ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Design of High Speed 2's Complement Multiplier-A Review

Mr. Ankit Bhatt
Student of ME, ENTC,
Dept of VLSI and Embedded systems,
Matoshri College of Engineering and Research Centre, Nashik, India.

Mr. Sachin D. Pable
Associate Professor,
Dept of VLSI and Embedded systems,
Matoshri College of Engineering and Research Centre, Nashik, India.

Abstract:

This paper presents the design and implementation of high speed 2s complement multiplier for 8bit. Two's complement multipliers are important for a wide range of applications. This reduction provides faster compression of the partial product array and regular layouts in two's complement multiplier. The modified Booth Encoder circuit generates half the partial products in parallel. By extending sign bit of the operands and generating an additional partial product the AMBE multiplier is obtained. The Carry Save Adder (CSA) tree and the final Carry Look ahead (CLA) adder used to speed up the multiplier operation. Since signed and unsigned multiplication operation is performed by the same multiplier unit the required hardware and the chip area reduces and this in turn reduces power dissipation and cost of a system. The Multiplier design implemented using Xilinx. This based on a rough theoretical analysis and on logic synthesis showed its efficiency in terms of both area and delay.

Index Terms:

Multiplication, Partial Product Array, Modified Booth Encoding.

IINTRODUCTION:

Multipliers play an important role in today's digital signal processing and various other applications. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, low power consumption, regularity of layout and hence less area or even combination of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation. In parallel multipliers number of partial products to be added is the main parameter that determines the performance of the multiplier.

In the past, the major concerns of the VLSI designer were area, performance, cost and reliability, power consideration was mostly of only secondary importance. In recent years, however, this has begun to change and, increasingly, power is being given comparable weight to area and speed considerations. Several factors have contributed to this trend. Perhaps the primary driving factor has been the remarkable success and growth of the class of personal computing devices (portable desktops, audio- and video-based multimedia products) and wireless communications systems (personal digital assistants and personal communicators) which demand high-speed computation and complex functionality with low power consumption. In these applications, average power consumption is a critical design concern. The projected power budget for a battery-powered, portable multimedia terminal, when implemented using off-the-shelf components not optimized for low-power operation, is about 40 W. In the absence of low-power design techniques then, current and future portable devices will suffer from either very short battery life or very heavy battery pack.

Multiplication:

If a positional numeral system is used, a natural way of multiplying numbers is taught in schools as long multiplication, sometimes called grade-school multiplication: multiply the each digit of the multiplier and then add up all the properly shifted results. It requires memorization of the multiplication table for single digits. This is the usual algorithm for multiplying larger numbers by hand in base. Computers normally use a very similar shift and add algorithm in base. A person doing long multiplication on paper will write down all the products and then add them together; an abacus-user will sum the products as soon as each one is computed.

Multiplication Algorithm:

A multiplication algorithm is an algorithm (or method) to multiply two numbers.

Depending on the size of the numbers, different algorithms are in use. Efficient multiplication algorithms have existed since the advent of the decimal system

General Principle of Binary Multiplication:

A multiplier plays an important role in various digital systems such as computer, process controller and signal processor. Consider two unsigned binary numbers X and Y. The basic algorithm is similar to the one used in multiplying the numbers on pencil and paper. The main operations involved are shift and add. Recall that the `pencil-and-paper' algorithm is inefficient as in that, each product term (obtained by multiplying each bit of the multiplier to the multiplicand) has to be saved till all such product terms are obtained. In machine implementations, it is desirable to add all such product terms to form the partial product. Also, instead of shifting the product terms to the left, the partial product is shifted to the right before the addition takes place. In other words, if Pi is the partial product after i steps and if Y is the multiplicand and X is the multiplier, then
Pi ← Pi + Xj * Y
And
Pi+1 ← Pi * 2-1
and the process repeats.

Note that the multiplication of signed magnitude numbers requires a straight forward extension of the unsigned case. The magnitude part of the product can be computed just as in the unsigned magnitude case. The sign p0 of the product P is computed from the signs of X and Y as
P0 ← x0 + y0

2's Complement Multiplication - Robertson's Algorithm: Consider the case that we want to multiply two 8 bit numbers X = x0x1:::x7 and Y = y0y1:::y7. Depending on the sign of the two operands X and Y, there are 4 cases to be considered:

X0 = y0 = 0, that is, both X and Y are positive. Hence, multiplication of these numbers is similar to the multiplication of unsigned numbers. In other words, the product P is computed in a series of add-and-shift steps of the form
Pi ← Pi + Xj * Y
Pi+1 ← Pi * 2-1

