IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-2, Issue-5 E-ISSN: 2347-2693

Design and Implementation of a Fast Unsigned 32-bit Multiplier Using Verilog HDL

D.Venu Gopal[1*] and M. Mohan Reddy[2]

[1*, 2] *Santhiram Engineering College, Nandyal*

www.ijcseonline.org

Received: 07/04/2014 Revised: 22/04/ 2014 Accepted: 16/05/May 2014 Published: 31/05/2014

Abstract: This project deals with the comparison of the VLSI design of the carry look-ahead adder (CLAA) based 32-bit unsigned integer multiplier and the VLSI design of the carry select adder (CSLA) based 32-bit unsigned integer multiplier. Both the VLSI design of multiplier multiplies two 32-bit unsigned integer values and gives a product term of 64-bit values. The CLAA based multiplier uses the delay time of 99ns for performing multiplication operation whereas in CSLA based multiplier also uses nearly the same delay time for multiplication operation the overall simulation can be observed by using model sim and the synthesis report can be given by using Xilinx ise.

***Keywords*-**CLAA; CSLA; Delay; Area; Array Multiplier; HDL Modeling & Simulation

I. INTRODUCTION

The multiplier is a vital module of a computer system and can be considered one of the fundamental arithmetic functions. However, multiplication is not as a simple operation as addition or subtraction, because it takes more time to perform two subtasks, addition and shifting. Typically, a multiplication operation takes between2 and 8 cycles [1]. Therefore, using high-speed multipliers is a critical requirement for high performance processors. Multipliers use th e addition operation for all the partial products. The adder can be a ripple-carry adder, a carry look-ahead adder, or any other adder [2, 3]. However, using a fast adder to complete the Multiplication operation improves the overall performance of the computer system. Our study is focused on multipliers using unsigned data. VHDL, a Very high speed integrated circuit Hardware Description Language, was used to model and simulate the multiplier design. Several researchers had addressed the adder performance issues and others did the same with regard to the multiplier performance. Sertbas and Özbey worked on the performance analysis of classified binary adder architectures. They compared the ripple adder, carry-look-ahead adder, carry select adder and the conditional sum adder. They used VHDL implementation for their designs and comparison studies.

II. CARRY LOOK-AHEAD ADDER

To reduce the delay caused by the effect of carry propagation in the ripple carry adder, we attempt to evaluate the carry-out for each stage (same as carry-in to next stage) concurrently with the computation of the sum bit [5, 6]. The two Boolean functions for the sum and carry are as follows [7, 8]:

$$SUM = A_i \oplus B_i \oplus C_i$$
$$C_{out} = C_{i+1} = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_i$$

$$C_{i+1} = G_i + P_i \cdot C_i$$

$$C_1 = G_0 + P_0 \cdot C_0$$
$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$
$$C_3 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$$
$$C_4 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

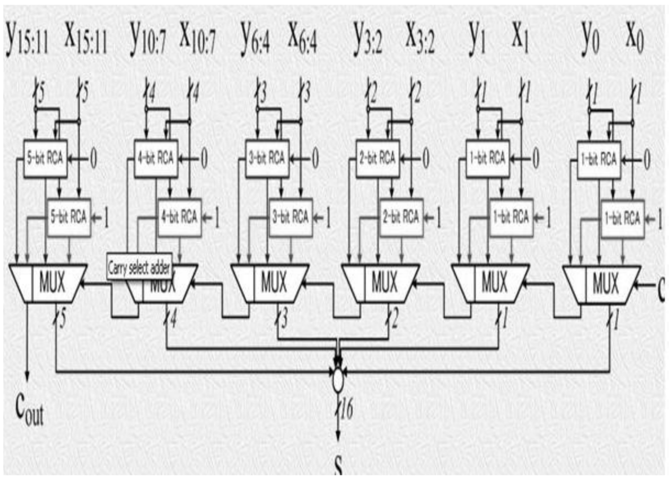
In general, we can write:
The sum function:

$$SUM_i = A_i \oplus B_i \oplus C_i = P_i \oplus C_i$$

The carry function

$$C_{i+1} = \sum_{j=0}^{i} (G_j \prod_{k=j+1}^{i} P_k) + \prod_{k=0}^{i} P_k Cin$$

III. CARRY SELECT ADDER



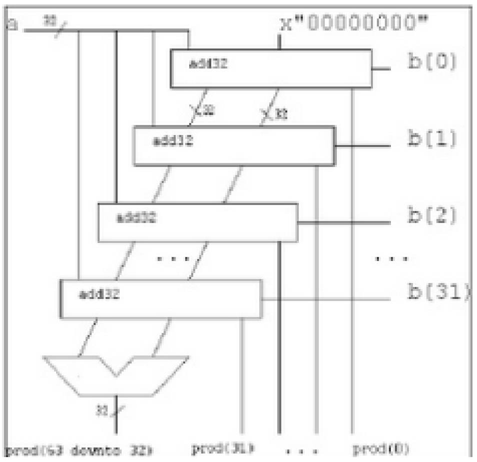
The carry select adder comes in the category of conditional sum adder. Conditional sum adder works on some condition. Sum and carry are calculated by assuming input carry as 1 and 0 prior the input carry comes. When actual carry input arrives, the actual calculated values of sum and carry are selected using a multiplexer. The conventional carry select adder consists of k/2 bit adder for the lower half of the bits i.e. least significant bits and for the upper half i.e. most significant bits (MSB's) two k/ bit adders. In MSB adders' one adder assumes carry input as one for performing addition and another assumes carry input as zero. The carry out

Corresponding Author: *D.Venu Gopal*
Santhiram Engineering College, Nandyal

calculated from the last stage i.e. least significant bit stage is used to select the actual calculated values of output carry and sum. The selection is done by using a multiplexer. This technique of dividing adder in to stages increases the area utilization but addition operation fastens. It is composed of two four-bit ripple carry adders per section. Both sum and carry bits are calculated for the two alternatives of the input carry, ―0‖ and ―1‖.The carry out of each section determines the carry in of the next section, which then selects the appropriate ripple carry adder. The very first section has a carry in of zero. Time delay: time to compute first section + time to select sum from subsequent sections.

