

Design and Development High-Performance Multiplier

Pavan Jadhav^{1*}, K. B. Ramesh²

¹Student, Department of Electronics and Instrumentation,
R.V. College of Engineering, Bengaluru, Karnataka, India

²Associate Professor, Department of Electronics and Instrumentation,
R.V. College of Engineering, Bengaluru, Karnataka, India

***Corresponding Author**

E-Mail Id: pavanjadhav.ei20@rvce.edu.in

ABSTRACT

The performance of multiplication in terms of speed and power is crucial for many Digital Signal methodology (DSP) applications. Many researchers have returned up with various multipliers like Associate in nursing associate array, Booth, carry-save, Wallace tree, and altered Booth multipliers. The amount is to boot one of the foremost necessary blocks of the various VLSI applications. So, it's required to vogue a superior range to boost up the performance of those circuits and systems. Inside the applying of the digital signal, methodology multipliers play a big role. With advances in technology, several researchers have tried and tried to vogue multipliers that provide high speed, low power consumption, regularity of layout, and so less space or perhaps the mixture of them in one multiplier factor. During this paper, the look of multipliers that's a smaller quantity advanced and power-consuming is made of basic electronic components like gates and adders. This vogue lowers the standard of the circuit and works on the essential principle of multiplication and an awfully few types of transistors. The results show the multiplier factor isn't sophisticated and works in massive multiplications applications.

Keyword: Half adder, full adder, Quine McCluskey method, k-map, and propagation delay

INTRODUCTION

Multipliers play a very important role in several DSP applications like convolution, quick Fourier rework (FFT), Discrete circular function rework (DCT), and filtering. The speed of the DSPs for the most part depends on the number block. This in turn will increase the demand for top speed multipliers.

Over the past few years, several researchers have developed numerous multipliers exploitation many algorithms like an array, Booth, carry-save, Wallace tree, and changed Booth algorithms. Several architectures even have been planned to support these algorithms that embrace parallel, serial and serial-parallel multipliers. In associate degree array

numbers, multiplication relies on shift and add. The partial product is generated by the multiplication of the number with one number bit. The partial product is shifted consistently with their bit orders then additional with carrying propagate adder. Array number is simple to styledue to its regular structure.

However, the disadvantage of the array number is that because the breadth of the number increases the delay becomes additional. This can result in the worst-case delay of the number proportional to the breadth of the multiplier. The delay of the number is additionally reduced by the arrangement of adders, exploitation Carry Save Array (CSA) methodology, and a Wallace tree adder methodology. In CSA

methodology, add signal is passed to the adders of the following stage. The final product is obtained during a final adder by exploitation quick adder (Carry Lookahead Adder). The CSA number has a regular structure to style however, there's some issue in achieving high speed. Different approaches and architectures are projected to design a numbered circuit to boost the performance [8-9].

Wallace tree is one of all the oldest and common techniques used to style number circuits and such multipliers area unit known as Wallace number [10-12]. The recognition of the architecture of Wallace's number is because of the high speed of operation. In standard Wallace number, partial products area unit generated, then the partial merchandise area unit processed and eventually, final addition is performed.

With the help of Wallace tree methodology, the partial product bits square measure summed up in parallel by means that of a tree of carry-save adders. the most advantage of this methodology is that it achieves high-speed operation. However, in Wallace's tree methodology the circuit layout is complicated and has irregular wires. To beat the disadvantage concerning the speed of the same algorithms, [1] given new number style approaches supported religious text arithmetic. The partial merchandise square measure calculated ahead then additionally based on the religious text maths approach to get the ultimate product. Furthermore to boost the speed of number, in [2]– [6] authors have given new styles, wherever the partial products square measure additional by exploitation ripple carry adder, carry-save adder, and carry-lookahead added.

In [7], a mechanical device primarily based on Vedic number (CVM) has been designed, to additional enhance the speed

by exchanging the full adders and half adders with compressors. However, in today's applications, one among the foremost challenges for superior DSP applications is that the power dissipation, each static and dynamic. Therefore, there is a need to notice the associate degree optimum between speed and power, instead of targeting them severally. This can be portrayed by the average energy dissipated for one change event and it's known as a power-delay product.

Perfection computation is the key to all computations. Rounding off a number isn't an option as it affects the accuracy of the computations and when it comes to large computations rounding a number can't be permitted as it affects the accuracy of the computation which further leads to a chain effect of small crimes in computation. To avoid this we need perfect circuits which can give us the exact results of the operation to be performed.

The addition is an important computation operation that is performed extensively to save the CPU from doing repetitious addition where addition is needed. But for this, we need to design a high-performance perfection multiplier that would give the exact value of the figures multiplied.

Designing of smaller multipliers with fewer range inputs and outputs will be simply done with exploitation reduction techniques like Quine McCluskey methodology and k-map reduction, however, once it comes to a better range of inputs say a 16-bit binary input or 32-bit binary input it'd result in massive k-maps or extended tables which would not be possible for getting the Boolean equation indeed if we get it some way the equation would be so high and complex that the fewest error will make the circuit give wrong Outputs.

