

Design and Verification of High Speed Multiplier

V V S Vijaya Krishna, K Sai Krishna

Abstract—Multiplier is one of the essential element for all digital systems such as digital signal processors, microprocessors etc. In this paper, a new high speed multiplier using booth recoding technique is presented. This algorithm can be implemented by using the radix-8 booth recoding process. The proposed multiplier reduces the partial product array by almost $3/4^{\text{th}}$ the size of the bits. This reduction increases the speed of the multiplier. The proposed method can be extended to any higher radix encodings, as well as to any size square and rectangular multipliers. The proposed multiplier is compared with the standard multiplier and two's complement multiplier using radix-4 MBE technique, demonstrated the good delay performance. These results show that the proposed multiplier is faster compared to other multipliers. The performance of the proposed multiplier is examined using verilog simulator in XILINX 12.4 version.

Index Terms- Multiplication, radix-8 booth recoding, partial product array

I. Introduction

Multiplier is one of the basic hardware block used for many digital and high performance systems such as FIR filters, digital signal processors and microprocessors etc. Many high speed low power multiplication algorithms and architectures have been proposed. Advances in technology have permitted many researchers to design multipliers which offer both high-speed and regularity of layout, thereby making them suitable for VLSI implementation. Digital signal processing requires efficient multiplication operations with the highest possible speed without compromising the power budget. In general, basic multiplication algorithm can be divided into three following steps. 1) partial product (pp) generation, 2) partial product reduction and 3) final carry propagated addition [1-2]. In the first step, a set of partial product rows is generated where each one is the result of the product of one bit of the multiplier by multiplicand. For example, if we consider the multiplication $X \times Y$ with both X and Y on n bits and of the form $x_{n-1} \dots x_0$ and $y_{n-1} \dots y_0$, then the i^{th} row is, in general, a proper left shifting of $y_i \times X$, i.e., either a string of all zeros when $y_i = 0$, or the multiplicand X itself when $y_i = 1$. In this case, the number of PP rows generated during the first phase is clearly n [1-4]. Recoding of binary numbers was first hinted at by Booth [5] four decades ago. MacSorley [6] proposed a modification of Booth's algorithm a decade after. Modified booth encoding (MBE) [6] is a technique that has been introduced to reduce the no of pp rows with a maximum height of $\lceil n/2 \rceil + 1$ rows. More specifically, Two's complement multiplier [7] using radix-4 MBE generates a pp array with a maximum height of $\lceil n/2 \rceil$ rows without any increase of delay, each row of the pp array follows the one of the following possible values: all zeros, $\pm X$, $\pm 2X$ [8].

This pp reduction may increases the speed of the multiplier. During pp reduction phase, all pp rows are reduced by using compression tree [9-10]. Since the 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.

In this work, we introduce an idea to reduce the pp array with a maximum height of $\lceil n/3 \rceil$ rows by using radix-8 booth recoding process. A similar study aimed at the reduction of the maximum height to $\lceil n/3 \rceil$ but using a different approach has recently presented interesting results in [11]. Thus, in the following, we will evaluate and compare the proposed approach with the technique in [7].

The paper is organized as follows: in section II, the multiplication algorithm based on radix-8 booth recoding process is briefly reviewed and analyzed. In section III, proposed multiplier structure implementation. In section IV explains simulation results and finally in section V ends with conclusion.

II. Radix-8 booth recoding

Booth multiplication is a technique that allows for smaller, faster multiplication circuits, by recoding the numbers that are multiplied. It is the standard technique used in chip design, and provides significant improvements over the "long multiplication" technique. One of the solutions of realizing high speed multipliers is to enhance parallelism which helps to decrease the number of subsequent calculation stage.

To increase the speed of the multiplication, a variation of the Modified Booth Encoding (MBE) approaches has to be employed, which reduces the number of partial product rows to half the size of the bits [5]. Radix-8 booth recoding is one of the techniques used to reduce the partial product arrays to $3/4^{\text{th}}$ the size of the bits. To multiply, multiplicand 'X' by multiplier 'Y' using the radix-8 booth recoding process. First group the multiplier bits 'Y' by four bits and encoding into one of $\{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$. Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) and Most Significant Bit (MSB) of the multiplier. Table 1 shows the rules to generate the encoded signals by MBE scheme.

