International Journal of Innovative Research in Electronics and Communications (IJIREC)
Volume 1, Issue 9, December 2014, PP 9-14
ISSN 2349-4042 (Print) & ISSN 2349-4050 (Online)
www.arcjournals.org

Low Power High Speed Multiplier and Accumulator Based on Radix-4 Booth's Algorithm

J. Nagaraju, K. Sai Sudheer

[1]Assistant Professor, MLR Institute of Technology, Hyderabad, A.P, India,
jnrchmr2k11@gmail.com

[2]Assistant Professor, Usharama College of Engineering and Technology, Telatrolu, Gannavaram, AP, India,
saisudheer_k@yahoo.in

Abstract: *With the recent rapid advances in multimedia and communication systems, real-time signal processing like audio signal processing, video/image processing, or large-capacity data processing are increasingly being demanded. The multiplier and multiplier-and-accumulator (MAC) are the essential elements of the digital signal processing such as filtering, convolution, transformations and Inner products. There are different entities that one would like to optimize when designing a VLSI circuit. These entities can often not be optimized simultaneously, only improve one entity at the expense of one or more others The design of an efficient integrated circuit in terms of power, area, and speed simu ltaneously, has become a very challenging problem. Power dissipation is recognized as a critical parameter in modern the objective of a good multiplier is to provide a physically compact, good speed and low power consuming chip.*

In this paper, we proposed a new architecture of multiplier-and-accumulator (MAC) for high- speed arithmetic. By combining multiplication with accumulation and devising a hybrid type of carry save adder (CSA), the performance was improved. Since the accumulator that has the largest delay in MAC was merged into CSA, the overall performance was elevated. The proposed CSA tree uses 1's-complement-based radix-2 modified Booth's algorithm (MBA) and has the modified array for the sign extension in order to increase the bit density of the operands. The CSA propagates the carries to the least significant bits of the partial products and generates the least significant bits in advance to decrease the number of the input bits of the final adder. Also, the proposed MAC accumulates the intermediate results in the type of sum and carry bits instead of the output of the final adder, which made it possible to optimize the pipeline scheme to improve the performance.

Keywords: *Booth multiplier, carry save adder (CSA) tree, computer arithmetic, digital signal processing(DSP), multiplier and-accumulator (MAC),Multiplier, MSP(Most Significant Part), LSP(Least Significant Part), PP(Partial Product)*

1. INTRODUCTION

With the recent rapid advances in multimedia and communication systems, real-time signal processings like audio signal processing, video/image processing, or large-capacity data processing are increasingly being demanded. The multiplier and multiplier-and-accumulator (MAC) are the essential elements of the digital signal processing such as filtering, convolution, and inner products. Most digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transform (DWT).Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic's determines the execution speed and performance of the entire calculation. Because the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined by the multiplier, in general. For high-speed multiplication, the modified radix-4 Booth's algorithm (MBA) is commonly used.

In general, a multiplier uses Booth's algorithm and array of full adders (FAs), or Wallace tree [instead of the array of FAs., i.e., this multiplier mainly consists of the three parts: Booth encoder, a tree to compress the partial products such as Wallace tree, and final adder. Because Wallace tree is to add the partial products from encoder as parallel as possible, its operation

time is proportional to, where is the number of inputs. It uses the fact that counting the number of 1's among the inputs reduces the number of outputs into. In real implementation, many (3:2) or (7:3) counters are used to reduce the number of outputs in each pipeline step. The most effective way to increase the speed of a multiplier is to reduce the number of the partial products because multiplication proceeds a series of additions for the partial products. To reduce the number of calculation steps for the partial products, MBA algorithm has been applied mostly where Wallace tree has taken the role of increasing the speed to add the partial products. To increase the speed of the MBA algorithm, many parallel multiplication architectures have been researched.

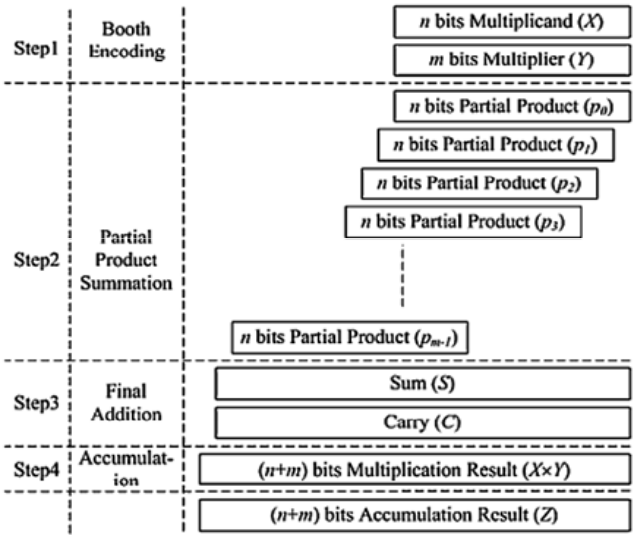
One of the most advanced types of MAC for general-purpose digital signal processing. It is an architecture in which accumulation has been combined with the carry save adder (CSA) tree that compresses partial products. In this architecture, the critical path was reduced by eliminating the adder for accumulation and decreasing the number of input bits in the final adder. While it has a better performance because of the reduced critical path compared to the previous MAC architectures, there is a need to improve the output rate due to the use of the final adder results for accumulation.

