Design and Implementation of Low Power Multiplier using VLSI Techniques

1Preeti B, 2S S Kerur
3Mr. K Zacharaiah Joseph, 4Mr. Avinash Rathod,
5Mr. Shrekar Vaidhya, 6Mr. Vasu K
1, 2Assistant Professor, 3, 4, 5, 6UG Student
Department of Electronics & Communication Engineering,
SDM College of Engineering & Technology,
Dharwad
Email ID: prtbellerimath69@gmail.com,
kerurss@gmail.com,
jsunnyjoseph@gmail.com,
avinashrathod6661@gmail.com,
vaidhyashreekar@gmail.com,
vasudeva6868@gmail.com

June 11, 2018

Abstract

A multiplier is one of the most important building block that is widely used in processor, embedded, VLSI applications, Application specific integrated circuits and most of the DSP applications. The three main thrust parameters of any VLSI design lies in speed, area and power. Low power is an emerging trend which intern can maximize the lifespan of battery operating time. In this paper an attempt is made to balance and optimize the performance of Wallace multiplier which consumes less power. The two main sources of power consumption are static power dissipation and dynamic power dissipation. The multiplier has been designed and simulated using various VLSI techniques. The
main objective of the paper is to design and implement a low power multiplier used for various VLSI applications. The work includes designing of basic gates, half adder and full adder with operating voltage 1.8V. To design and implement multiplier, standard gpdk180 technology library is used. The multiplier block is implemented using Cadence tool 180nm CMOS technology. The logic styles used in our proposed design of the multiplier are CMOS (Complementary Metal Oxide Semiconductor Logic) and MCML (Mos Current Mode Logic). The power analysis has been carried out and measured on both the logic styles.

1 Introduction

A Multiplier is one of the most important blocks in any processor now days. A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. A variety of computer arithmetic techniques can be used to realize a digital multiplier. Most techniques involve computing a set of fractional products, and then summing the fractional products together. This process conducting long multiplication on base-10 integers, but have been modified here for Application to a number of base two systems. As more transistors per chip became available due to larger-scale integration, it became possible to put enough adders on a single Chip to sum all the fractional products at once, fairly than use again a single adder to handle each partial product one at a time. As the common digital signal processing algorithms spend most of their time multiplying, the processors spend a lot of chip area in order to make the multiplication as fast as possible. Hence a non conventional however very efficient Vedic mathematics is used for making a High performance multiplier. Vedic Mathematics deals mainly with various Vedic mathematical formulae and their applications for carrying out large arithmetical operations easily. To evaluate the performance of the new multiplier, the multiplier is compared with the already existing digital multipliers on various parameters as power consumption and speed of operation.

[1] Types of multiplier

A typical processor central processing unit devotes a consid-
erable amount of processing time in performing arithmetic operations, particularly multiplication operations. Multiplication is one of the basic arithmetic operations and it requires substantially more hardware resources and processing time than addition and subtraction. MULTIPLIERS can be designed by using any of the following methods. The main objective is to design low power consumption, consuming less area and a high speed multiplier.

- Booth multiplier.
- Combinational multiplier.
- Wallace tree multiplier.
- Array multiplier.
- sequential multiplier

We prefer Wallace tree multiplier because it provides the only efficient multiplier which has substantial hardware savings, higher speeds, less propagation delay, has reduced schematic layout and occupies less area. Hence Wallace is very often preferable for multipliers.

2 Theoretical discussion on multiplier

![Fig.1 Block diagram of a multiplier](image_url)  

In any signal processing system, multiplier is an important and a basic building block element. The performance of these types of processing systems depends on the performance of the inbuilt multiplier. So it is a challenging task for any designer to design a high performance multiplier. There are different factors that drive for high performance electronic system design in terms of low power dissipation and high speed. The block diagram of a multiplier is shown in Fig. A basic multiplier consists of three stages: generation of partial product, addition of partial product and final addition.
A multiplier based on Wallace-tree structure is called Wallace multiplier. It is substantially faster than other multiplier architecture. The operation of a Wallace multiplier is carried out in three different steps. After generating the partial product, these are accumulated in different stages. [2]

Basic Building Blocks of Multiplier

![Flow chart](image.png)

Function and Algorithm of the Modules[3]

a) 2’s Complement Generator:

- Function: The 2’s Complement Generator Takes the multiplicand MD and MR as its input and produces MD and - MR as its output in case of negative numbers.

- Algorithm: 2’s complement is generated by inverting all bits of the multiplicand and then adding 1 using a ripple carry adder.

b) Partial Product Generator:

- Function: The Partial Product Generator generates appropriate partial products to be added with a Wallace tree.