From an operational point of view, multiply by zero means the multiplicand is multiplied by "0". Multiply by "1" means the product still remains the same as the multiplicand value. Multiply by "-1" means that two's complement of the multiplicand value. Multiply by "2" means just left shift the multiplicand by one place. Multiply by "-2" is to shift left one bit the two's complement of the multiplicand value. Multiply by 3 means adding the multiplicand value and left shift the multiplicand value.

Manuscript published on 30 November 2013.

*Correspondence Author(s)

V V S Vijaya Krishna, ECE Dept, Vignan University, Vadlamudi- AP, India.

K Sai Krishna, ECE Dept, CVSR College of Engg, Hyderabad- AP, India.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license <http://creativecommons.org/licenses/by-nc-nd/4.0/>

TABLE I. Radix-8 booth recoding

Y_{2i+2}	Y_{2i+1}	Y_{2i}	Y_{2i-1}	Generated partial product
0	0	0	0	$0 \times X$
0	0	0	1	$1 \times X$
0	0	1	0	$1 \times X$
0	0	1	1	$2 \times X$
0	1	0	0	$2 \times X$
0	1	0	1	$3 \times X$
0	1	1	0	$3 \times X$
0	1	1	1	$4 \times X$
1	0	0	0	$(-4) \times X$
1	0	0	1	$(-3) \times X$
1	0	1	0	$(-3) \times X$
1	0	1	1	$(-2) \times X$
1	1	0	0	$(-2) \times X$
1	1	0	1	$(-1) \times X$
1	1	1	0	$(-1) \times X$
1	1	1	1	$0 \times X$

Multiply by “-3” means two’s complement of the partial product “3” value. Multiply by “4” means shift left two bits the multiplicand value and multiply by “-4” is to shift left two bits the two’s complement of the multiplicand value

III. Proposed architectures for radix 8 technique

The proposed approach is general and, for the sake of clarity, will be explained through the practical case of 8×8 multiplication. As briefly outlined in the previous sections, the main goal of our approach is to produce a partial product array with a maximum height of $\lceil n/3 \rceil$ rows, without introducing any additional delay.

In radix 8 booth recoding technique, multiplier will be scanned as a four bit window. For each group of four bits (y_{2i+2} , y_{2i+1} , y_{2i} , y_{2i-1}), only one partial product row is generated according to the encoding in Table 1. A possible implementation of radix-8 and of corresponding partial product generation is shown in Fig. 1 and Fig. 2. For each partial product row, Fig. 1 produces the one, two, three, four and neg signals. These signals are then exploited by the logic in Fig. 2, along with the appropriate bits of the multiplicand, in order to generate the whole partial product array.