In this paper, a new architecture for a high-speed MAC is proposed. In this MAC, the computations of multiplication and accumulation are combined and a hybrid-type CSA structure is proposed to reduce the critical path and improve the output rate. It uses MBA algorithm based on 1's complement number system. A modified array structure for the sign bits is used to increase the density of the operands. A carry look-ahead adder (CLA) is inserted in the CSA tree to reduce the number of bits in the final adder. In addition, in order to increase the output rate by optimizing the pipeline efficiency, intermediate calculation results are accumulated in the form of sum and carry instead of the final adder outputs.

2. MAC

In this section, basic MAC operation is introduced. A multiplier can be divided into three operational steps. The firs t is radix-2 Booth encoding in which a partial product is generated from the multiplicand (X) and the multiplier (Y). The second is adder array or partial product compression to add all partial products and convert them into the form of sum and carry. The last is the final addition in which the final multiplication result is produced by adding the sum and the carry. If the process to accumulate the multiplied results is included, a MAC consists of four steps, as shown in Figure 1.

A general hardware architecture of this MAC is shown in Figure 2. It executes the multiplication operation by multiplying the input multiplier and the multiplicand. This is added to the previous multiplication result as the accumulation step. The N-bit 2's complement binary number can be expressed as



$$X = -2^{N-1}x_{N-1} + \sum_{i=0}^{N-2} x_i 2^i, \quad x_i \in 0,1 \tag{1}$$

If (1) is expressed in base-4 type redundant sign digit form in order to apply the radix-2 Booth's algorithm.

$$X = \sum_{i=0}^{\frac{N}{2}-1} d_i 4_i \quad (2)$$

$$d_i = -2x_{2i+1} + x_{2i} + x_{2i-1} \quad (3)$$

If (2) is used, multiplication can be expressed as

$$X \times Y = \sum_{i=0}^{\frac{N}{2}-1} d_i 2^{2i} Y \quad (4)$$

If these equations are used, the afore-mentioned multiplication- accumulation results can be expressed as

$$P = X \times Y + Z = \sum_{i=0}^{\frac{N}{2}-1} d_i 2^i Y + \sum_{j=0}^{2N-1} Z_i 2^i \quad (5)$$

Each of the two terms on the right-hand side of (5) is calculated independently and the final result is produced by adding the two results. The MAC architecture implemented by (5) is called the standard design If N-bit data are multiplied, the number of the generated partial products is proportional to N. In order to add them serially, the execution time is also proportional to N. The architecture of a multiplier, which is the fastest, uses radix-2 Booth encoding that generates partial products and aWallace tree based on CSA as the adder array to add the partial products. If radix-2 Booth encoding is used, the number of partial products, i.e., the inputs to the Wallace tree, is reduced to half, resulting in the decrease in CSA tree step. In addition, the signed multiplication based on 2's complement numbers is also possible. Due to these reasons, most current used multipliers adopt the Booth encoding.

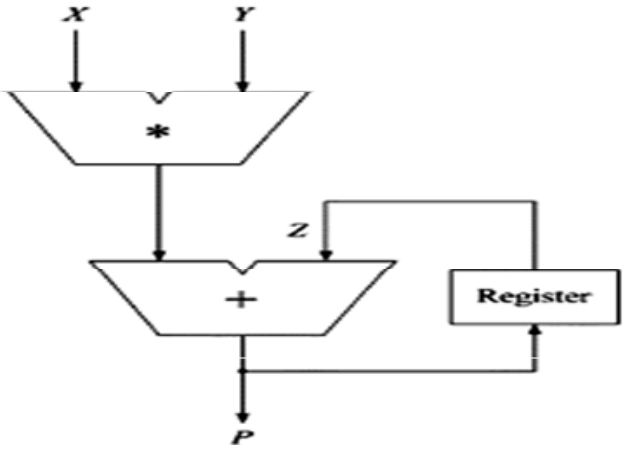


Figure2. *Hardware Architecture of General MAC*

Booth's Algorithm and Rules

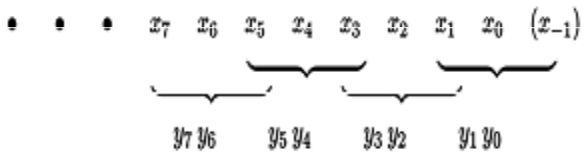
x_i	x_{i-1}	Operation	Comments	y_i
0	0	shift only	String of zeros	0
1	1	shift only	String of ones	0
1	0	subtract and shift	Beginning of a string of ones	1
0	1	add and shift	End of a string of ones	1

