ISSN No: 2309-4893
International Journal of Advanced Engineering and Global Technology
Vol-03, Issue-07, July 2015

Design and Implementation of 32 Bit Unsigned Multiplier Using CLAA and CSLA

K.Bhaskar, Athula Maleha Mirza, Mangipudi Venu

Asso.Prof in ECE Department
Sarada Institute Of Technology and Science,
Khammam, A.P, India

Abstract

In computational circuits the adders plays a key role in arithmetic operations. Adders like Ripple carry adder, Carry lookahead adder, Carry select adder, Carry skip adder and carry save adder etc. In this paper, a high performance and low power 32bit unsigned multiplier is proposed using adders. The design of 32bit unsigned multiplier using CLAA and 32bit unsignedmultiplier using CSLA multiplies and gives the product term of 64bit. The CLAA based multiplier and CSLA based multiplieruses the same delay for multiplication operation. These two 32-bit unsigned multipliers are simulated using Modelsim10.1band synthesized using Xilinx14.2.

Keywords: Low Power**,** Ripple carry adder, Carry look ahead adder (CLAA), Carry select adder (CSLA), Unsigned Multiplier, Xilinx.

1. Introduction

The design of low power, low area and high performance logic systems are most essential in VLSI system design. Thedigital systems such as embedded systems, Digital signal processing (DSP), Data process unit and Communication network the arithmetic operations like addition, subtraction; multiplication and division are mostly used and plays akey role in various applications. In electronics, the adder is a digital circuit. The adder can be used to perform the addition of binary numbers. In many computers and different type of processors and controllers, adders are not onlyused in the arithmetic logic unit and not only perform the addition operation. They are used to calculate addresses, registers and different type of operations. Multiplication is one of the basic arithmetic operations. Multiplicationoperation is also called as a adding and shifting method. Multiplication operation involves two methods one is Generation of partial products and another one is summation. The speed of multiplication is mainly depends on thePartial product generation and/or summation. The multiplication speed will be high when the generation of partialproducts is less. In this, we are going to implement the Two 32-bit unsigned multipliers using adders. . In many processors Carry select adder is used to perform the fast arithmetic operations. The carry propagation delay time is veryhigh in Ripple carry adder. To overcome this problem Carry look ahead adder is proposed. This type of adder does notrequire the carry propagation step by step. The CLAA and CSLA adders have the similarity properties. For themultiplication process the both adders (CLAA & CSLA) will have the nearly same delay speed. Here The Two 32-bit unsigned multipliers multiplies (N*N) and gives the 64 bit (2N) output.

2. Adders

In electronics, an adder is digital circuit which is used to perform the addition of binary digits. In vlsi system design using adders we are increasing the performance of the module. In this section we will review the different types ofadders and their characteristics and performance.

2.1 Half adder

The half adder adds two binary inputs a,b and its have the two binary outputs Sum and carry. The logic diagram of halfadder is shown in below figure1.

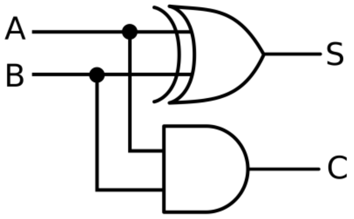


Figure 1 Half adder logic diagram

2.2 Full adder

The full adder adds three binary inputs a,b,cin and its have the two binary outputs Sum and Carry. The logic diagram of full adder as shown in figure 2.

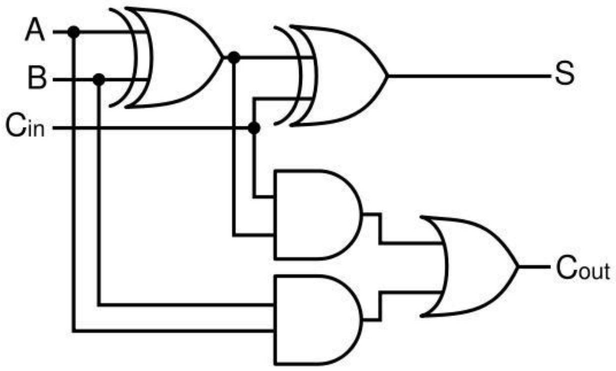


Figure 2 Full adder logic design

2.3 RIPPLE CARRY ADDERS

The well known adder architecture, ripple carry adder is composed of cascaded full adders for n-bit adder, as shown in figure.3 .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.

