International Journal of Research Studies in Science, Engineering and Technology
Volume 2, Issue 12, December 2015, PP 22-28
ISSN 2349-4751 (Print) & ISSN 2349-476X (Online)

64 Bit Pipelined Hybrid Sparse Kogge-Stone Adder Using Different Valance

Gangula Thirupathi Reddy[1], Dr. M. K Charan[2]

[1]M. Tech, ECE Department, Malla Reddy Engineering College (Autonomous), Hyderabad, India
[2]Associate Professor, Department of ECE, MREC (Autonomous), JNTUH, Hyderabad, India
[1]thiru.steve@gmail.com, [2]kavicharan@mrec.ac.in

Abstract: *Present architectures concerns to the advancement in the area, power and speed of different adders, different designs starting from the ripple carry adder to the carry select adder and carry tree adders. There are many carry tree structures like kogge-stone, sparse kogge-stone, spanning tree, brunt Kung and linder-fishcher but these consider only two input black cell or white cell there by the stages will be increased when input length increases. In order to overcome the disadvantages that occurred in previous adder methods a novel method called different valency block cells is adopted. With this we design basic adders along with Valance structure and characteristic evaluation can be carried out. Finally 64 bit hybrid sparse kogge-stone adder with different valency block cells can be designed.*

1. INTRODUCTION

Adder is an important operational block that is associated with many applications which demands advance accelerated adder.

Basically ripple carry adder is the primary parallel adder structure. This gives good result when processing bit length is small. As the bit length increases the design produces long carry propagation chain introducing latency [7].

The next level carry skip and carry chain adders that were proposed also could not meet the requirements. There by for better performance we make use of carry select adder and carry tree adder. Even though the carry select adder performs well it requires two pairs of the adder with default carry ‘0’ and ‘1’ as inputs thereby increasing the area considerably [6].

Hence carry tree adders are promising alternatives. In carry tree adder also many structures are proposed which are used for achieving better performance comparatively. On the whole these much one foot back towards the number of levels (black cell and white cell).

For the advancement referring to the reduction of the number of black cells we propose the advanced concept called as the Valance node for the longer bit length. This has the additional feature of pipelining. In our proposed method we designed pipelined hybrid carry tree structure based on the Valance nodes.

This paper proceeds as follows: SectionI deals with the basic introduction followed by Section II dealing with the literature survey of the carry tree adder. Section III pertains to the proposed method, SectionIV deals with the results generated and discussion followed by SectionV ofconclusion and references.

2. CARRY TREE ADDERS

Carry tree adder is also referred as the parallel prefix adder. It basically consists of the three sub sections namely [2].

- Pre-processing
- Carry generation
- Final addition

In preprocessing, propagate and generate of the corresponding inputs can be calculated. The equations are

$$G=a\&b$$

$$P=a\wedge b$$

If A, B are the inputs, generate G is produced by performing logical AND operation to the inputs while propagate P is produced by performing the logical XOR operation to the inputs.

The carry generation stage consists of the black cells and white cells.

Black cell takes GP value-pairs as (G_{left}, P_{left}) and (G_{right}, P_{right}), then the combined block has the GP values as follows:

$$G_{left,\ right} = G_{left} \mid (P_{left}\&G_{right})$$

$$P_{left,\ right} = P_{left}\&P_{right}$$

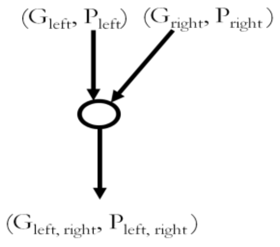


Fig1. *Black Cell*

White cell is similar to that of the black except the propagate output i.e. it also takes two pair of GP but produces only the G.

Black cell is connected to another black cell or white cell bit. Black cell is not the final state but white cell is the final state.

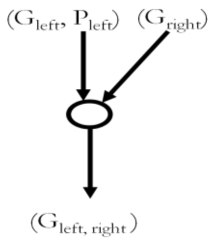


Fig2. *White Cell*

2.1. Kogge-Stone Adder

The kogge-stone adder can be constructed using the combination of black and white cells based on the fundamental carry operator (fco)

$$(gL, pL) \circ (gR, pR) = (gL + pL \bullet gR,\ \ pL \bullet pR)$$

From the design shown below we can easily observe that the count of the black cell and white cell are high. It is designed in such a way that the carry will be generated as the old traditional method. Thereby the carry is generated for the all inputs. This increases the number of stages thereby increasing the area and power [3].

