International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume 2, Issue 4 (July-Aug 2014), PP. 93-95

COMPARISON ANALYSIS OF 16-BIT ADDERS

Sangeeta Rani[1], Er. Sachin Kumar[2]
[1]M.Tech Student, [2]Faculty
Department of Electronics & Communication M. Tech Student
MERI COLLEGE OF ENGINEERING & TECHNOLOGY
Sampla

***Abstract*— Adders are one of the most widely digital components in the digital integrated circuit design and are the necessary part of Digital Signal Processing (DSP) applications. With the advances in technology, researchers have tried and are trying to design adders which offer either high speed, low power consumption, less area or the combination of them. The addition of the two bits is very Based on the various speed-up schemes for binary addition, a comprehensive overview and a qualitative evaluation of the different existing basic adder architectures are given in this paper. In addition, their comparison is performed in the thesis for the performance analysis. We will synthesize the adders - Ripple Carry adder, Carry look- ahead Adder, Carry Save Adder in ISE XIILINX 10.1 by using HDL - Verilog and will simulate them in Modelsim 6.4a. We will Compare above mentioned adders in terms of Delay, Slices Used and Look up tables used by the adder architecture.**

***Key words*— Ripple Carry Adder, Carry Look Ahead adder, Carry Save adder.**

I.INTRODUCTION

Adders are commonly found in the building blocks of microprocessor and digital signal processing chips. Adders not only does addition in the processors and other digital circuits but also does subtraction, multiplication and division. Thus adder is the most basic arithmetic operation of any digital circuit. Chen et. Al found that addition is the most frequently used operation [5]. Basic unit used in any adder structure is Full adder. By cascading the full adders, we have a ripple carry adder. This kind of networks is called an iterative [1] logic array. In terms of iterative logic array, a ripple carry adder is referred to as an array, and the full adder is referred to as a cell. A basic adder structure is the ripple-carry adder, there are other adder structures providing higher speed than Ripple Carry Adder and Speed of the adder structure depends on the length of the operands. The other adder structure described in this paper are Ripple Carry Adder, Carry Look ahead adder and Carry Save adder, Ripple carry adder is the slowest of all adders.

In this work, comparison analysis of three adders is presented. The Adders presented in this paper are all modeled by using Verilog for 16-bit data ISE. XILINX v10.1 is used as synthesis tool. Modelsim 6.2a is used to get timing simulation.

II.RIPPLE CARRY ADDER:

Ripple carry addition was used in the first electronic computers. Ripple carry adder uses the O(n) area and O(n) Delay where n is the width of operand [2]. Ripple carry adders uses the propagation of the carry from least significant bit to most significant bit. i.e. carry out of ith stage is fed to the input of the i+1th stage along with the inputs Ai+1 andBi+1. Hence the delay is introduced automatically due to propagation of the carry from LSB to MSB. For n-Bit Ripple carry adder, n full adders are used. Figure 1 shows the n bit ripple carry adder using n full adders. Because of the propagation of carry from right to left, delay is maximum in this adder.

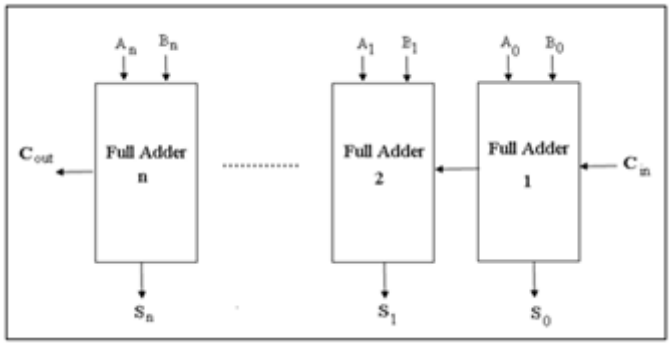

Figure 1: n Bit Ripple Carry Adder using n full bit adders

There also exists Probabilistic Ripple carry adder which calculates that a bias voltage scaling (BIVOS) technique can yield much better energy saving than simply supplying all FAs with the same voltage.