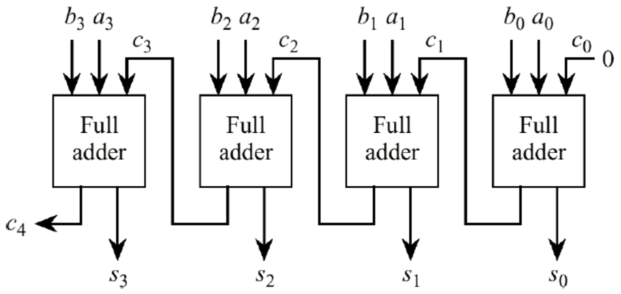


Figure 3 Ripple Carry Adder

Disadvantages:

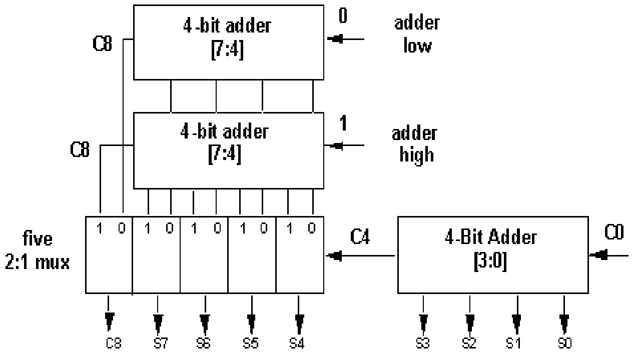
- Not very efficient when large number bit numbers are used.
- Delay increases linearly with bit length.

2.4 CARRY SELECT ADDERS

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 recomputed 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.

Because of multiplexers larger area is required.

- Have a lesser delay than Ripple Carry Adders (**half delay of RCA**).
- Hence we always go for Carry Select Adder while working with smaller no of bits.



compute the high order sums in parallel
one addition assumes carry in = 0
the other assumes carry in = 1

Figure 4 Carry Select Adders

2.5 CARRY LOOK AHEAD ADDERS

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.

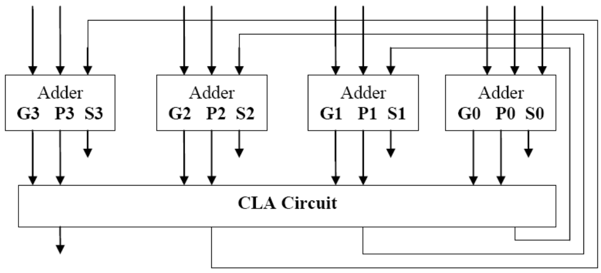


Figure 5 4-Bit CLA Logic equations

Let **ai** and **bi** be the augends and addend inputs, **ci** the carry input, **si** and **ci+1** , the sum and carry-out to the **ith** bit position. If the auxiliary functions, **pi** and **gi** called the *propagate* and *generate* signals, the sum output respectively are defined as follows. pi = ai + bi gi = ai bi si = ai **xor** bi **xor** ci **ci+1** = gi + pici .

- As we increase the no of bits in the Carry Look Ahead adders, the complexity increases because the no. of gates in the expression **Ci+1** increases. So practically its not desirable to use the traditional CLA shown above because it increase the Space required and the power too.

- Instead we will use here Carry Look Ahead adder (less bits) in levels to create a larger CLA. Commonly smaller CLA may be taken as a 4-bit CLA. So we can define ***carry look ahead*** over a group of 4 bits. Hence now we redefine terms ***carry generate*** as [***Group Generated Carry***] **g[i,i+3]** and ***carry propagate*** as [***Group Propagated Carry***] p**[i,i+3]** which are defined below.