Note that all partial products are non-negative. Hence, leading 0s are introduced during right shift of the partial product.

x0 = 1; y0 = 0, that is, X is negative and Y is positive. In this case, the partial product is always positive (till the sign bit x0 is used). In the final step, a subtraction is performed. That is, $P \leftarrow P-Y$
x0 = 0; y0 = 1, that is, X is positive and Y is negative. In this case, the partial product is positive and hence leading 0s are shifted into the partial product until the _rst 1 in X is encountered. Multiplication of Y by this 1, and addition to the result causes the partial product to be negative, from which point on leading 1s are shifted in (rather than 0s).x0 = 1; y0 = 1, that is, both X and Y are negative. Once again, leading 1s are shifted into the partial product whenever the partial product is negative. Also, since X is negative, the correction step is also performed. Recall the difference in the correction steps between multiplication of two integers and two fractions. In the case of two integers, the correction step involves subtraction and shift right. Whereas, in the case of fractions, the correction step involves subtraction and setting Q(7)0.

II. LITERATURE REVIEW:

Martin S. Schmookler (1991) describes about the AltiVec technology is an extension to the Power PC architecture which provides new computational and storage operations for handling vectors of various data lengths and data types. The first implementation using this technology is a low cost, low power processor based on the acclaimed Power PC 750 microprocessor. This describes the micro architecture and design of the vector arithmetic unit of this implementation [1] . D.S.Dawoud (1965) describes the performance of multiplication is crucial for multimedia applications such as 3D graphics and signal processing systems, which depend on the execution of large numbers of multiplications. Many algorithms are proposed to implement high speed parallel multipliers. These algorithms mainly focused on rapidly reducing the partial products rows down to final sums and carries used for the final accumulation. Fewer partial products rows means lowering the overall operation time. The second technique uses the conventional way of getting the 2's complement but a simple hardware is proposed to implement the "add 1" operation without any carry propagation. In addition to the speed improvement, our algorithms result in a true diamond-shape for the partial product tree, which is more efficient in terms of implementation. The simulation of our proposed techniques showed large improvement in speed and in power consumption when compared to conventional multiplication algorithms.

Kyeongsoon Cho (2010) describes the pipeline architecture of high-speed modified booth multipliers. The multiplier circuits are based on the modified booth algorithm and the pipeline technique which are the most widely used to accelerate the multiplication speed. In order to implement the optimally pipelined multipliers, many kinds of experiments have been conducted. The speed of the multipliers is greatly improved by properly deciding the number of pipeline stages and the positions for the pipeline registers to be inserted. We described the proposed modified Booth multiplier circuits in Verilog HDL and synthesized the gate-level circuits using 0.13um standard cell library. The resultant multiplier circuits show better performance than others. Since the proposed multipliers operate at GHz ranges, they can be used in the systems requiring very high performance .Reto Zimmermann (2003) describes the latest approach to data path synthesis from RTL, data paths are extracted into largest possible sum-of-product (SOP) blocks, thus making extensive use of carry-save intermediate results and reducing the number of expensive carry propagations to a minimum.

The sum-of-product blocks are then implemented by constraint- and technology-driven generation of partial products, carry-save adder tree and carrypropagate adder. A smart generation feature selects the best among alternative implementation variants Special data path library cells are used where available and beneficial. All these measures translate into better performing circuits for simple and complex data paths in cell-based design [4]. Paul F. Stelling (1998) present new design and analysis techniques for the synthesis of parallel multiplier circuits that have smaller predicted delay than the best current multipliers[8]. In Oklobdzija, etal suggested a new approach, the Three-Dimensional Method (TDM), for Partial Product Reduction Tree (PPRT) design that produces multipliers that outperform the current best designs. The goal of TDM is to produce a minimum delay PPRT using full adders.