Note: Ripple carry adders are useful when n is small.

III.CARRY LOOK-AHEAD ADDER

It is a synchronous addition which was introduced by Weinberger & Smith in 1956. It is based on the principle that even if the number of inputs is increased the carry propagation can still be reduced [4]. Its concept is to propagate the carry as fast as possible to the last stage for all operand values. In the carry look ahead adder Carry Generation signal and carry propagate signals are generated. A carry is generated when

both inputs A & B are 1 and Carry is propagated when any of the input A or B is 1.

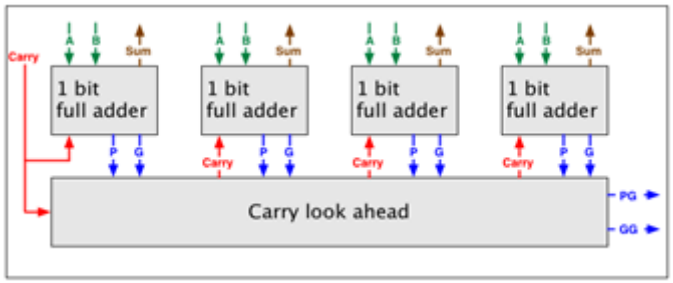

Figure 2: 4 Bit Carry Look Ahead Adder

$Gi = Ai \cdot Bi$ called the generate function ……(1)
$Pi = Ai \oplus Bi$ called the propagate function ……(2)

Equations of Full adder are
$S = A \oplus B \oplus C = Pi \oplus Ci$ ……..(3)
$C = A.B + B.C + C.A = A.B + C(A+B)$ ……..(4)

We can also write the (4) eq. as
$C = A.B + C(A \oplus B)$
$C = G+P.C$
And

$C_{i+1} = G_i + P_i C_i$ …….(5)

Now

$C0 = G0 + P0.Cin$

$C1 = G1 + P1\ G0 + P1\ P0\ .Cin$

$C2 = G2 + P2\ .\ G1 + P2\ .\ P1\ .\ G0 + P2\ .\ P1\ .\ P0\ .\ Cin$

$C3 = G3 + P3.G2 + P3.P2.G1 + P3.P2.P1\ .G0 + P3.P2.P1.P0.Cin$

.
.
.
.

$C_{n+1} = Gn + Gn\text{-}1.Pn + Gn\text{-}2.Pn.Pn\text{-}1 + \ldots\ldots\ldots + G0.PnPn\text{-}1 \ldots P1 + PnPn\text{-}1 \ldots P1P0Cin$ ……..(6)

One of the most popular methods to reduce delay is to use a carry look-ahead mechanism. By using carry look-ahead mechanism, the propagation delay is reduced to four-gate level irrespective of the number of bits in the adder. [7]

IV. Carry Select Adder (CSA)

In the CSA, addition is divided into m modules of k bits each. Each module has two k- bit wide adders (generally, CRAs), which perform two additions at the same time: one with Cin=0 (Cin means carry-in) and another with Cin =1. CSA is one of the fastest adders. On the other hand, the CSA area is more than twice as that of the RCA, for a given bit-width. The time and area complexities of the CSA are O(n/k) and O(2n), respectively [3]. Carry select adder is not area efficient because it uses multiple pairs of Ripple Carry Adders. The carries within each module are computed in parallel by using "generate." and "propagate." in order to anticipate the module carry out.

There is another adder called as A1CSA and it is inspired by the CSA. Also the numbers of adder used in this adders is half as that of the CSA because the adder with Cin=1 in a CSA is replaced by "A1" block which is less expensive logic than the CSA.

Using the idea of CSA iteratively, the delay of the adder can significantly be reduced. It is proven that if the delay of multiplexers is negligible, the delay of the iterative CSA will grow with the square root of the number of bits. [8]

V. Carry Save Adders (CSAA)

There are many cases where it is desired to compute the sum of more than two numbers together. The straightforward way of adding together m numbers (all n bits wide) is to add the first two, then add that sum to the next, and so on. This requires a total of m−1 additions, for a total gate delay of O (m lg n) Instead, a tree of adders can be formed, taking only O (lg m • lg n) gate delays. Using carry save addition, the delay can be reduced further still. The idea is to take 3 numbers that we want to add together, x + y + z, and convert it into 2 numbers c + s such that x + y + z = c + s, and do this in O(1) time. The reason why addition cannot be performed in O(1) time is because the carry information must be propagated. In carry save addition, we refrain from directly passing on the carry information until the very last step.

