International Journal of Electronics Engineering (ISSN: 0973-7383)
Volume 11 • Issue 2 pp. 11-17 June 2019-Dec 2019 www.csjournals.com

Performance Comparison of Adder Architectures

D A Madhura[1], Rakesh M B[2]
[1]Student, [2]Assistant Professor, Dept. of Telecommunication, Siddaganga Institute of Technology, Tumkur, karnataka

Abstract: Adders are one of the most desirable entities in processor data path architecture. Since long, VLSI engineers are working towards optimization and miniaturization of adder architectures to ultimately improve the performance of processors. As the technology is scaling down, challenges towards optimization are also increasing. Three well known adders are Carry Look Ahead adder, Koggestone adder and Brent Kung adder, which are all parallel adders. The limitation of rippling effect of carry in Ripple Carry Adder(RCA) is overcome by CLA,but the area required is more than RCA. Koggestone Adder(KSA) has a lesser propagation delay and has lower fan-out at each stage which in turn increases performance. However,the area required for this adder increases. Brent Kung Adder(BKA) has less wiring congestion with better performance and also requires less area for the implementation than KSA. But the propagation delay increases. In this paper the comparison of these three adders is done in terms of propagation delay and area.These adders' design is implemented using the Verilog code on XILINX 14.7 tool.

Keywords: Adder,Verilog,Xilinx,Propogarion delay,Area,VLSI.

1. Introduction

Adders are the key components in general purpose microprocessors and digital signal processors. They also find use in many other functions such as subtraction, multiplication and division. As a result, it is very pertinent that its performance augers well for their speed performance. Furthermore, for the applications such as the RISC processor design, where single cycle execution of instructions is the key measure of performance of the circuits, use of an efficient adder circuit becomes necessary, to realize efficient system performance. Additionally, the area is an essential factor which is to be taken into account in the design of fast adders. Towards this end, high-speed, low power and area efficient addition and multiplication have always been a fundamental requirement of high-performance processors and systems. The major speed limitation of adders arises from the huge carry propagation delay encountered in the conventional adder circuits, such as ripple carry adder and carry look adder.

Ripple carry adder: Multiple full adder circuits can be cascaded in parallel to add an N-bit number. For an N-bit parallel adder, there must be N number of full adder circuits. It includes a series of full adders equivalent to the number of bits. The first full adder will be provided with first bits of both two numbers say (A (0) and B (0)) along with input carry say Cin. The output of first full adder will be the first bit of sum and a carryout, which will be rippled to the next full adder, and this process continues. Hence, the name Ripple Carry Adder. Propagation delay is time elapsed between the application of an input and occurrence of the corresponding output. Although the area consumption of RCA is less, the delay in the circuit is high. The RCA is the combination of low area consumption and high delay time compared with other adders [1].

Carry Look Ahead adder: The Carry Look Ahead adder provides a better speed in obtaining the result, as the carries in the intermediate stages will be calculated beforehand using carry generate and carry propagate regardless of input carry. Hence, it is called as Carry Look Ahead adder. The extra blocks of this adder are the carry propagate and carry generate, where the carry propagate will be propagated to the next stages and the carry generate is responsible for the advance generation of carry irrespective of input carry given to the first stage. The drawback of this adder is that it involves complex circuitry and the hardware gets complicated as the number of bits increases [2]. Ripple Carry Adder consumes less area but takes more time for execution. Carry Look Adder consumes same area as that of RCA but executes the operation in lesser time in comparison. Carry Save Adder consumes more area as well as more time for execution [1].

Koggestone adder: The Koggestone adder is a parallel prefix form carry look-ahead adder. The architecture consists of three blocks, which are pre-processing, carry generator and post processing blocks. In KSA, for each block a PG (Propagation Generation) block is generated. Propagation P is calculated by Xor operation and Generation G is calculated by AND operation. Lastly, first level propagate bits are Xored with carry bits to obtain sum. This is the fastest adder at the cost of area [4].The propagation delay is reduced compared to other form of adders. The KSA requires the highest cell area[3]. This is expected since extra FCO blocks are required in order to generate the sum out and Cout results in parallel.

