International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 01 | Jan 2020 www.irjet.net p-ISSN: 2395-0072

MAC Unit by Efficient Grouping of Partial Products along with Circular Convolution Operation

Ananda R[1], B K Venugopal[2]

[1]ME Student, Dept. of Electronics and Communication Engineering, UVCE
Bangalore University, Karnataka, India
[2]Assistant Professor, Dept. of Electronics and Communication Engineering, UVCE
Bangalore University, Karnataka, India

Abstract - MAC units plays a vital role in many of the digital signal processing applications, such as filtering, convolution, inner products and so on. To perform all these applications there should be a need for an efficient MAC unit, usually conventional MAC units are easy to design, but due to involvement of pipeline stages lot of logic units and adoptable LUTs required which intern increases the area required to implement on FPGA, also as these conventional MAC units uses carry propagate adders, the delay is more. To enhance the performance and to reduce the resource usage we are implementing a MAC unit along with the circular convolution and comparing the resource usage and performance characteristics of the proposed structure with the conventional pipeline design and with the redundant carry save MAC unit. The redundant MAC have 16x16 multiplier with booth encoding to enhance the speed and the 40 bit accumulator for the accumulated result, the proposed MAC unit is just the extension of the redundant mac unit. And the design are synthesised in Altera Stratix III FPGA's to utilise the fast carry chain for a better performance

Key Words: pipeline stages, LUT's, resource usage, Circular convolution, booth encoding, partial product

1. INTRODUCTION

MAC (Multiply and Accumulate) Units are extensively used in Digital Signal Processing Applications such as multiplication, convolution, Filtering etc. And Field Programmable Gate Arrays, that is reconfigurable systems are mostly used to perform the various Digital Signal Processing Applications. For the multiplication operations, multiplicand and multiplier produces summands also known as partial products, and each of those are shifted by the radix of the bits which we use. The summands generated can be added using the multi bit operand adders with the carry propagation along with various pipeline stages or we can also reduce the partial products, using the reduction operators such as compressor trees using carry save arithmetic, booth encoding and counters[1,2,3]. The compressor trees and the reduction operators require more area than the carry propagate adders But the ripple carry delay in carry propagate addition is linearly increases with the operand bit width, thus if the length of the operands is very large then the fast carry chains utilised for the effectiveness of carry propagate addition becomes less effective. In our work we are utilising both the fast carry chains and the reduction operators to optimize the performance and also the area as compared to conventional MAC unit.

The flow of the paper is as follows. We design a MAC unit to perform multiply operation of N bit operands which usually generates N partial products, but using the Radix 4 Booth encoding we reduce the number of partial products to N/2, by grouping those partial products and use a suitable multi bit adders of size less than the N bit operand adders, along with the carry save addition. And we are implementing the circular convolution operation, by utilising the partial products which are already generated, as the circular convolution operation includes the same multiply operation by rearranging the partial products in a particular order.

2. LITERATURE SURVEY

In the reference paper[4], a pipelined Elliptic curve scalar multiplication using the pipeline architecture, here they make use of MAC unit of finite field of order. In our paper to reduce the area further, using reduction operators and carry save technique

In the reference Paper [5], explained on how to use the fast carry chains to build the compressor trees for partial product reduction and for the accumulation using the FPGA Logic. About fast carry chains there is a perception that these can only be used to implement the carry propagate adders with the accompanying circuits like subtracters. But by using the LUT (look up tables) to stop the propagation of the carry at appropriate points, enable to use fast carry chains in compressor trees. In our paper we are also utilising this advantage along with further reduction in area

And to reduce the area further in our work we are referring [6] where they have discussed to implement reduction operators in a slightly different manner. In our Paper we are reducing the partial products in stages using network of counters.

In the reference Paper [7], depending on the FPGA characteristics low precision fused multiply and addition units are discussed to use in DSP's efficiently. Here the

shifting is done before the multiply and add stage, using this idea in slightly different manner in our paper we have realised the circular convolution operation of the multiplier and multiplicand bits.