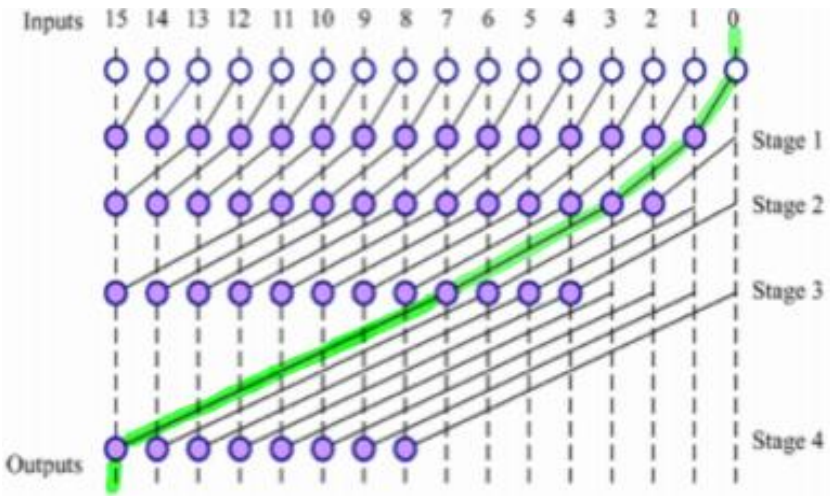


Fig3. *16 bit Kogge-Stone adder*

By segmenting all the inputs as sub blocks the carry will be generated for the specified blocks only. Those sub blocks will propagate carry forming the full structure. Thus, sparse kogge-stone adder was introduced. [10],[6].

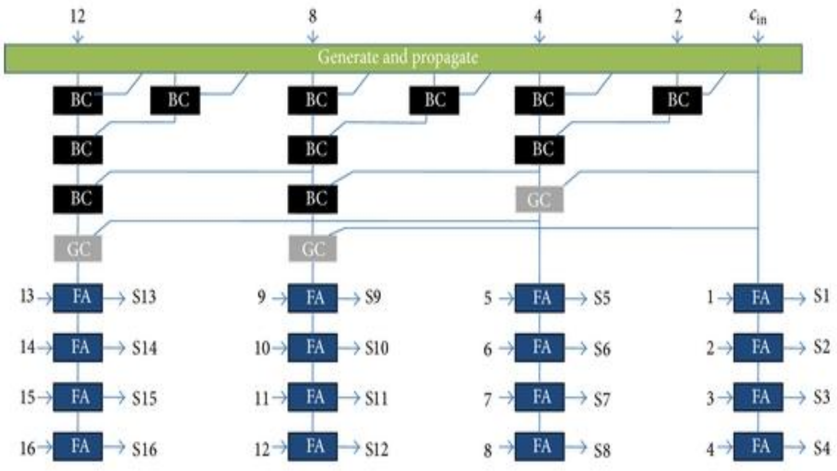


Fig4. *16-bit SparseKogge-Stone adder*

3. Proposed Method

The black cell will be basic building block for the carry tree adder design. Presently it takes only two pairs of propagates and generates and increases as the higher bits are increased.

In order to overcome the demerits we espoused new method as the advance of the black node where we increase the input pairs of PG to reduce the number of levels.

The corresponding equations for the node can be shown below

$$G = (g0 \mid (g1\&p0) \mid (g2\&p0\&p1))$$

$$P = p1\&p2\&p0$$

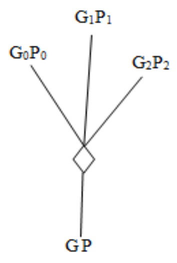
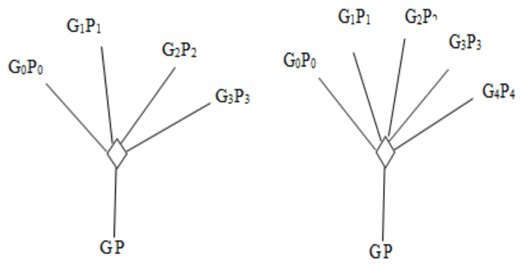


Fig5. *Black cell with valency-3*

Different valencies like valency-3, valency-4, valency-5 etc. can be constructed to form a huge architecture made pipeline with less stages.



