© 2017 IJSRSET | Volume 3 | Issue 2 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Design and Performance Investigation of Binary Signed Digit Adder

Sharmila Hemanandh*, Aishwarya Gopinath , S. Karthika, B. Nandhini

ECE, JEPPIAAR SRR Engineering College , Chennai, Tamil Nadu, India

ABSTRACT

An Adder is a basic building block of a digital system. In general, an adder is a digital circuit that performs addition of numbers. Adders play a vital role in deciding the overall performance of digital signal processing applications. They are also utilized in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators etc. The type of number representation used in the design of adders influence the performance of the adder and hence the digital system. This paper proposes the design of Binary Signed Digit (BSD) adder based on signed digit number representation. Binary Signed Digit representation results in fast and propagation free addition. The carry free addition offers many advantages in the implementation of arithmetic circuits. The advantages and disadvantages of Fast adders such as Carry Save Adder (CSA) and Carry Look Ahead Adder (CLA) are studied. The adders are discussed and the performance parameters such as area and power are compared. The adders are designed using Xilinx and implemented with XC9572XL-5-TQ100.The results prove that the proposed Binary Signed Digit adder is highly efficient than the other two fast adders.

Keywords: Fast Adders, CSA, CLA, BSD

I. INTRODUCTION

A very common and useful combinational logic circuit which can be constructed using just a few basic logic gates allowing it to add together two or more binary numbers is the Binary Adder. A basic Binary Adder circuit can be made from standard AND and Ex-OR gates allowing us to "add" together two single bit binary numbers, A and B. The addition of these two digits produces an output called the SUM of the addition and a second output called the carry or Carry-out,(C_{OUT}) bit according to the rules for binary addition. There are many types of adders such as half adder, full adder, ripple carry adder, carry save adder, carry look-ahead adder, carry save adder etc. Carry look ahead adder and Carry save adders are studied in this paper and the results are compared with the proposed Binary Signed Digit (BSD) Adder.

Carry Look-ahead Adder (CLA) is more efficient than carry save adder in its operation by which in CLA, the carry signals are calculated in advance based on the input signals. Whereas in ripple carry adder we need not wait for the propagation of carries to get the sum and hence the propagation time is reduced and also has a faster addition logic.

The Binary Signed Digit Adder (BSD) makes it possible to perform addition with carry propagation chains limited to a single digit position and has to be used to speed up the arithmetic operations. In order to cope with the problem of carry propagation the most appropriate approach is elimination of carry propagation. Hence this results in carry free addition, less delay and higher efficiency. This paper proposes Binary Signed Digit Adder.

II. CARRY LOOKAHEAD ADDER

A carry-look ahead adder improves speed by reducing the amount of time required to determine carry bits. It is similar to that of ripple carry adder in 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 shown in Fig 1 calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits.

The Kogge-Stone adder and Brent-Kung adder are examples of this type of adder.

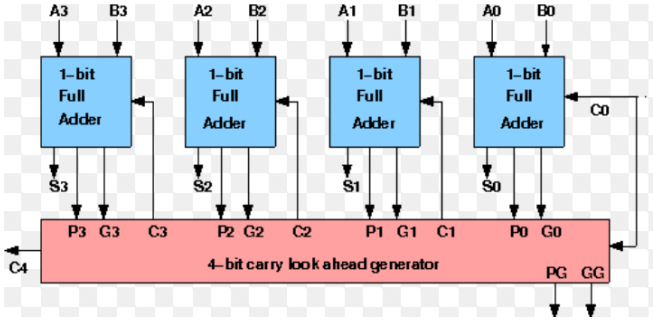

Figure 1. Block diagram of carry look ahead adder

The computation time to add two binary numbers can be reduced by using carry look ahead adder. They work by creating two signals P and G known to be Carry Propagator and Carry Generator. The carry propagator is used to propagate the carry to the next level whereas the carry generator is used to generate the output carry, regardless of input carry.

