ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Design of Modified Booth Multiplier using Reversible gate logic for Radix-8

Baljinder Kaur[1], Manmeet Kaur[2]
[1]Research Scholar, Chandigarh Engineering College, Landran, Mohali,India
[2]Asstt. Professor, ECE Department, Chandigarh Engineering College, Landran, Mohali,India
cheemabindu@gmail.com

Abstract

Abstract— This paper describes the concept of multiplication by using modified booth algorithm and reversible logic functions for radix-8. By using Modified booth algorithm, less delay is produced compared to normal multiplication process. This booth algorithm also reduces the number of partial products which will reduce maximum delay count at the output. Reversible logic has the advantage of reducing the gate count, garbage outputs as well as constant inputs. Results are compared with Radix-2 and Radix-4 Booth multiplier. This modified booth algorithm is synthesized and simulated by using Xilinx 8.1 ISE simulator and ModelSim.

Keywords: Booth multiplier, Reversible logic gate, Multiplier and-Accumulator (MAC), Modified booth algorithm.

Introduction

Increasing demands of high speed data signal processing motivated the researchers to seek fastest multipliers or processors. The advancement of the microelectronic technologies makes better use of input energy, to encode the data more efficiently, to transmit the information faster and reliable, etc. In these applications, a multiplier is a fundamental arithmetic unit and used in almost every circuit especially in Digital signal processing (DSP) applications. The multiplier is implemented in very-large-scale integration (VLSI) designs in large area. The building blocks of the processor are multiplier and multiplier-and- accumulator (MAC) and have a great impact on the speed of the processor. Multiplier consists of three operational steps: Booth Encoder, Wallace Tree and final adder. Booth multiplication allows for the smaller, faster multiplication circuits through encoding the signed bits to 2's complement which is also the standard technique in chip design and provide improvement by reducing the partial products. Although the partial products are further reduced by using higher radix Booth Encoder which improves the performance. In order to increase the speed, two major parameters are considered, one is partial product reduction technique that is used in multiplication block and other is the accumulator. Both of the stages require the addition of large operands that require long paths for carry propagation. Wallace Tree and compressors are used to add the partial products. The last step is the final addition in which final result is produced by addition of sum and carry. If the process to accumulate is also considered then a MAC consists of four operational steps as shown in Figure 1.

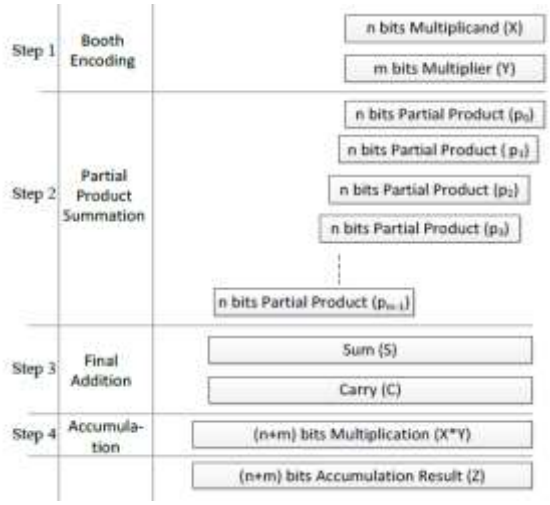

Figure 1: Basic arithmetic steps for multiplication and Accumulation

Components of booth multiplier

Wallace Tree Multiplier: A fast process for multiplication of two numbers was developed by Wallace. Using this method, a three step process is used to multiply two numbers; the bit products are formed, the bit product matrix is reduced to a two row matrix where sum of the row equals the sum of bit products, and the two resulting rows are summed with a fast adder to produce a final product. In this architecture, all the bits of all partial products in each

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

column are added together to a set of counters in parallel without propagating the carries. Another set of counters reduces this new matrix until a two row matrix is generated. A fast adder is at the end produces the final result. Conclusion: In the Wallace tree method, the circuit layout is not easy although the speed of the operation is high since the circuit is quite irregular. Wallace tree styles are generally avoided for low power applications, since excess of wiring is likely to consume extra power.

