Efficient Design for Multiplication using Booth Algorithm

Sanjeev Kumar[1] Sukhvinder Kaur[2]
[1]M. Tech. Student [2]Assistant Professor
[1,2]Department of Electronics and Communication Engineering
[1,2]Swami Devi Dyal Institute of Engg. And Technology, (Affiliated to Kurukshetra University)

Abstract— We introduce here a multiplication approach for 16 bit input using modified booth algorithm. This reduces the number of digits in a multiplication compared with a binary multiplication. In this paper we will compare the complexity of binary against ternary multi-pliers. The design is simulated using Model Sim 6.5e and synthesized using Xilinx ISE design .Results obtained from proposed design is in terms of area, delay and speed have been compared with conventional multiplier design.

Key words: Booth Algorithm, Design for Multiplication

I. Introduction

Arithmetic operations are fundamental to modern digital computer. Addition and multiplication are used ubiquitously in various applications and their importance cannot be overemphasized. For carry-propagation addition (CPA), although there have been many algorithms developed to achieve optimal performance, new algorithms are still being proposed to further improve the performance and they are always claimed to be superior to the old ones based on some benchmarks.

However, none of them gives formal proof at algorithm level. Therefore we develop a set of operators and notations to describe CPA algorithms at first. Based on the derived equations and notations, we find that it is possible to achieve optimal performance for CPA by the scheduling of bit-level Boolean operations. To solve the scheduling problem, dual-bit forward prediction (DFP)and generalized earliest-first (GEF) bit-level scheduling algorithms have been proposed herein.

For parallel multiplication, although there have been various previous works focus on the design of multiplier, many of them do not provide quantitative analysis at algorithm level. Therefore we divide multiplication operation into three steps and analyze each of them to obtain more accurate model. The first step is partial product generation, the second one is partial product reduction, and the third one is final addition. For partial product generation, two popular approaches are evaluated. The first approach, direct-generation, generates partial products directly via two-input AND gates. The second approach is based on modified-Booth encoding (MBE).

II. Introduction about Multiplication

Multiplication (often denoted by the cross symbol "×") is the mathematical operation of scaling one number by another. It is one of the four basic operations in elementary arithmetic (the others being addition, subtraction and division).

A. *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 multiplicand by 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 10. Computers normally use a very similar shift and add algorithm in base 2. 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.

1) *Example:*

This example uses long multiplication to multiply 23,958,233 (multiplicand) by 5,830 (multiplier) and arrives at 139,676,498,390 for the result (product).

```
       23958233
          5830 ×
    ------------
00000000 (=   23,958,233 ×    0)
71874699  (=    23,958,233 ×   30)
   191665864  (=     23,958,233 ×   800)
  119791165   (=     23,958,233 × 5,000)
    ------------
  139676498390 (= 139,676,498,390 )
```

2) *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

3) *Types of Multiplication Algorithms*

- Booth's Algorithm
- Modified Booth's Algorithm
- Wallace Tree Algorithm

4) *Booth's Algorithm:*

Booth's algorithm is a multiplication algorithm which worked for two's complement numbers. It is similar to our paper-pencil method, except that it looks for the current as well as previous bit in order to decided what to do. Here are steps

- If the current multiplier digit is 1 and earlier digit is 0 (i.e. a 10 pair) shift and sign extend the multiplicand, subtract with previous result.
- If it is a 01 pair, add to the previous result.
- If it is a 00 pair, or 11 pair, do nothing.

Let's look at few examples.

```
        4 bits
         0110   <- 6
      x  0010   <- 2
   -------------
      00000000
    -   0110
  --------------
      11110100
    +   0110
  --------------
  (1) 00001100 <- 12     (overflow bit ignored)
```


8 bits

In Booth's algorithm, if the multiplicand and multiplier are n-bit two's complement numbers, the result is considered as 2n-bit two's complement value. The overflow bit (outside 2n bits) is ignored. The reason that the above computation works is because

0110 x 0010 = 0110 x (-0010 + 0100) = -01100 + 011000 = 1100.

