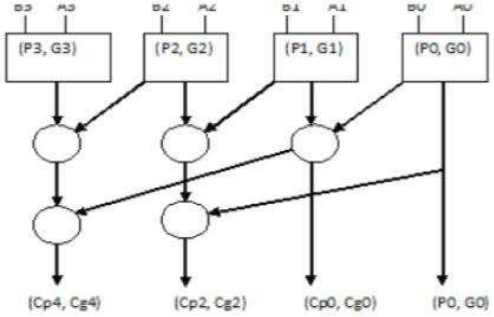

Fig10: structural diagram of 4 bit KSA

In this proposed method modification is done by replacing the parameter 4-bit carry skip adder with 4- bit carry look ahead adder, 4-bit Kogge-Stone adders[5]. By using this logic we can reduce delay and area. The figure9&11 shows structure of modified sparse-tree adder using CLA and Kogge-stone adder logic.

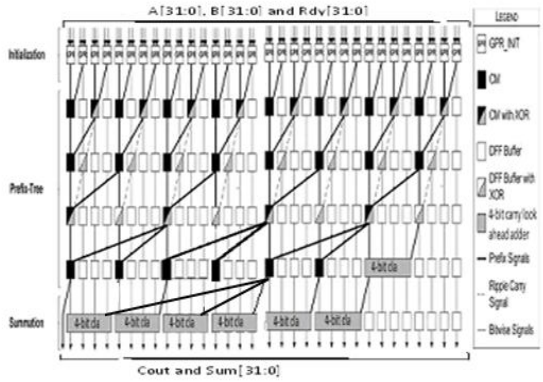

Fig 11: sparsetree adder with kogge_stone adderadder

5. SIMULATION RESULTS AND COMPARISIONS

The design of different tree adders are coded in Verilog Hardware Description Languageusing structural modeling in Xilinx ISE Design Suite and all the simulation results are verified using XILINX ISIM simulator. Synthesized for the Spartan-3 FPGA XC3S500 with the speed grade of 4.. In this proposed architecture, the implementation code for modified 32-bit sparse tree adder by using carry skip adder, Kogge-Stone, and carry look Ahead adders were developed and corresponding values of delay and area were observed. Table1 shows the comparison of adders. The simulatedoutputs of 32-bit proposed adders are shown inFigure 12,13&14.

Table2: Comparisons of Adders

Topology	Delay	No. of LUTs	No of slices
Sparse tree adder with CSA	16.545ns	89	51
Sparse tree adder with CLA	16.498ns	91	51
Sparse tree adder with KSA	15.766ns	100	56

Thus the above table tells that, if speed is main factor then sparse tree adder with kogge stone is preferred. If area is considered sparse tree adder with CSA is preferred.

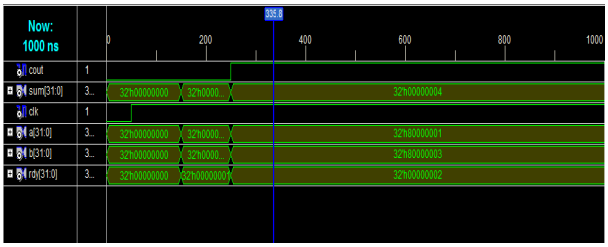

Fig12: Simulated output of 32-bit Sparse-tree with CLA

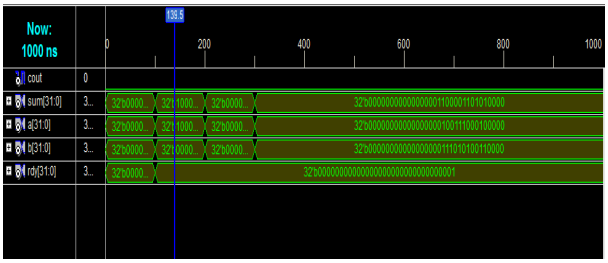

Fig13: Simulated output of 32-bit Sparse-tree with CSA

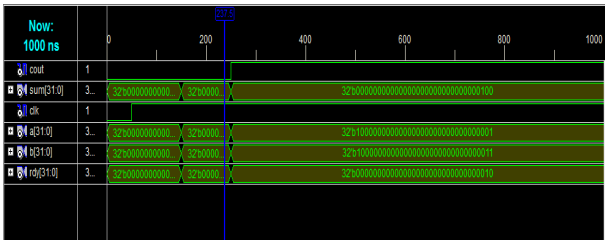

Fig14: Simulated output of 32-bit Sparse-tree withkogge stone adder

6. CONCLUSION

The proposed adders are faster because of less delay and area efficient compared to other basic adders. Among these three prefix adders Sparse-tree adder with carry skip adder

