

AREA EFFICIENT ALU DESIGN FOR LOW LATENCY APPLICATIONS USING RADIX-8 MODIFIED BOOTH ENCODING

1. T.SATYA, 2.K.YAMINI

1. PG Scholar, Dept. of ECE, Eluru College of Engineering and Technology, Eluru, A.P.
2. Assistant Professor, Dept. of ECE, Eluru College of Engineering and Technology, Eluru, A.P.

ABSTRACT:

The main concept of this project is to design an efficient arithmetic logic unit. The reason behind choosing this topic as a research work is that, ALU is the key element of digital processors like as microprocessors, microcontrollers, central processing unit etc. Every digital domain based technology depends upon the operations performed by ALU either partially or whole. That's why it highly required designing high speed ALU, which can enhance the efficiency of those modules which lies upon the operations performed by ALU. The performance of ALU greatly depends upon the design of multiplier. There are so many multiplication algorithms exist now-a-days at algorithmic and structural level. Vedic multiplication technique is one of the best algorithms in terms of speed. Further, this project is enhanced by using Radix-8 modified booth encoding algorithm to reduce area and time.

KEYTERMS: Arithmetic logic unit, Vedic Sutras, Urdhva Triyakambakam, Density, Micro controllers, Radix, Modified booth encoding.

INTRODUCTION: Among the forceful investigation in the field of low power, high speed digital applications due to the growing demand of systems like phones, laptop, palmtop computers, cellular phones, wireless modems and portable multimedia applications etc has directed the VLSI technology to scale down to nano-regimes, allowing additional functionality to be incorporated on a single chip. The designer's novel purpose in the field of multifaceted digital circuit design is minimization of power consumption. These investigations are responsible for special design techniques for digital circuits distant from conventional CMOS design style. A large body of investigate has been performed to expand and advance conventional Complementary Metal Oxide Semiconductor (CMOS) techniques for the fabrication of ULTRA low power integrated circuits (ICs). The purpose of this study is to expand a faster, lower power, and reduced area substitute to standard CMOS logic circuits. Methods of multiplication have been documented in the Egyptian, Greek, Babylonian, Indus Valley and Chinese civilizations.[1] In early days of Computers, multiplication was implemented generally with a sequence of addition, subtraction and shift operations. There exist many algorithms proposed in literature to perform multiplication, each offering different advantages and having trade off in terms of delay, circuit complexity, area occupied on chip and power consumption. For multiplication algorithms performing in DSP applications, latency and throughput are two major concerns from delay perspective. Latency is the real delay of computing a function. Simply it's a measure of how long the

inputs to a device are stable is the final result available on outputs. Throughput is the measure of how many multiplications can be performed in a given period of time. Multiplier is not only a high delay block but also a major source of power dissipation. So, if one aims to minimize power consumption, it is of great interest to reduce the delay by using various optimization methods. Our ALU architecture consists of a set of fast and slow functional units. There are many advantages and plus points to the design of our ALU. Not only does it consume minimal power during runtime, it does not require real time process to monitor performance. Neither is a hardware circuit needed to tune the supply voltage. Compared with other models operating on the supply voltage reduction principle, the ALU we have designed is far simpler. Low power and High speed are the design trade-offs in VLSI industry. Power consumption, area, speed, noise immunity has emerged as a primary design constraints for integrated circuits (ICs). The VLSI designers always targets on three basic design goals such as minimizing the transistor count, minimizing the power consumption and increasing the speed. Most of the Very Large Scale IC (VLSI) applications, Full adder circuit is functional building block and most critical component of complex arithmetic circuits like microprocessors, digital signal processors or any ALUs. Almost every complex computational circuit requires full adder circuitry. The entire computational block power consumption can be reduced by implementing low power techniques on full adder circuitry. In this paper, from different existed base papers several full adder circuits based on different low power techniques

have been proposed targeting Static CMOS gates are very power efficient because they dissipate nearly zero power when idle. Earlier, the power consumption of CMOS devices was not the major concern while designing chips. Factors like speed and area dominated the design parameters. As the CMOS technology moved below sub-micron levels the power consumption per unit area of the chip has risen tremendously. In the era of growing technology and scaling of devices up to nanometer regime, the arithmetic logic circuits are to be designed with compact size, less power and propagation delay. Arithmetic operations are indispensable and basic functions for any high speed low power application digital signal processing, microprocessors, image processing etc. Addition is most important part of the arithmetic unit rather approximately all other arithmetic operation includes addition. Thus, the primary issue in the design of any arithmetic logic unit is to have low power high performance adder cell. There are various topologies and Methodologies proposed to design full adder cell efficiently. With rapid development of portable digital applications, the demand for increasing speed, compact implementation, and low power dissipation triggers numerous research efforts [1]–[3]. The wish to improve the performance of logic circuits, once based on traditional CMOS technology, resulted in the development of many logic design techniques during the last two decades [17]. One form of logic that is popular in low-power digital circuits is pass-transistor logic (PTL).

