Design Tradeoff Analysis and Implementation of Digital Binary Adders Using Verilog

Priyanka Pawar*, Dr. Vaibhav Neema**

Department of Electronics, Institute of Engineering & Technology, DAVV, Indore, M.P. India
*priyankapawar001@gmail.com, **vaibhav.neema@gmail.com

Abstract: Hardware designs targeting communication and DSP applications consists of a large number of data path elements such as adders, multipliers, comparators, shifter etc. These elements are main contributors to the power consumption of digital circuits. At present, most of the popular hardware synthesis tools give higher priority to delay. So the synthesis tools tend to generate data path architecture for faster implementation.

An Adder is one of the key hardware blocks in most digital and high performance systems such as FIR filters, digital signal processors and microprocessors etc. With advances in technology, many researchers have tried and are trying to design Adders which offer either of the following- high speed, low power consumption, regularity of layout and hence less area or even combination of them in Adders. The delay of an adder circuit often determines the clock cycle time of a processor, especially if it falls in the critical path of the design. One of the primary causes, for the delay of an adder is the rippling nature of the carry. The key to fast addition is to compute carry bits for every bit position in parallel. Thus making them suitable for various high speed, low power, and compact VLSI implementations. However area and speed are two conflicting constraints. So improving speed results always in larger areas. So here we try to find out the best trade off solution among the both of them. Hence in this paper we have first tried to design different adders and then compare their speed and complexity of circuit i.e. the area occupied.

While comparing the adders we found out that Ripple Carry Adder had a smaller area while having lesser speed, in contrast to which Carry Select Adders are high speed but possess a larger area. And a Carry Look Ahead Adder is in between the spectrum having a proper tradeoff between time and area complexities. The result of our project helps us to make a proper choice of different adders in fabricating in different arithmetic units as well as making a choice among different adders in different digital applications according to requirements. All the programs and results have been given in the following sections. Further work on Low Power Techniques on different multipliers using these adders needs to be done in order to make us choose a proper multiplier in accordance with the requirements by making the best possible trade off choice between Speed and Power in different circumstances.

Index Terms – Binary Adders, Verilog, Xilinx ISE 12.1

I. Introduction

Regarding the efficient implementation of an arithmetic unit, the binary adder structures become a very critical hardware unit. In any book on computer arithmetic, someone looks that there exists a large number of different circuit architectures with different performance characteristics and widely used in the practice. Although many researches dealing with the binary adder structures have been done, the studies based on their comparative performance analysis are only a few. In this project, qualitative evaluations of the classified binary adder architectures are given. Among the huge member of the adders we wrote Verilog code for Ripple-carry, Carry-select and Carry-look ahead along with one more adder that is Manchester carry adder to emphasize the common performance properties belong to their classes. In the following section, we give a brief description of the studied adder architectures.

With respect to asymptotic delay time and area complexity, the binary adder architectures can be categorized into four primary classes as given in
Table-1. The given results in the table are the highest exponent term of the exact formulas, very complex for the high bit lengths of the operands. The first class consists of the very slow ripple-carry adder with the smallest area. In the second class, the carry-skip, carry-select adders with multiple levels have small area requirements and shortened computation times. From the third class, the carry-look-ahead adder and from the fourth class, the parallel prefix adder represents the fastest addition schemes with the largest area complexities.

Complex (A)	Delay (T)	Product (A*T)	Adder Class Schemes
O(n)	O(n)	O(n2)	Ripple carry(1)
O(n)	$O(n^{1/*1+1})$	$O(n^{1+2/+1})$	Carry select(2) Carry Skip(2)
O(n)	O(log n)	O(nlogn)	Carry look Ahead(3)

Table – 1 Categorization of addersw.r.t to delay time and class

II. POWER CONSUMPTION IN CMOS CIRCUITS

Power has always been one of the foremost issues in system design. No matter what the design scale, there is a direct correspondence between power dissipation and performance/functionality, battery life, cost and size. A hand held device, for example, must be small. Similarly, a personal computer should be inexpensive; few are willing to pay for exotic cooling technologies. In fact, high performance processors have already reached the power density limit for cost effective cooling.
All these things limit the amount of power a processing chip can burn. Chip Power can be divided into two main components:

a) Dynamic power/Switching Power
b) Static Leakage Power

a) Dynamic Power

Dynamic power dissipation, ignoring short circuit current which is usually a small fraction of dynamic power is given by,

$$P = (1/2)\, CV f$$

Where,
C is the average total on chip capacitance switched per cycle.
f is the clock frequency and
V is the Supply Voltage.

b) Static Power

Static power dissipates when current flows through a transistor even when the transistor is off. The short circuit power is dissipated by a CMOS gate during a short period of time when both pull up and pull down networks conduct current. Static power is consumed even when chip is quiescent.