Redefined Equations:
For 4-bit adder, we can compute the carry for all the stages as shown below:
C1 = Go + Po.Co
C2 = G1 + P1.C1 = G 1 + P 1.GO + P 1.PO.CO
C3 = G2 + P2.C2 = G2 + P2.G1 + P2.P I.GO + P2.P1.PO.C O
C4 = G3 + P3.C3 = G3 + P3.G2 + P3.P2 .G 1 + P3.P2.P1.GO + P3.P2.Pl.PO.CO
Now the modified block diagram for the Carry Look ahead Adder (8-bit) using levels (of 4-bit CLA) will be as block diagram below.

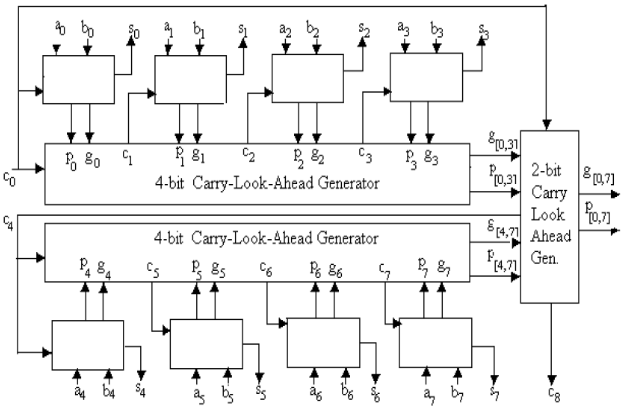

a0 b0 s0 a1 b1 s1 a2 b2 s2 a3 b3 s3
p0 g0 c1 p1 g1 c2 p2 g2 c3 p3 g3
4-bit Carry-Look-Ahead Generator
c0
c4
g[0,3]
p[0,3]
2-bit Carry Look Ahead Gen.
g[0,7]
p[0,7]
4-bit Carry-Look-Ahead Generator
g[4,7]
p[4,7]
p4 g4 c5 p5 g5 c6 p6 g6 c7 p7 g7
a4 b4 s4 a5 b5 s5 a6 b6 s6 a7 b7 s7
c8

Figure 6 8-BIt Carry Look Ahead Generator

2.5. Unsigned Multiplier

The multiplier is one of the hardware key blocks in Digital signal processing techniques. The multiplier involves generation of partial products and summation. The n-bit multiplier multiplies with the n-bit multiplier and it gives the final product term is a 2n bit value.

The two 32 bit unsigned multipliers multiplication output will be shown in below figure 7.