(a) (b)

Fig6. *a) Black cell with valency-4, b) Black cell with valency-5*

The equations for the valency-4 node can be shown below

$$G = (g1 \mid (g2\&p1)\mid (g3\&p1\&p2)\mid (g4\&p1\&p2\&p3))$$

$$P = p1\&p2\&p3\&p4$$

The equations for the valency-5 node can be shown below

$$G = (g1 \mid (g2\&p1)\mid (g3\&p2\&p1)\mid$$

$$(g4\&p1\&p2\&p3)\mid (g5\&p1\&p2\&p3\&p4));$$

$$P = p1\&p2\&p3\&p4\&p5;$$

In our design we are not using the traditional full adder rather we areusing a modified full adder.

Ourfull adder takes only two inputs carry and propagate input which was calculated at the first stage of the operation.

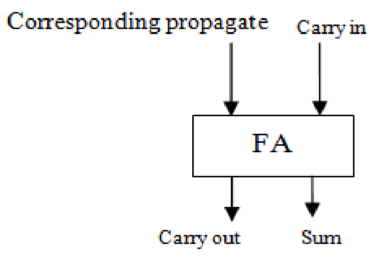
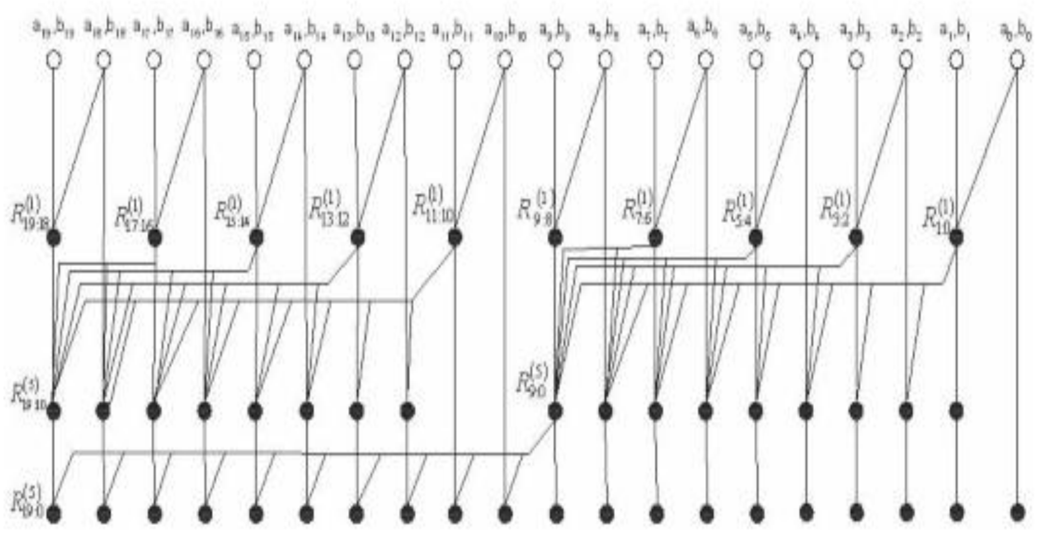


Fig7. *Modified full adder*

The proposed 20 bit adder is designed using the different valency. We use the design methods based on the two, five, two valency. In second stage it includes valency-3, valency-4,valency-5. The pipelined method can be a concern to the advancement of the specified concept for improvement of the featured specifications.

The valency will be design by equal segmentation and by this method we generate all the carry’s for the summation. This proposed design will give pipelined parallel design with different values. This shows less gate count as compared to the different structures of carry tree adder gate count, more over our proposed methodoperation time also less when compared to the other carry tree designs [8],[11],[12].



.

Fig8. *Proposed 20-bit adder carry tree with valency 2*5*2*

In our proposed adder method the concept is similar to that of the normal Sparse Kogge-Stone adder but we utilized the different valency black cell as the elements.

From the design we know the carry’s generated for four bits. This carry is given to the block of four full adders which will propagate carry through it. This design consumes less gate count compared with the present existing different adder methods.