- Algorithm: The Partial Product Generator uses the table for each multiplier bit. Depending on the value of MD or -MD or MR or -MR is assigned to partial product. 4 bit is then extended till the 7th bit for appropriate sign extension.

c) Carry Look-ahead Adder:

- Function: Carry Look-ahead Adder (CLA) add two numbers with very lower latency.
• Algorithm: By extend the Cin with the corresponding inputs, the carry and sum are independent of the previous bits.

Wallace multiplication
Function: The Wallace tree module adds the 4 partial products and generates final two intermediate operands for final addition

Fig. 3 Algorithm [2]

3 Cadence Tool

The intention for this manual is to serve as an introduction to the Cadence design environment and describe the methodology used when designing integrated circuits. The Cadence tool kit consists of several programs for different applications such as schematic drawing, layout, verification, and simulation. These applications can be used on various computer platforms. The open architecture also allows for integration of tools from other vendors or of own design. The integration of all these tools is done by a program called Design Framework II (DFW). When a command is performed, from a form or a menu, the system is executing functions written in the SKILL language. SKILL is developed by Cadence and is based on Lisp. The Cadence tools are using SKILL for internal communication and for the tool-design communication. SKILL is also accessible for the designers. Commands can be written in the CIW window or placed in command files for execution. It can be used for simple tasks like...
executing a command or building more complex functions to perform various tasks. The step Create the Design consists of drawing schematic views of all cells and blocks. The schematic view contains transistor symbols, and maybe other components such as resistors and capacitances, and wires connecting them. From the schematic view the symbol view is created (almost automatically) so that the cell can be used on a higher level in the hierarchy. [4]

Fig.4 Design flow

Introduction to CMOS Logic

Complementary metal-oxidesemiconductor, abbreviated as CMOS, is a technology for constructing integrated circuits. CMOS technology is used in microprocessors, microcontrollers, static RAM, and other digital logic circuits. CMOS technology is also used for several analog circuits such as image sensors (CMOS sensor), data converters, and highly integrated transceivers for many types of communication. In 1963, while working for Fairchild Semiconductor, Frank Wanlass patented CMOS (US patent 3,356,858).

CMOS is also sometimes referred to as complementary-symmetry metal-oxidesemiconductor (or COS-MOS). The words "complementary-symmetry" refer to the fact that the typical design style with CMOS uses complementary and symmetrical pairs of p-type and n-type
metal oxide semiconductor field effect transistors (MOSFETs) for logic functions

Two important characteristics of CMOS devices are high noise immunity and low static power consumption. Since one transistor of the pair is always off, the series combination draws significant power only momentarily during switching between on and off states. Consequently, CMOS devices do not produce as much waste heat as other forms of logic, for example transistor-transistor logic (TTL) or NMOS logic, which normally have some standing current even when not changing state. CMOS also allows a high density of logic functions on a chip. It was primarily for this reason that CMOS became the most used technology to be implemented in VLSI chips.

The phrase "metal-oxide-semiconductor" is a reference to the physical structure of certain field-effect transistors, having a metal gate electrode placed on top of an oxide insulator, which in turn is on top of a semiconductor material. Aluminum was once used but now the material is polysilicon. Other metal gates have made a comeback with the advent of high-k dielectric materials in the CMOS process, as announced by IBM and Intel for the 45 nanometer node and beyond. [5] CMOS (complementary metal oxide semiconductor) logic has a few desirable advantages:

- High input impedance. The input signal is driving electrodes with a layer of insulation (the metal oxide) between them and what they are controlling. This gives them a small amount of capacitance, but virtually infinite resistance. The current into or out of CMOS input held at one level is just leakage, usually 1 nano Ampere or less.

- The outputs actively drive both ways.

- CMOS logic takes very little power when held in a fixed state. The current consumption comes from switching as those capacitors are charged and discharged. Even then, it has good speed to power ratio compared to other logic types.

- CMOS gates are very simple. The basic gate is an inverter, which is only two transistors. This together with the low power consumption means it lends itself well to dense inte-
gration. Or conversely, you get a lot of logic for the size, cost and power.

Implementation of wallace multiplier using CMOS Logic
Design and implementation of basic gates
NAND gate:-

As shown in Fig.6 the NAND gate operates as an AND gate followed by a NOT gate. It acts in the manner of the logical operation "and" followed by negation. The output is "false" if both inputs are "true." Otherwise, the output is "true."

NOT gate:-

A logical inverter in Fig 7 sometimes called a NOT gate to differentiate it from other types of electronic inverter devices, has only one input. It reverses the logic state.