Properties

- Multiplication starts from least significant bit
- If started from most significant bit - longer adder/subtractor to allow for carry propagation
- No need to generate recoded SD multiplier (requiring 2 bits per digit)
- Bits of original multiplier scanned - control signals for adder/subtractor generated
- Booth's algorithm can handle two's complement multipliers
- If unsigned numbers multiplied - 0 added to left of multiplier (xn=0) to ensure correctness

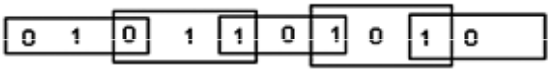
Radix-4 Booth Algorithm

- Bits *xi* and *xi* – 1 recoded into *yi* and *yi*–1 - *xi*-2 serves as reference bit
- Separately - *xi*-2 and *xi*-3 recoded into *yi*-2 and *yi*-3 - *xi*-4 serves as reference bit
- Groups of 3 bits each overlap - rightmost being *x*1 *x*0 (*x*–1), next *x*3 *x*2 (*x*1), and so on



- To Booth recode the multiplier term, we consider the bits in blocks of three, such that each block overlaps the previous block by one bit. Grouping starts from the LSB, and the first block only uses two bits of the multiplier (since there is no previous block to overlap):

Figure 4 shows a computing example of Booth multiplying two numbers "2AC9" and "006A". The shadow denotes that the numbers in this part of Booth multiplication are all zero so that this part of the computations can be neglected. Saving those computations can significantly reduce the power consumption caused by the transient signals.



BLOCK	PARTIAL PRODUCT
0 0 0	0
0 0 1	1*multiplicand
0 1 0	1*multiplicand
0 1 1	2*multiplicand
1 0 0	-2*multiplicand
1 0 1	-1*multiplicand
1 1 0	-1*multiplicand
1 1 1	0

Figure3. *Grouping of bits from the multiplier term*

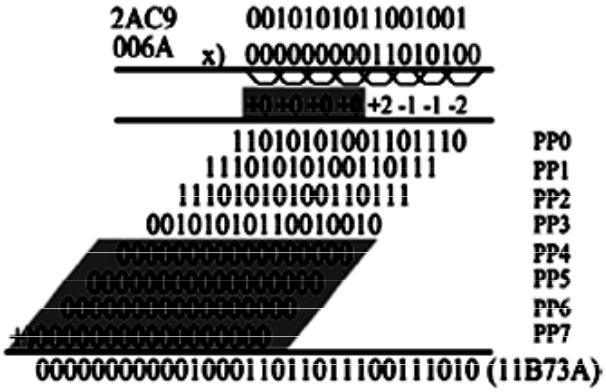


Figure 4. *Multiplication using modified Booth Encoding*

3. CSA

The architecture of the hybrid-type CSA that complies with the operation of the proposed MAC is shown in Figure 5, which performs 8*X* 8-bit operation.

- $A + B \Rightarrow S$
- Save carries $A + B \Rightarrow S$, Cout
- Use *Cin* $A + B + C \Rightarrow S1, S2$ (3# to 2# in parallel)

Used in combinational multipliers by building a Wallace Tree It was formed based on (12).

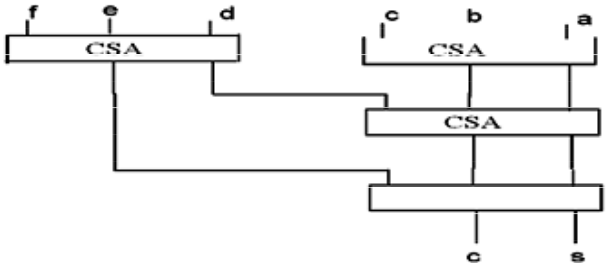


Figure5. *Carry Save Adder*

Advantages

- Booth multipliers save costs (time and area) for adding partial products
- With the higher radix the number of additions is reduced and the redundant Booth code reduces costs for generating partial products in a higher radix system.
- Low power consumption is there in case of radix 4 booth multiplier because it is a high speed parallel multiplier.

Applications

- Multimedia and communication systems,
- Real-time signal processing like
- audio signal processing,
- Video/image processing, or large-capacity data processing.
- The multiplier and multiplier-and-accumulator (MAC) are the essential elements of the digital signal processing such as filtering, convolution, and inner products.
- The MAC on specific processor cannot be run at 100% efficiency.
- Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic's determines the execution speed and performance of the entire calculation.

