INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

IMPLEMENTATION OF LOW POWER, LOW DELAY FUSED ADD-MULTIPLY OPERATOR USING PREFIX ADDERS

Gaddam Sushma[1], Manchu Naresh Babu[2]

[1]PG Student, VLSI, Sree Vidyanikethan Engineering College, Tirupati, Chittoor, A.P, India
[2]Assistant Professor, Dept. of ECE, Sree Vidyanikethan Engineering College, Tirupati, Chittoor, A.P, India

Abstract: - In many Digital Signal Processing (DSP) applications, complex arithmetic operations are used. To increase the performance and to reduce the complexity of arithmetic operations, we designed a Fused Add-Multiply operator which directly recodes the sum of two numbers in its modified booth (MB) form and uses Wallace Carry save adder for the partial product addition. The proposed technique focus on FAM design by using prefix adders at the last stage of partial product addition which reduces both power consumption and delay leading to more efficient design for the purpose of low power applications.

Keywords: Modified Booth algorithm, Wallace Carry Save adder, Parallel Prefix adders.

1. Introduction

The continuous advance in modern consumer electronics makes use of Digital Signal Processing (DSP). Low power and high performance are the main parameters to meet the requirements of various applications. The performance of DSP systems is based on arithmetic operations as their implementation depends on Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), Finite Impulse Response (FIR) filters and signals convolution. To increase the performance of the arithmetic operations, faster and smaller multipliers are required. Modified Booth recoding is one used for this purpose.

In this work, we use a fused technique which performs both addition and multiplication operation at a time. For this, a specially designed signed bit adders are used. Here, we use a radix-4 modified booth's algorithm to increase the speed of multiplication. As the radix size increases, the number of partial products are reduced, which uses small adder tree for the addition of partial products, which in turn increases speed and reduces area. The fused operation is mainly performed in three steps: S-MB recoding, a Wallace tree, and a final adder. In the existing system carry look ahead or ripple carry adder is used at the last stage. Here, we replace it with parallel prefix adder which reduces delay and power as the addition is performed in parallel.

2. Existing Design

In conventional design of AM operator addition and multiplication are performed separately. First addition is done by using any conventional adder and then the input and the result of adder are driven to a multiplier to get the output. The disadvantage of using a separate conventional adder is that it inserts a significant critical path delay. As the bit width increases critical path delay also increases, because the carry signals are to be propagated inside the adder. To overcome this we used a Carry Look Ahead (CLA) adder, which however

increases area and power dissipation. In order to reduce this area and power dissipation, an optimized design of the AM operator which combines the adder and MB encoding unit into a single data path block by direct recoding the sum of two numbers into its modified Booth form as in figure 1.

The fused Add Multiply (FAM) operator uses only one adder at the last stage of addition; as a result it reduces both area and critical path delay. A way to increase the speed of multipliers is through the use of Modified Booth algorithm to generate the partial products. The generated partial products are added by using Wallace tree instead of normal full adders. The carry look ahead adder is performed for the sum and carry bit generated from the Wallace CSA at the last stage for the final result.

