

Performance evaluation of 64-Bit MAC Units using Different Multipliers In Terms of Area, Power Dissipation and Delay

Kunal Kumar¹, Prasantha R. Mudimela^{2*}

¹Department of Electrical Engineering and Graduate Institute of Photonics and Optoelectronics, National Taiwan University, Taipei, 10607, Taiwan

²School of Electronics and Electrical Engineering, Lovely Professional University, Phagwara, Punjab, India

*** prasantha.22708@lpu.co.in**

ABSTRACT

64-bit multiplier-and-accumulator unit is designed using different type of multipliers such as Array Multiplier, Wallace-tree Multiplier, Booth Multiplier and Vedic Multiplier. Implementation of design is programmed using Verilog coding and design synthesis is performed using Encounter RTL compiler-Cadence. Evaluating the performance of various multipliers is performed in terms of area, power dissipation and speed. The best multiplier is one that has high speed capacity, low power consumption and low power dissipation. Once implementation of MAC unit is done using all different multipliers, analysis of area, power and speed is done on Cadence tool. On the comparison of all the above mentioned parameter, a conclusion is made that Vedic multiplier has comparatively good performance among all multipliers.

Keywords: MAC unit, power dissipation, delay, cells

INTRODUCTION

The multiplier-and-accumulator (MAC) unit performs the multiply-accumulate operation which is a common step to get the product of a two number and then adds that product to an accumulator [1]. The MAC unit consists of multiplier which is realized in a combinational logic that is followed by an adder and an accumulator, where result is saved. The multiplication and addition of two binary numbers are most frequently used in data-processing applications, digital signal processors and specifically in integrated circuit. In multiplier-accumulator unit multiplication and addition forms the main blocks.

High speed and low power MAC unit is required especially in signal processing application such as convolutions, Fourier transforms, FIR filters fast Fourier transforms. These require efficient MAC units to achieve desired performance. Area and speed is also important factor of MAC unit but sometimes increasing speed leads to increase in power consumption, so there is always an upper bound of speed for a given power criteria. So there is a need for designing the MAC unit. In this work 64 bit MAC unit is implemented separately with four different types of multipliers such as array multiplierWallace tree, Booth and Vedic multipliersusing Verilog coding. Design synthesis isexecuted using Cadence Encounter RTL compiler.

ARRAY MULTIPLIER MAC UNIT

Initially 64-bit MAC unit is implemented using array multiplier. In array Multiplier, between the hardware structure and manual multiplication there is one to one topological correspondence. It requires $A \times B$ two bit AND gates for the generation of A partial products. For the addition of X partial product, most of the area of multiplier is dedicated to it which needs $(A-1) B$ bit adders [2]. Partial product shifting is necessary as shown in Figure 1 which indicates the 4×4 array multiplication process. As shift of partial product takes place in every step, this multiplier exhibits consistent structure.

