ISSN 2319 – 2518 www.ijeetc.com
Vol. 3, No. 1, January 2014
© 2014 IJEETC. All Rights Reserved

Research Paper

DESIGN AND IMPLEMENTATION OF FASTER PARALLEL PREFIX KOGGE STONE ADDER

Sunil M[1]*, Ankith R D[1], Manjunatha G D[1] and Premananda B S[1]

Corresponding Author: **Sunil M, ✉ msunil92@gmail.com*

In this paper we proposed a high speed Kogge-Stone adder by modifying the existing Kogge-Stone architecture (Pakkiraiah Chakali and Madhu Kumar Patnala, 2013). Currently speed of the multipliers is restricted by the speed of the adders used for partial products addition. Kogge-Stone is one of the fastest parallel prefix adders; we eliminated the redundant Black-Cells and performed rerouting thus minimizing the logic delay and routing delay. Coded modified architecture in Verilog HDL, simulated and synthesized using Xilinx ISIM and XST. Compared the architecture with Kogge-Stone adder, an improvement in the speed of 9.84% is achieved.

Keywords: Parallel-adders, Fan-out, Redundant black-cells, Grey cell, Kogge-Stone Adder (KSA)

INTRODUCTION

The fundamental operations involved in any Digital systems are addition and multiplication. Addition is an indispensible operation in any Digital, Analog, or Control system. Fast and accurate operation of digital system depends on the performance of adders (Anitha and Bagyaveereswaran, 2012). The main function of adder is to speed up the addition of partial products generated during multiplication operation. Hence improving the speed by reduction in area is the main area of research in VLSI system design. Over the last decade many types of adder architectures were studied, such as carry ripple adders, carry skip adder, carry look ahead adder, parallel prefix tree adders (Krishna Kumari *et al.*, 2013), etc.

In tree adders, carries are generated in parallel and fast computation is obtained at the expense of increased area and power. The main advantage of the design is that the carry tree reduces the number of logic levels (N) by essentially generating the carries in parallel.

The parallel-prefix tree adders are more favorable in terms of speed due to the complexity O($\log_2 N$) delay through the carry path compared to that of other adders. The prominent parallel prefix tree adders are

[1] Department of Telecommunication Engineering, R. V. College of Engineering, Bangalore 560059, India.

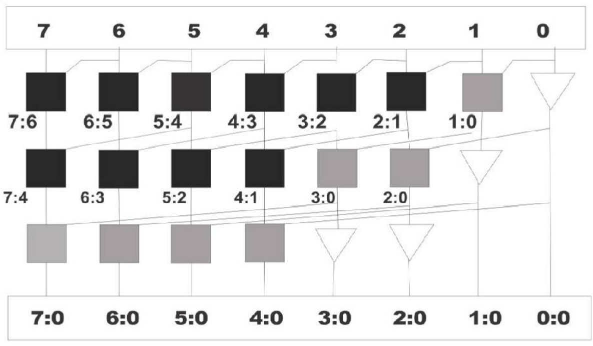
Kogge-Stone, Brent-Kung, Han-Carlson, and Sklansky (Nurdiani Zamhari *et al.*, 2012). Out of these, it was found from the literature that Kogge-stone adder is the fastest adder when compared to other adders. The adder priority in terms of worst-case delay is found to be Ripple-Carry, Carry-Look-Ahead, Carry-Select and Kogge-Stone. This is due to the number of “Reduced stages”. Kogge-Stone adder implementation is the most straightforward, and also it has one of the shortest critical paths of all tree adders. The drawback with the Kogge-Stone adder implementation is the large area consumed and the more complex routing (Fan-Out) of interconnects.

KOGGE-STONE ADDER

The Kogge-Stone adder is a parallel prefix form of carry look-ahead adder. It generates the carry signals in O($\log_2 N$) time, and is widely considered as the fastest adder design possible. It is the most common architecture for high-performance adders in industry. The Kogge-Stone adder concept was first developed by Peter M. Kogge and Harold S. Stone (Madhu Thakur and Javed Ashraf, 2012). In Kogge-stone adder, carries are generated fast by computing them in parallel at the cost of increased area.

Tree structures of carry propagate and generate signals in 8-bit Kogge-Stone Adder (KSA) is shown in Figure 1. Carry generation network is the most important block in tree adders, and it consists of three components such as Black cell, Grey cell and Buffer. Black cells are used in the computation of both generate and propagate signals. Grey cells are used in the computation of generate signals which are needed in the computation of sum in the post-processing stage. Buffers are used to balance the loading effect.