This is done by carefully modeling the relationship of the output delays to the input delays in an adder and, then, interconnecting the adders in a globally optimal way. Oklobdzija et al. suggested a good heuristic for finding the optimal PPRT, but no proofs about the performance of this heuristic were given. We provide a formal characterization of optimal PPRT circuits and prove a number of properties about them. For the problem of summing a set of input bits within the minimum delay, we present an algorithm that produces a minimum delay circuit in time linear in the size of the inputs [13]. Jung-Yup Kang (2002) proposes an innovative algorithm to the two's complement of a binary number. This method works in logarithmic time (O (log)) instead of the worst case linear time (O(N)) where a carry has to ripple all the way from LSB to MSB. The proposed method also allows for more regularly structured logic units which can be easily modularized and can be naturally extended to any word size. Our synthesis results show that our method achieves up to 2.8£ of performance improvement and up to 7.27£ of power savings compared to the conventional method. Stuart N.Wooters (2010) presents a fast energy-efficient level converter capable of converting an input signal from sub threshold voltages up to the nominal supply voltage. Measured results from a 130-nm test chip show robust conversion from 188 mV to 1.2 V with no intermediate supplies required. A combination of circuit methods makes the converter robust to the large variations in the current characteristics of sub threshold circuits. To support dynamic voltage scaling, the level converter can up convert an input at any voltage within this range to 1.2 V .Pramod M transistor level CMOS implementation of XOR3 and DFF is done in 130 nm CMOS process. The layout for the cells are designed and extracted for parasitic. The extracted net list is simulated for power, delay and input capacitance and the simulation results are used in standard library cell for synthesis of ALU core from the SPARC micro-processor.

III SYSTEM OVERVIEW:

In multimedia, 3D graphics and signal processing applications, performance, in most cases, strongly depends on the effectiveness of the hardware used for computing multiplications, since multiplication is, besides addition, massively used in these environments. The high interest in this application field is witnessed by the large amount of algorithms and implementations of the multiplication operation, which have been proposed in the literature (for a representative set of references, see [1]). More specifically, short bit-width (8-16 bits) two's complement multipliers with single-cycle throughput and latency have emerged and become very important building blocks for high-performance embedded processors and DSP execution cores [2], [3]. In this case, the multiplier must be highly optimized to fit within the required cycle time and power budgets. Another relevant application for short bit-width multipliers is the design of SIMD units supporting different data formats [3], [4].

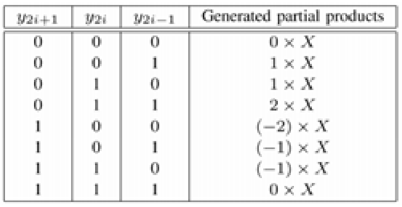
In this case, short bit-width multipliers often play the role of basic building blocks. Two's complement multipliers of moderate bit-width (less than 32 bits) are also being used massively in FPGAS. All of the above translates into a high interest and motivation on the part of the industry, for the design of high-performance short or moderate bit-width two's complement multipliers. The basic algorithm for multiplication is based on the well-known paper and pencil approach [1] and passes through three main phases: 1) partial product (PP) generation, 2) PP reduction, and 3) final (carry-propagated) addition.

During PP generation, a set of rows is generated where each one is the result of the product of one bit of the multiplier by the multiplicand. For example, if we consider the multiplication X Â Y with both X and Y on n bits and of the form xnà1 . . . X0 and ynà1 . . . Y0, then the ith row is, in general, a proper left shifting of yi* X, i.e., either a string of all zeros when yi= 0, or the multiplicand X itself when yi= 1. In this case, the number of PP rows generated during the first phase is clearly n.Modified Booth Encoding (MBE) is a technique that has been introduced to reduce the number of PP rows, still keeping the generation process of each row both simple and fast enough.

One of the most commonly used schemes is radix-4 MBE, for a number of reasons, the most important being that it allows for the reduction of the size of the partial product array by almost half, and it is very simple to generate the multiples of the multiplicand. More specifically, the classic two's complement n * n bit multiplier using the radix-4 MBE scheme, generates a PP array with a maximum height of [n/2]+1 rows, each row before the last one being one of the2 following possible values: all zeros, +-X;+-2X. The last row, which is due to the negative encoding, can be kept very simple by using specific techniques integrating two's complement and sign extension prevention [1].

The PP reduction is the process of adding all PP rows by using a compression tree [6], [7]. Since the knowledge of intermediate addition values is not important, the outcome of this phase is a result represented in redundant carry- save form, i.e., as two rows, which allows for much faster implementations. The final (carry-propagated) addition has the task of adding these two rows and of presenting the final result in a non redundant form, i.e., as a single row.

TABLE 3.1: Modified Booth Encoding (Radix-4)



y_{2i+1}	y_{2i}	y_{2i-1}	Generated partial products
0	0	0	$0 \times X$
0	0	1	$1 \times X$
0	1	0	$1 \times X$
0	1	1	$2 \times X$
1	0	0	$(-2) \times X$
1	0	1	$(-1) \times X$
1	1	0	$(-1) \times X$
1	1	1	$0 \times X$

In this work, we introduce an idea to overlap, to some extent, the PP generation and the PP reduction phases. Our aim is to produce a PP array with a maximum height of [n/2] rows that is then reduced by the compressor tree stage.2As we will see for the common case of values n which are power of two, the above reduction can lead to an implementation where the delay of the compressor tree is reduced by one XOR2 gate keeping a regular layout. Since we are focusing on small values of n and fast single-cycle units, this reduction might be important in cases where, for example, a high computation performance through the assembly of a large number of small processing units with limited computation capabilities are required, such as 8 Â 8 or 16 Â 16 multipliers .