In Fig.2 two variables as carry generate Gi and carry propagate Pi are defined as,

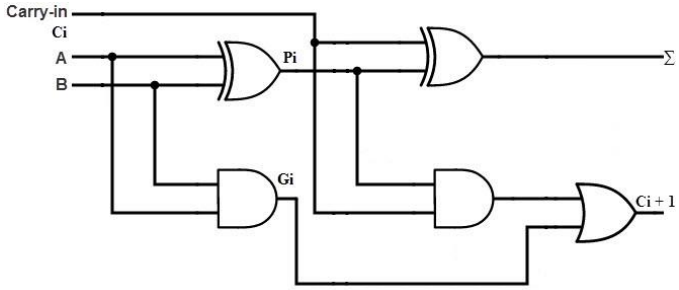

Figure 2. Circuit diagram of carry look ahead adder

TABLE 1. Truth Table of Carry Look Ahead Adder

A	B	Ci	C i+1	Condition
0	0	0	0	No carry generate
0	0	1	0	
0	1	0	0	
0	1	1	1	No carry propagate
1	0	0	0	
1	0	1	1	
1	1	0	1	Carry generate
1	1	1	1	

Table. 1 gives the truth table of carry look ahead adder and hence $P_i = A_i \oplus B_i$ and Gi = Ai Bi. The sum output and carry output can be expressed as Si = Pi ⊕Ci C i +1 = Gi + Pi Ci. Where Gi is a carry generate which produces the carry when both Ai, Bi are one regardless of the input carry. Pi is the carry propagate and it is associated with the propagation of carry from Ci to Ci +1. The carry output Boolean function of each stage in a 4 stage carry-Look ahead adder can be expressed as C1 = G0 + P0 Cin, C2 = G1 + P1 C1, C2= G1 + P1 G0 + P1 P0 Cin, C3 = G2 + P2 C2, C3= G2 + P2 G1+ P2 P1 G0 + P2 P1 P0 Cin.

From the above Boolean equations we can observe that C4 does not have to wait for C3 and C2 to propagate but actually C4 is propagated at the same time as C3 and C2. Since the Boolean expression for each carry output is the sum of products so these can be implemented with one level of AND gates followed by an OR gate.One main advantage of carry look ahead adder is that we need not wait for the propagation of carries to get the sum and hence the propagation time is reduced and also has a faster addition logic. Moreover for very large numbers (hundreds or even thousands of bits) carry look ahead logic does not become any more complex, because more layers of super groups and sub super groups can be added as necessary.

III. CARRY SAVE ADDER

Carry save adder is a type of digital adder, used in computer micro architecture to compute the sum of n-bit numbers in binary. It differs from other digital adders in that it outputs two numbers of the same dimensions as the inputs, one which is sequence of partial sum bits and another which is a sequence of carry bits. Generally the carry save unit consists of n full adders, each of which computes a single sum and carry bit. It is made up of a ladder of standalone full adders, and carries out a number of partial additions. The principal idea is that the carry has a highest power of 2 and thus is routed to next column. As shown in Fig.3 a 4 bit carry save adder has four full adders and a ripple carry adder. Here three bits are added parallel at a time. The carry is not propagated to the next stage. Instead the carry is stored in present stage and updated as added value in next stage. Hence delay due to carry is reduced.

Carry save adders are used to calculate the partial products in integer multiplication. This allows for

architectures, where a tree of carry-save adders is used to calculate the partial products very fast. One normal adder is then used to add the last set of carry bits to the last partial products to give the final multiplication result. Usually, a very fast carry-look ahead or carry-select adder is used for this last stage, in order to obtain the optimal performance.

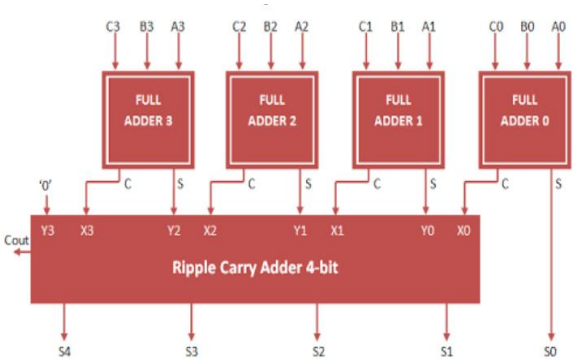

Figure 3. Block diagram of Carry Save Adder

Carry save adders applied in partial product lines of an array multiplier circuit used to speed-up the summation of partial products in order to speed up the carry propagation along the array. The use of the carry save signal representation is a powerful technique in the high speed implementation of arithmetic circuits that solve the joint module selection and retiming problem. Thus using carry save adders avoids carry propagation and will result in higher throughput. The CSA design automatically avoids the delay in the carry out bits. It is well known that carry save arithmetic is a useful technique in the implementation of high speed arithmetic functions since it saves logic and time.