1V. MULTIPLIER USING ADDER

Multiplication involves the generation of partial products, one for each digit in the multiplier, as inFig.1. These partial products are then summed to produce the final product. The multiplication of two n-bit binary integers results in a product of up to 2n bits in length [2].



We used the following algorithm to implement the multiplication operation for unsigned data:

V. MULTIPLICATION ALGORITHM

Let the product register size be 64 bits. Let the multiplicand registers size be 32 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 32 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.
3Shift-in the carry bit into the most significant bit of the product register

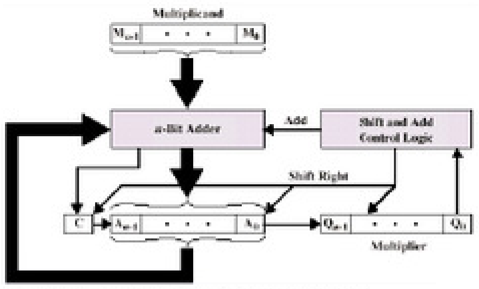


Figure 2. Multiplier of two n-bit values.

VI. SIMULATION RESULTS.

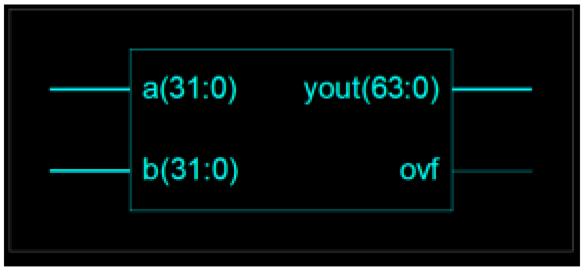


FIG: RTL SCHEMATIC OF CLAA

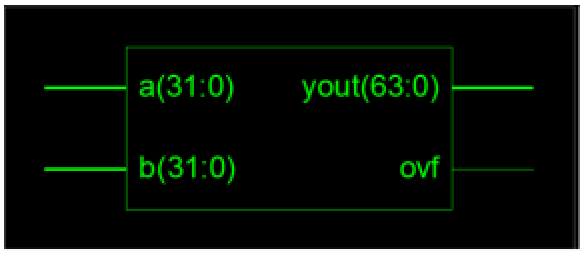


FIG:RTL SCHEMATIC OF CSLA

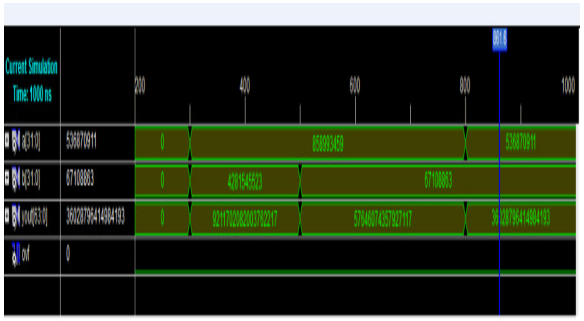


FIG: TIMING DIAGRAM OF CLAA

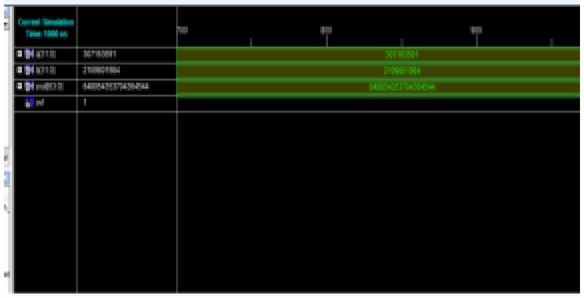


FIG: TIMING DIAGRAM OF CSLA

VII. CONCLUSION

A design and implementation of a HDL-based32-bit unsigned fast multiplier was presented. VHDL, a Very High Speed Integrated Circuit Hardware Description Language, was used to mode land simulate our multiplier. Using a fast adder improves the overall performance of the multiplier.

REFERENCES

[1]. P. Asadi and K. Navi, "A novel highs-speed 54-54 bit multiplier*", Am. J. Applied Sci., vol.4 (9), pp. 666-672.2007.* http://www.scipub.org/fulltext/ajas/ajas49666-672.pdf.

[2]. W. Stallings, *Computer Organization and Architecture Designing for Performance*, 7th ed., Prentice Hall, Pearson Education International, USA, 2006, ISBN: 0-13-185644- 8.

[3]. J. F. Wakerly, *Digital Design-Principles and Practices*, 4th ed., Pearson Prentice Hall, USA, 2006. ISBN: 0131733494.

[4]. A. Sertbas and R.S. Özbey, "A performance analysis of classified binary adder architectures and the VHDL simulations", *J. Elect. Electron. Eng*., Istanbul, Turkey, vol. 4, pp. 1025-1030, 2004. http://www.istanbul.edu.tr/eng/ee/jeee/main/pages/issues/is41/41005.pdf .

[5]. P. S. Mohanty, "Design and Implementation of Faster and Low Power Multipliers", *Bachelor Thesis*. National Institute of Technology, Rourkela, 2009. http://ethesis.nitrkl.ac.in/213/1/10509019_final.pdf.pdf.

[6]. S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with VHDL Design*, 2nd ed., McGraw-Hill Higher Education, USA, 2005. ISBN: 0072499389.