1. Rationed circuit's burn power in fight between ON transistors.
2. Leakage draws power from nominally OFF devices.

III. LOW POWER ADDER DESIGN

As we get closer to the limits of scaling in Complementary metal-oxide semiconductor (CMOS) circuits, power and heat dissipation issues are becoming more and more important. In recent years, the impact of pervasive computing and the internet have accelerated this trend. The applications for these domains are typically run on battery-powered embedded systems. The resultant constraints on the energy budget require design for power as well as design for performance at all layers of system design. Thus reducing power consumption is a key design goal for portable computing and communication devices that employ increasingly sophisticated and power hungry signal processing techniques. Flexibility is another critical requirement that mandates the use of programmable components like FPGAs in such devices.

The speed of addition is limited by the time required to propagate a carry through the adder in digital adders. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated into the next position.

The CSA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum .In Carry select adder, blocks of bits are added in two ways: one assuming a carry-in of 0 and the other with a carry-in of 1.This results in two pre- computed sum and carry-out signal pairs. The correct output is selected based on the original carry-in. Generally multiplexers are used to propagate carries.

The RCA uses multiple full adders to perform addition operation. Each full adder inputs a carry-in, which is the carry-out of the proceeding adder. This selects the corresponding sum bit from the next block of data.
However, the CSA is not area efficient because it uses multiple pairs of Ripple Carry Adders to generate partial sum and carry by considering carry input Cin=0 and Cin=1, then the final sum and carry are selected by the multiplexers.

Thus, the carry select adder achieves higher speed of operation at the cost of increased number of devices used in the circuit. This in turn increases the area and power consumed by the circuits of this type of structure. Demands of low power devices growth of battery powered systems, Mobility, Portability, Reliability and Cost, Environmental effects.

Ripple Carry Adders (RCA)

The well-known adder architecture, ripple carry adder is composed of cascaded full adders for n-bit adder, as shown in figure-1. It is constructed by cascading full adder blocks in series. The carry out of one stage is fed directly to the carry-in of the next stage. For an n-bit parallel adder it requires n full adders.

i. Not very efficient when large number bit numbers are used.
ii. Delay increases linearly with bit length.

Logic Equations:

$$gi = ai\ bi$$
$$p = ai \text{ xor } bi$$
$$C\ i{+}1 = gi + pici$$
$$Si = pi \text{ xor } ci$$

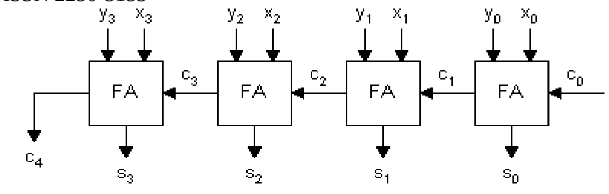

Figure – 1 ***Parallel adder, A 4 bit Ripple Carry Adder***

Carry Select Adders (CSA)

In Carry select adder scheme, blocks of bits are added in two ways: one assuming a carry-in of 0 and the other with a carry-in of 1.This results in two precomputed sum and carry-out signal pairs (s0i-1:k , c0i ; s1i-1:k , c1i) , later as the block's true carry-in (ck) becomes known, the correct signal pairs are selected. Generally multiplexers are used to propagate carries.

i. Because of multiplexers larger area is required.
ii. Have a lesser delay than Ripple Carry Adders (half delay of RCA).
iii. Hence we always go for Carry Select Adder while working with smaller no of bits.

Logic Equations:

$$S_{i-1:k} = C_k S^0_{i-1:K+} C_k S^1_{i-1:K} \quad C_i = C_k C^0_{i+} C_k C^1_i$$

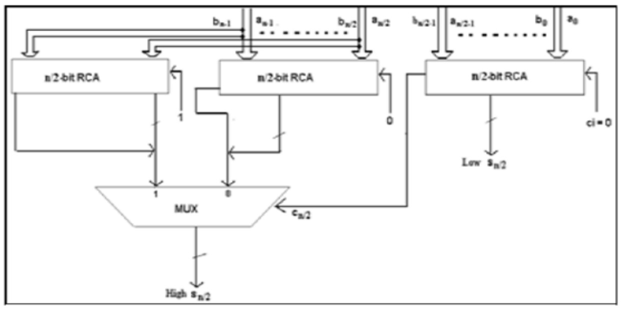

Figure – 2 n-bit *Carry Select Adder*

Carry Look Ahead Adders (CLA)

Carry Look Ahead Adder can produce carries faster due to carry bits generated in parallel by an additional circuitry whenever inputs change. This technique uses carry bypass logic to speed up the carry propagation.