In the reference Paper [8], mentioned about redundant carry save and hybrid architecture to design MAC unit, So in our Paper we are also taking up the ideas discussed in the latter reference paper along with implementing circular convolution operation on the bits of the operands

3. METHODOLOGIES

3.1 Booth Encoding

In normal multipliers, the multiplication result is computed by the partial products generated in a Radix 2 manner, that is at a time by witnessing one bit of the multiplier. In our Paper we are utilising the Radix-4 booth encoding to reduce the 16 partial products to half of 16 bit operands, here each of the partial products depend on the 2 bits of the multiplier. If Y(multiplicand) and X(multiplier) are the two operands, then each of the partial product is 0, Y, 2Y and -2Y which is discussed as modified booth encoding[20] and an example for the same is mentioned for the 6 bit operands $Y=011001_2$ and $X=100111_2$ in the Fig. 1

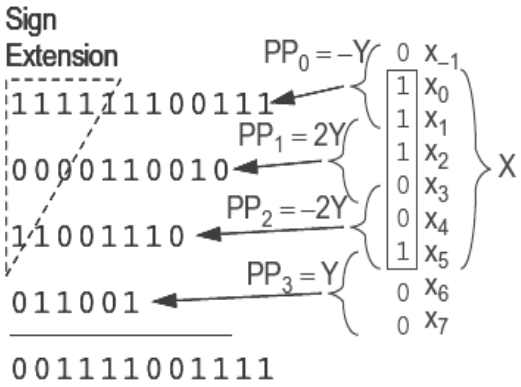

Fig -1: Booth Encoding example.

Simulation Result for the Booth encoded output can be seen in the Fig.2. In this A and b are the multiplier and the multiplicand and p1 to p9 are the partial products generated based on the modified radix-4 booth encoding algorithm

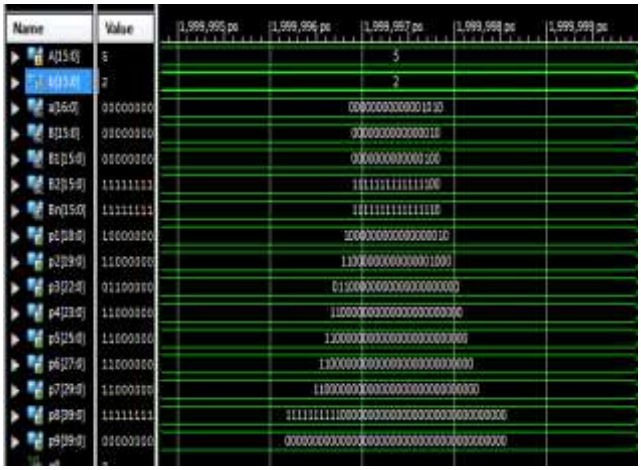

Fig -2: Verilog simulation result for booth encoding

3.2 Carry Save redundant Addition

The most usual method for addition is Carry Propagate Adder to add k N-bit words, which takes large area and also it is slow. So the better approach is to preserve the sum and carry of the each addition and by using those results output can be computed instead of propagating the carry, as the carry output is preserved, this is called carry save redundant form. And an example for the CSA is shown in Fig. 3.

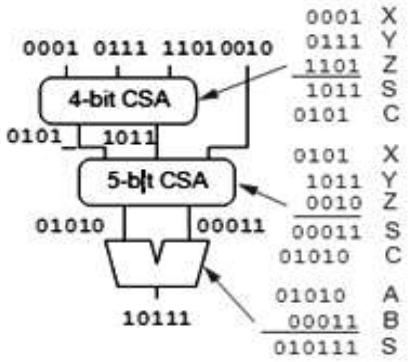

Fig – 3: CSA example.

4. CONVENTIONAL MAC UNIT

An accumulator adder and multiplier needed together to construct a MAC unit. Typically Carry-select or Carry-save adders are mostly implemented due to the DSP application necessitates 'fast speed', The memory fetches the inputs from its location to multiplier for additional multiply-accumulate operations. The produced result of MAC unit is stored at a relevant memory location. The situation demands that this complete process must be carried out in a single clock cycle.