A long example:

```
          10011100    <- -100
        x   01100011    <- 99
     -------------------
       00000000 00000000
     -  11111111 10011100
     -------------------
       00000000 01100100
+ 11111110 011100
     -------------------
       11111110 11010100
     -  11110011 100
     -------------------
       00001011 01010100
+ 11001110 0
     -------------------
       11011001 01010100    <- -9900
```

Note that the multiplicand and multiplier are 8-bit two's complement number, but the result is understood as 16-bit two's complement number. Be careful about the proper alignment of the columns. 10 pair causes a subtraction, aligned with 1, 01 pair causes an addition, aligned with 0. In both cases, it aligns with the one on the left. The algorithm starts with the 0-th bit. We should assume that there is a (-1)-th bit, having value 0.

Booth Algorithm Advantages and Disadvantages

- Depends on the architecture
  - Potential advantage: might reduce the # of 1's in multiplier
- In the multipliers that we have seen so far:
  - Doesn't save in speed (still have to wait for the critical path, e.g., the shift-add delay in sequential multiplier)
  - Incr
  - Eases area: recoding circuitry AND subtraction

5) Modified Booth:

- Booth 2 modified to produce at most n/2+1 partial products.
- Algorithm: (for unsigned numbers)

1) Pad the LSB with one zero.
2) Pad the MSB with 2 zeros if n is even and 1 zero if n is odd.
3) Divide the multiplier into overlapping groups of 3-bits.
4) Determine partial product scale factor from modified booth 2 encoding table.
5) Compute the Multiplicand Multiples
6) Sum Partial Products

- Can encode the digits by looking at three bits at a time
- Booth recoding table:
  - Must be able to add *multiplicand* times –2, -1, 0, 1 and 2
- Since Booth recoding got rid of 3's, generating partial products is not that hard (shifting and negating)

III. LITERATURE REVIEW

Considering the requirement of the complete idea of the booth algorithm, Author collected additional knowledge on various adders and filters through the literatures in order to make improved performance. P.song et.al in [4] implementations of high-performance parallel multipliers are discussed. Circuit building blocks required for partial-product reduction are analyzed and two schemes leading to highly regular layouts are proposed. The circuit implementations related to the first-scheme in three different BiCMOS technologies are discussed. Ron.S et.al in [5] Wallace high-speed multipliers use full adders and half adders in their reduction phase. Half adders do not reduce the number of partial product bits. Therefore, minimizing the number of half adders used in a multiplier reduction will reduce the complexity. adders.Renteria et.al in [6] This paper presents the design of 8192-bit Montgomery multipliers based on radix-8 Booth encoding and coded-digit. Both multipliers use a systolic architecture and simultaneously carry out two multiplications. Kumre, L et.al in [7]. Here we presents Modified Gate Diffusion Input technique used to implement radix 4 Booth Multiplier. Use of Booth algorithm as compare to other multiplication algorithms in multiplication process shows less computation and less complexity as it minimizes the total number of partial products to half of it.

IV. SIMULATION RESULTS

The below figures shows the simulation results of test cases applied to the Multiplier

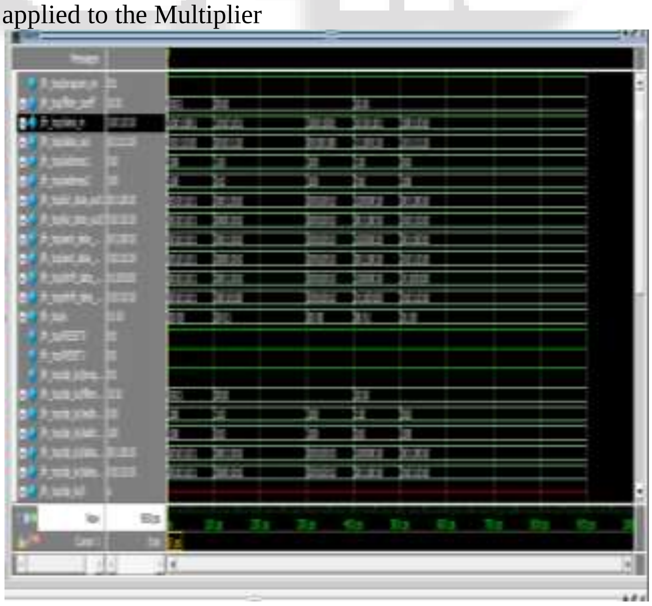

Fig. 1: Simulation Result of Multiplier

A. Synthesis Result:

The figure below shows the RTL schematic after synthesise the results.