IV. BSD ADDER

Addition is the most important and frequently used arithmetic operation in computer systems. Generally, two methods can be used to speed up the addition operation. One is to explicitly shorten the carry-propagation chain by using circuit design techniques, such as detecting the completion of the carry chain as soon as possible, carry look-ahead, etc. Another is to convert the operands from the binary number system to a redundant number system, e.g., the signed-digit number system or the residue number system, so that the addition becomes carry-free (CF). In second method, it implicitly eliminates the Carry-propagation chain so that fast addition can be done, at the expense of conversion between the binary number system and the redundant number system. Fast adder can be designed using ripple carry or carry look ahead adder. But in case of ripple carry adder the delay will be more as the carry should be propagated entire bit width. So speed will be reduced. Carry look ahead adders are faster than ripple carry adders but the complexity of the circuitry increases as the number of bits increases. Use of non-conventional number systems in designing Adder is gaining attention in recent years because of their facility to provide carry free addition thus enhancing the achievable processing speed. The property of carry propagation chain elimination tends to make the processing faster.

The binary signed number system is used implicitly or explicitly to speed up arithmetic operations in many digital system. This is due to its capability of carry free addition and regular layout. Also, with the proper selection of the encoding scheme used to encode the BSDN digit set D= {-1, 0, 1} into binary bits, an error detection capability feature can be gained. The redundant Signed radix 2 number digit set contains {-1, 0, 1}. The number of digits present in the digit set will be more than the radix. So each number can be represented in many ways. Example: Number 9 in decimal can be represented in redundant binary as follows.

1 0 0 1 ------- 9

1 -1 0 0 1 ------- 9 (bsd)

In case of conventional binary addition there will be carry propagation. Carry will be propagated till the end. In order to cope with the problem of carry propagation the most appropriate approach is elimination of carry propagation. Addition in case of signed digit (SD) representation is carry free. steps involved in adding the two operands using SD are as below.

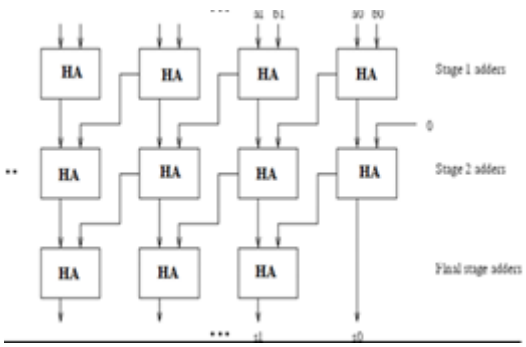

Figure 4. Block diagram of Binary Signed Digit adder

Step 1. Both the operands are added to get the position sum, pi. Step 2. Interim sum wi and transfer

digit ti are determined. The interim sum and transfer digit are selected in such a way that they should be within the digit set i.e. -1,0,1 and after adding interim sum and transfer digit the final sum also should be in the selected digit set. Step 3. Transfer digit is added with the interim sum to get final sum

TABLE 2. Transfer Digit and Interim Sum for Binary Signed Digit Radix 2

X_i	Y_i	X_i+Y_i	t_i	w_i
-1	-1	-2	-1	0
-1	0	-1	-1	1
0	-1	-1	0	-1
-1	1	0	0	0
1	-1	0	0	0
0	0	0	0	0
0	1	1	0	1
1	0	1	1	-1
1	1	2	1	0

xi = 1 0 0 1----9

Yi = 0 1 0 0 ---4

-1 1 0 -1

1 0 0 1

1 -1 1 1 -1

The binary signed digit number makes it possible to perform addition with carry propagation chains limited to a single digit position and has to be used to speed up the arithmetic operations. The main advantages of BSD adder are Carry free addition, Independent of word length, highly efficient and Reduction in delay.

V. RESULTS AND DISCUSSION

The carry save adder, carry look ahead adder and binary signed digit adders are simulated using Modelsim and implemented in Xilinx. A comparison of the hardware complexity of the three adders is given in the table below. The performance of the adders is compared based on the number of macro cells and functional block inputs used. The table shows that the resource utilization of the BSD adder is lesser than the other two adders.