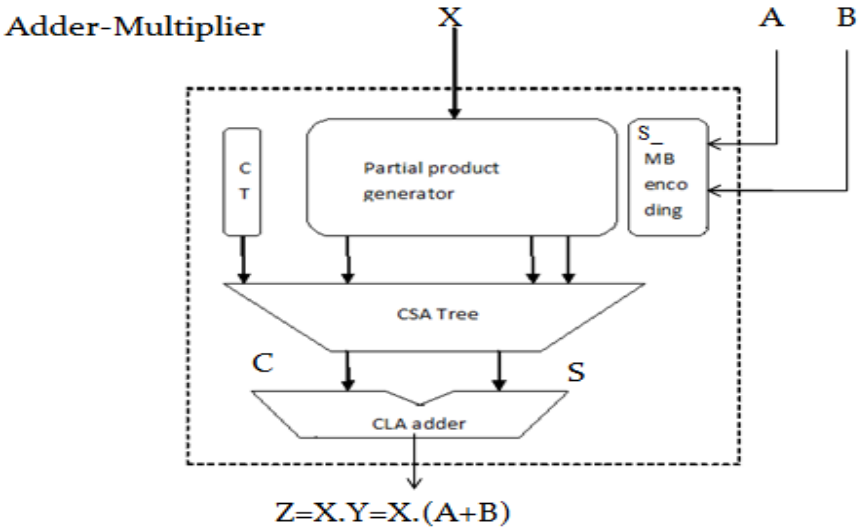

Figure 1: Fused design with direct recoding of A and B in its MB representation

CLA calculates the carry in advance based on input bits. The carry is obtained in two cases: In first case when both inputs are 1 and in the second case when either of the input is 1 and the carry from the previous stage is 1. It generates two internal signals: carry propagate and generate signals. The carry propagate signal denotes whether the carry is passed to the next stage or not. It happens when either of the input is 1.The carry generate signal is 1 when both inputs are 1.

3. Proposed Design

This paper focus on the efficient design of the fused AM unit which implement the operation Z=X. (A+B). The parallel prefix adder is used at the last stage of fused Add-Multiply (FAM) as shown in figure 2.

3.1 S-MB recoding

In this we recode the sum of two consecutive bits of the input A with two consecutive bits of the input B into one MB digit y_j^{MB}. S-MB recoding is performed in 3different schemes for this a set of bit level Half Adders (HAs) and Full Adders (FAs) are developed.

3.1.1 S-MB1 Recoding

In S-MB1 recoding technique, we used conventional and signed FA for both odd and even number of bit width of input numbers. In order to form the MB digit y_j^{MB}, $0 \le j \le k-1$ we need 3 bits (s_{2j+1}, s_{2j}, c_{2j}) which are the outputs of j^{th} recoding cell having the inputs a_{2j}, a_{2j+1} and b_{2j}, b_{2j+1}. The sum bits s_{2j+1} and s_{2j} are extracted from the j^{th} recoding cell and c_{2j} bit is extracted from the conventional FA having the inputs a_{2j-1}, b_{2j-1} and b_{2j-2}.

3.1.2 S-MB2 Recoding

In S-MB2 recoding technique, we used signed FA and HA for both odd and even number of bit width of input numbers. Initially we consider $c_{0,1} = c_{0,2} = 0$. In order to form the MB digit y_j^{MB}, $0 \le j \le k-1$ we need 3 bits (s_{2j+1}, s_{2j}, $c_{2j,2}$) which are the outputs of jth recoding cell. As in the S-MB1 recoding scheme, we use a FA to

produce the sum s_{2j} and the carry c_{2j+1} with a_{2j}, b_{2j}, $c_{2j,\ 1}$ as inputs. The bit $c_{2j,\ 1}$ is the output carry of the conventional HA of the previous recoding cell and has the bits a_{2j-1}, b_{2j-1} as inputs. The output bit s_{2j+1} is produced from the HA* which is negatively signed bit.

3.1.3 S-MB3 Recoding

The third recoding scheme is S-MB3. In this scheme, we use a conventional FA, signed HA and FA. Initially we consider $c_{0,\ 1} = c_{0,\ 2} = 0$. As same in the previous recoding schemes, we use a FA to produce sum s_{2j} and the carry c_{2j+1} with a_{2j}, b_{2j}, $c_{2j,\ 1}$ as inputs. The bit $c_{2j,\ 1}$ is the output carry of the signed HA (HA*) of the previous recoding cell and the output bit s_{2j+1} is produced from HA** which is negatively signed.