Brent Kung adder: BKA is a type of parallel prefix adder that consists of BKA pre-processing stage, BKA parallel prefix network and BKA post processing stage for computation of summation and output carry of input bits [5]. Brent–kung has the lowest propagation delay compared to all the existing adders. They are based on reducing carry computation to a “prefix” computation. The Brent–Kung adder has been chosen first for efficient QCA realization in view of the (relatively)small growth in the number of associative operations as a function of the adder size[6].This adder requires fewer modules to implement than the Koggestone adder. It is much simpler to build and contains far fewer connections to other modules, which also contributes to its simplicity. Major disadvantage of this adder is fan-out. Fan-out may split and weaken the current propagating through the adder.

2. Adders

An adder is a digital logic circuit that executes an arithmetic operation such as addition, subtraction. It is also used in processor to calculate table indices, addresses, and similar operation.The basic adder types are half adder (HA) and full adder (FA) as shown in Figure 1. Half adder basically adds two binary digits a and b and produce two output signals sum s and carry C. The carry signal indicates an overflow into the next digit of a multi-digit addition. HA can be implemented using XOR gate and AND gate according to equations 1 and 2. In contrast, FA adds three binary digits, often written as a, b and c, and produce two output signals sum s and carry C. FA can be implemented using equations 3 and 4.[6]

$$S = a \oplus b$$
$$C = a.b$$
$$S = (a \oplus b) \oplus c$$
$$C = (a \oplus b) \oplus c + a.b$$

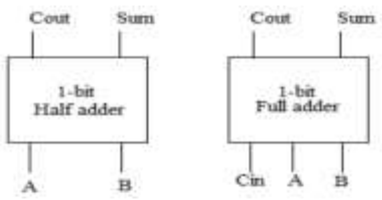

Figure 1. Modules of basic adders

A.Carry Look ahead Adder (CLA)

CLA can be constructed using two levels[2]. The first level is called a Partial Full Adder (PFA). This part is responsible for generate and propagate the carry to the second level. For N-bit CLA, the two n-bit inputs a[n − 1 : 0] and b[n − 1 : 0] to be added are used to generate the carry propagate p[n − 1 : 0] and carry generate g[n − 1 : 0] signals to be supplied to the CLA at bit i.

$$pi = ai \oplus bi$$

$$gi = ai \cdot bi$$

The output sum can be expressed according to the below equation, where ci is the carry output of each stage.

$$si = pi \oplus ci$$

Thus Pi and Gi are generated from pre-processing block. This stage is the part that differentiates the performance of an adder. On addition of pre-processing block delay of the adder can be reduced. The signals then proceeds to the next stage, Prefix Carry Tree to generate Ci. On the other hand, the final stage that is post-processing block which is aimed to get the final adder result Sum along with the output carry as shown in figure 2.

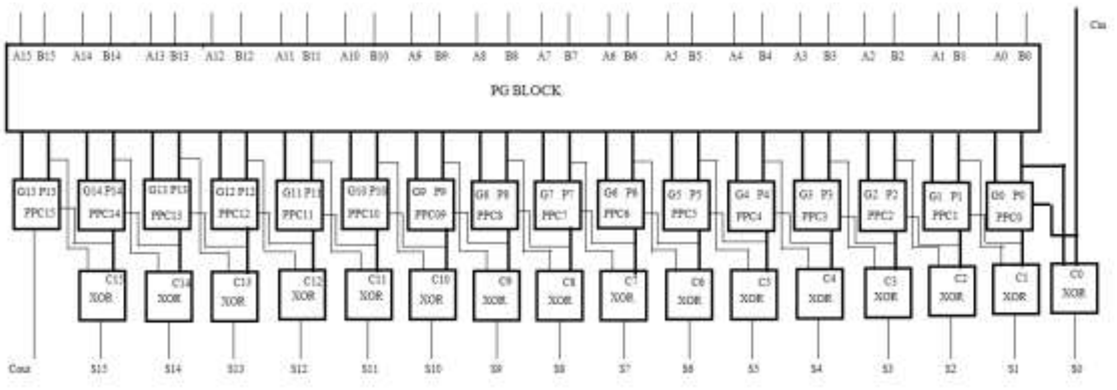

Figure 2.16-bit carry look ahead adder

B.Kogge Stone Adder (KSA)