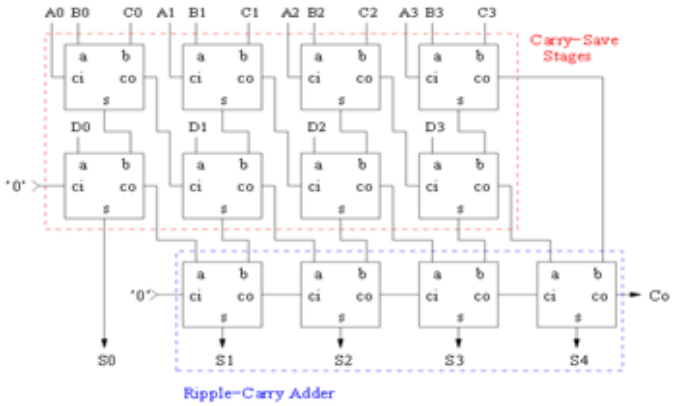

Figure 2: Carry save adder

The carry save approach breaks this process down into two steps. The first is to compute the sum ignoring any carries: The final addition is then computed as:

1. Shifting the carry sequence C left by one place.
2. Placing a 0 to the front (MSB) of the partial sum sequence S.
3. Finally, a ripple carry adder is used to add these two together and computing the resulting sum. [6]

VI. Simulation Results:

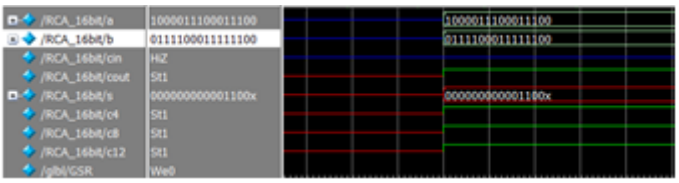

Figure 3: Addition using 16 Bit Ripple Carry adder

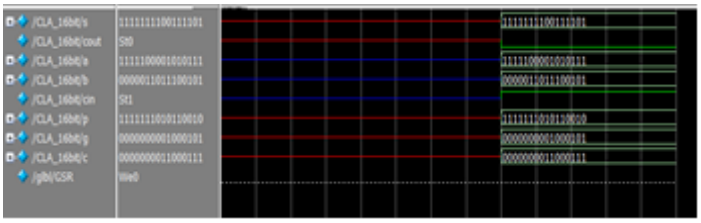

Figure 4: Addition using 16 Bit Carry Look-Ahead adder

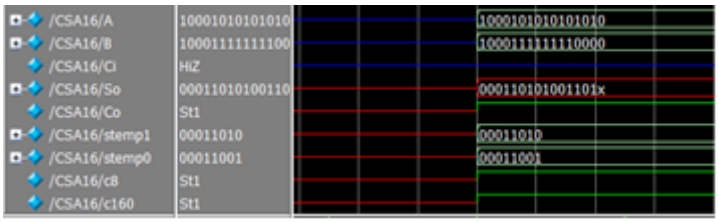

Figure5: Addition using 16 Bit Carry Save adder

(Target Device: 3s100evq100-4)

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	32	1,920	1%	
Logic Distribution				
Number of occupied Slices	24	960	2%	
Number of Slices containing only related logic	24	24	100%	
Number of Slices containing unrelated logic	0	24	0%	
Total Number of 4 input LUTs	32	1,920	1%	
Number of bonded IOBs	50	66	75%	

TABLE I. DEVICE UTILIZATION SUMMARY OF RIPPLE CARRY ADDER

Maximum combinational path delay: 24.686ns

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	18	960	1%
Number of 4 input LUTs	32	1920	1%
Number of bonded IOBs	50	66	75%