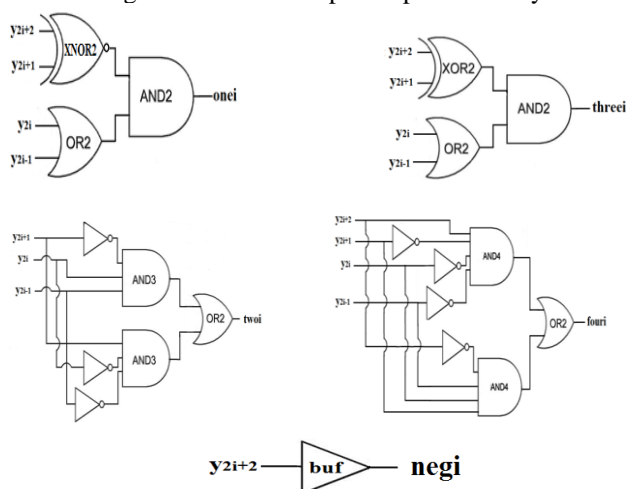


Fig. 1. Gate-level diagram for booth encoded signals generation

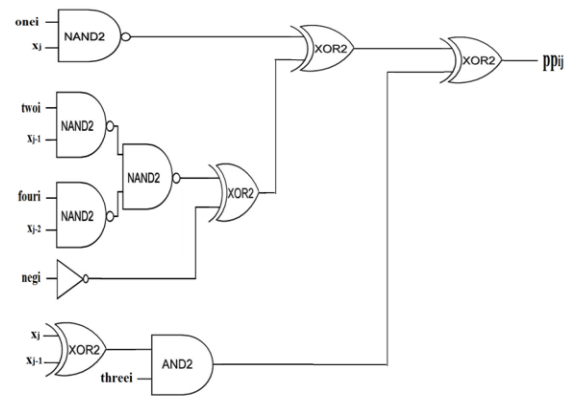


Fig. 2. Gate-level diagram for partial product generation

In particular, we observe that, by direct comparison of Figs. 1 and 3, the generation of booth encoded signals for the first row is simpler, and theoretically allows for the saving of the delay of one AND4 gate. As we know that, in radix8 booth recoding process, only three bits are required to obtain the first row partial product array. So, we have implemented new circuits for the first row to increase the speed of the multiplier, although it could require a small amount of additional area. As we see in the following, this issue hardly has any significant impact on the overall design, since this extra hardware is used only for the four most significant bits of the first row, and not for all the other bits of the array.

The high-level description of our idea is as follows:

1. Generation of the bits for the first row: possible implementations can use the circuitry of Fig. 3, for each bit of the first row.
2. Parallel generation of the bits of the other rows: possible implementations can use the circuitry of Fig.1, replicated for each bit of the other rows.

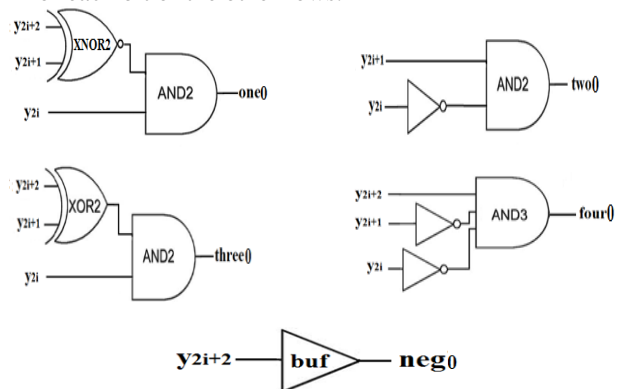


Fig. 3. Gate-level diagram for first row booth encoded signals generation

IV. SIMULATION RESULTS

The multiplier unit design applying the proposed radix-8 recoding algorithm has been done thinking in its VLSI implementation. Proposed multiplier is designed and verified using verilog simulator in XILINX 12.4 version. A comparison with existing or already reported designs is included which shows the advantage of the proposed design good delay performance and the results are shown in Tab II.

The top module and input, output waveforms are shown in Fig. 4 & 5. These waveforms show that the proposed multiplier reduces the partial product array with a maximum height of $\lceil n/3 \rceil$, thus making this multiplication algorithm attractive in high performance digital signal processing applications.

The input data is 8bit multiplicand and 8 bit multiplier i.e.

Multiplicand = 8'b01010010

Multiplier = 8'b01010101

The output generated is: 0001101100111010

Multiplicand = 8'b00101010

Multiplier = 8'b11001101

The output generated is: 0010000110100010

Multiplier	Delay (ns)
Standard multiplier	9.9
Two's complement [9]	11.9
Two's complement [pr]	9.850
Proposed multiplier	9.586

Table II. Performance comparison of the proposed and alternative multipliers.