Figure 1: 8-Bit Kogge-Stone PG Network



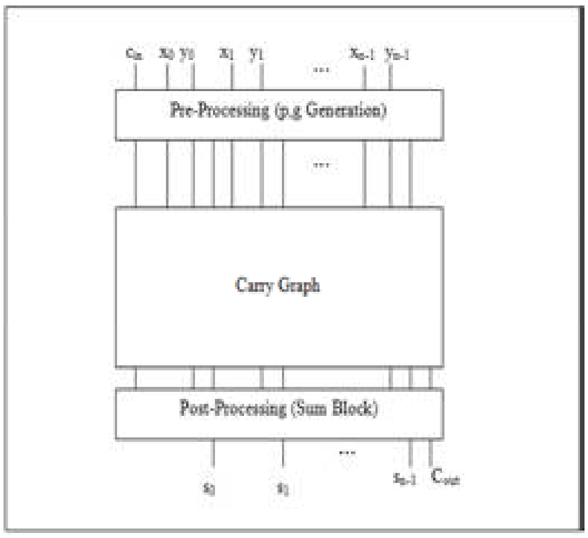
Working of Kogge-Stone Adder

Kogge-Stone Adder has three processing stages for calculating the sum bits, they are:

1. Pre-processing stage
2. Carry generation (PG) network
3. Post-processing stage

The above steps involved in the operation of Kogge-Stone adder are as shown in Figure 2.

Figure 2: Architecture of Kogge-Stone Adder


C_{in} x_0 y_0 x_1 y_1 ... x_{n-1} y_{n-1}
Pre-Processing (p,g Generation)
...
Carry Graph
Post-Processing (Sum Block)
...
s_0 s_1 s_{n-1} C_{out}

Preprocessing

This step involves computation of generate and propagate signals corresponding to each pair of bits in A and B.

$Pi = Ai$ x or Bi ...(1)

$Gi = Ai$ and Bi ...(2)

Carry Generation Network

In this stage we compute carries corresponding to each bit. Execution of these operations is carried out in parallel. 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 and 4):

$CPi{:}j = Pi{:}k + 1$ and $Pk{:}j$ ...(3)

$CGi{:}j = Gi{:}k + 1$ or ($Pi{:}k + 1$ and $Gk{:}j$) ...(4)

Post Processing

This is the final step and is common to all adders of this family (carry look ahead). It involves computation of sum bits.

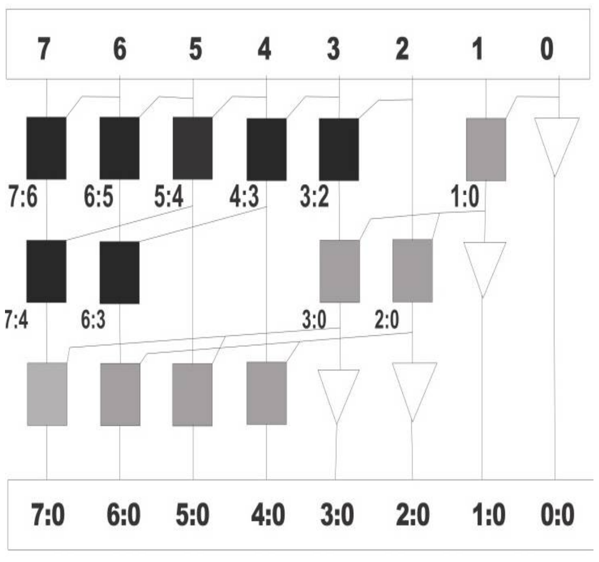
$C_{i-1} = (P_i$ and $C_{in})$ or G_i ...(5)

$S_i = P_i$ x or C_{i-1} ...(6)

Modified Kogge-Stone Adder

The Kogge-Stone adder in Figure 1 is faster than other well known parallel prefix adders and has fan-out of 2 in all stages. We can reduce the computation by eliminating the redundant cells thus compensating for the increased delay. The Kogge-Stone adder can be modified by reducing the Black cells and rerouting to compensate the functionality of adder. Propagate-Generate (PG) network of modified 8-bit Kogge-stone adder is shown in Figure 3. Delay of the Kogge-Stone adder can also be decreased by only rerouting the wires but this is not so effective because area remains same. We can increase the speed of adder also by eliminating redundant Black cells which results in reduction in area of the adder.

