International Journal of Engineering & Technology, 7 (3) (2018) 1189-1192

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i3.12945
Research paper

Analysis and comparison of fast multiplier circuits based on different parameters

Mr. Aaron D'costa [1] *, Dr. Abdul Razak [2], Dr. Shazia Hasan [3]

[1] Electrical And Electronics Dept. BITS Pilani, Dubai Campus. Dubai, United Arab Emirates
[2] Hod. Electrical And Electronics Dept. BITS Pilani, Dubai Campus. Dubai, United Arab Emirates
[3] Faculty Electrical And Electronics Dept. BITS Pilani, Dubai Campus. Dubai, United Arab Emirates
**Corresponding author E-mail: dr.shaziahasan@gmail.com*

Abstract

Digital multiplier circuits are used in computers. A multiplier is an electronic circuit used in digital electronics to multiply two binary numbers. Multiplier circuits are used in ALU for binary multiplication of signed and unsigned numbers. The delay, area and power consumption are the 3 most important design specifications a chip designer has to consider. Delay of the circuit is directly proportional to the delay of a multiplier. Increased delay in the multiplier leads to higher delay in the circuit. Therefore research is carried out as to how to reduce the delay of the multiplier block so as to reduce the delay of whole circuit. The main purpose is to deal with high speed and lower power consumption even after decreasing the silicon area. This makes them well-suited for numerous complex and convenient VLSI circuit implementations. The fact however, remains that area and speed are two contradictory performance restrictions. Hence, increase in speed always results in the use of more and complex hardware. Different arithmetic techniques can be used to implement different multiplier circuits. The focus of this paper is to implement various multiplier circuit and compare them. The timing signals can be observed using software such as Modelsim and Xilinx.

Keywords: *Tree Multiplication; Array Multiplication; Booth's Algorithm; Wallace Tree Algorithm; Vedic Multiplier; Adder Circuits; Area; Delay.*

1. Introduction

Multipliers expect a basic part in many processing applications. It is a block which has a high time delay and significant amount of dissipation of energy. Because of progression in innovation, numerous analysts have attempted and are attempting to outline and execute multipliers which offer rapid, low power utilization, consistency of design and henceforth less hardware or even an amalgamation of all these characteristics in one multiplier therefore making the multiplier circuits reasonable for different fast, low power consumption and conservative VLSI usage. Numerous different types of multiplier circuits have been published during the past few years. The multiplier is one of the most vital piece of hardware block and it is utilized in high performance systems such as 'digital signal processors' and 'microprocessors'. Multiplicative operations are used more often as compared to that of addition subtraction and convolution. With the recent development in technology, many researchers have worked on the design of more efficient multipliers. Many real time and image processing applications today require high speed processors [1].Thus speedier math operations are required. In such sort of uses barring all different arithmetic operations, multiplication is the most important operation and henceforth improvement of multiplier circuit has been a zone of research over decades [1]. One way to implement a faster multiplier circuit could be by using different multiplier algorithms, the other way can be by using different types of faster adder circuits as most of the multiplying operations are based on repetitive additions

2. Booth's algorithm

2.1. Description

The Booth calculation manages operations such as shifting, addition and subtraction and is exceptionally famous in managing the multiplication of signed binary numbers [3]. In many applications, keeping in mind the end goal is to decrease the intricacy of the 'binary coded-digit products' while preforming arithmetic operations, the radix (base) of r is set to 2. With the expansion in r value, the complexity of partial products expands which leads to a noteworthy increment in the speed and cost required for equipment. In the conventional Booth encoding as the Radix-2, its expression of partial product: $PP=A\times B=A\times (\sum (-Xi+Xi-1)) \times 2i)$, the bits of multiplication B is n, so that n-bit partial products are produced [4].