Booth Multiplier: The Booth recoding multiplier is one such multiplier; it scans the three bits at a time to reduce the number of partial products. These three bits are: the two bit from the present pair; and a third bit from the high order bit of an adjacent lower order pair. After examining each triplet of bits, the triplets are converted by Booth logic into a set of five control signals used by the adder cells in the array to control the operations performed by the adder cells. The method of Booth recording reduces the numbers of adders and hence the delay required to produce the partial sums by examining three bits at a time. To speed up the multiplication Booth encoding performs several steps of multiplication at once. Booth's algorithm takes advantage of the fact that an adder subtracted is nearly as fast and small as a simple adder. Conclusion: The drawbacks of booth multiplier are number of add subtract operations and the number of shift operation becomes variable and becomes inconvenient in designing parallel multipliers. The algorithm becomes inefficient when there are isolated 1's.which results in more power consumption due to large number of adders. Summing the partially redundant partial products requires as much hardware as representing them in the fully redundant form along the evaluation path.

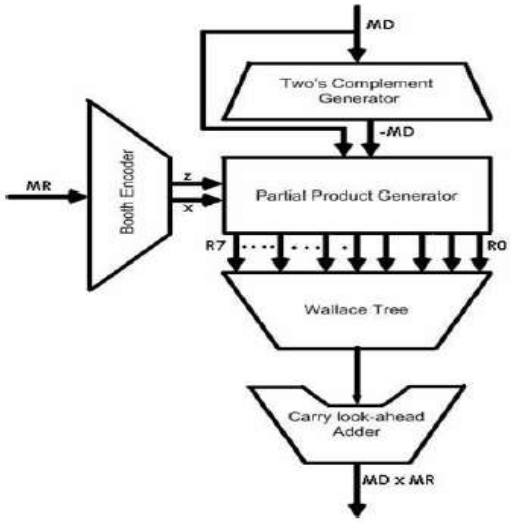

Figure 2: Block diagram of booth multiplier

Radix 2 Booth's Algorithm:
Radix 2 multiplication algorithm is multiplication algorithm that multiplies two signed binary numbers in two's compliment notations.
The Booth's algorithm serves two purposes:
1) Fast multiplication
2) Signed multiplication
Booth's algorithm requires examination of the multiplier bits, and shifting of the partial product. Prior to the shifting, the multiplicand may be added to partial product or subtracted from the partial product, or left unchanged according to the following rules:
I. First step is to add '0' to the LSB of the multiplier i.e.B_{-1} is 0.
II. Second step is to look at the bits B_x and B_{x-1} and perform operation according to the table given below.
0 0 Shift only
1 1 Shift only
0 1 Add Y to U, and shift
1 0 Subtract Y from U, and shift or add (-Y) to U and shift
III. The next step is to set two registers, which we name u and v, to be zero. These are going to be the registers where we store our product throughout the working of the problem.
IV. Once added, we then do an arithmetic right shift on u and v, with the last bit of v dropping off, and a circular right shift on x, also copying the LSB of x into x-1.
V. Now repeat steps for n number of times where n is number of bits in multiplier and multiplicand. And finally we have result of A x B.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

Drawbacks of Radix-2 Booth's Algorithm

1) Variable number of add/subtract operations and of shift operations between two consecutive add/subtract operations, Inconvenient when designing a synchronous multiplier.
2) Algorithm inefficient with isolated 1's.

Radix-4 Modified Booth's Algorithm:

Booth's 2 modified to produce at most n/2+1 partial products.MBE is widely been adopted in parallel multipliers since it can reduce the number of partial product rows to be added by half, thus reducing the size and enhancing the speed of the reduction tree.
The booth's encoding algorithm is a bit-pair encoding algorithm that generates partial products which are multiples of the multiplicand. The booth's algorithm shifts and/or complements the multiplicand (X operand) based on the bit patterns of the multiplier (Y operand). Essentially, three multiplier bits [Y (i+1), Y (i) and Y (i-1)] are encoded into nine bits that are used to select multiples of the multiplicand {-2X, -X, 0, +X, +2X}. The three multiplier bits consist of a new bit pair [Y (i+1), Y (i)] and the leftmost bit from the previously encoded bit pair [Y (i-1)] as shown in figure 3.1.Partial products can be generated with the help of table 1.

- Separately: x_{i-2} and x_{i-3} recoded into y_{i-2} and $y_{i-3} - x_{i-4}$ serves as reference bit.
- Groups of 3 bits each overlap - rightmost being x1 x0 (x_{-1}), next x3 x2 (x_1), and so on.
- Bits xi and x_{i-1} recoded into y_i and $y_{i-1} - x_{i-2}$ serves as reference bit.