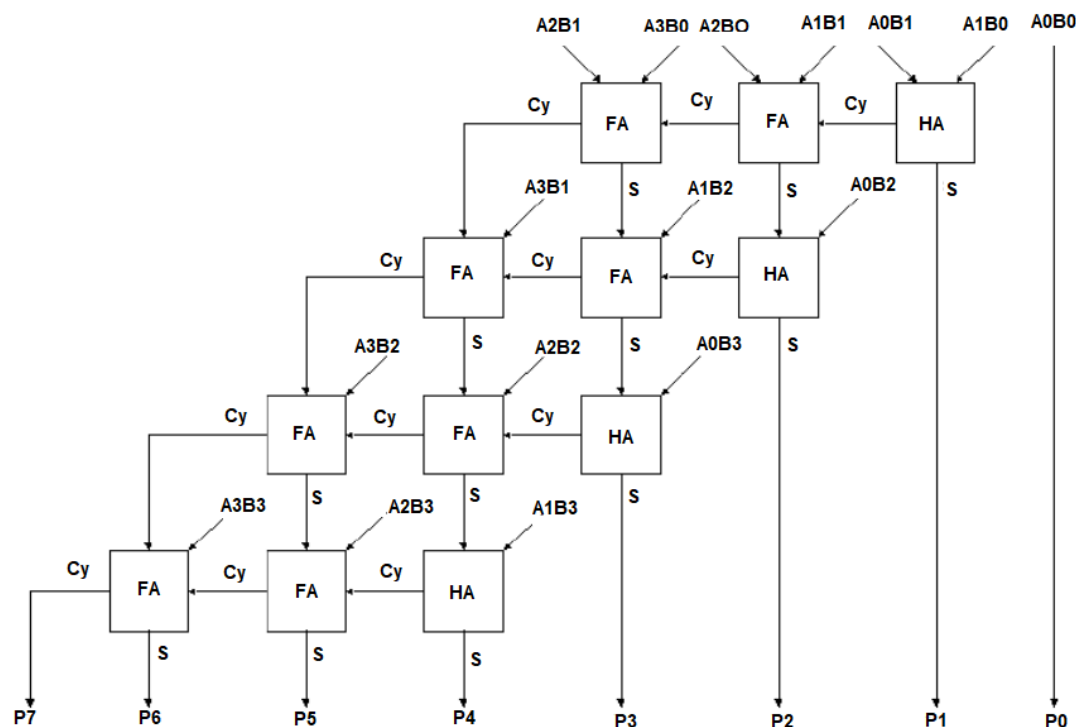


Figure 1. Block diagram of array multiplier for 4 bit numbers

In simple term it can be said that every partial product is added using adder and the sum from each adder block is taken as final product and carry is transferred to the following block as an input. For 32×32 array multiplier required 1024 AND gates, 960 full adders and 32 half adders. This structure needs to be implemented in the Verilog code so that to obtain desired output. An adder used over here can be of any type depending on the need of applications.

In order to verify code and check whether the implementation is done correctly. The simulation will show the output for each input given and hence the output is verified over here. When inputs are given, the multiplied value will be displayed and when next inputs are encountered, the sum of its multiplication and previous product will be displayed. Hence different values can be given and checked according to the application required. The simulation results of 64-bit MAC unit which is designed using array multiplier are shown in Figure 2.

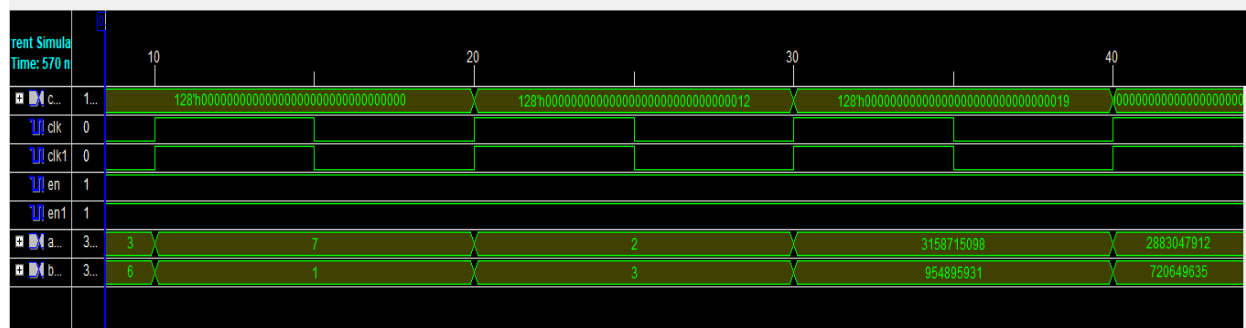


Figure 2: The simulation of 64-bit MAC unit using array multiplier.