LITERATURE SURVEY:

There are different types and designs of full adder which is discussed in various papers at state of the art level and process and circuit level. Twelve state of the art full adder cells are: conventional CMOS, CPL, TFA, TG CMOS, C2MOS, Hybrid, Bridge, FA24T, N-Cell, DPL and Mod2f. R. Shalem, E. John, and L.K. John, proposed a conventional CMOS full adder consisting of 28 transistors [1]. Later, the number of transistor count is reduced to have less area and power consumption. A. Sharma, R Singh and R. Mehra, Member, IEEE, have improved performance with Transmission Gate Full adder using CMOS nano technology where 24 transistors are used [2]. The Complementary Pass transistor Logic (CPL) full Adder contains the 18 transistors. The power consumption of this structure is $2.5\mu\text{W}$ [3]. A Transmission Function Full Adder (TFA) based on the transmission function theory has 16 transistors. The power consumption of this structure is $12\mu\text{W}$. N-CELL contains the 14 transistors and utilizes the low power XOR/XNOR circuit. The power consumption of this structure is $1.62\mu\text{W}$. Mod2f Full Adder contains the 14 transistors, generates full swing XOR and XNOR signals by utilizing a pass transistor based DCVS circuit. The power consumption of this structure is $2.23\mu\text{W}$ [3]. Saradindu Panda, N. Mohan Kumar, C.K. Sarkar, optimized the full adder circuit to 18

Transistor using Dual Threshold Node Design with Submicron Channel Length [4]. T. Vigneswaran, B. Mukundhan, and P. Subbarami Reddy, designed 14 transistor high speed CMOS full adder and significantly improved threshold problem to 50% [5]. Gate Diffusion Input Technique is a new method of reducing power dissipation, propagation delay with less area. T. Esther Rani, M. Asha Rani, Dr. Rameshwar Rao, designed an area optimized low power arithmetic and logic unit in which Arithmetic Logic Unit is implemented using logic gates, pass transistor logic, as well as GDI technique [6]. Manish Kumar, Md. Anwar Hussain, and L.L.K. Singh explained a Low Power High Speed ALU in 45nm Using GDI Technique and Its Performance Comparison [7]. We have designed ALU in different way by using GDI cells to implement multiplexers and full adder circuit. The input and output sections consist of 4x1 and 2x1 multiplexers and ALU is implemented by using full adder. A. Morgenshtein described new design GDI cell that allows reducing delay, area and power dissipation [2], [3].

ALU DESIGN:

Now days we are living in digital world, where all operations get performed with more reliably and highest accuracy by digital signal processor. The key element of all the processors like Microprocessor, Microcontroller, DSP processor etc is ALU. Every digital domain based technology depends upon the operations performed by ALU either partially or whole. Speed is the most prominent factor of processor and controllers being used recently. [7], [8] describe the Vedic mathematics from beginning and discuss all operations. By improving the ALU unit we can develop efficient the Digital Signal Processor, for that proposed Arithmetic unit appears very useful [6]. One of the major purposes of Vedic mathematics is to execute the difficult calculations in simple way, even manageable orally without much use of pen and paper.

EXISTING TECHNIQUE:

VEDIC MULTIPLIER ARCHITECTURE :

The hardware architecture of 2X2, 4x4 and 8x8 bit Vedic multiplier module are displayed in the below sections. Here, “Urdhva-Tiryagbhyam” (*Vertically and Crosswise*) sutra is used to propose such architecture for the multiplication of two binary numbers. The beauty of Vedic multiplier is that here partial product generation and additions are done concurrently. Hence, it is well adapted to parallel processing. The feature makes it more attractive for binary multiplications. This in turn reduces delay, which is the primary motivation behind this work

VEDIC MULTIPLIER FOR 2X2 BIT MODULE:

The method is explained below for two, 2 bit numbers A and B where $A = a_1a_0$ and $B = b_1b_0$ as

shown in Fig. 2. Firstly, the least significant bits are multiplied which gives the least significant bit of the final product (vertical). Then, the LSB of the multiplicand is multiplied with the next higher bit of the multiplier and added with, the product of LSB of multiplier and next higher bit of the multiplicand (crosswise). The sum gives second bit of the final product and the carry is added with the partial product obtained by multiplying the most significant bits to give the sum and carry. The sum is the third corresponding bit and carry becomes the fourth bit of the final product. The 2X2 Vedic multiplier module is implemented using four input AND gates & two half-adders which is displayed in its block diagram in Fig. 3. It is found that the hardware architecture of 2x2 bit Vedic multiplier is same as the hardware architecture of 2x2 bit conventional Array Multiplier [2]. Hence it is concluded that multiplication of 2 bit binary numbers by Vedic method does not made significant effect in improvement of the multiplier's efficiency. Very precisely we can state that the total delay is only 2-half adder delays, after final bit products are generated, which is very similar to Array multiplier. So we switch over to the implementation of 4x4 bit Vedic multiplier which uses the 2x2 bit multiplier as a basic building block. The same method can be extended for input bits 4 & 8. But for higher no. of bits in input, little modification is required.

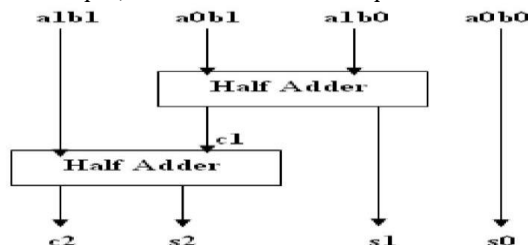


Fig. 3 Block Diagram of 2x2 bit Vedic Multiplier

The 4x4 bit Vedic multiplier module is implemented using four 2x2 bit Vedic multiplier modules as discussed in Fig. 3. Let's analyze 4x4 multiplications, say $A = A_3 A_2 A_1 A_0$ and $B = B_3 B_2 B_1 B_0$. The output line for the multiplication result is - $S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0$. Let's divide A and B into two parts, say $A_3 A_2$ & $A_1 A_0$ for A and $B_3 B_2$ & $B_1 B_0$ for B. Using the fundamental of Vedic multiplication, taking two bit at a time and using 2 bit multiplier block, we can have the following structure for multiplication as shown in Fig. 4.

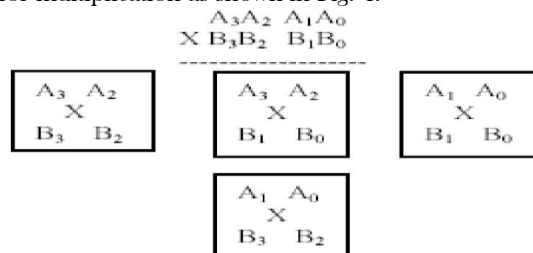


Fig. 4 Sample Presentation for 4x4 bit Vedic Multiplication

Each block as shown above is 2x2 bit Vedic multiplier. First 2x2 bit multiplier inputs are $A_1 A_0$ and $B_1 B_0$. The last block is 2x2 bit multiplier with inputs $A_3 A_2$ and $B_3 B_2$. The middle one shows two 2x2 bit multiplier with inputs $A_3 A_2$ & $B_1 B_0$ and $A_1 A_0$ & $B_3 B_2$. So the final result of multiplication, which is of 8 bit, $S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0$. To understand the concept, the Block diagram of 4x4 bit Vedic multiplier is shown in Fig. 5. To get final product ($S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0$), four 2x2 bit Vedic multiplier (Fig. 3) and three 4-bit Ripple-Carry (RC) Adders are required. The proposed Vedic multiplier can be used to reduce delay. Early literature speaks about Vedic multipliers based on array multiplier structures. On the other hand, we proposed a new architecture, which is efficient in terms of speed. The arrangements of RC Adders shown in Fig. 5, helps us to reduce delay. Interestingly, 8x8 Vedic multiplier modules are implemented easily by using four 4x4 multiplier modules

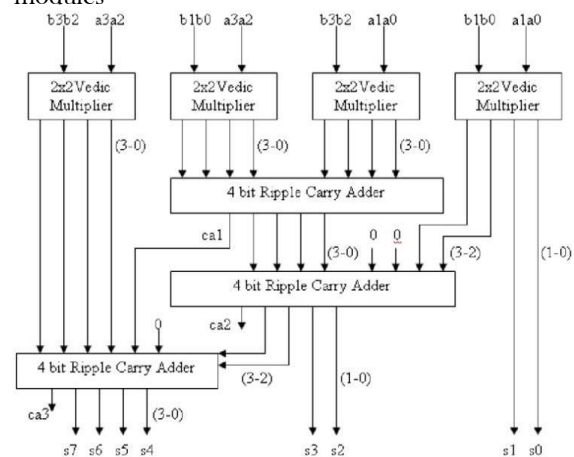


Fig. 5 Block Diagram of 4x4 bit Vedic Multiplier

PROPOSED TECHNIQUE:

RADIX-8 MODIFIED BOOTH ALGORITHM:

The Booth algorithm consists of repeatedly adding one of two predetermined values to a product P and then performing an arithmetic shift to the right on P.