To overcome this complexness we will use a distinct methodology in which we tend to use adders and AND gates to style a number that works on the essential methodology of multiplication that would result in a way less advanced circuit while not moving the accuracy of the number.

PROPOSED METHODOLOGY

The principle this works is that the same of the technique by which we tend to multiply 2 numbers i.e. take one bit from one input and multiply the second input bit then take the second bit from 1st input and multiply the second input by it and shift the solution left by one bit then until we tend to get the results of the multiplication of last bit of 1st input and second input then add up all the numbers.

This principle is tedious for little bit multiplications however because the range of input bits will increase 16-bit or 32-bit this is often the simplest suited principle because it is a smaller amount longer than account the mathematical equation for all the outputs and as we all know the biggest

range of bits in outputs of two binary numbers of size n is 2 times the size of n i.e. the output is double of the scale of input.

Thus once it involves multiplication of enormous numbers we are going to confirm mathematical equations that are double the number of inputs and with a sizable amount of time-consuming variables and it gets harder with the increase in input size as an example for an eight-bit multiplier factor we are going to confirm sixteen mathematical equations with sixteen variables.

Thus by exploiting this coming up with the technique we will construct multiplier factor simply by exploitation AND gates and adder circuits without the necessity of mathematical equations. By the employment of AND gates we tend to first multiply 2 bits because the initiative is to multiply the input bits with one another then comes to the role of an adder that adds up the input applied.

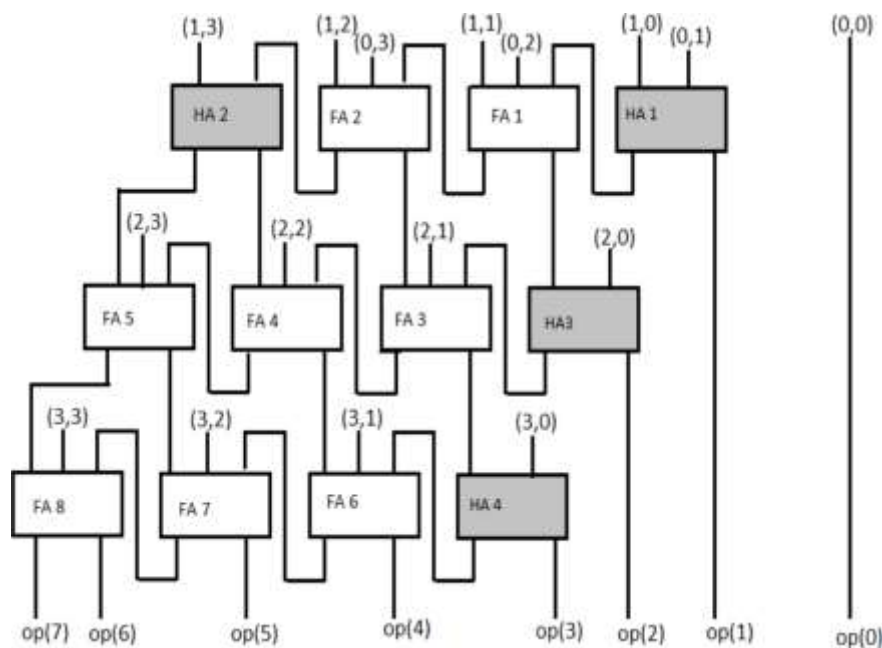


Fig. 1: Block Diagram of an 8-bit Multiplier(11).

Talking about the design of the multiplier. These ideas are referred from the paper by *Prajwal Gaharwar* and *Ayoush Johari*. Designing this multiplier takes AND gates and adders both half and full. The role of AND gate is to take the input bits and multiply them. These gates are used to multiply two bits and that's what we want we have to multiply the first bit of the first input with other bits of the second input individually so the gate is best suited for this operation.

After the first step of individual multiplication, the next step is the addition of the multiplied bits after the addition takes place the output is obtained. The module is designed such that it can be easily understood that on what part which input is applied. The adder modules are in the same number as the number of input bits in each row. For example, If we are making a 4-bit multiplier we place 4 adder circuits in a row such that the output values from AND gates are in order i.e. the first row will only use the individually multiplied values of the first bit and second bit of first input with the bits of the second input.

The second row contains the multiplied values of the third bit of the first input with the bits of the second input and then the third row has the third bit of the first input multiplied with bits of the second input and so on up to the last bit of first input getting multiplied with bits of the second input.

Now talking about the complexity of the design. The circuits made using this principle are not complex irrespective of the size of the input. On increasing the size input the number of adders and AND gates increases. The number of AND gates required in the circuit for multiplication SCIECS 2016 is the product of the size of both inputs. Similarly, the number of adders required is equal to the size of the

first input bus multiplied by one less than the size of the second input bus. For example, if we are making a 4-bit multiplier then the number of AND gates required will be 16 in number and the number of adders required would be 12. Both half and full adders are used in this circuit. Half adders are used at places where only 2-bit addition is required to be done and Full adders are used at places where 3-bit addition is to take place. The number of half and full adders also depends on the input size. if we are multiplying two 4 bit numbers then the number of half adders required would be 4.