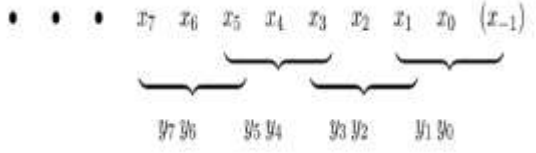

Figure 3: Recoding in Radix 4

Table 1: Booth's Radix-4 Algorithm Table

x_i	x_{i-1}	x_{i-2}	Operation	Comments
0	0	0	+0	String of zeroes
0	0	1	+A	A single 1
0	1	0	+2A	Beginning of 1's
0	1	1	+A	Beginning of 1's
1	0	0	-2A	End of 1's
1	0	1	-A	End of 1's
1	1	0	-A	A single 0
1	1	1	+0	String of 0's

The modified Booth's algorithm (radix-4 recoding) starts by appending a zero to the right of X 2(multiplier LSB). Triplets are taken beginning at position X –1 and continuing to the MSB with one bit overlapping between adjacent triplets. If the number of bits in X (excluding x –1) is odd, the sign (MSB) is extended one position to ensure that the last triplet contains 3 bits. In every step we will get a signed digit that will multiply the multiplicand to generate a partial product entering the reduction tree. The meaning of each triplet can be seen in figure 3. Fig 4 shows array of partial products for signed number.

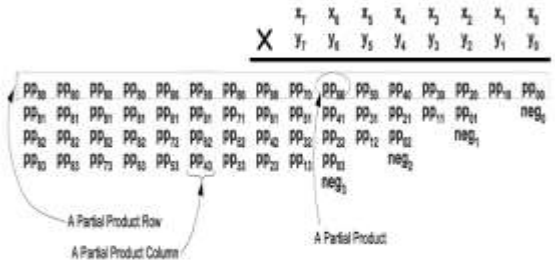

Figure.4 The array of partial products for signed multiplication with MBE

Radix-8 Booth's Algorithm:

Recoding extended to 3 bits at a time - overlapping groups of 4 bits each. Only n/3 partial products generated - multiple 3A needed - more complex basic step. Example: recoding 010(1) yields $y_i\ y_{i-1}\ y_{i-2}$=011. Technique for simplifying generation and accumulation of ±3A exists.
Radix-8 recoding applies the same algorithm as radix-4, but now we take quartets of bits instead of triplets. Consequently, a multiplier based on this radix-8 scheme generates fewer partial products than a radix-4 multiplier, but the computation of each partial product is more complex. In particular, a partial product corresponding to an encoding x=+3 requires the computation of 3x, and therefore a full addition.
Here we have an odd multiple of the multiplicand 3Y, which is not immediately available. To generate it we need to perform this previous add: 2Y+Y=3Y.The multiplication of two binary numbers, 24-bit length, 2s-complement and using the algorithm with radix-8 recoding of the multiplier presents the following features:

- Radix-8 recoding of the multiplier implies a reduction in the number of digits to 8.
- The partial products multiplexer must choose one out of nine possibilities depending on the value of the corresponding signed-digit.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

A		00	01	00	01		+17
X		00	00	10	10		+10
Add A		00	10	00	10		
3 bit shift		00	00	10	00	10	
Add A		00	01	00	01		
	00	01	01	01	01	0	+170

Figure 5 shows example of radix 8 algorithm

Comparison of radix 2, radix 4 and radix 8 algorithm:

• The shortcoming of Radix 2 Booth's algorithm is that it becomes inefficient when there are isolated 1's. For example, 001010101(decimal 85) gets reduced to 01- 11-11-11-1(decimal 85), requiring eight instead of four operations. 001010101(0) recoded as 011111111, requiring 8 instead of 4 operations. This problem can be overcome by using high radix Booth's algorithms.

• As we move towards Radix 8 less number of partial products are generated but more number of operations are required to generate {+1,+2, +3, +4, -1, -2, -3, -4}. In Radix 4 we need to save {+2, -2, +1, -1}.

• Speed of Radix 8 is highest among Radix 2, 4 and 8.

Another point that makes interesting the use of a radix-8 recoding is the less number of transistors resulting in a reduced power dissipation and active area size, compared to radix-4 architecture.