Logic equations:

The Propagate P and generate G in a full-adder, is given as:

Pi = Ai Bi Carry propagate
Gi = AiBi Carry generate

Notices that both propagate and generate signals depend only on the input bits and thus will be valid after one gate delay. The new expressions for the output sum and the carryout are given by:

Si = Pi Ci-1
Ci+1= Gi + PiCi

These equations show that a carry signal will be generated in two cases:
1) If both bits Ai and Bi are 1
2) If either Ai or Bi is 1 and the carry-in Ci is 1.

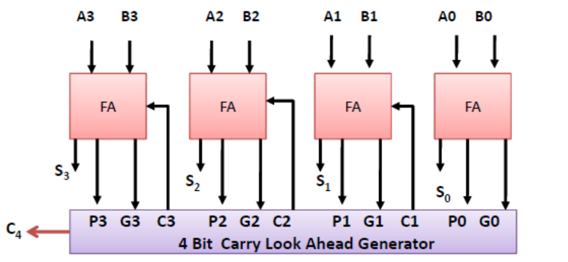

Figure – 3 **4 bit *Carry Look Ahead Adder***

MANCHESTER ADDER

The propagation time, when calculating the sum of two binary strings ***A*** and ***B*** using any generic parallel adder, can be speed up significantly if we utilize a **Manchester cell** in the design of that particular adder.

Generation and Propagation
Here we provide a brief summary of the underlying mechanics behind the decision to propagate or generate a carry out (refer to *carry skip mechanics* for a thorough explanation)

Boolean Equations:

Gi = Ai⊕ Bi	-- carry generate of ith stage
Pi = Ai⊕ Bi	-- carry propagate of ith stage
Si = Pi ⊕Ci	-- sum of ith stage
Ci+1 = Gi + PiCi	-- carry out of ith stage

The condition for a carry generate (generation of a new carry) to occur at any stage of the addition is ***Ai = Bi*** making the carry out, ***Ci+1***, depends solely on ***Gi*** (i.e. ***Ci+1 = Gi***). A carry propagate, on the other hand, has the requirement that ***Ai Bi***, hence producing ***Ci+1 = Ci***.

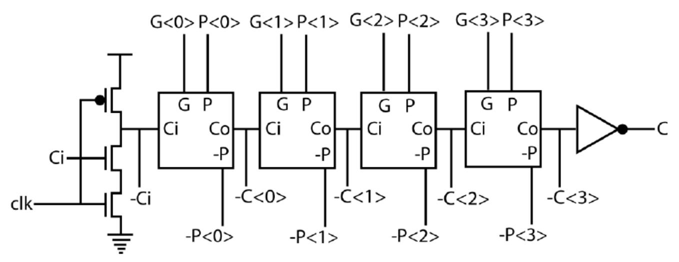

Figure – 4 4-bit Machester carry section

IV. PRACTICAL IMPLEMENTATION

Using Verilog programming language and Xilinx ISE 12.1 as synthesis tool we have implemented the adders mentioned above for different word length and then generated synthesis report for comparing delay & device utilization of different adders.

V. RESULTS

ADDERS	NUMBER OF SLICES	NUMBER OF 4 I/P LUTS
4 bit Ripple carry Adder	4	8
8 bit Ripple carry Adder	9	16
4 bit Carry look ahead adder	7	13
16 bit Carry look ahead adder	19	33
4 bit Carry select adder	6	11
16 bit Carry select adder	29	34
4 bit Manchester adder	1	1
16 bit Manchester adder	18	32