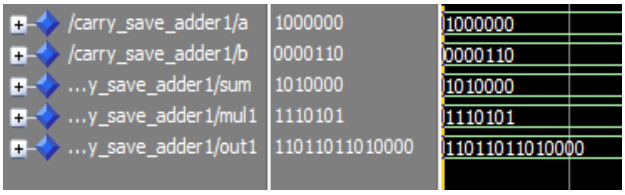

Figure 5. Simulation result of Carry Save Adder

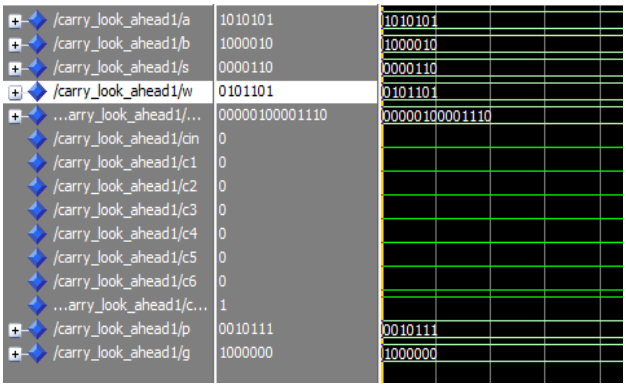

Figure 6. Simulation result of Carry Look Ahead adder

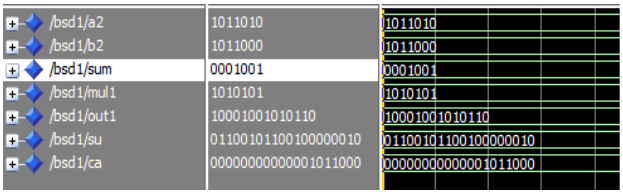

Figure 7. Simulation result of Binary Signed Digit adder

Table 2 Resources Summary

Adder	Macro cells Used	Function Block Inputs Used
Carry Save	27/72 (38 %)	65/216 (31%)
Carry Lookahead	21/72 (30 %)	46/216 (22%)
BSD	18/72 (25 %)	43/216 (20%)

V. CONCLUSION

This paper describes the design and implementation of CLA, CSA and BSD adder. The adders are simulated using Modelsim. The performance of the BSD adder is better than the other two adders. The BSD adder significantly requires fewer numbers of resources and hence it is cost effective when compared to the design of the other two adders. Since the proposed adder satisfies the most essential parameter needed in an adder, it is chosen to be the most efficient adder

VI. REFERENCES

[1] L.Wanhammer, (1999). "DSP Integrated Circuits," Academic Press, ISSN No.

[2] J.Melander, (1997) "Design of SIC FFT Architectures," Linkoping Studies in Science and Technology, Thesis No.618, Linkoping University, Sweden.

[3] K.K.Parhi, (2001), "VLSI Digital Signal Processing Systems: Design and Implementation," John Wiley & Sons..

[4] S. Knowles, (2001) "A family of adders," Proceeding of 15th Symp. Computer Arithmetic, pp. 277–281.

[5] J. Park, H. C. Ngo, J. A. Silberman, and S. H. Dong, (2000) , "470 ps 64 bit parallel binary adder," Symp. VLSI Circuits, pp. 192–193.

[6] Ning Zhu; Wang-Ling Goh; Kiat-Seng Yeo, (2009). "An enhanced low-power high-speed Adder For Error-Tolerant application," Integrated Circuits, ISIC '09. Proceedings of the 12th International Symposium. pp. 69-72.

[7] G.Wang, M.Ozaydin, and M.Tull, (2002). "High-Performance Divider Using Redundant Binary Representation," in Proc. 45th Midwest Symp. On Circuits andSystems,Vol.1, Aug.2002, pp.471-474.

[8] Kharbash, F. Chaudhry, G. M., "Binary Signed Digit Number Adder Design with Error Detection Capability," Sharjah 9th International Symposium on signal processing and Its Applications, ISSPA, pp.1-4,

[9] A.Avizenis, "Signed-Digit Number Representations for Fast Parallel Arithmetic 1961, " IRE Trans. On Electronic Computers, pp.389-400.

[10] Y.Harata, Y.Nakamura, H.Nagase, M.Takigawa, and N.Takagi, (1987) "A High Speed Multiplier Using a Redandant Binary Adder Tree," IEEE J. of Solid-State Circuits, Vol. 22, no. 1, pp. 28-31.