Here we are designing this conventional MAC unit to make the comparison with the proposed architecture, and that conventional MAC unit is shown in the Fig. 4.

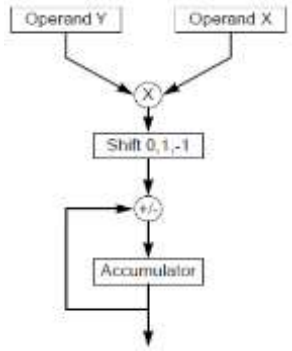

Fig – 4: conventional MAC unit with pipeline stages

5. REDUNDANT CARRY SAVE MAC ARCHITECTURE

This redundant structure is designed before the proposed structure so as to distinguish the technique which we are using in our proposed work.

The redundant carry-save MAC architecture is shown in Fig. 5. The MAC output is in redundant carry-save form, that is each digit has the value set of (0, 1, 2), each of the multiply and add operation be made of one Booth encoding stage and two (6, 3) counter stages and a (3, 2) counter stage. The redundant multiply accumulate operation does not have pipeline stage explicitly. The structure works at high speed as compared to conventional multiply accumulate structures, even with no pipeline in the operation. Only for the redundant to normal binary conversion stage, clock delay appears which is simply a two operand adder. In Fig. 5. Inside the dotted box the redundant to binary conversion stage is shown. In most of the digital signal processing applications, normal binary conversion is not required often. For example, 100-tap finite impulse response filter (FIR) stage,

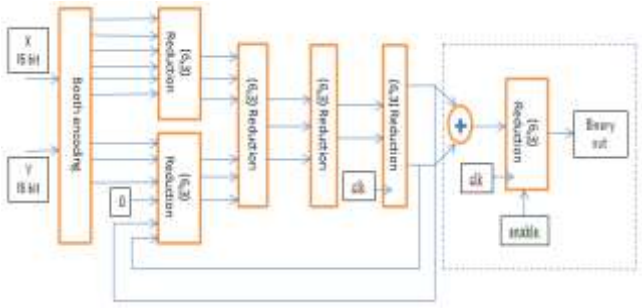

Fig -5: Redundant MAC architecture

After every 100 multiply-add operations, redundant to binary conversion is required. Even for matrix multiplication, general convolution operations etc. of digital signal processing this condition is true.

6. PROPOSED MAC UNIT AND CIRCULAR CONVOLUTION USING PARTIAL PRODUCTS

6.1 Circular convolution Operation

This Operation is similar to two decimal number multiplication and just by rearranging the partial products we are able to get the circular convoluted outputs [9].

In our project we are designing this operation by considering the length of the multiplier and multiplicand same which is N. And based on the circular convolution Operation the result length is equal to the minimum of the length of the two inputs, to show the circular convolution operation, instead of passing different sequences to the MAC unit, we are showing the same operation with the binary digits of the two operands. That is by representing two operands which is same as multiplicand and multiplier to the MAC unit in a binary format and by considering each bit as a number in a digital sequence. So in this way we are able to observe the exact circular convolution operation.

In this project we are demonstrated the inputs lengths varying from N=3 to N=10 and we are able to observe the corresponding output Xilinx ISE simulation window. For example if we consider 'a' and 'b' as input and 'y' as output for N=5, then we can represent the circular convolution as below

y[0]=(a[0]&b[0])+(a[1]&b[4])+(a[2]&b[3])+(a[3]&b[2])
+(a[4]&b[1]);

y[1]=(a[0]&b[1])+(a[1]&b[0])+(a[2]&b[4])+(a[3]&b[3])
+(a[4]&b[2]);

y[2]=(a[0]&b[2])+(a[1]&b[1])+(a[2]&b[0])+(a[3]&b[4])
+(a[4]&b[3]);

y[3]=(a[0]&b[3])+(a[1]&b[2])+(a[2]&b[1])+(a[3]&b[0])
+(a[4]&b[4]);

y[4]=(a[0]&b[4])+(a[1]&b[3])+(a[2]&b[2])+(a[3]&b[1])
+(a[4]&b[0]);

And the above example is constructed based on the (1)

$$y(n) = x(n) * h(n) = \sum_{m=0}^{N-1} x(m)y(m-n) \bmod N \quad \text{- - (1)}$$

And the simulation result for the circular convolution can be seen in the Fig. 6.