TABLE II. DEVICE UTILIZATION SUMMARY OF CARRY LOOK-AHEAD ADDER

Maximum combinational path delay: 24.686ns

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	27	960	2%
Number of 4 input LUTs	48	1920	2%
Number of bonded IOBs	50	66	75%

TABLE III. DEVICE UTILIZATION SUMMARY OF CARRYS SAVE ADDER

Maximum combinational path delay: 19.092ns

CONCLUSION

Logic Utilization	RCA	CLA	CSAA
No.of Slices	24	18	27
Number of 4 input LUTs	32	32	48
Number of Bonded IOBs	50	50	50
Delay	24.686ns	24.686ns	19.092ns

TABLE IV. COMPARISON ANALYSIS OF 16-BIT ADDERS

Simulation of Ripple Carry adder, Carry look- ahead Adder and Carry Save Adder is done in ISE XIILINX 10.1 by using HDL - Verilog and Modelsim 6.4a. By comparing them in terms of delay i have reached the conclusion that Carry Save adder has the lowest delay among them and hence it is fastest of them all. Also Ripple carry adder and Carry look ahead adder has the same and maximum delay but according to theory carry look ahead adder has less delay than Ripple carry adder (CLA does not wait for the carry to propagate on each stage but uses direct equations to calculate sum and carry). And number of Look Up Tables used is minimum in Ripple carry adder, and Carry look–ahead adder but maximum in carry save adder. Also number of IOB is same in all the adders. No. of slices used by Carry Look ahead adder is minimum.

REFERENCES

[1] Seji Kahihara and Tsutomu Sasao, "On the adders with minimum tests", IEEE Proceedings of the 5th Asian Test Symposium 1997.

[2] Chetana Nagendra, Mary Jane Irwin and Robert Michael Owens, "Area-Time-Power Tradeoffs in Parallel Adders", IEEE Transactions on circuits and systems-II: Analog and Digital signal Processing, Vol. 43, No.10, oct. 1996, pp. 689 – 702

[3] Jucemar Monteiro, José Luís Güntzel Luciano Agostini, "A1CSA: An Energy-Efficient Fast Adder Architecture for Cell-Based VLSI Design" Electronics, Circuits and Systems (ICECS), 2011 18th IEEE International Conference on 11-14 Dec. 2011 pp. 442 – 445.

[4] James Levy and Jabulani Nyathi, "A High Performance, Low Area Overhead Carry Lookahead Adder"

[5] D. C. Chen, L. M. Guerra, E. H. Ng, M. Potkonjak, D. P. Schultz, and J. M. Rabaey, "An intcgrated system for rapid prototyping of high performancc algorithm specific data paths," in Proc. Application Specific Array Processors, Aug 1992, pp. 134-148.

[6] Raminder Preet Pal Singh, Parveen Kumar, Balwinder Singh.International Performance "Analysis of 32-Bit Array Multiplier with a CarrySave Adder and with a Carry-Look-Ahead Adder", Journal of Recent Trends in Engineering, Vol 2, No. 6, November 2009.

[7] May Phyo Thwal, Khin Htay Kyi, and Kyaw Swar Soe; "Implementation of Adder-Subtracter Design with Verilog HDL" World Academy of Science, Engineering and Technology Vol:2 , March 2008.

[8] Behnam AmelifardFarzan FallahMassoud Pedram, "Closing the Gap between Carry Select Adder and Ripple Carry Adder: A New Class of Low-power High-performance Adders"

[9] Padma DeviAshima Girdher," Improved Carry Select Adder with Reduced Area and Low Power Consumption", International Journal of Computer Applications (0975 – 8887) Volume 3 – No.4, June 2010.

[10] M. Morris Mano; "Digital logic and computer design"; Prentice hall of India Private limited (2003).

[11] Chip-Hong Chang, Jiangmin Gu, Mingyan Zhang, "A Review of 0.18-um Full adder Performances for Tree Structured Arithmetic Circuits," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 13, no. 6, pp. 686-695, June 2005.