IV CONCLUSION:

Multipliers are a major component of DSP circuits. Multipliers mainly comprise as the power consuming elements. For a good quality DSP circuit we need to implement an high speed multipliers with good precision and range for good multiplication. Multipliers consume a lot of power as the gate count of these multipliers increases exponentially with the width of the operands. In this paper, all modules are realized using verilog HDL. The synthesis and power analysis of all modules are done by using of Xilinx ISE 12.1 .The Proposed high speed multiplier architectures are very deserved for implementation of multipliers. There by multipliers performs the functions with less power dissipation while maintaining the speed. The radix-4 booth's multiplier architecture can further be improved by using fast adders to add the partial products.

References:

1)M.S. Schmookler, M. Putrino, A. Mather, J. Tyler, H.V. Nguyen, C.Roth, M. Sharma, M.N. Pham, and J. Lent, "A Low-Power, High-Speed Implementation of a PowerPC Microprocessor Vector Extension," Proc. 14th IEEE Symp. Computer Arithmetic, pp. 12-19,1999.

2)H. Lee, “A High-Speed Booth Multiplier”, ICCS, (2002).

3)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.

4)R. Zimmermann and D.Q. Tran, “Optimized Synthesis of Sum-of-Products,” Proc. Conf. Record of the 37th Asilomar Conf. Signals,Systems and Computers, vol. 1, pp. 867-872, 2003.

5)C.S. Wallace, “A Suggestion for a Fast Multiplier,” IEEE Trans.Electronic Computers, vol. EC-13, no. 1, pp. 14-17, Feb. 1964.D.E. Shaw, “Anton: A Specialized Machine for Millisecond-ScaleMolecular Dynamics Simulations of Proteins,” Proc. 19th IEEE Symp. Computer Arithmetic, p. 3, 2009.

6)J.-Y. Kang and J.-L. Gaudiot, “A Simple High-Speed Multiplier Design,” IEEE Trans.Computers, vol. 55, no. 10, pp. 1253-1258, Oct.2006.

7)F. Lamberti, N. Andrikos, E. Antelo, and P. Montuschi,“Speeding-Up Booth Encoded Multipliers by Reducing the Size of Partial Product Array,” internal report, http://arith.polito.it/ir_mbe.pdf, pp. 1-14, 2009.

8)P.F. Stelling, C.U. Martel, V.G. Oklobdzija, and R. Ravi, “Optimal Circuits for Parallel Multipliers,” IEEE Trans. Computers, vol. 47,no. 3, pp. 273-285, Mar. 1998.

9)R.Hashemian and C.P.Chen, “A New Parallel Technique for Design of Decrement/Increment and Two’s Complement Circuits,” Proc. 34th Midwest Symp. Circuits and Systems, vol. 2,pp. 887-890, 1991.

10)M.D. Ercegovac and T. Lang, Digital Arithmetic. Morgan Kaufmann Publishers, 2003.

11)H. Kaul, M.A. Anders, S.K. Mathew, S.K. Hsu, A. Agarwal, R.K.Krishnamurthy, and S. Borkar, “A 300 mV 494GOPS/W Reconfi-gurable Dual-Supply 4-Way SIMD Vector Processing Accelerator in 45 nm CMOS,” IEEE J. Solid State Circuits, vol. 45, no. 1, pp. 95-101, Jan. 2010

12)S.K. Hsu, S.K. Mathew, M.A. Anders, B.R. Zeydel, V.G.Oklobdzija, R.K. Krishnamurthy, and S.Y. Borkar, “A 110GOPS/ W 16-Bit Multiplier and Reconfigurable PLA Loop in 90-nm CMOS,” IEEE J. Solid State Circuits, vol. 41, no. 1, pp. 256-264, Jan.2006.

13)V.G. Oklobdzija, D. Villeger, and S.S. Liu, “A Method for Speed Optimized Partial Product Reduction and Generation of Fast Parallel Multipliers Using an Algorithmic Approach,” IEEE Trans.Computers, vol. 45, no. 3, pp. 294-306, Mar. 1996.