WALLACE-TREE MULTIPLIER MAC UNIT

By rearranging the partial-sum adders in a tree like fashion, the requirement of adder cells and the critical path are reduced. This tree like structure is known as Wallace-tree multiplier. It is important to have substantial hardware therefore tree multipliers help in savings from larger multiplier. It also helps in reducing the propagation delay. As Wallace-tree Multiplier help from using large multipliers but it also has the drawback of being irregular as a result it obscures the completion of well-organized layout structure.

In Wallace-tree architecture, multiplication of two given numbers are performed by reduction of partial products into 2×2 matrix. This can be done by the usage of half adder, full adder carry save adders. A set of counters are used in parallel to add partial products in each column without any need of propagation of carriers. Another set of counters is used to reduce this new matrix and these goes on until the two rows matrix is produced. In this method full adders and half adders are used for addition of 3 bit and 2 bit respectively. Due to prior addition of partial products, the speed of Wallace-tree multiplier is very high [3].

Figure 3 illustrates the typical diagram of 8 bit Wallace-tree multiplier. In this multiplier partial products are added in Wallace block. The results from addition are the final product bits, sum and carry bits which are summed in the final adder. In 8×8 multiplication process four stages are needed to implement the reduction is done in four stages, each with delay of full adder [4]. Following the above architecture, 32-bit Wallace tree multiplier can be implemented. Finally 64-bit MAC unit is implemented using 32-bit Wallace multiplier and the obtained simulations are shown in Figure 4.

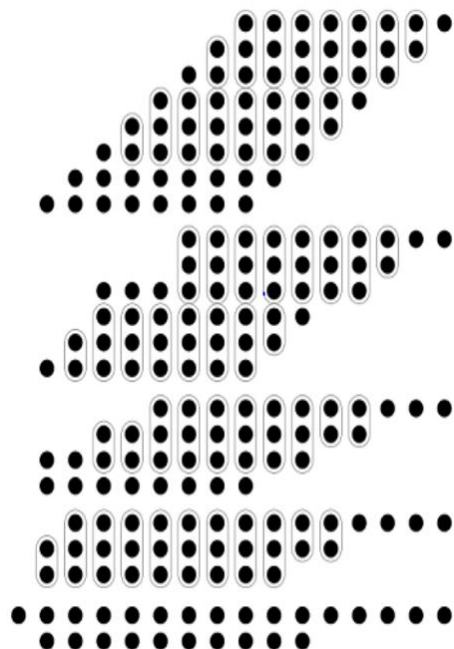


Figure 3.8-bit Wallace tree multiplication process [5-6]

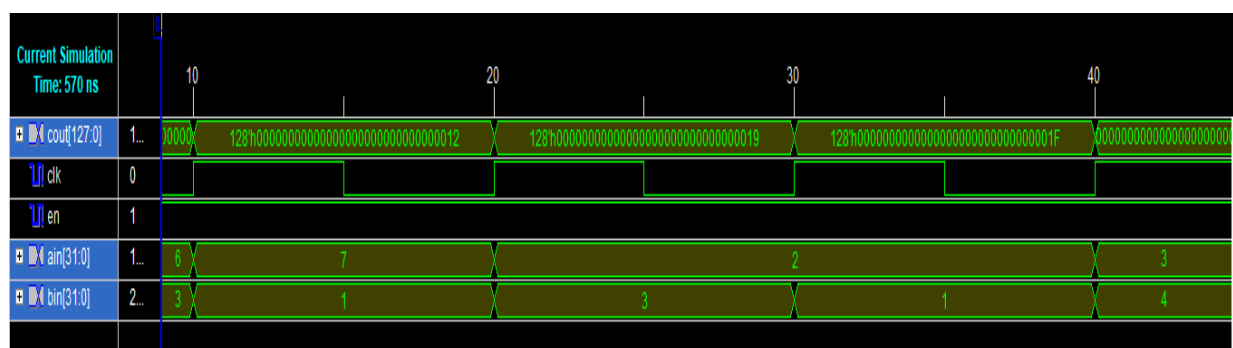


Figure 4. The simulation of 64-bit MAC unit using Wallace tree multiplier.