Proposed work

Modified Booth Multiplier:

Booth encoding is a method of reducing the number of partial products required to produce the multiplication result. To achieve high-speed multiplication, algorithms using parallel counters like modified Booth algorithm has been proposed and used. Modified Booth Algorithm (MBA) is most commonly used for high speed multiplication, in which partial product is generated from Multiplicand (X) and Multiplier (Y). This type of fast multiplier operates much faster than an array multiplier for longer operands because it's time to compute is proportional to the logarithm of the word length of operands. The number of partial products is reduced to half, by using the technique of Booth recoding. Reduction in the number of partial products depends upon how many bits are recoded and on the grouping of bits. Multiplication consists of three steps:

1) The first step to generate the partial products; 2) the second step to add the generated partial products until the last two rows are remained; 3) the third step to compute the final multiplication results by adding the last two rows. The modified Booth algorithm reduces the number of partial products by half in the first step.

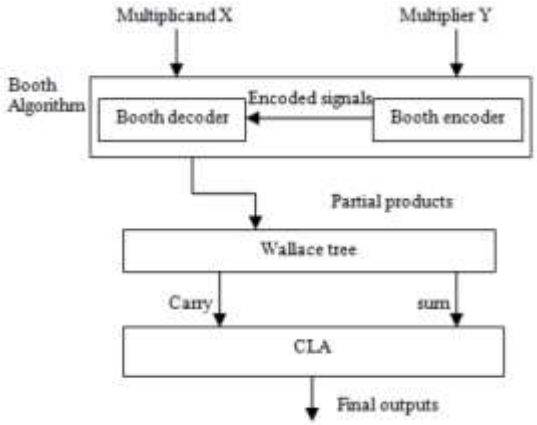

Figure 6: Architecture of the modified Booth multiplier

Reversible Logic Gate:

A reversible logic circuit should have the following features:

· Use minimum number of reversible gates.
· Use minimum number of garbage outputs.
· Use minimum constant inputs

Reversible Gates are circuits in which number of outputs is equal to the number of inputs and there is a one to one correspondence between the vector of inputs and outputs. It not only helps us to determine the outputs from the inputs but also helps us to uniquely recover the inputs from the outputs.

A 4* 4 reversible DKG gate that can work singly as a reversible Full adder and a reversible Full subtractor is shown in Fig 6a. It can be verified that input pattern corresponding to a particular output pattern can be uniquely determined. If input A=0, the proposed gate works as a reversible Full adder, and if input A=1, then it works as a reversible Full subtractor. It has been proved that a reversible full-adder circuit requires at least two garbage outputs to make the output combinations unique.

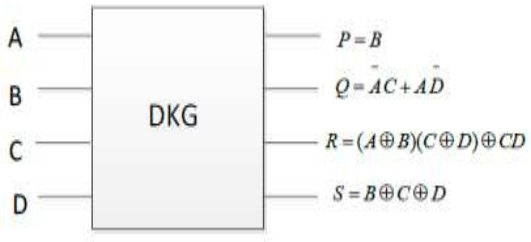

Figure 7: Reversible DKG gate

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

Experimental results

The proposed architecture was simulated and synthesized by using Xilinx tool and ModelSim.