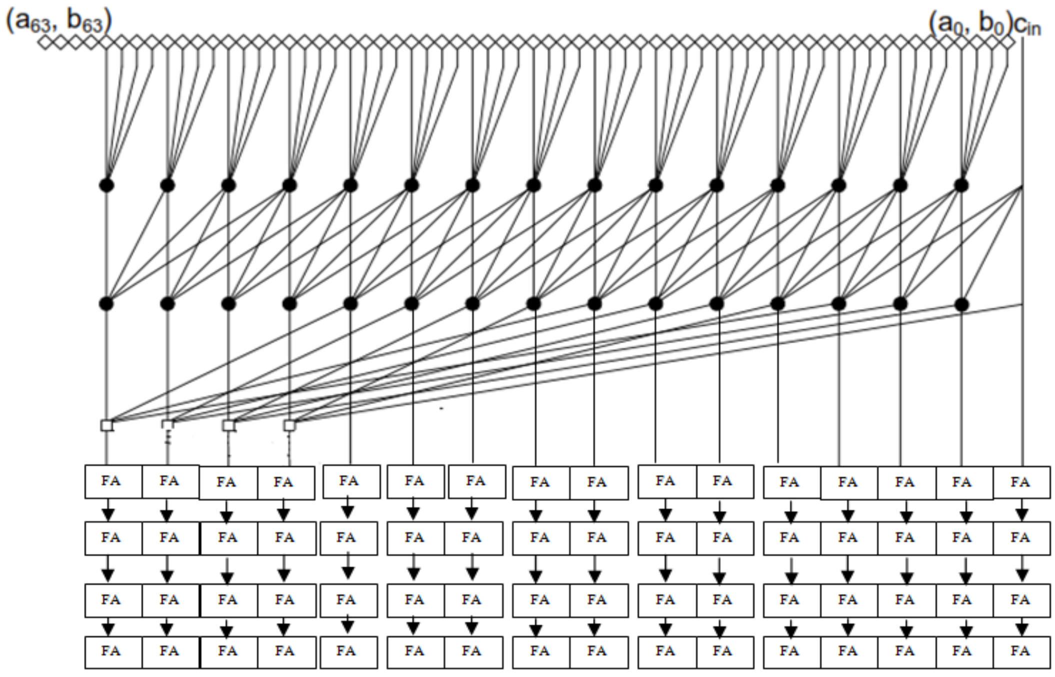
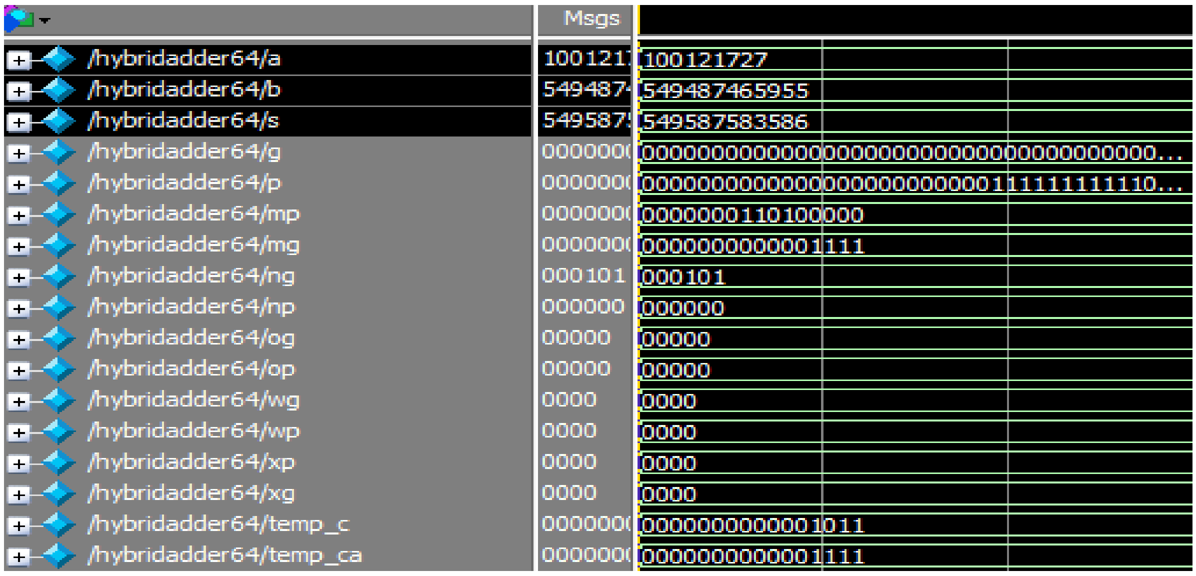