BOOTH MULTIPLIER MAC UNIT

Booth Multiplier algorithm is a means of achieving multiplication by addition, subtraction and shifting. In this method involves multiplication of two signed binary numbers in 2's complement notation. Booth multiplier need 4 registers which is named as Accumulator (A), Multiplier (Q), Initial bit (Q-1) and Multiplicand (M). The accumulator is cleared so that it is empty with value zero. Multiplier register stores the second number, from which we want to multiply the multiplicand. Initial bit register can hold only one bit at a time. Multiplicand register store the first number [7]. In this multiplier both positive and negative numbers are equally treated.

It involves add-shift operation, every multiplier bit produces one multiple of the multiplicand that can be summed in partial product. For very large multipliers large number of multiplicands are required. Delay of Booth multiplier depends on the number of additions performed. The performance of booth multiplier can be improved if a way is found to minimize the number of additions performed.

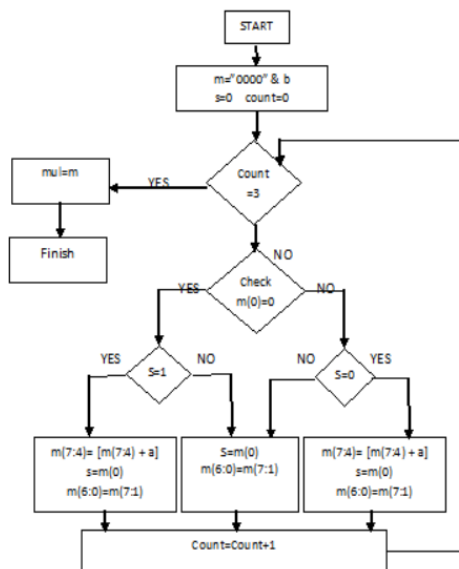


Figure 5. Booth algorithm which are used for the multiplication of two binary numbers [8]

Figure 5 shows the algorithm of Booth multiplier which needs to be followed to multiply two binary numbers. Four registers are used in this algorithm. Accumulator (A) which is 0 as the register is cleared, Initial bit (Q-1) it is also 0 initially, Multiplier register (Q) it stores the number from which we need to multiply the multiplicand, Multiplicand (M) it stores the number with which need to multiply. In the flow chart after the value are stored in register, the comparison of initial bit register and the LSB bit of multiplier is done and accordingly operations is performed in each step. It also has a Count register (C) which stores the count level. When count becomes 0, the algorithm stop and the final output is generated.

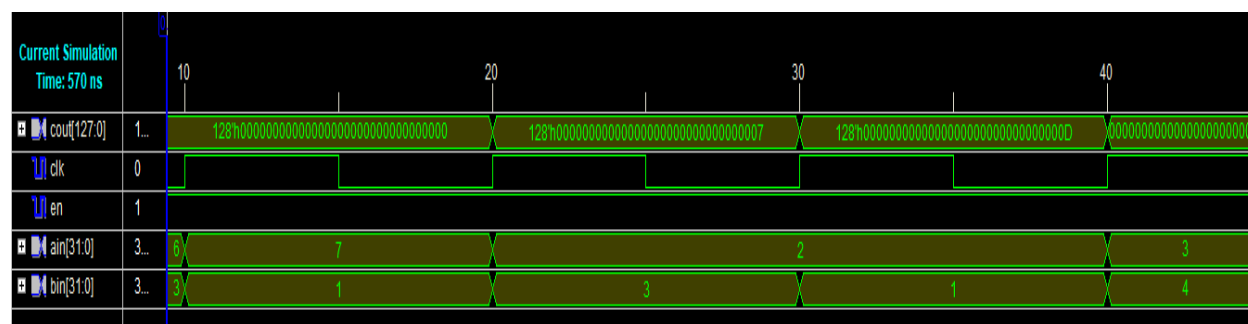
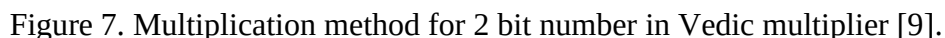