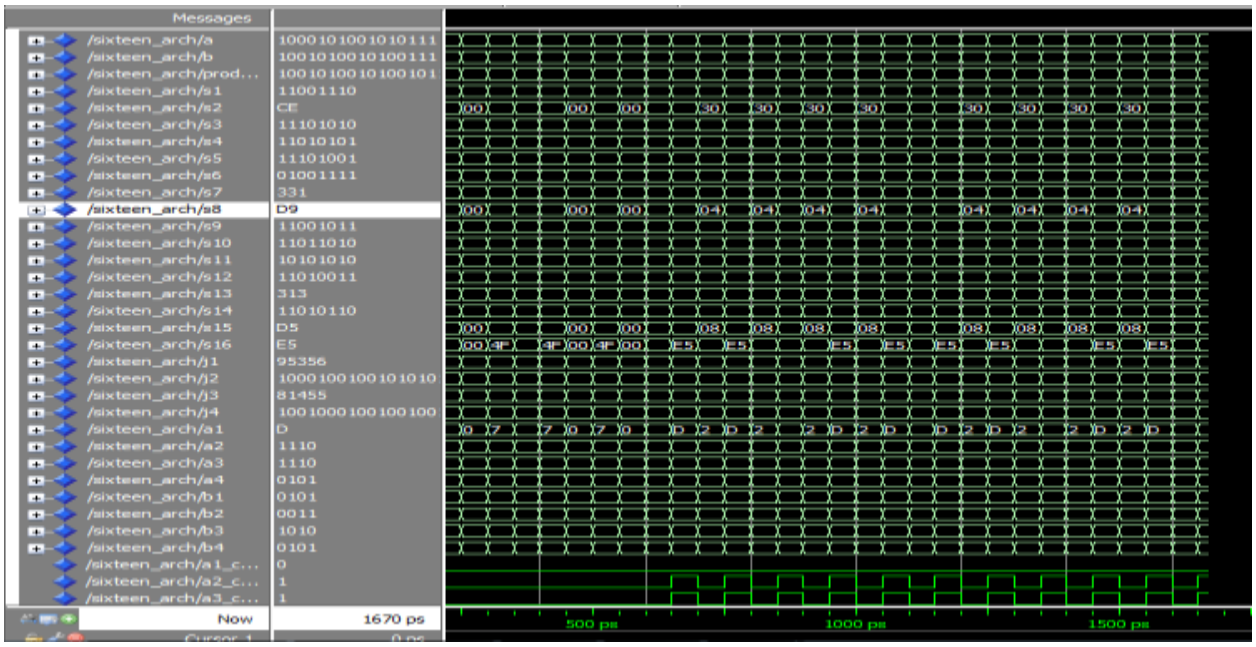

Figure 8: Output of Modified booth algorithm with reversible gate logic

We compared the proposed architecture with that the previous booth multipliers. Because of the difficulties in comparing other factors, only delay is compared. The multipliers were implemented by using Xilinx 8.1 simulator. The delay of ours was 21.18ns while in previous it was 34.35ns and 30.38ns, which means that ours improved about 9.2ns of the delay performance. This improvement is mainly due to the reversible logic gate.

Table 2.Comparison table of different parameters

Parameter	Radix-2	Radix-4	Radix-8
No. of slices	397	71	56
No. of bonded input/output	16	16	16
Delay	34.535ns	30.38ns	21.18ns

Conclusion

Radix-8 Modified Booth Multiplier is implemented here; the complete process of the implementation is giving higher speed of operation. The Speed and Circuit Complexity is compared, Radix-8 Booth Multiplier is giving lesser delay as compared to Radix-2 and Radix-4 Booth Multiplier and Circuit Complexity is also less as compared to these. The architecture include a normal full adders that will consumes maximum delay for that we used is very effective and gives maximum performance. The proposed architecture can be used effectively where we requiring high throughput and speed such as a real-time digital signal processing. In future, by replacing reversible logic gate with other gate or adder further improves the performance.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

References

1. *K.Nagarjun, S.Srinivas, "A New Design of Multiplier using Modified Booth Algorithm and Reversible Gate Logic," International Journal of Computer Applications Technology and Research, Vol.2, Issue.6, pp.743-747, 2013*
2. *Sukhmeet Kaur, Suman and Manpreet Signh Manna, "Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix-2)" Advance in Electronic and Electric Engineering. ISSN 2231-1297, Volume 3, 2013, pp. 683-690.*
3. *Nishat Bano, "VLSI Design of Low Power Booth Multiplier" International Journal of Scientific & Engineering Research, Volume 3, Issue 2, 2012 ISSN 2229-5518.*
4. *K.V.Ganesh, T. Sudha Rani, P.N.Venkateswara Rao, K.Venkatesh," Constructing a low power multiplier using Modified Booth Encoding Algorithm in redundant binary number system" International Journal of Engineering Research and Applications Vol. 2, Issue 3, May-Jun 2012, pp.2734-2740.*
5. *Soojin Kim and Kyeongsoon Cho, "Design of high-speed modified booth multipliers operating at GHz range", World academy of science, Engineering and Technology 61, 2010.*
6. *P. Assady, "A new multiplication algorithm using high speed counters", European journal of scientific research, ISSN 1450-216X Vol.26 No.3 (2009), pp.362-368.*
7. *G.Jaya Prada, N.C. Pant, "Design and Verification of Faster Multiplier", international journal of engineering Research and Applications(IJERA), Vol. 1, Issue 3, pp.683-686.*
8. *Razaidi Hussin, Ali Yeon Md. Shakaff, Norina Idris, Zaliman Sauli, Rizalafande Che Ismail and Afzan Kamarudin, " An Efficient Modified Booth Multiplier Architecture" International Conference on Electronic Design, 2008, Penang, Malaysia.*
9. *C.N. Marimuthu, P. Thangaraj, "Low power high performance multiplier", ICGST-PDCS, Volume 8, Issue 1, December 2008.*
10. *O. L. MacSorley, "High speed arithmetic in binary computers," Proc.IRE, Vol.4, Issue.2, pp. 67–91, February 1999.*