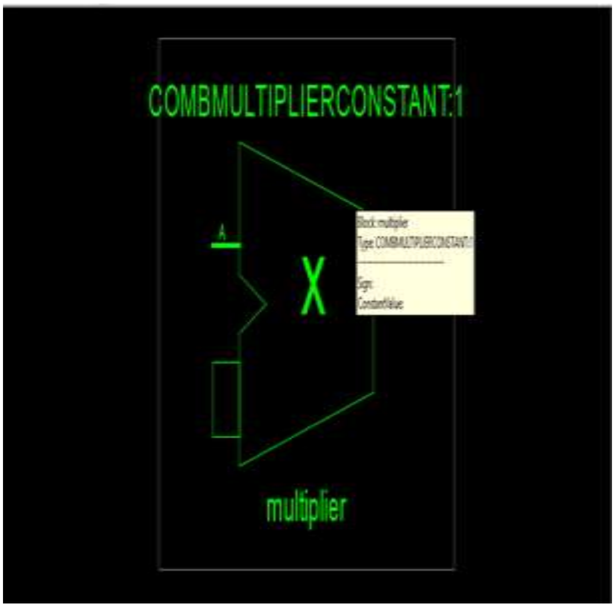

Fig. 2: combinational multiplier

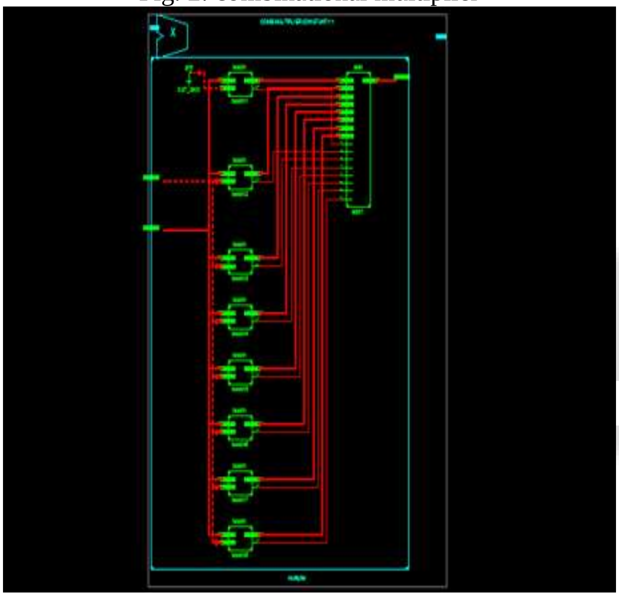
Fig. 3: internal circuit of multiplier

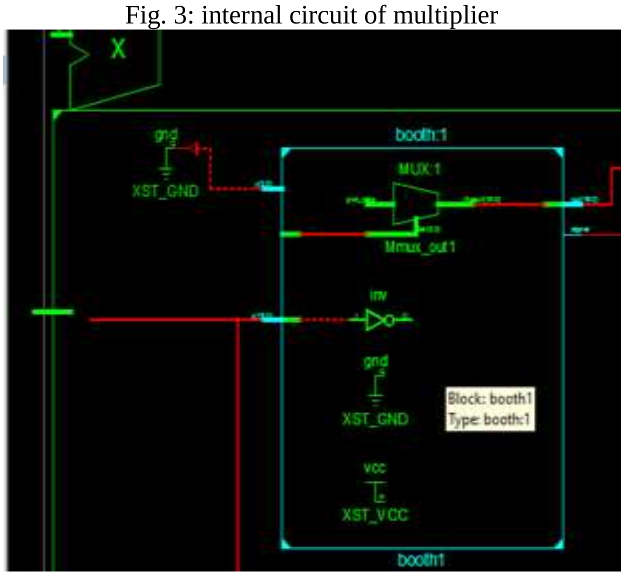

Fig. 4: Internal circuit of booth multiplier

B. Design Summary

multiplier Project Status (06/07/2015 - 12:44:38)			
Project File:	[illegible]	Parser Errors:	No Errors
Module Name:	multiplier	Implementation State:	Synthesized
Target Device:	[illegible]	• Errors:	No Errors
Product Version:	ISE 12.1	• Warnings:	8 Warnings (8 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	358	2400	14%
Number of fully used LUT-FF pairs	1	358	0%
Number of bonded IOBs	64	102	62%

Fig. 5 Detail Summary

C. Analysis:

Comparison of Area Occupied and Speed of Various

NO. OF BITS	DELAY (ns)	AREA (Number of LUTs)
8 bit existing (ternary)	19.33	-
3 bit existing	2.864	210
8 bit existing	2.863	331
12 bit existing	2.863	423
16 bit proposed	1.437	358

Table 1: Delay and area comparison for Multiplier Architectures

V. CONCLUSION

The complete design and implementation booth multiplier for 16 bit discussed here.

This new approach provides us better results in terms of Area, delay and speed than conventional multiplier.

The power consumption is reduced by improving partial product compression ratio. The power and delay is optimization is main aim of all designs. Various techniques for applied for achieving this.

REFERENCES

[1] M.D. Ercegovac 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 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.

[4] P. Song and G. De Michelli "Circuit and Architecture Tradeoffs for High Speed Multiplication", IEEE Journal of Solid State Circuits, vol. 26, no. 9, 1991