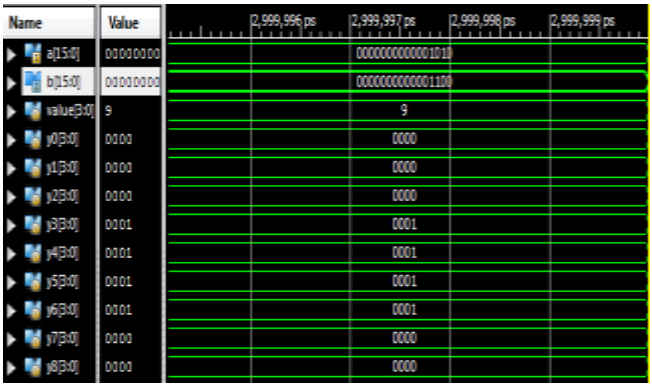

Fig -6: Verilog Simulation Result for Circular convolution

a and b are the two inputs in binary and for the value N=9, the result of the circular convolution operation on those binary inputs is displayed in Fig. 6. Like this we can perform this operation to any value of N, but in this project for the demonstration we have designed upto N=10 from N=3.

6 .2 By grouping the partial products

Redundant carry-save architecture as depicted in Fig. 5 provides high performance, due to the fact that it is free from carry propagation. However, the carry-save structure necessitates twice the area than ripple-carry structures. Furthermore, ripple carry arithmetic offers high performance in most of the FPGA systems due to the fact that fast carry chains boosts the performance. However, as the operand bit sizes increases, the structure becomes less efficient.

In the proposed structure, the carry propagation delay is narrowed by breaking the partial product or summands array into smaller blocks. The division is carried based on the structure of the partial product array which has a diamond shape, getting thick in the middle

The partial product array is divided into four multi-operand adder blocks of rectangular shape with almost equal delays as shown in Figure 5, by dividing the carry propagation chain in small and equivalent delay blocks,. In this partial products after the Booth encoding are divided into equivalent delay rectangular blocks, and each of the rectangular blocks is fed into multi-operand adders. In FPGA development tools, it is easy to configure multi-operand adders. In this work, for the automatic generation of various multi-operand adder blocks, AlteraTM's Quartus II software is used. The generated multi-operand blocks also have the benefit of fast carry chain for the horizontal addition stage. Also, these blocks can be pipelined further to increase throughput. Much larger multi-operand adder (both in horizontal and vertical size) would be vital, if these blocks are not divided into sub-blocks. The MAC unit output is in double carry-save format, that is the output is a arrangement of three components. Each rectangular blocks is normal carry-propagate multi-operand adders. The empty space in each multi-operand adder is expanded with zero's in Fig. 7. And the feedback path of the accumulate stage ,the (3, 2) counter array is placed.