Similarly, if we are multiplying 16-bit numbers then the number of half adders would be 16. As the size of the multiplier increases from 4 to 32 bit, the number of adders also increases by a large amount. In a 16 bit multiplier, there are 16 half adders and 240 full adders, and 256 AND gates but its circuit is as simple as that of a 4-bit multiplier. Only the number of input increases and which leads to an increase of other components but the process remains the same i.e. the input bits are multiplied bit by bit using an AND gate and then the results are added up. Apart from being less complex the circuit is easy to design and the area increases less as compared to other multiplier circuits [20].

For example, if we increase the circuit input from 4 to 8 bits the area of the circuit increases by 4 times the area of the 4-bit multiplier. The same principle applies on the transition from 8 bit to 16 bit the area of the circuit increases by 4 times.

EXPECTED RESULT

This result is taken from the paper by *Prajwal Gaharwar* and *Ayoush Johari* paper. The simulation results show that the planning works utterly well in each tiny bit multiplications and enormous

multiplications. 32-bit multiplications may also be performed with a similar exactness and accuracy as 4-bit multiplications. The simulation results of the projected style thereby prove that the planning is capable of activity multiplications. The circuit of the 32-bit number includes a total power consumption of 45.31mW and a delay of 105.3ns excluding the delay of D-flip flops that have a delay of 5.02ns. The key issues featured during this style are the dross values that are obtained throughout the propagation delay.

his is neglected by the utilization of the D-flip flop that is employed to not enable the dross values to suffer it and permit solely the first result to pass. This will be done by initiating a scan cycle once a delay that is comparable to the propagation delay of the planning. The planning is additionally increased by the utilization of adders like carry look-ahead adders that perform addition with less delay as compared to traditional adders. The delay may also be reduced by employing a smaller technology which might modify quicker switch of transistors thereby reducing the delay.

CONCLUSION

We conferred a style of number exploitation adders and gates having the options like less quality, fewer transistors, and high accuracy. With these options, we will say that such multipliers will be used at places wherever high accuracy is expected regardless of the dimensions of multiplication performed which is that the most significant part of calculations.

REFERENCES

1. Akhter, S. (2007, August). VHDL implementation of fast NxN multiplier based on vedic mathematic. In *2007 18th European Conference on Circuit Theory and Design* (pp. 472-475). IEEE.
2. Tiwari, H. D., Gankhuyag, G., Kim, C. M., & Cho, Y. B. (2008, November). Multiplier design based on ancient Indian Vedic Mathematics. In *2008 International SoC Design Conference* (Vol. 2, pp. II-65). IEEE.
3. Ramalatha, M., Dayalan, K. D., Dharani, P., & Priya, S. D. (2009, July). High speed energy efficient ALU design using Vedic multiplication techniques. In *2009 International Conference on Advances in Computational Tools for Engineering Applications* (pp. 600-603). IEEE.
4. Jaina, D., Sethi, K., & Panda, R. (2011, October). Vedic mathematics based multiply accumulate unit. In *2011 International Conference on Computational Intelligence and Communication Networks* (pp. 754-757). IEEE.
5. Kunchigi, V., Kulkarni, L., & Kulkarni, S. (2012, March). High speed and area efficient vedic multiplier. In *2012 International Conference on Devices, Circuits and Systems (ICDCS)* (pp. 360-364). IEEE.
6. Bansal, Y., Madhu, C., & Kaur, P. (2014, March). High speed vedic multiplier designs-A review. In *2014 Recent Advances in Engineering and Computational Sciences (RAECS)* (pp. 1-6). IEEE.
7. Huddar, S. R., Rupanagudi, S. R., Kalpana, M., & Mohan, S. (2013, March). Novel high speed vedic mathematics multiplier using compressors. In *2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s)* (pp. 465-469). IEEE.
8. Kuo, K. C., & Chou, C. W. (2006, December). Low Power Multiplier with Bypassing and Tree Strucuture. In *APCCAS 2006-2006 IEEE Asia Pacific Conference on Circuits and Systems* (pp. 602-605). IEEE.

9. Hwang, Y. T., Lin, J. F., Sheu, M. H., & Sheu, C. J. (2006, December). Low power multiplier designs based on improved column bypassing schemes. In *APCCAS 2006-2006 IEEE Asia Pacific Conference on Circuits and Systems* (pp. 594-597). IEEE.
10. Wen, M. C., Wang, S. J., & Lin, Y. N. (2005, May). Low power parallel multiplier with column bypassing. In *2005 IEEE International Symposium on Circuits and Systems* (pp. 1638-1641). IEEE.
11. Gaharwar, P., & Johari, A. (2016, March). Design and implementation of multipliers. In *2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS)* (pp. 1-4). IEEE.
12. Hussain, I., & Kumar, M. (2017). A Fast and Reduced Complexity Wallace Multiplier. *Journal of Active & Passive Electronic Devices*, 12.
13. Khan, S., Kakde, S., & Suryawanshi, Y. (2013, December). VLSI implementation of reduced complexity wallace multiplier using energy efficient CMOS full adder. In *2013 IEEE international conference on computational intelligence and computing research* (pp. 1-4). IEEE.