KSA can be easily implemented by analysing it in terms of three parts:

• Pre-processing : This step includes computation of generate and propagate signals that corresponding to each pair of bits in a and b. The generate and propagate signals are given by the equations below:

$$p_i = a_i \oplus b_i$$

$$g_i = a_i \cdot b_i$$

• Prefix carry tree: This part differentiates KSA from other adders and is the reason behind its high performance. This step includes computation of carries that corresponding to each bit. This part uses group propagate and generate signals which are given by the equations below:

$$p_{i:j} = p_{i:k+1} \cdot p_{k:j}$$

$$g_{i:j} = g_{i:k+1} + (p_{i:k+1} \cdot g_{k:j})$$

The notations $p_{i:j}$ and $g_{i:j}$ denote to group-propagate and group-generate respectively and for the group that includes bit positions from i to j. k represents the logic level from where the input is produced.

• Post processing stage: This step is the final step to all adders of the (carry look ahead) family. It includes computation of sum bits which is given by the equation below:

$$c_i = g_i \,|\, (p_i . c_{in})$$

$$s_i = p_i \oplus c_i \text{ [4]}$$

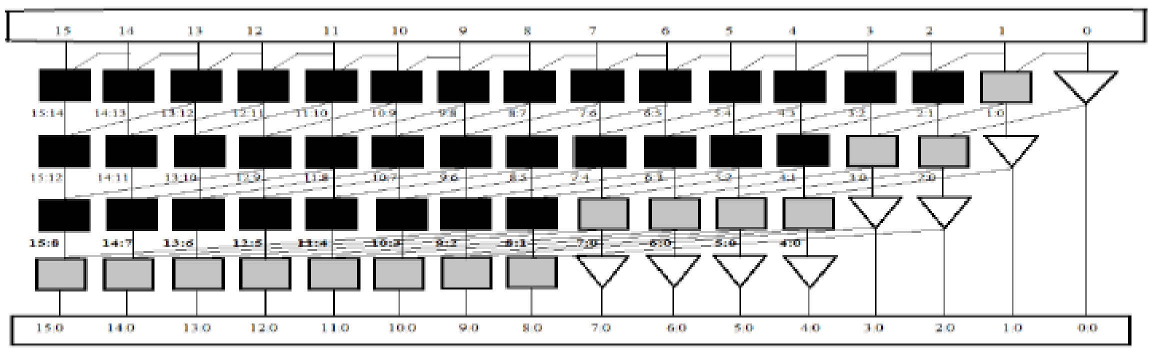

Figure 3. 16-bit KoggeStone adder

C. Brent Kung Adder (BKA)

Brent-Kung Adder (BKA) is a parallel prefix adder that was developed by Brent and Kung in 1982. The idea of Brent and Kung adder is to combine propagate signals and generate signals into groups of two by using the associative property only. BKA is also belonged to carry look ahead adder family and can be implemented by analysing it into three parts as following:

• Pre-processing: This step includes computation of generate and propagate signals that corresponding to each pair of bits in a and b, and it is the same process of other CLA family. The generate and propagate signals are given by the equations below:

$$p_i = a_i \oplus b_i$$

$$g_i = a_i \cdot b_i$$

• Prefix carry tree: This part includes computation of carries that corresponding to each bit, and it is different from other CLA family. The equations below shows how propagate and generate signals are calculated in BKA

$$p_{i:j} = p_{i:k} \cdot p_{k-1:j}$$

$$g_{i:j} = g_{i:k} + (p_{i:k} \cdot g_{k-1:j})$$

The smart idea of this design is to compute prefixes for 2-bit groups first. These are then used to find prefixes for 4-bit groups, and turn to find prefixes for 8-bit groups, etc. The issue of this design is that the propagate and generate signals take more stages than KSA to be calculated. Figure 4 shows 16-bit BKA.