Table –2 Device utilization comparison of Adders

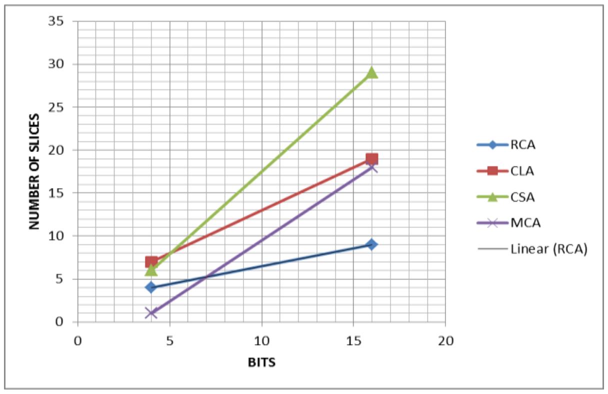

Figure – 5 ***Comparison of utilization of number of Slices***

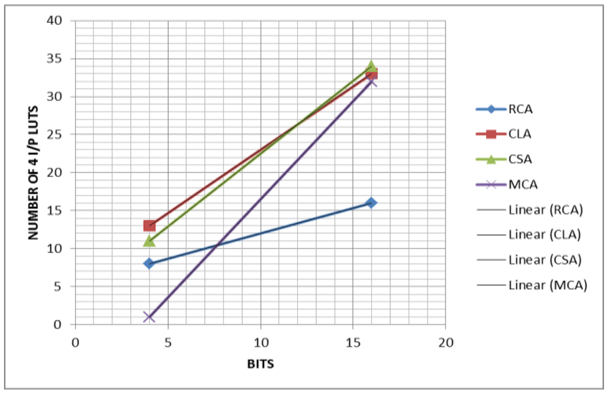

Figure – 6 ***Comparison of utilization of number of 4 i/p LUTS***

ADDERS	Logic Level	Delay (in ns)
4 bit Ripple carry Adder	6	5.959
8 bit Ripple carry Adder	10	13.203
4 bit Carry look ahead adder	5	7.962
16 bit Carry look ahead adder	17	20.388
4 bit Carry select adder	6	8.580
16 bit Carry select adder	11	14.725
4 bit Manchester adder	3	5.895
16 bit Manchester adder	18	21.690

Table –3 Delay comparison of Adders

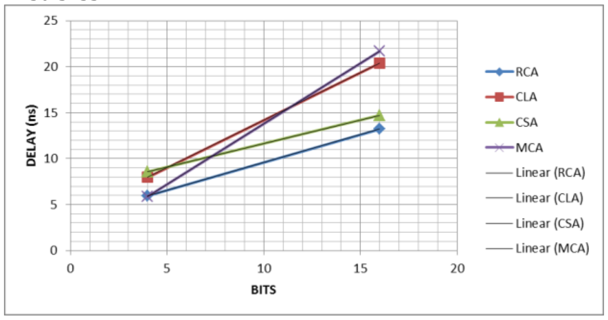

Figure – 7 ***Comparison of delay***

VI. CONCLUSION

We studied about different adders theoretically as well as practically and by implementation and among compared them by different criteria like Area, Delay and then Area-Delay Product etc. so that we can judge to know which adder was best suited for situation. Comparing the performance metrics of adders for different word lengths using Verilog and Xilinx as synthesis tools, and the tradeoffs becomes apparent. As can be seen there exist an inverse relationship between time delays, operating speed, and circuit area, in this case the number of slices / LUT (measure of the area). The ripple carry adder, the most basic of flavors, is at the one extreme of this spectrum with the least amount of LUTs but the highest delay. The carry select adder on the other hand, is at the opposite corner since it has the lowest delay (half that of the ripple carry's) but with a larger area required to compensate for this time gain. Finally, the carry look-ahead is middle ground. Along with that another adder which is Manchester adder can be used for designing of larger adder designs by slight modification in its architecture thus all we came to a conclusion that Carry Select Adders are best suited for situations where Speed is the only criteria. Similarly Ripple Carry Adders are best suited for Low Power Applications. But Among all the Carry Look Ahead Adder had the least Area-Delay product that tells us that, it is suitable for situations where both low power and fastness are a criteria such that we need a proper balance between both as is the case with our research.

REFERENCES

[1] M. J. Flynn, S. F. Oberman, Advanced Computer Arithmetic Design. John Wiley & Sons, Inc, 2001.

[2] Digital Core Design, http://www.dcd.pl/

[3] Dr. A. Sahu , "CS221: Digital Design Data Path Component Adder" Dept of Comp. Sc. & Engg. Indian, Institute of Technology Guwahati

[4] Lecture 2," Parallel Adders" by Electronics and Communication Engineering Department , Concordia University

[5] M. Morris Mano, Digital Design second edition, Prentice Hall, 1991

[6] N.H.E. Weste and K. Eshraghian, Principle of CMOS VLSI Design, Addison-Wesley Company, 1992

[7] Peter Pirsch, Architectures for digital signal processing, John Wiley & Sons, 1998