Due to the reasons of lower speed of MAC, To improve speed of MAC on specific processor, MAC needs to be fast must have special algorithm for "multiplication" instruction.

4. RESULTS AND DISCUSSION

In this paper we are evaluating the performance of the proposed high speed low power MAC implemented in VERILOG coding. For getting power reports we are synthesizing using Cadence, Xilinx. The simulation results are given in Figure 6.

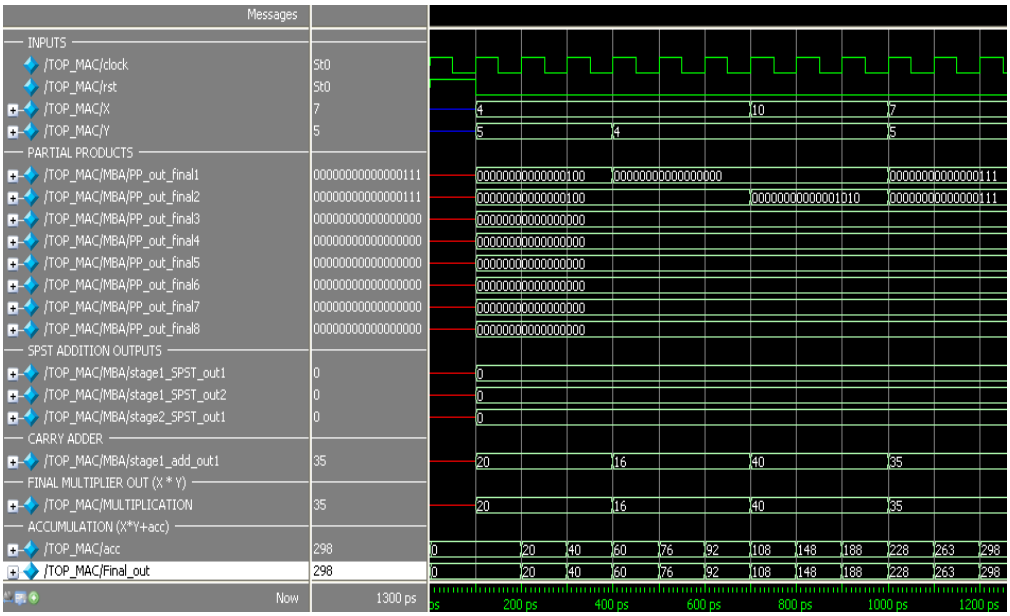


Fig 6. *Simulation Results for M*

Power & Area Analysis with Cadence tool (180nm Technology)

Instance	Cells	Leakage Power(nW)	Dynamic Power(nW)	Total Power(nW)
MULTIPLIER	**193**	**235.141**	**590468.264**	**590703.405**
BE	123	77.732	117052.091	117129.822
BM	123	77.732	117052.091	117129.822
CSA	36	67.413	200086.433	200153.846
ADD1	19	42.814	73254.76	73297.574
ADD2	0	0	0	0
ADD3	0	0	787.32	787.32
acc	17	54.351	64593.194	64647.545
Total Area	**193**	**2109** combinational	**4524** logic	**6633**

5. CONCLUSIONS

Our paper gives a clear concept of different multiplier. We found that the parallel multipliers are much option than the serial multiplier. We concluded this from the result of power consumption and the total area. In case of parallel multipliers, the total area is much less than that of serial multipliers. Hence the power consumption is also less. This is clearly depicted in our results. This speeds up the calculation and makes the system faster.

While comparing the radix 2 and the radix 4 booth multipliers we found that radix 4 consumes lesser power than that of radix 2. This is because it uses almost half number of iteration and adders when compared to radix 2. Multipliers are one the most important component of many systems. So we always need to find a better solution in case of multipliers. Our multipliers should always consume less power and cover less power. So through our paper we try to determine which of the three algorithms works the best. In the end we determine that radix 4 modified booth algorithm works the best.

REFERENCES

[1] A New VLSI Architecture of Parallel Multiplier-Accumulator Based on Radix-2 Modified Booth Algorithm - Young-Ho Seo, and Dong-Wook Kim.

[2] A Spurious-Power Suppression Technique for Multimedia/DSP Applications - Kuan-Hung Chen, Yuan-Sun Chu.

[3] A Fast Parallel Multiplier-Accumulator Using the Modified Booth Algorithm - Fayaz Elguibaly.

[4] Parallelized Booth-Encoded Radix-4 Montgomery Multipliers - Nathaniel Pinckney, Philip Amberg and David Money Harris.

[5] A.D. Booth, "A Signed Binary Multiplication Technique", *Quart. J.Math.*, 4, pp. 236-240, 1952.