• Post processing stage: This step is the final step to all adders of the (carry look ahead) family. It includes computation of sum bits which is given by the equation below:

$$c_i = g_i$$

$$s_i = p_i \oplus c_i$$

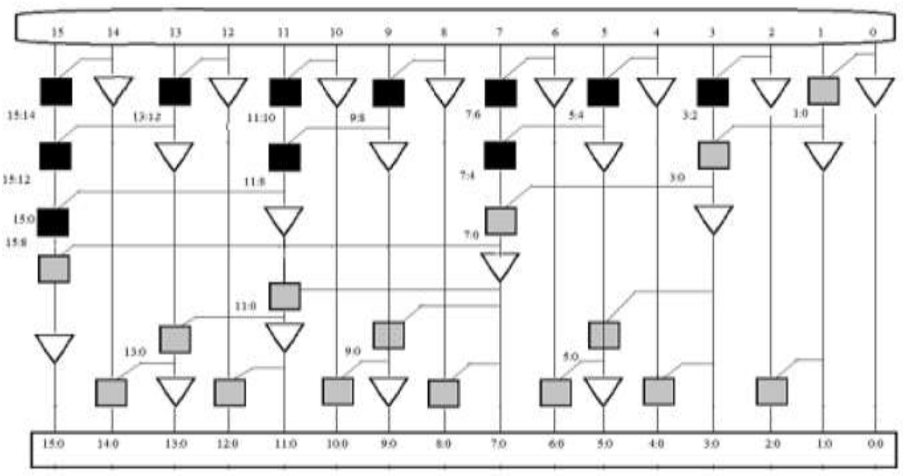

Figure 4.16-bit Brent kung adder

Brent-Kung Adder features with low network complexity comparing to Koggestone Adder. The low network complexity assists to reduce the area of adder resulting in reducing the power consumption as well. This feature makes BKA more efficient than KSA, which has more black competition nodes and long wires. On the other hand, BKA has more stages (logic levels) compare to KSA. Having more competition stages leads to a slower adder. For example, as shown in Figure 4, 16-bit KSA needs only three stages to calculate the carries while BKA in Figure 3.14 needs five stages to get the carries calculated. Hence, KSA is more efficient than BKA in terms of speed.

The number of blocks required in the BKA is less compared to KSA. Thus it overcome the limitation of the Koggestone adder by consuming less area than KSA. But the path delay of the brent kung adder is more than the Koggestone adder. Thus, there is a trade off between the delay and the area required.

3. Results

All the three adders are simulated and synthesized using Xilinx 14.7 and hardware kit Spartan 6.

A. Carry look ahead adder:

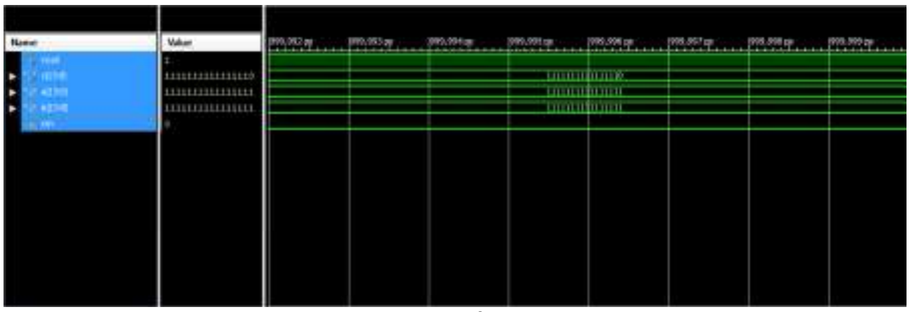

Figure 5. Simulation result of Carry look ahead adder

The waveform shows the addition of two 16 bit numbers. The sum is generated using Verilog code of carry look ahead adder. Two inputs are considered as a= 16’b1111111111111111, b=16’b1111111111111111 and Cin=0 which gives the sum as, S=16’b1111111111111110 and Cout=1.