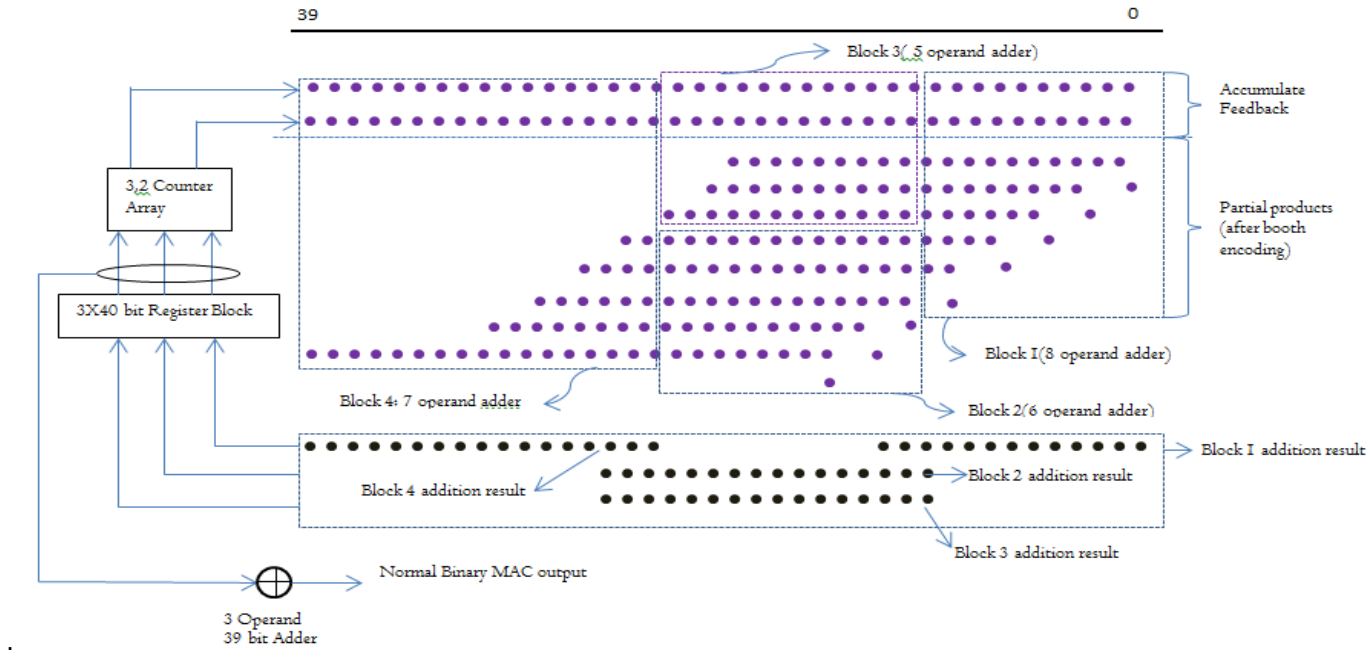

Fig -7: MAC unit by efficient grouping

Still, it does not disturb throughput as for the preparation of the partial products booth encoding for the next multiplication operation also needs a LUT array delay. With the benefit of fast carry structure, carry propagation is limited and compared to regular carry propagate structures considerably higher performance can be attained.

In Fig. 7, the output is made of double carry-save format. And to get the normal two's complement output, the output should be fed into 'three operand' regular adder block. Though, regular binary conversion is the final result and it is needed after many computations in most of the DSP operations such as, matrix multiplication, convolution etc. Conventional binary format conversion from double carry-save can be handled similar to Figure 4 output scheme, with three operand adder other than two-operand adder at the output. However, both three and two operand adders has the same performance in Altera's Stratix family FPGA's, so that the throughput will not be affected.

And the simulation result for the MAC unit (Fig.7) is shown in Fig. 8 which contains the output of Proposed multiplier architecture with input 'a' and 'b' with output 'out'. Here partial products follows booth algorithm and for addition operation carried by counters and finally accumulated outputs. 'out' is final output, 'out1' is accumulated outputs with having extended sign bits as 1's

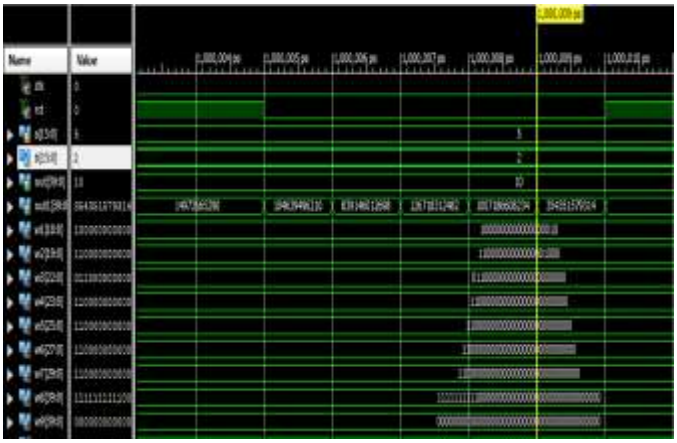

Fig -8: Verilog simulation result for the Mac unit by efficient grouping

7. RESULTS AND DISCUSSIONS

By comparing different Mac structures, the Table 1 represents the different trade off in design, which is been obtained by running the synthesis report in Xilinx. And it is clear from those comparison that the conventional MAC unit with different pipeline stages require more Area, and also as it involves complete carry propagate adders even the delay is more.