TOP MODULE:

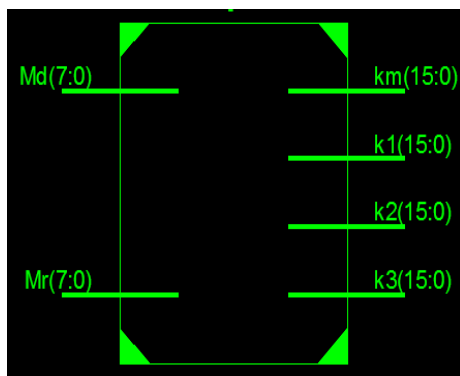


Fig 4: Top Module-Symbol of Multiplier

TOP MODULE WAVEFORM:

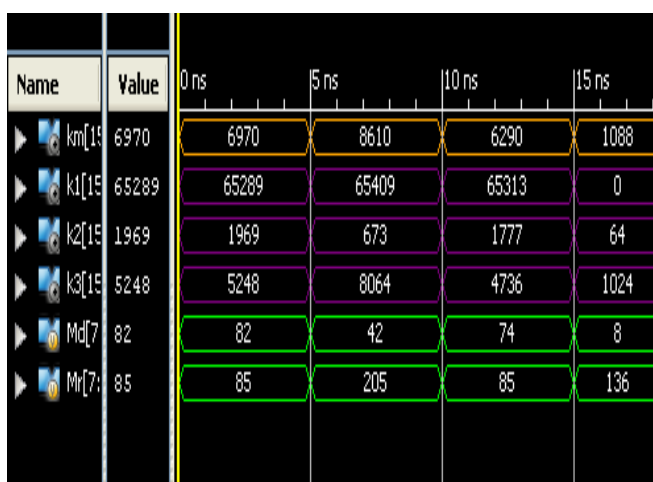


Fig. 5. Simulated input & output waveforms of proposed multiplier

V. CONCLUSION

Two's complement multiplier using radix-4 Modified Booth Encoding produces the partial product array has a maximum height of $\lceil n/2 \rceil$. We presented a scheme that produces a partial product array with a maximum height of $\lceil n/3 \rceil$, without introducing any extra delay. The outcome of the above is that the reduction of the maximum height of the partial product array by almost $3/4^{\text{th}}$ the size of the bits may simplify the partial product reduction tree, both in terms of delay and regularity of the layout. We have also compared our approach with a recent proposal with the same aim, considering results using a widely used synthesis tool and a modern technology library, and concluded that our approach may improve the performance of the square multiplier designs. The proposed approach also applies with minor modifications to rectangular multipliers.

REFERENCES

1. M.D. Ercegovic and T. Lang, Digital Arithmetic. Morgan Kaufmann Publishers, 2003.
2. 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.
3. H. Kaul, M.A. Anders, S.K. Mathew, S.K. Hsu, A. Agarwal, R.K. Krishnamurthy, and S. Borkar, "A 300 mV 494GOPS/W Reconfigurable 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.
4. 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.
5. A.D. Booth, "A signed binary multiplication technique," Quarterly J. Mechan. Appl. Math., vol. 4, no. 2, pp. 236-240, 1951.
6. O.L. MacSorley, "High Speed Arithmetic in Binary Computers," Proc. IRE, vol. 49, pp. 67-91, Jan. 1961.
7. Fabrizio Lamberti, Nikos Andrikos, Elisardo Antelo, Paolo Montuschi, "Reducing the computation time in (Short Bit-Width) Two's Complement Multipliers," IEEE Trans. Computers, vol. 60, no. 2, pp. 148-156, Feb. 2011.
8. G. Jaya Prada1, N.C. Pant, "Design and Verification of Faster Multiplier," Proc. IJERA, Vol. 1, Issue 3, pp.683-686.
9. L. Dadda, "Some Schemes for Parallel Multipliers," Alta Frequenza, vol. 34, pp. 349-356, May 1965.
10. C.S. Wallace, "A Suggestion for a Fast Multiplier," IEEE Trans. Electronic Computers, vol. EC-13, no. 1, pp. 14-17, Feb. 1964.
11. J.A. Hidalgo, V. Moreno-Vergara, O. Oballe, A. Daza, M.J. Martín-Vázquez, A.Gago, "A Radix-8 Multiplier unit Design for Specific Purpose,"