2.2. Working

The "booth's algorithm" implemented in this paper consists of different arithmetic functions such as shifting adding and subtracting of numbers. An arithmetic Right shift operation is used for shifting the bits.
'M' is the multiplicand and 'Q' is the multiplier. 'A' is the accumulator and is initially given the value 0.
The conditions for booth algorithm are as follows:

Q0	Q-1	Operation
0	0	Shifting
1	1	Shifting

Copyright © 2018 Mr. Aaron D'costa et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

0	1	A+M A; Then shift
1	0	A-M A (2's complement); Then shift

The size of the multiplier and accumulator and the no. of steps are decided by the no of bits to be multiplied for ex. If a 4x4 bit multiplication process is to be performed the size of the multiplier is '4+1'=5 bits in size and the accumulator size is also decided as 5 bits. The multiplication will end in 4 steps. The size of M is 4 bits. The last two bits (LSB) decide the operation to be carried out. In the first step the Q-1 bit is set to 0.

Let us consider a 4x4 bit multiplication process

Ex: M=0111 (7)

Q=0011 (3)

Product =10101 (21)

A=0000

Step 1:

A Q Q-1

0000 0011 0

Since the LSB bits are '10', Subtraction of A-M is to be carried out and the result is to be stored in A. Subtraction is done using 2's complement.

A=0000

M=0111

M (2's complement) =1000+1=1001 .This value is stored in the accumulator.

A Q Q-1

1001 0011 0

1100 1001 1 … (Shifting)

Step 2:

Since the last two bits are '11', only shifting is to be carried out.

A Q Q-1

1100 1001 1

1110 0100 1

Step 3:

The last two bits are '01'.A+M is to be performed and the result is to be stored in A.

A=1110

M=0111

A+M=10101 (If there's a carry, it has to be discarded, since its 2's complement)

A Q Q-1

0101 0100 1

0010 1010 0 … (Shifting)

Step 4:

The last two bits are '00' and only shifting operation is performed.

A Q Q-1

0010 1010 1

0001 0101 0

The final value is 00010101 (21) and the Q-1 bit is ignored.

3. Wallace tree algorithm

3.1. Description

Implementing a tree multiplier is an efficient way of reducing the delay. Unlike array multiplication tree multiplier performs parallel addition operation unlike the shifting and adding operation as used in serial multiplier. One major drawback of a tree multiplier is increase in the area. This can be overcome by using the Wallace tree algorithm in which the adders are placed in an efficient way to reduce the delay as well as the area [5]. Wallace tree algorithm is an efficient hardware implementation of a digital circuit used to multiply two numbers. "Each partial product is generated by the multiplication of the multiplicand with one multiplier bit" [6]. The partial products generated are shifted and then added.

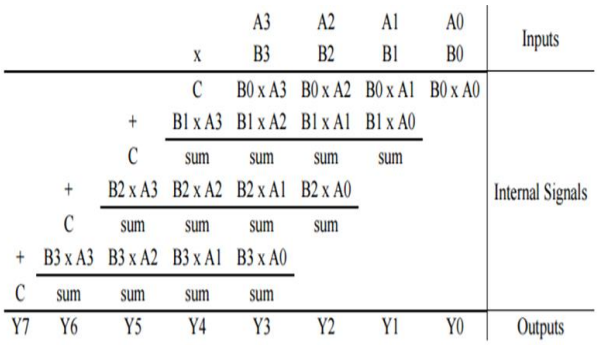


Fig. 1: Wallace Tree Multiplication.

3.2. Working

Wallace tree algorithm consists of three basic steps namely:

1) And-ing of two numbers (multiplication) to generate partial products.
2) Grouping of these partial products
3) Shifting and Adding of these partial products.

The logic implemented in the circuit demonstrated in this paper will be shown through an example .Let us consider an example of multiplying two 4 bit numbers.