Fig.6 Implementation of NAND gate

Fig.7 Implementation of NOT gate
XOR GATE:-

The XOR (exclusive-OR) gate acts in the same way as the logical "either/or." The output is "true" if either, but not both, of the inputs are "true." The output is "false" if both inputs are "false" or if both inputs are "true." Another way of looking at this circuit is to observe that the output is 1 if the inputs are different, but 0 if the inputs are the same.
The half adder adds two single binary digits A and B. It has two outputs, sum (S) and carry (C). The carry signal represents an overflow into the next digit of a multi-digit addition. The value of the sum in decimal system is $2C + S$. The simplest half-adder design, pictured on the right, incorporates an XOR gate for S and an AND gate for C. The Boolean logic for the sum (in this case S) will be $A'B + AB'$ whereas for carry (C) will be $AB$. With the addition of an OR gate to combine their carry outputs, two half adders can be combined to make a full adder. The half adder adds two input bits and generates a carry and sum, which are the two outputs of a half adder.

Design and implementation of full adder

A full adder circuit can be implement with the help of two half adder circuits. The first will half adder will be used to add A and B to produce a partial Sum. The second half adder logic can be used to add CIN to the Sum produced by the first half adder to get the final S output. If any of the half adder logic produces a carry, there will be an output carry. Thus, COUT will be an OR function of the half-adder Carry outputs.

Fig.11 Implementation of FULL ADDER

Symbol

Fig.12 Symbol of FULL ADDER
Implementation of Wallace multiplier

Wallace multiplier is designed using nand gates, half adder and full adder. Nand gates are used to generate partial products and adders are used to calculate the sum of the partial products to generate the final results.

Introduction to MCML Logic:

The MCML gate consists of four main blocks, i.e., the logic function block, current source ICs, power switch, and the load resistors RL. Differential pair of NMOS transistors is used to implement the logic function. Depending on the complexity of the function levels, the NMOS transistors have to be stacked one upon the other to implement the logic function. The constant tail current $I_{ss}$ is provided by using a current source. This current will be switched based on the logic function to one of the output branch, which finally reach voltage level ($V_{dd}-I_{ss}R_{d}$), which corresponds to logic 0 due to the entire current flowing through the load resistor. The other output will stay at logic 1. The operation is elaborate in the next section. During sleep mode, the power switch is used to cut the current, which will force both the outputs to logic 1 since there will be no current in the output branch. In this MCML circuit, the design parameters include the voltage gain, total power dissipation, circuit delay and voltage swing. These parameters can be controlled by the variables such as bias current, current source transistor size, differential pull-down network transistor sizes, and the current source bias voltage. [6]
Advantages of MCML Logic:

- High speed since it does not operate in the cut-off region.
- Low switching noise because of the constant current source.
- High noise immunity because of its differential nature.
- Capability to operate at low supply voltage.
- Efficient implementation of arithmetic circuits (xor styles).
- Controllable output voltage swing.
- Small output swing efficiency in driving data buses, and high load signals, which reduces dynamic power dissipation and delay.

Implementation of multiplier using MCML Logic
Design and implementation of basic gates
Not gate:
Fig. 15 Implementation of NOT gate

Nor Gate:

Fig. 16 Implementation of NOR gate

Nand Gate:

Fig. 17 Implementation of NAND gate
Design and implementation of full adder

Sum:

Fig. 18 Implementation of FULL ADDER (SUM)

Carry:

Fig. 19 Implementation of FULL ADDER (CARRY)

Full adder Circuit:
Fig. 20 Implementation of FULL ADDER

Implementation of Wallace multiplier

Fig. 21 Implementation of Wallace multiplier

4 RESULTS

Schematic of multiplier

Fig. 22 Schematic of multiplier

Simulation result
5 CONCLUSION

Thus, the schematic is designed for a four-bit multiplier for which we provide two four-bit data input lines and obtain the product of 8-bit. The designed multiplier is tested for various data and functionally verified. The multiplier is implemented using both logic styles (CMOS, MCML). The proposed work is operated with a supply voltage of 1.8V. The Wallace tree multiplier is designed and implemented using gpdk180 standard library file. The schematic is designed using virtuoso editor. Cadence tool is used for the implementation of the multiplier design. The schematic is designed for a 4-bit multiplier. The complete schematic of a 4-bit multiplier is functionally verified and implemented using Cadence tool. The virtuoso tool Version 6
is used for schematic. The proposed design of the multiplier consumes 60.94mW for CMOS technology and 151.24uW for MCML technology. By comparative study one can conclude that MCML style consumes less power than CMOS style. A low power and efficient multiplier is designed and implemented for various VLSI applications.

References


[7] Neha Maheshwari, A Design of 4X4 Multiplier using 0.18 um Technology, International Journal of Latest Trends in Engi-