B. Koggestone adder:

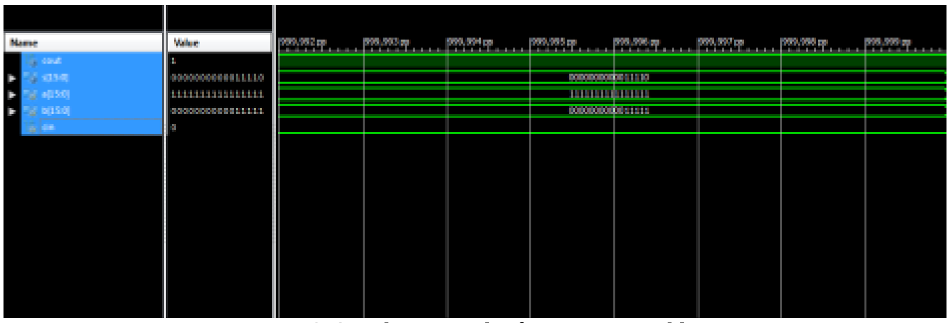

Figure 6. Simulation result of Koggestone adder

The waveform shows the addition of two 16 bit numbers. The sum is generated using Verilog code of Koggestone adder. Two inputs are considered as a= 16’b1111111111111111, b=16’b0000000000011111 and Cin=0 which gives the sum as, S=16’b0000000000011110 and Cout=1.

C. Brentkung adder:

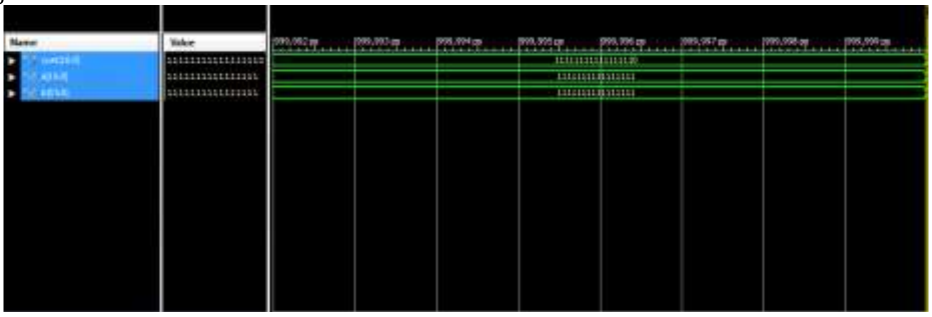

Figure 7. Simulation result of Brentkung adder.

The waveform shows the addition of two 16 bit numbers. The sum is generated using Verilog code of Brent kung adder. Two inputs are considered as a= 16’b1111111111111111, b=16’b1111111111111111 and Cin=0 which gives the sum as, S=16’b1111111111111110.

The carry look ahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of larger number of bits. The speed of this adder is independent of the number of bits. However, the process in obtaining carry generate and propagate bits depends on the cout from the previous bit to complete. Hence, the time delay is not solved completely by Carry look ahead adder. Area and power consumption required for CLA is less compared to KSA and BKA.
KSA and BKA overcome the time delay compared to CLA. In terms of propagation delay KSA is a better choice than BKA. In terms of area between the two parallel prefix adders, BKA proves to be a better choice. Though the area increases in BKA when number of bits increases, it doesn't increase as drastically as KSA. The power consumption required for BKA is less compared to KSA. Thus there is a tradeoff between BKA and KSA.