Ex:

A=0110(6), B=1011(11)

Step 1:

0 1 1 0

```
        0 1 1 0
X       1 0 1 1
      -----------
        0 1 1 0     ---> P1
      0 1 1 1 0     ---> P2
    0 0 0 0 0 0     ---> P3
  0 1 1 0 0 0 0     ---> P4
```

Step 2: Z1= P1 + P2

```
  1 1
      0 1 1 0
+   0 1 1 0 0
    ---------
    1 0 0 1 0   = Z1
```

Step 3: Z2 = Z1 + P3

```
    1 0 0 1 0
+ 0 0 0 0 0 0
  -----------
  0 1 0 0 1 0 = Z2
```

Step 4: Z3= Z2 + P4

```
  1 1
      0 1 0 0 1 0
+ 0 1 1 0 0 0 0
  -------------
  1 0 0 0 0 1 0   = Z3 = 66
```

4. Vedic multiplication

4.1. Description

Vedic mathematics can be used to greatly reduce the delay area and thereby reducing the power consumption in the multiplier block reducing the delay of the entire circuit. There are a total of 13 up sutras and 14 down sutras in Vedic mathematics. This paper will be focusing on the Urdhva tiryagbhyam Sutra.

Urdhva Tiryagbhyam Sutra is utilized to process the partial products. For a 2x2 bit binary multiplication, the first step is to multiply the vertical parts of LSB of both the multiplicands. The second step is to multiply the bits of the multiplicands diagonally and the addition of the partial products generated from this. The final step is to multiply the MSB bit of the multiplicands vertically and addition with the carry produced in the second step.

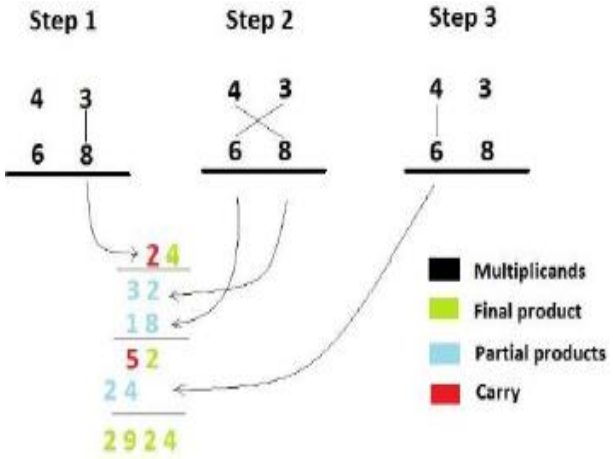


Fig. 2: Vedic Multiplication.

4.2. Hardware implementation of a 2-bit vedic multiplier

In a multiplier of 2x2 bit, the multiplicand comprises of 2 bits and the consequence of the multiplication is 4 bits. The range of input data vary from 00 (0) to 11(3) and output data vary from 0000(0), 0001(1), 0010(2), 0011(3), 0100(4), 0110(6) and 1001(9). Both the multiplicands a, b are taken and the initial step of the procedure is multiplication of vertical (LSB) bits of each of the multiplicands. The second step is criss cross or diagonal multiplication of the two bits of the multiplicands and addition of partial products produced in this step. The third step does the vertical multiplication of MSB of the multiplicands and is then added with the carry created in the second step [2].

As shown below in the figure 4 AND gates and 2 half adders are used for the multiplication of a 2x2 bit binary number rather than using 2 half adders and 1 full adder in normal multiplier circuits . In this way the hardware is reduced which reduces the area and the time delay is also reduced. As the number of bits increases, a considerable amount of change in delay can be observed.

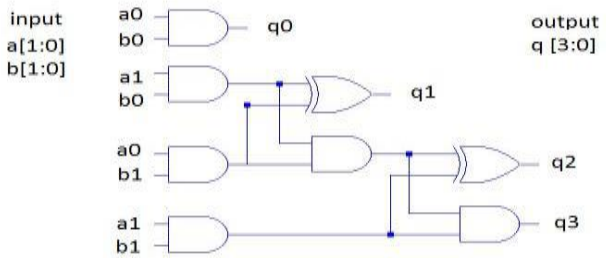


Fig. 3: Hardware Used for 2x2 Vedic Multiplication.