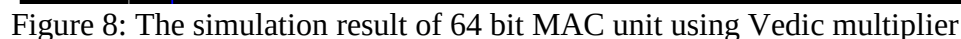
Figure 6. The simulation of 64-bit MAC unit using Booth multiplier

VEDIC MULTIPLIER MAC UNIT

To get high performance and high speed ample number of modification is done on standard Booth multiplier, Wallace-tree Multiplier and other multipliers to make the design and also people are working on several new techniques. Among all these, Vedic Multiplier which is based on Vedic Mathematics is dominating and is under focus because of its high speed and is considered as fastest and also due to low power multiplier. “UrdhvaTiryakbhyam” is considered the most efficient one among sixteen sutras in Vedic multiplication because of its speed.



For designing higher order Vedic multiplier, directly designing of 32 bit is not possible. Initially 2 bit multiplier is designed. For 4 bit Vedic multiplier, 2 bit multiplier is called inside the main module of 4 bit Vedic multiplier. For 8 bit multiplier, 4 bit multiplier is called inside the main module of 8 bit multiplier and so on the process goes. For implementation of 64-bit MAC unit, two 32 bit number input are given and four 16 bit Vedic multipliers are called in order to implement 32 bit Vedic multiplier. For adding the partial products 32 bit adders are used to get the final products. The simulation results of 64-bit MAC unit using Vedic multiplier is show in Figure 8.



Test synthesis is performed using Encounter RTL compiler-Cadence once after completion of implementation part. In synthesis part area, power and delay of the different multiplier is calculate for 45nm technology node, which represents the minimum feature size of the transistor, to do synthesis in slow and fast mode. The smaller the size is the larger the number of transistor can be fabricated on the chip. It can also be concluded that this number represent the minimum channel length that can be used in fabrication. Using Encounter RTL compiler-Cadence tool power, area and delay are evaluated.

Area	Array Multiplier MAC unit		Wallace-tree Multiplier MAC unit		Booth Multiplier MAC unit		Vedic Multiplier MAC unit	
(Cells)	Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
Sequential	320	320	256	256	319	319	256	256
Inverter	146	190	140	190	7785	7783	141	190
Logic	2356	2146	2394	2145	3603	3419	3785	3586
Total	2822	2656	2790	2591	11707	11521	4182	4032

Table 1: Comparison of multiplier with respect to cells

The comparison of MAC unit on the basis of area is shown in the table 1. This result is of 45 nm technologies using slow as well as fast mode. In Wallace-tree multiplier MAC unit in slow mode the total cells used are 2790 and in case of fast mode the total cells used are 2591. MAC unit total cells used in both modes are quite less in comparison to Booth and Vedic multiplier MAC units. The total number of cells required in Array multiplier MAC unit is very close to number of cell required in Wallace-tree multiplier MAC unit. Booth multiplier MAC unit requires more area among all other MAC units studied in this work as it requires more number of cells both in fast and slow modes. Vedic multiplier requires 4182 cells in slow mode and 4032 cells in fast mode.

	Array MultiplierMAC unit		Wallace-tree MultiplierMAC Unit		Booth Multiplier MAC unit		Vedic MultiplierMAC unit	
Power	Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
Dynamic	1707.0608	2355.0662	2302.1775	2774.5017	6145.4319	9392.6991	2016.5378	2407.3949
Leakage	0.52662	1.52474	0.48540	1.39218	1.32220	3.91203	0.72320	2.24194

Table 2. Comparing all multiplier with respect to Power (μ W)

	Array Multiplier MAC Unit		Wallace-tree Multiplier MAC		Booth Multiplier MAC Unit		Vedic Multiplier MAC Unit	
Delay	Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
Worst path	8053.20	6550.90	8047.90	6476.60	7091.80	6227.00	8345.30	6424.80