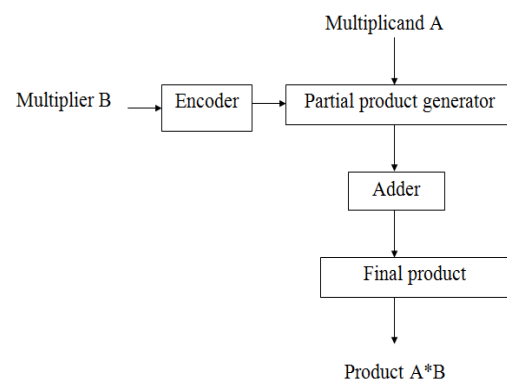


Fig.6 Booth algorithm

The multiplier architecture consists of two architectures, i.e., Modified Booth. By the study of different multiplier architectures, we find that Modified Booth increases the speed because it reduces the partial products by half. Also, the delay in the multiplier can be reduced by using Wallace tree. The energy consumption of the Wallace Tree multiplier is also lower than the Booth and the array. The characteristics of the two multipliers can be combined to produce a high-speed and low-power multiplier. The modified stand-alone multiplier consists of a modified recorder (MBR). MBR has two parts, i.e., Booth Encoder (BE) and Booth Selector (BS). The operation of BE is to decode the multiplier signal, and the output is used by BS to produce the partial product. Then, the partial products are added to the Wallace tree adders, similar to the carry-save-adder approach. The last transfer and sum output line are added by a carry look-ahead adder, the carry being stretched to the left by positioning.

Table . 4. Quartet coded signed-digit table

Quartet value	Signed-digit value
0000	0
0001	+1
0010	+1
0011	+2
0100	+2
0101	+3
0110	+3
0111	+4
1000	-4
1001	-3
1010	-3
1011	-2
1100	-2
1101	-1
1110	-1
1111	0

Here we have a multiplication multiplier, 3Y, which is not immediately available. To Generate it, we must run the previous addition operation: $2Y + Y = 3Y$. But we are designing a multiplier for specific purposes and then the multiplier belongs to a set of previously known numbers stored in a memory chip. We have tried to take advantage of this fact, to relieve the radix-8 bottleneck, that is, 3Y generation. In this way, we try to obtain a better overall multiplication time or at least comparable to the time, we can obtain using a radix-4 architecture (with the added benefit of using fewer transistors). To generate 3Y with 21-bit words you just have to add $2Y + Y$, ie add the number with the same number moved to a left position. A product formed by multiplying it with a multiplier digit when the multiplier has many digits. Partial products are calculated as intermediate steps in the calculation of larger products.

The partial product generator is designed to produce the product multiplying by multiplying A by 0, 1, -1, 2, -2, -3, -4, 3, 4. Multiply by zero implies that the product is "0 ". Multiply by " 1 " means that the product remains the same as the multiplier. Multiply by "-1" means that the product is the complementary form of the number of two.

Multiplying with "-2" is to move left one as this rest as per table.

4 SIGN EXTENSION CORRECTOR:

The Sign Extension Corrector is designed to increase the Booth multiplier capacity by multiplying not only the unsigned number but also the signed number.

The principle of the sign extension that converts the signed multiplier not signed as follows. When unisgn is signalled $s_u = 0$, it indicates the multiplication of the unsigned number and when $s_u = 1$, it shows the multiplication of the signed number. When a bit signal is called unsigned bit (s_u), it is indicated whether the multiplication operation is an unsigned number or number.

Table.5.Sign extension corrector

Sign-unsigned	Type of operation
0	Unsigned multiplication
1	Signed multiplication

Example:

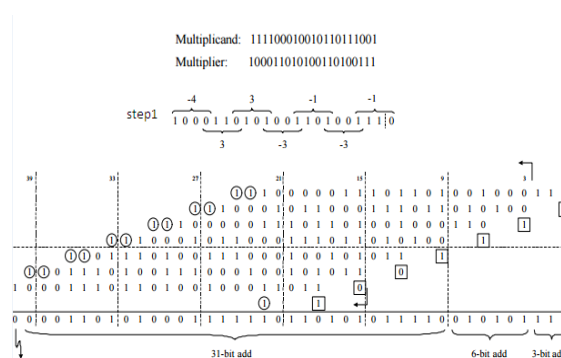
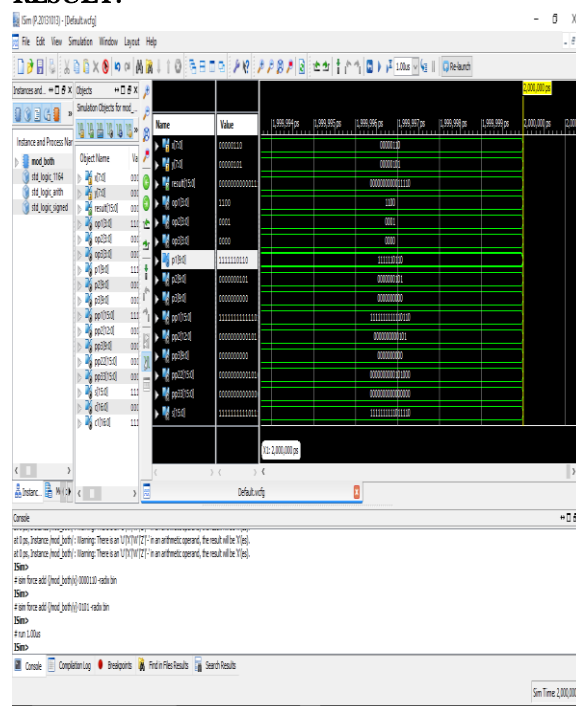


Fig.. Example of modified booth algorithm

RESULT:



CONCLUSION:

Arithmetic and Logic Unit forms an important part of the digital system design and various architectures are proposed which reduces the area or the timing delay of the circuit in recent years. The highest frequency could be got from the timing report. Power count of the involvement is proportional to the relevant frequency. Optimal timing delay done using Vedic mathematics in implementation of multipliers as the speed of ALU depends significantly on the speed of multiplier. Radix-8 modified booth encoding algorithm is used here to develop multiplier design for improved results.

REFERENCES :

- [1] Jagadguru Swami, Sri Bharati Krisna, Tirthaji Maharaja, "Vedic Mathematics or Sixteen Simple Mathematical Formulae From the Veda, Delhi (1965)", Motilal Banarsidas, Varanasi, India,
- [2] M. Morris Mano, "Computer System Architecture", 3rd edition, Prentice-Hall, New Jersey, USA, 1993, pp. 346-348.
- [3] H. Thapliyal and H.R Arbania. "A Time-Area-Power Efficient Multiplier and Square Architecture Based On Ancient Indian Vedic Mathematics", Proceedings of the 2004 International Conference on VLSI (VLSI'04), Las Vegas, Nevada, June 2004, pp. 434-439.
- [4] P. D. Chidgupkar and M. T. Karad, "The Implementation of Vedic Algorithms in Digital Signal Processing", Global J. of Engg. Edu, Vol.8, No.2, 2004, UICEE Published in Australia.
- [5] Thapliyal H. and Srinivas M.B, "High Speed Efficient NxN Bit Parallel Hierarchical Overlay Multiplier Architecture Based on Ancient Indian Vedic Mathematics", Transactions on Engineering, Computing and Technology, 2004, Vol.2.
- [6] Harpreet Singh Dhillon and Abhijit Mitra, "A Reduced- Bit Multiplication Algorithm for Digital Arithmetics" International Journal of Computational and Mathematical Sciences
- [7] Honey Durga Tiwari, Ganzorig Gankhuyag, Chan Mo Kim and Yong Beom Cho, "Multiplier design based on ancient Indian Vedic Mathematician", International SoC Design Conference, pp. 65- 68, 2008.
- [8] Parth Mehta and Dhanashri Gawali, "Conventional versus Vedic mathematics method for Hardware implementation of a multiplier", International conference on Advances in Computing, Control, and Telecommunication Technologies, pp. 640-642, 2009.
- [9] Ramalatha, M.Dayalan, K D Dharani, P Priya, and S Deoborah, "High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques", International Conference on Advances In Computational Tools for Engineering Applications (ACTEA) IEEE, pp. 600-603, July15-17, 2009.
- [10] Sumita Vaidya and Deepak Dandekar, "Delay-Power Performance comparison of Multipliers in

VLSI Circuit Design", International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, pp 47-56, July 2010.

[11] S.S.Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A "Implementation of Vedic Multiplier For Digital Signal " International conference on VLSI communication & instrumentation

[12] Pushpalata Verma, K. K. Mehta" Implementation of an Efficient Multiplier based on Vedic Mathematics Using EDA Tool" International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 - 8958, Volume-1, Issue-5, June 2012