4.3. Implementation of a 4x4 bit multiplier

4 bit multiplication can be also implemented in a similar way. The bits are grouped together in groups of two and then multiplied as shown in the 2x2 bit multiplication. Instead of performing serial addition, the addition tree has been modified to as Wallace tree thereby reducing the levels of addition to two instead of three levels. Two lower bits of q0 passes directly to output whereas the upper bits of q0 are fed into the addition tree [2].

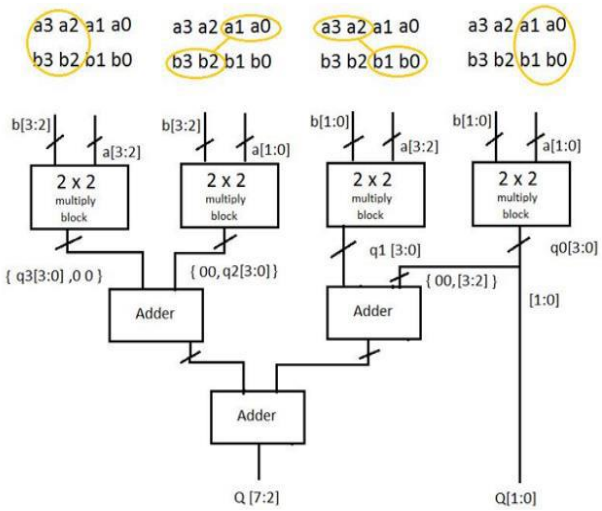


Fig. 4: Hardware Implementation 4x4 Vedic Multiplication.

This simultaneous generation and addition of partial products reduces the delay. In the same way an 8x8 bit Vedic multiplier can be implemented.

5. Results and discussions

5.1. Booth's algorithm

The time delay of the booths algorithm is 31.401ns and the area used is 16% of the total area available. The delay here is the highest because of the shift and add operation which takes place.

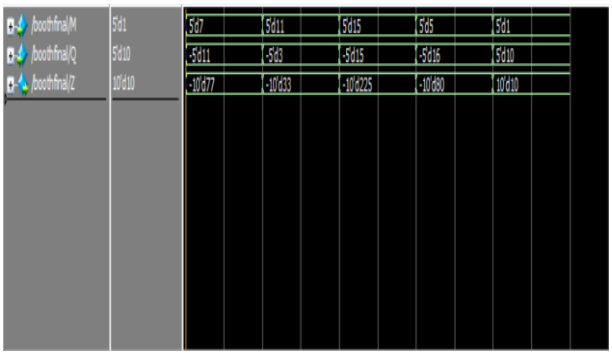


Fig. 5: Simulation Result for Booth's Multiplier.

Simulation of Booth's Multiplier:
Inputs given are Multiplicand M= 00111(5'd7) and Multiplier Q = 10101(-5'd11) . Output is 10110011(-5'd77).

5.2. Wallace tree algorithm

The time delay of a tree algorithm is always less. In case of Wallace tree algorithm it is 17.988ns and the area used is also 12%. The delay can be further reduced by using different types of adders like the carry save adder and the carry look-ahead adder.

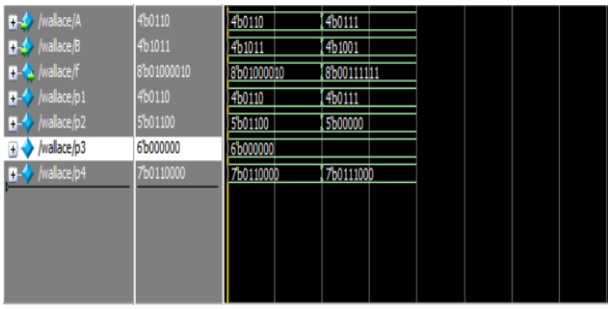


Fig. 6: Simulation Result for Wallace Tree Multiplier.

Simulation of Wallace tree Multiplier:
Inputs given are A= 0110(6) and B = 1011(11). Output is f=01000010(66).

5.3. Multiplier vedic mathematics

The Vedic multiplier makes use of the tree adder approach. So the area consumed is almost the same which is around 12%.This paper shows the implementation of the multiplier using a 4 bit ripple carry adder and a 4 bit carry look ahead adder.