Figure 3: Modified Kogge-Stone Adder PG Network



Routing Technique

The one important concept in all tree adders is that each cell has its own task in the overall output, for example in the second stage of the 8-bit Kogge-Stone adder on the left the generate bit is calculated if only bit-6 and -7 is present. It only requires generate bit from bit-0 to -5, when these two generate bits are combined it gives the generate bit for the overall bit-0 to -7. The important concept is that generate bit of 6 and 7 need generate bits from the calculation of bit 0 to 5 but instead, can give it the generate bit calculated using bit 0 to 6 and will obtain the same result. This is because generate bit from 6 is already included but is being added again from the computation of generate bit of 0 to 6 and this does not affect the end result. But final generate bit from bit-i cannot take any generate bits higher than i, if so the end result will be affected as we are included the generate value of the higher bits.

The above mentioned method is a good start but we can even take it a step further by trying to remove some of the redundant cells. Redundant cells can be removed without any adverse consequences but it has to be properly compensated, if not the delay will increase considerably. And this can be compensated by changing the routing (wiring). Performing the rerouting in only first and the second stages produced an improvement in speed but doing the same in all 3 stages of carry-generate network increases delay.

Elimination of Redundant Black Cells

Grey cells are required for computation of generate bit in final stage and thus cannot be removed. Black cells are the only redundant cells in tree adders. The removal of redundant cells can be performed in different ways, but not all changes give the desired results. Thus all changes which are going to be done have to be done in perspective of speed. Analyzing from the last stage gives us a much better understanding of the redundant cells. In the last stage there are no redundant cells as it contains only grey cells and hence none of them can be removed.

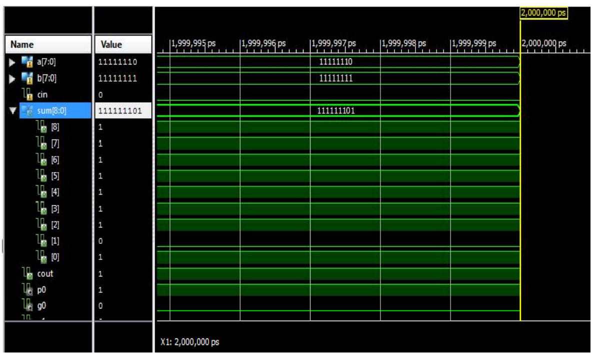
In the second stage there are 2 redundant black cells, black cell 5:2 and black cell 4:1 can be removed and this is compensated by directly giving the output of black cell 5:4 and grey cell 3:0 to the grey cell 5:0 in the last stage.

Black cell 4:1 is compensated by giving the of black cell 4:3 and grey cell 2:0 to the grey cell 4:0 in the last stage. Both the grey cell 2:0 and grey cell 3:0 take generate bit form grey cell 1:0. Only one redundant cell is present in the first stage that is the black cell 2:1. This can be removed without any consequence as it was only going to be used in the black cell 4:1 since it was removed, even the black cell 2:1 can be removed.

IMPLEMENTATION

The design of different tree adders are coded in Verilog Hardware Description Language using structural modeling in Xilinx ISE Design Suite 14.2 and all the simulation results are verified using XILINX ISIM simulator. Synthesized for the Spartan-3 FPGA XC3S400 with the speed grade of 4. Inputs are generated using Verilog HDL test bench. Simulated result for the addition of 8-bit inputs "11111110" and "11111111" is shown in Figure 4. Output sum = "11111101" and carry Cout = '1'.

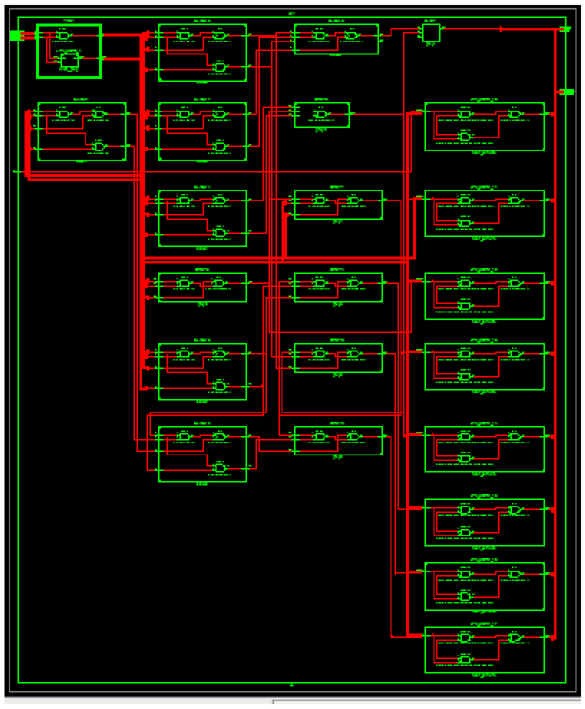
Figure 4: 8-Bit Kogge-Stone Simulation Result