Table 3: Comparison of different multipliers in term of Delay (ps)

Table 2 depicts the comparison of all the multipliers on the basis of power dissipation. By comparing all MAC units, the Array multiplier MAC unit exhibits less dynamic power dissipation than other MAC units. It is worth noting that in fast mode the Vedic multiplier MAC unit has slightly more dynamic power in comparison with array multiplier MAC unit. Wallace tree multiplier and array multiplier have less leakage power among all other multipliers. Vedic multiplier MAC has leakage power of 2.24194 μ W which is lower than Booth multiplier MAC unit. In the category of dynamic power, array and Vedic multiplier exhibits good performance.

The reason for higher leakage power. This is due to number cells of Vedic multiplier is more i.e., 4032 in compare with array and Wallace tree multiplier MAC units. Booth multiplier shows large dynamic and leakage power dissipation as it has more than the other MAC units.

The most important part of this work is the time taken for the worst path. Table 3 shows worst path delay of the MAC units. Booth multiplier MAC unit has lowest worst delay of 7091.80ps among MAC units. Vedic multiplier MAC unit less worst delay in comparison with array and Wallace-tree MAC units. But at the same time Vedic MAC unit has highest worst path delay in slow mode than other MAC units.

CONCLUSION

Booth multiplier MAC unit exhibits less worst path delay among all multipliers. But Booth multiplier MAC unit has large power dissipation other MAC units. Wallace tree multiplier MAC unit requires less number of cells, less power dissipation and reasonably low worst path delay. But it has irregular structure and making layout is difficult in realizing its structure. So, Wallace tree multiplier MAC unit is not feasible to implement practically. Array multiplier MAC unit and Vedic multiplier MAC unit are topmost contenders. Vedic MAC unit requires more number of cells than array multiplier. Vedic MAC unit has less dynamic power dissipation which could be useful in low power applications.

REFERENCES

- [1] A. Johnson, B. Manohar, A.P. Mathew, "Modified MAC based FIR Filter Using Carry Select Adders," International Journal of Engineering Science and Innovative Technology, vol. 4 (3), May 2015.
- [2] A.P. Ramesh, A.V.N. Tilak, A.M. Prasad, "Efficient Implementation of 16-Bit Multiplier-Accumulator Using Radix-2 Modified Booth Algorithm and SPST Adder Using Verilog," International Journal of VLSI design & Communication Systems vol. 3 (3), pp. 107-118, Jun., 2012.
- [3] P. Ramanathan, P. Kowsalya, P. Anitha, "Modified low power Wallace Tree multiplier using higher order compressors," International Journal of Electronics Letters, vol. 5 (2), pp. 177-188, 2017.
- [4] B. Pratap Singh, R. Kumar, "Design & Implementation 8-Bit Wallace Tree Multiplier," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol. 5 (4), pp.2307-2312, Apr., 2016.
- [5] C.S. Wallace, "A Suggestion for fast multipliers," IEEE Trans. Electronics Computer., vol. EC-13, pp. 14-17, Feb. 1964.
- [6] H. Bansal, K. G. Sharma, T. Sharma, "Wallace Tree Multiplier Designs: A Performance Comparison Review," Innovative Systems Design and Engineering, vol.5 (5), pp. 60-67, 2014.
- [7] S-Rong Kuang, J-P Wang, "Design of Power-Efficient Configurable Booth Multiplier," IEEE Transactions on Circuits and Systems-I: Regular Papers, vol. 57 (3), pp. 568-580, Mar. 2010.

- [8] R.S.Misra, P. Gour, B.B. Soni, “Design and Implements of Booth and Robertson’s Multipliers Algorithm on FPGA,” vol. 1 (3), pp.905-910, 2011.
- [9] L.S. Jie, S.H. Ruslan, “A 2x2 bit Vedic Multiplier with Different Adders in 90nm CMOS Technology”, AIP Conference Proceedings 1883, 020017, pp.1-7, Sep. 2017.