[5] Ron S. Waters, Member, IEEE, and Earl E. Swartz-lander, Jr., Fellow, IEEE," A Reduced Complexity

Wallace Multiplier Reduction," IEEE transactions on computers, vol. 59, no. 8, August 2010.
[6] Renteria-Mejia, C.P. Trujillo-Olaya, V. ; Velasco-Medina, J. "design of 8192-bit Montgomery multipliers based on radix-8 Booth encoding and coded-digit " Circuits and Systems (LASCAS), 2013 IEEE Fourth Latin American Symposium pp.-1 – 4, 2013
[7] Kumre, L.,Somkuwar, A. ; Agnihotri, G. Modified Gate Diffusion Input technique used to implement radix 4 Booth Multiplier, Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy (AICERA/ICMiCR), 2013 Annual International Conference on pp-1-5,2013.
[8] Rakesh Kumar, Neelam Sharma "Design of Fast Pipelined Multiplier using Modified Redundant Adder" I.J. Intelligent Systems published in MECS and Applications, 47-53,2012
[9] Hsiao, Shen-Fu, Ming-Roun Jiang, and Jia-Sien Yeh, "Design of highspeed low-power 3-2 counter and 4-2 compressor for fast multipliers,"IEEE Electronics Letters, vol. 34, no.4, pp. 341-343, Feb. 1998.
[10] I.S. Abu I.S.Abu-Khater, A. Bellaouar, and M. I. Elmastry, "Circuit techniques for CMOS low-power high-performance multipliers," IEEE J. Solid-State Circuits, vol. 31, pp. 1535-1546, Oct. 1996.
[11] Arkadiy Morgenshtein, Alexander Fish, and Israel A. Wagner, "Gate- Diffusion Input (GDI)-A Power-Efficient method for digital combinatorial circuits" IEEE Transactions on VLSI systems, vol. 10, No. 5, October 2002
[12] Yuan-Ho Chen; Chung-Yi Li; Tsin-Yuan Chang,"Area-Effective and Power-Efficient Fixed-Width Booth Multipliers Using Generalized Probabilistic Estimation BiasEmerging and Selected Topics in Circuits and Systems", IEEE Journal on Volume: 1, Issue: 3,pp- 277 - 288, 2011