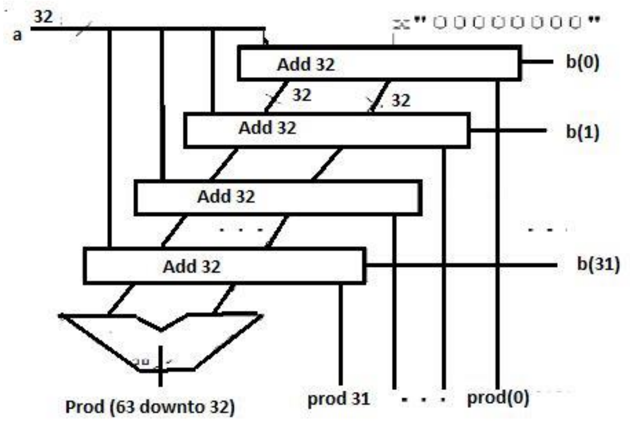


Figure 7 Unsigned Multiplier

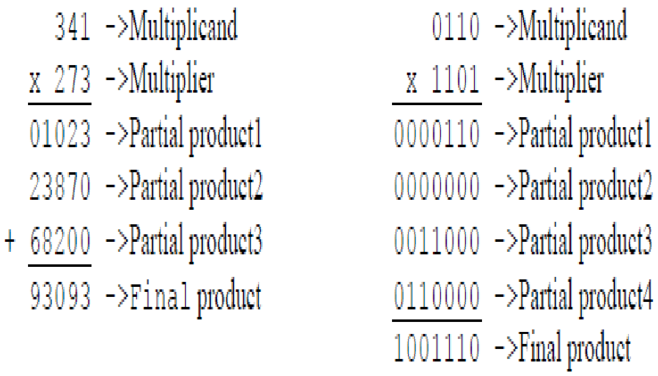
3. MULTIPLICATION ALGORITHM

A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It is built using binary adders. A variety of computer arithmetic techniques can be used to implement a digital multiplier. Most techniques involve computing a set of partial products, and then summing the partial products together. This process is similar to the method taught to primary schoolchildren for conducting long multiplication on base-10 integers, but has been modified here for application to a base-2 (binary) numeral system.

The first step in developing the multiplier is to understand the algorithm used for multiplication. The basic algorithm to be used is the same as the one we use to multiply decimal numbers.

We multiply each digit of the multiplier times the multiplicand to form a partial product and we add all of the partial products together to form the final product. This is illustrated in the figure .Ordinarily, we produce all of the partial products and then add all of them together; however, this is not necessary and, instead, as each partial product is produced, it can be added to the sum of previous ones. Since adders in computers usually can only add two numbers together, the addition of partial products must be done in the latter way.

Binary multiplication of positive numbers can be done using the same algorithm, but it is simpler since each partial product is either zero or equal to the multiplicand properly aligned with respect to the multiplicand. This is illustrated on the right side of Fig.6 for a 4-bit multiplicand and 4-bit multiplier.



```
    341 ->Multiplicand            0110 ->Multiplicand
  x 273 ->Multiplier            x 1101 ->Multiplier
  01023 ->Partial product1     0000110 ->Partial product1
  23870 ->Partial product2     0000000 ->Partial product2
+ 68200 ->Partial product3     0011000 ->Partial product3
  93093 ->Final product        0110000 ->Partial product4
                               1001110 ->Final product
```

Figure 8 array multiplier

This is much simpler than in the decimal system, as there is no table of multiplication to remember: just shifts and adds.

This method is mathematically correct and has the advantage that a small CPU may perform the multiplication by using the shift and add features of its arithmetic logic unit rather than a specialized circuit. The method is slow, however, as it involves many intermediate additions. These additions take a lot of time. Faster multipliers may be engineered in order to do fewer additions; a modern processor can multiply two 64-bit numbers with 6 additions (rather than 64), and can do several steps in parallel.

Let the product register size be 16 bits. Let the multiplicand registers size be 8 bits. Store the multiplier in the least significant half of the product register. Clear the most significant half of the product register. Repeat the following steps for 8 times:
1. If the least significant bit of the product register is "1" then add the multiplicand to the most significant half of the product register.
2. Shift the content of the product register one bit to the right (ignore the shifted-out bit.)
3. Shift-in the carry bit into the most significant bit of the product register. Figure 9.Shows a block diagram for such a multiplier