Name	Value
a[7:0]	11111110
b[7:0]	11111111
cin	0
sum[8:0]	111111101
[8]	1
[7]	1
[6]	1
[5]	1
[4]	1
[3]	1
[2]	1
[1]	0
[0]	1
cout	1
p0	1
g0	0

RTL block diagram of the proposed 8-bit modified Kogge-Stone adder is shown in Figure 5. The red path in Figure 5 are the routing and data path and the green blocks

Figure 5: RTL Schematic of Modified 8-Bit Modified Kogge-Stone Adder



represent the grey and black cells and the gates inside the cells are represented in green.

Comparison with Various Adder Architectures

The proposed adder design is compared with various architectures in terms of number of logic levels and overall delay, the results are tabulated in Table 1. From Table 1 it is evident that there is reduction in overall delay in the proposed modified Kogge-Stone adder by eliminating redundant Black-cells. The logic delay and routing delay of modified Kogge-Stone adder are 8.669 ns and 4.998 ns respectively. When compared to normal Kogge-Stone adder architecture there is an improvement by 9.84% in the overall delay.

Table 1: Comparison of Various Adder Architectures

Architecture	Logic Levels	Speed (ns)
Ripple Carry	N-1	31.744
Brent-Kung	$2\log_2 N-1$	19.059
Han-Carlson	$\text{Log}_2 N+1$	16.943
Sklansky	$\text{Log}_2 N$	15.604
Kogge-Stone	$\text{Log}_2 N$	15.160
Kogge-Stone (Rerouting)	$\text{Log}_2 N$	15.017
Kogge-Stone (Reducing Black Cells)	$\text{Log}_2 N$	13.667

CONCLUSION

This paper presents an innovative way of modifying the already existing Kogge-Stone adder by re-routing (wiring) and Black-cell reduction for increasing the speed of execution. The design originates from the principle of removing the redundant black cells and compensating this removed cells by re-routing. The above design has a delay which is much less than the architectures that it is being compared with. The number of logic levels used is also found to be less. There is a reduction in both logic and routing delay and improvement in the speed by 9.84% to that of normal Kogge-Stone adder. So parallel prefix adders of this type are the best choice in many VLSI applications where speed is the main constraint.

REFERENCES

1. Ahmet Akkas and Michael J Schulte (2011), "A Decimal Floating-Point Fused Multiply-Add Unit with a Novel Decimal Leading-Zero Anticipator", AMD IEEE.

2. Anitha R and Bagyaveereswaran V (2012), "High Performance Parallel Prefix Adders with Fast Carry Chain Logic", *International Journal of VLSI Design & Communication Systems*, Vol. 3, No. 2, pp. 01-10, ISSN 0976-6499.

3. David Jeff Jackson and Sidney Joel Hannah (1993), "Modeling and Comparison of Adder Designs with Verilog HDL", 25th South-Eastern Symposium on System Theory, March, pp. 406-410.

4. Kogge P and Stone H (1973), "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Relations", *IEEE Transactions on Computers*, Vol. C-22, No. 8, pp. 786-793.

5. Krishna Kumari V, Sri Chakrapani Y and Kamaraju M (2013), "Design and Characterization of Koggestone, Sparse Koggestone, Spanning Tree and Brentkung Adders", *International Journal of Scientific & Engineering Research*, Vol. 4, No. 10, pp. 1502-1506, ISSN 2229-5518.

6. Madhu Thakur and Javed Ashraf (2012), "Design of Braun Multiplier with Kogge-Stone Adder & It's Implementation on FPGA", *International Journal of Scientific & Engineering Research*, Vol. 3, No. 10, pp. 03-06, ISSN 2229-5518.

7. Nurdiani Zamhari, Peter Voon, Kuryat iKipli, Kho Lee Chin and Maimun Huja Husin (2012), "Comparison of Parallel Prefix Adder (PPA)", Proceedings of the World Congress on Engineering, WCE, Vol. 2, July 4-6, London, UK.

8. Pakkiraiah Chakali and Madhu Kumar Patnala (2013), "Design of High Speed Kogge-Stone Based Carry Select Adder", *International Journal of Emerging Science and Engineering (IJESE)*, Vol. 1, No. 4, ISSN: 2319-6378.

9. Weste Neil, Harris David and Banerjee Ayan (2009), "CMOS VLSI Design: A Circuits and System Perspective", Pearson Education.