TABLE -1: Comparison of proposed MAC with other MAC

Mac units	conventional	redundant	proposed
Number of Slice LUT's	414	169	205
Number of bonded IOBs	12% (72/600)	47% (286/600)	13% (78/600)
Propagation Delay	5.513ns	1.756ns	4.476ns

The resource usage can be analysed by looking at the number of LUT's utilised in the design which ultimately specifies the area required for the given design. So both proposed and the redundant design require less area as compared to conventional design. By considering 1000 LUT's as reference for our proposed structure, the conventional MAC unit utilises 41.4% area where as our proposed structure utilises around 21%. Also the proposed structure utilises less number of input output pins out of the available pins in Virtex7 FPGA. Also the proposed structure has the delay lessor than the conventional but more than the redundant structure, as mentioned in the TABLE 1, so these kind of area and performance trade off exist between Redundant and Proposed MAC unit, but the proposed MAC unit using less logic units as it require lessor number of input output pins, Both redundant and proposed using the carry save architecture, the only difference is that by grouping those partial products to reduce the number of logic units required for the design. And clearly as compared to Conventional MAC unit the proposed architecture is far better both in terms of area and performance.

And with the help of partial products we are generating the circular convoluted output for the given set of inputs and also extending this to perform the MAC operation by efficient grouping of those partial products generated.

8. CONCLUSION

In this Project we have designed a MAC unit by utilising the booth encoding and using carry save architecture as in Redundant carry save MAC unit along with circular convolution , In this proposed MAC unit first with the help of partial products we show the results of the circular convolution and then with slightly different grouping of those partial products designed the MAC unit which is far better than conventional MAC unit in terms of resource usage also we can observe that its performance is improved.

REFERENCES

[1] A High Performance CMOS Redundant Binary Multiplication-and-Accumulation (MAC) Unit. Huang, X, Liu, W and Wei, B. 1, 1994, IEEE Trans. Circuits & Syst.-I, Vol. 41, pp. 33-44.

[2] Redundant Arithmetic: Algorithms and Implementations. Gonzalez, A F and Mazumder, P. December 2000, INTEGRATION, the International VLSI Journal, Vol. Vol.30, pp. 13-53.

[3] Parhami, B. Algorithms and Design Methods for Digital Computer Arithmetic. s.l.: Oxford University Press, 2012.

[5] EXPLOITING FAST CARRY-CHAINS OF FPGAS FOR. Parandeh-Afshar, Hadi, Brisk, Philip and Ienne, Paolo. s.l. : IEEE, 2009. International Conference on Field Programmable Logic and Applications.

[6] Efficient synthesis of compressor trees on FPGAs. Parandeh-Afshar, H, Brisk, p and Ienne, P. Seoul : s.n., 2008. Asia-South Pacific Design Automation Conf. pp. 138-143.

[7] "Low-precision DSP-based floating-point multiply-add fused for Field Programmable Gate Arrays". Amaricai, Alexandru, Boncalo, Oana and Elena Gavriliu, Constantina. 2013. IET Computers & Digital Techniques.

[8] "MAC unit for reconfigurable systems using multi operand Adders with double carry save encoding". Ugur Cini, Olcay Kurt. 2016, IEEE, DTIS

[9] J. Merin Philip and Chalil, A., "An FPGA Implementation of linear & Circular Convolution", Amrita Vishwa Vidyapeetham, in IEEE National Conference on Electrical & Electronics Engineering (NC3E-2014), Banglore, 2014.