The delay of the 4 bit ripple carry adder is 18.959 which is lesser than the delay of the 4 bit Carry look ahead adder which is 20.423. This is because a ripple carry adder has the least time delay for a 4 bit number. However increase in the bits will cause a drastic change in the delay of a ripple carry adder. So for higher bits it could be advantageous to use a 4 bit CLA.



Fig. 7: Simulation Result for Vedic multiplier.

Simulation of Vedic Multiplier:
Inputs given is y= 1001(-7) and B = 0111(7) . Output is f=11001111(-49).

Table 1: Simulation Results

Algorithm used	Delay	No. of LUT's used	Area
Booth's algorithm	31.401 ns	88	16 %
Wallace tree algorithm	17.988 ns	30	12 %
Vedic multiplier using ripple adder	19.959 ns	36	10%
Vedic multiplier using CLA	20.423 ns	45	12%

6. Conclusion

As shown in the above simulation results the delay in the Wallace tree algorithm is lesser than that of booths algorithm. Also the hardware required is minimum in Wallace tree as compared to that of booth. The delay is less due to the fact that booth algorithm is a serial multiplier in which the steps are processed one by one whereas Wallace tree is a parallel multiplier and more than one step is performed at a time.

In terms of area Wallace tree utilizes less area as compared to that of booth's algorithm compared to other tree multipliers. However booth's algorithm can be used for both positive and negative numbers and there is a regularity in the process which is not found in Wallace tree since the number of half adders and full adders may vary in each stage.

Also increasing the number of bits in Wallace tree will increase the hardware drastically. This can be avoided by making use of the Vedic multiplier which provides the best tradeoff between area and delay. Vedic multiplier can also be used for signed and unsigned binary multiplication by adding a 2's complement generator.

The delay of the Vedic multiplier can be reduced by using different types of adders such as carry save adder, carry look ahead adder etc. As shown in the simulation results table, the delay from the Vedic multiplier is quite close to that of Wallace tree however as the number of bits to be multiplied increases the delay of Vedic multiplier will be quite low as that compared to all other multiplier circuits.

Despite the fact that Urdhva Tiryagbhyam Sutra is quick and proficient however one reality worth seeing is that the 2x2 multiplier is the key block of multiplier of 4x4 bit multiplier. This prompts that a substantial number of partial products will be generated and obviously, extensive fan-out for input signals. To handle this issue, a 4x4 multiplier can be framed utilizing different calculations utilized for quicker multiplication possible.

References

[1] K. D. P. S. a. B. R. Pichhode, "FPGA implementation of vedic multiplier," in *Internation conference on Information Processing(ICIP)*, 2015.

[2] S. V. R. Raju, "Design and implementation of low power and high performance vedic multiplier," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, 2016. https://doi.org/10.1109/ICCSP.2016.7754210.

[3] Booth, "A signed binary multiplication technique," *Modern Electronics Techniques*, vol. 4, no. E98-C, 1951.

[4] L. S. J. W. J. X. Chendong Liang, "An innovative Booth algorithm," in 2016 IEEE *Advanced information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, 2016.

[5] C. Wallace, "A suggestion for a fast multiplier," *IEEE Transaction on Electronic Computers*, vol. 4, no. 13, pp. 14-17.

[6] B. V. V. P. K. a. M. K. Dinesh, "Comparison of regular and tree based multiplier architectures with odified booth encoding for 4 bits on layout level using 45 nm technology," in *2014 International Conference on Green Computing Communication and Electrical Engineering(ICGCCEE)*, 2014.

[7] 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.

[8] Mohammed Hasmat Ali, Anil Kumar Sahani "Study, Implementation and Comparison of Different Multipliers based on Array, KCM and Vedic Mathematics Using EDA Tools" , International Journal of Scientific and Research Publications, Volume 3 , Issue 6 , June 2013.

[9] D. W. Neil Harris, CMOS VLSI DESIGN: A Circuits and Systems Perspective 4th edition, McGraw-Hill.