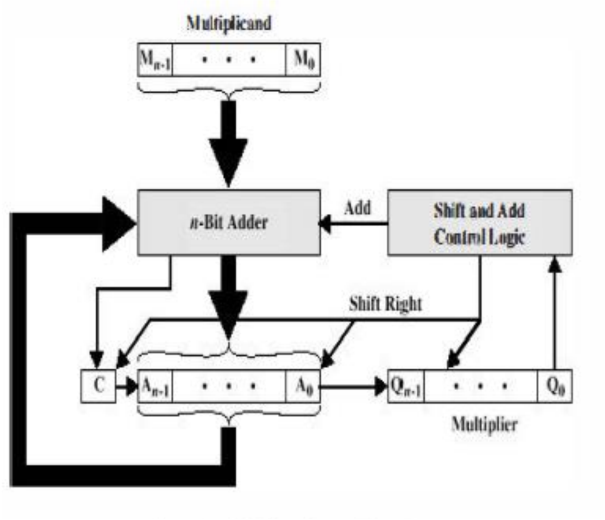
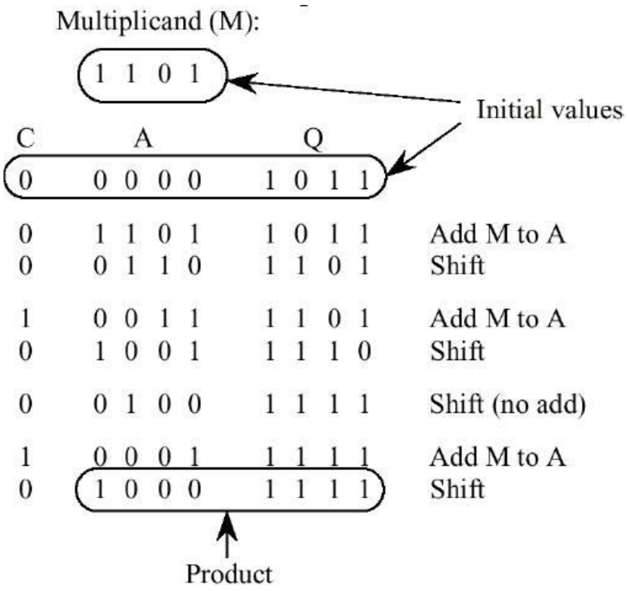


Figure 9 Multiplier of two n-bit values



C	A	Q	
0	0 0 0 0	1 0 1 1	
0	1 1 0 1	1 0 1 1	Add M to A
0	0 1 1 0	1 1 0 1	Shift
1	0 0 1 1	1 1 0 1	Add M to A
0	1 0 0 1	1 1 1 0	Shift
0	0 1 0 0	1 1 1 1	Shift (no add)
1	0 0 0 1	1 1 1 1	Add M to A
0	1 0 0 0	1 1 1 1	Shift

Figure 10 Multiplication example

4. CONCLUSION

A design and implementation of a VHDL-based 32-bit unsigned multiplier with CLAA and CSLA was presented. VHDL, a Very High Speed Integrated Circuit Hardware Description Language was used to model and simulate our multiplier. Using CLAA and CSLA improves the overall performance of the multiplier.

5. REFFERENCES

[1] W. –C. Yeh and C. –W. Jen, "High Speed Booth encoded Parallel Multiplier Design," IEEE transactions on computers, vol. 49, no. 7, pp. 692-701, July 2000.

[2] Shiann-Rong Kuang, Jiun-Ping Wang, and Cang-Yuan Guo, "Modified Booth multipliers with a Regular Partial Product Array," IEEE Transactions on circuits and systems-II, vol 56, No 5, May 2009.

[3] Li-Rong Wang, Shyh-Jye Jou and Chung-Len Lee, "A well-tructured Modified Booth Multiplier Design" 978-1-4244-1617-2/08/$25.00 ©2008 IEEE.

[4] Soojin Kim and Kyeongsoon Cho "Design of High-speed Modified Booth Multipliers Operating at GHz Ranges" World Academy of Science, Engineering and Technology 61 2010.

[5] Magnus Sjalander and Per Larson-Edefors. "The Case for HPM-Based Baugh-Wooley Multipliers," Chalmers University of Technology, Sweden, March 2008.

Authors Profile

K.Bhaskar M.Tech (P.hd pursuing) is working as Associate Professor i**n** Department of Electronic, Communication & Engineering; He had 14 years of teaching experience. His research interested in Digital Electronics and VLSI

Mrs.Athula Maleha Mirza B.Tech (ECE), M.Tech (DECS) is currently working as an Associate Professor, in Department of Electronic, Communication & Engineering, in Sarada Institute Of Technology and Science, Khammam, A.P, India. Her Research interests on Digital Electronics and VLSI. He had 7 years of Teaching Experience

Mr. Mangipudi Venu B.Tech (ECE), M.Tech (VLSI) is currently working as an Assosiate Professor, in Department of Electronic, Communication & Engineering, in Sarada Institute Of Technology and Science, Khammam, A.P, India. His Research interests on VLSI and wireless communication. He had 5 years of Teaching Experience.