has better performance compared to remaining adders. But in terms of area , sparse tree adder with kogge stone adder using less no. of LUTs. The performance comparisons between these adders are measured in terms of area and delay. It would be interesting to investigate the design of the 64 and 128 bit adders. These adders are popularly used in VLSI implementations

REFERENCES

[1] Mikhail Dorojevets, Christopher L. Ayala, Nobuyuki Yoshikawa, and Akira Fujimaki "16-Bit Wave-Pipelined Sparse-Tree RSFQ Adder" IEEE Trans on APPLIED SUPERCONDUCTIVITY, VOL. 23, NO. 3, JUNE 2013

[2] P.MKogge and H. S. Stone, "A parallelalgorithm for the efficient solution of ageneral class of recurrence equations," IEEETrans. Computer, vol.C-22, no. 8, pp. 786-793, Aug .1973..

[3] S. Mathew, M. Anders, R. K.Krishnamurthy, and S. Borkar, "A 4-GHz130-nm address generation unit with 32-bitsparse-tree adder core," IEEE J. Solid-State Circuits, vol. 38, no. 5, pp. 689–695, May2003..

[4] A. G. M. Strollo and E. Napoli, "A fast andarea efficient complimentary pass-transistorlogic carry-skip adder," in Proc. 21st Int.Conf. Microelectron., Sep. 1997, vol. 2, pp.701–704.

[5] V.KrishnaKumari , Y.SriChakrapani "Designing and Characterization of koggestone, Sparse Kogge stone, Spanning tree and Brentkung Adders", International Journal of Modern Engineering Research (IJMER) www.ijmer.com Vol. 3, Issue. 4, July-august. 2013 pp-2266-2270

[6] Reto Zimmermann. Binary AdderArchitectures for Cell-Based VLSI an theirSynthesis. Hartung-Gorre, 1998.

[7] Y. Choi, "Parallel Prefix Adder Design,"Proc. 17th IEEE Symposium on ComputerArithmetic, pp 90-98, 27th June 2005.

[8] D. Harris, "A taxonomy of parallel prefix networks," in Signals, Systems and Computers, 2003. Conference Record of Thirty Seventh Asilomar Conference on, vol. 2, the Nov. 2003,pp.2217.

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

[10] P.Chaitanyakumari, R.Nagendra "Design of 32 bit Parallel Prefix Adders" ,IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834,p- ISSN: 2278-8735. Volume 6, Issue 1 (May. - Jun. 2013), PP 01-06

[11] D. Gizopoulos, M. Psarakis, A. Paschalis, and Y.Zorian, "Easily Testable Cellular Carry Look ahead Adders," Journal of Electronic Testing: Theory and Applications19-285-298,2003.

[12] Sunil M, Ankith R D"Design and implementation of faster parallel Prefix kogge stone adder ", journal of IJEETC on vol 3 no. 1 2014.

AUTHORS

A Padma Priya received her B.Tech, Degree in Electronics and Communication Engineering and currently pursuing M.Tech at Shri Vishnu Engineering College for Women, bhimavaram, Andhra Pradesh, INDIA. Her main research interest includes vlsi design, adders,parallel prefix adder design, sparsetreeadders

M.Prema Kumar is currently Associate Professor in the Dept of ECE, SVECW, Bhimavaram, AP,India. He received his B.E from SRKR Engg College, Andhra University, Vishakhapatnam, India. M.Tech from IIT Madras, Chennai. He has 12 years of teaching experience to UG students and guided 5 PG theses. Presently pursuing Ph.D from Andhra University.