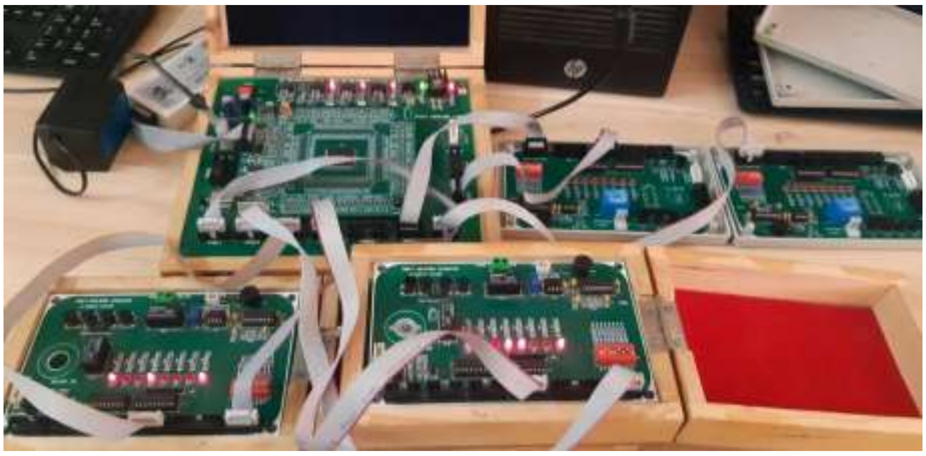
Figure 8. Implementation using FPGA.

The comparison between the adders are shown below,

Table 1: Comparison between adders

Adders	Propagation delay	Area
CLA(16 bit)	13.05ns	No. of LUTs: 25 Logic Slice: 25
KSA(16 bit)	9.54ns	No. of LUTs: 71 Logic Slice: 71
BKA(16 bit)	12.04ns	No. of LUTs: 24 Logic Slice: 24

4.Conclusions

It is shown that the results obtained for Parallel Prefix Adders are better than the serial adders in terms of delay and at the same time there is a trade-off with the area occupied.Koggestone and Brent-Kung adders are designed and implemented using FPGA kit. These high speed adders are compared against Carry look ahead adder. All of these adder architectures are implemented using verilog code with Xilinx14.7 tool. The propagation delay and chip area are calculated. Kogge Stone Adder architecture result shows improvement in propogation delay from Carry look ahead but area increases. Brent Kung adder requires less area compared to that of Koggestone adder but propogation delay increases.

References

[1]. Bhavani Koyada,N. Meghana,Md. Omair Jaleel and Praneet Raj Jeripotula,"A Comparative Study on Adders", International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET),pp. 2226-2230,2017.

[2]. Kartheek Boddireddy,Boya Pradeep Kumar and Chandra Sekhar Paidimarry," Design and implementation of area and delay optimized carry tree adders using FPGA", International Conference on Computing and Communication Technologies,pp.1-6,2014.

[3]. Nidhi Gaur,Devyani Tyagi,Deepika and Anu Mehra , "Performance comparison of adder architectures on 28nm FPGA", 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA),pp.1-5.2016.

[4]. Lee Mei Xiang, Muhammad Mun'im Ahmad Zabidi, Ainy Haziyah Awab and Ab Al-Hadi Ab Rahman," VLSI Implmentation of a Fast Kogge-Stone Parallel-Prefix Adder", Journal of Physics: Conference Series,Vol-1049,2018

[5]. N. Udaya Kumar,K. Bala Sindhuri,K. Durga Teja and D. Sai Satish," Implementation and comparison of VLSI architectures of 16 bit carry select adder using Brent Kung adder", Innovations in Power and Advanced Computing Technologies (i-PACT),pp.1-7,2017.

[6]. Kunjan D. Shinde,Jayashree C. Nidagundi," Design of fast and efficient 1-bit full adder and its performance analysis", International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), pp.1275-1279,2014.