Fig9. *Proposed 64 bit Sparse Kogge-Stone adder*

4. Results and Discussion



```
                              Msgs
/hybridadder64/a         100121?  100121727
/hybridadder64/b         549487·  549487465955
/hybridadder64/s         549587!  549587583586
/hybridadder64/g         000000(  00000000000000000000000000000000000000000...
/hybridadder64/p         000000(  000000000000000000000000000111111111110...
/hybridadder64/mp        000000(  0000000110100000
/hybridadder64/mg        000000(  0000000000001111
/hybridadder64/ng        000101   000101
/hybridadder64/np        000000   000000
/hybridadder64/og        00000    00000
/hybridadder64/op        00000    00000
/hybridadder64/wg        0000     0000
/hybridadder64/wp        0000     0000
/hybridadder64/xp        0000     0000
/hybridadder64/xg        0000     0000
/hybridadder64/temp_c    000000(  0000000000001011
/hybridadder64/temp_ca   000000(  0000000000001111
```

Fig10. *Proposed 64 bit simulation result*

There are five Pipelined stages in our design and each stage operates simultaneously. In our design we give a=100121727 as input in binary format, and give b=549487465955 also in binary format, when added together sum 54958758356 is produced. Different adders were designed and Maximum combinational path delay compared with XILINX 14.3.

Table1. *Maximum combinational path delay of different adders*

Type of adder	Maximum combinational path delay
16 bit Kogge_Stone[11]	14.041ns
16 bit Ripple_Carry[11]	24.686ns
16 bit Spanning_Tree[11]	21.720ns
16 bit Sparse_Kogge[11]	17.527ns
16 bit carry_select_adder[12]	17.501ns
16 bit carry_skip[8]	24.841ns
Proposed method	15.450ns

Our design shows the better speed performance compared with the different adder structures that are listed in the table. Our design shows the 1 ns delay is greater than the kogge-stone adder but kogge-stone adder takes 5% of LUT utilization but our design consumes only 3% of LUT.

5. Conclusion

In our proposal we designed different adders like kogge-stone, brunt kung, spanning tree, sparse kogge-stone and proposed hybrid adder of same bit width and the characteristic evaluation can be done, which shows our proposal gives good records. Moreover we designed the 64-bit sparse kogge-stone adder also designed. For designing Verilog HDL can be used, simulated using Model-Sim and synthesized by XILINX configured to Spartan-6 FPGA.By this, we can conclude that this method not only give the better result for the smaller numbers but also for the higher bit lengths. This will be well suited for the faster applications which demanding less power and area.

References

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

[2] R. P. Brent and H. T. Kung, “A regular layout for parallel adders,” *IEEE Trans. Comput.*, vol. C-31, pp.260-264, 1982.

[3] D. Harris, "A Taxonomy of Parallel Prefix Networks, "in*Proc. 37th Asilomar Conf. Signals Systems andComputers*, pp. 2213–7, 2003.

[4] P. M. Kogge and H. S. Stone, "A Parallel Algorithmfor the Efficient Solution of a General Class ofRecurrence Equations," *IEEE Trans. on Computers*, Vol. C-22, No 8, August 1973.

[5] P. Ndai, S. Lu, D. Somesekhar, and K. Roy, "Fine Grained Redundancy In Adders," *Int.Symp. OnQualityElectronicDesign,* pp.317-321, March2007.

[6] T. Lynch and E. E. Swartz lander, "A Spanning Tree Carry Look ahead Adder," *IEEE Trans. on Computers*, vol. 41, no. 8, pp. 931-939, Aug. 1992.

[7] D. Gizopoulos, M. Psarakis, A. Paschalis, and Y. Zorian, "Easily Testable Cellular Carry LookaheadAdders," *Journal of Electronic Testing: Theory andApplications 19*, 285-298, 2003.

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

[9] M. Becvár and P. Štukjunger, "Fixed-Point Arithmetician FPGA," *ActaPolytechnica*, vol. 45, no. 2, pp. 6772,2005.

[10] K. Vitoroulis and A. J. Al-Khalili, "Performance of Parallel Prefix Adders Implemented with FPGAtechnology," *IEEE Northeast Workshop onCircuitsand Systems*, pp. 498-501, Aug. 2007.

[11] David H. K. Hoe, Chris Martinez and Sri JyothsnaVundavalli"Design and Characterization of Parallel Prefix Adders using FPGAs"2011 IEEE.

[12] O. J. Bedrij, "Carry-select adder," *IRE Trans. Electron. Comput*, pp.340–344, 1962.