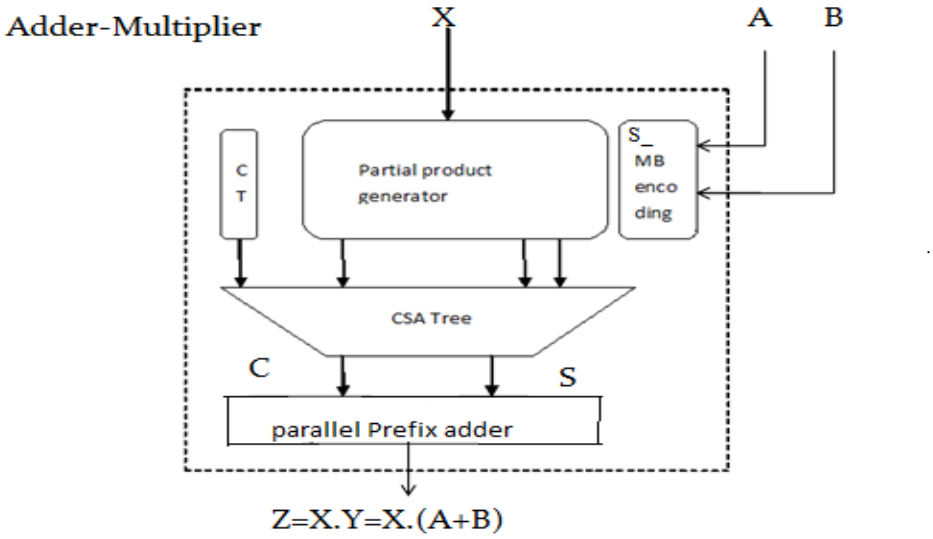

Figure 2: Fused design with parallel prefix adder

3.2 Radix-4 Modified Booth Recoding

Booth uses radix recoding to achieve high speed. As the radix size increases the number of partial products are reduced resulting in high speed. It is possible to reduce the number of partial products by half, using radix-4 booth recoding. It performs the process of recoding the multiplicand based on the multiplier bits. As we use radix-4, it will compare three consecutive bits at a time with overlappimg technique.

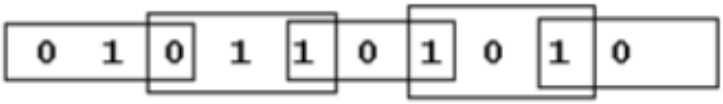

Figure 3: Grouping of bits for radix-4

Grouping of bits starts from the LSB. The first block only uses two bits of the multiplier and assumes a zero for the third bit. Then compare the bits with the booth recoding table shown in table1 to generate partial products. By using S-MB recoding schemes, we convert the sum of two consecutive bits of two inputs into three bits and then obtanied bits are compared with the booth table to generate partial products. After the partial products are generated, they are added through a Wallace Carry Save Adder (CSA) tree along with the correction bits.

4. Wallace CSA

For real time signal processing, high speed is always a key to achieve high performance in the DSP systems. The main consideration of add-multiply or MAC unit is to enhance its speed. That high speed is achieved through Wallace CSA. It is used to add the partial products that are generated from the booth recoding along with the correction bits into two rows, which are added for the final result.

Wallace CSA tree is implemented in two ways. One way among them is considering all bits in each column at a time and compresses them into two bits (a sum and a carry). Another way is to consider all bits in each four rows at a time and compresses them into two bits using 4:2 compressors, 3:2 compressors, full adders and half adders. The inputs to a column are the bits of the partial products and the carry bits from one column to the right and the sum bits that are generated within the same column. The outputs from a column are the carry bits to the column one to the left and the last two sum bits in that column that are passed to the prefix adder. Finally, the carry-save output of the Wallace CSA is leaded to a Parallel prefix adder to form the final result Z = X. (A+B).

X_{i+1}	X_i	X_{i-1}	Y	Partial product explanation
0	0	0	0	All 0's
0	0	1	1.A	$[A_{(n-1)}, A_{(n-1)}, A]$
0	1	0	1.A	$[A_{(n-1)}, A_{(n-1)}, A]$
0	1	1	2.A	$[A_{(n-1)}, A, 0]$
1	0	0	-2.A	$[A_{(n-1)}, -A, 0]$
1	0	1	-1.A	------- ------ $[A_{(n-1)}, A_{(n-1)}, -A]$
1	1	0	-1.A	------- ------- $[A_{(n-1)}, A_{(n-1)}, -A]$
1	1	1	0	All 0's

Table 1: Booth recoding strategy for radix-4

5. Parallel Prefix adder

Parallel prefix adder differs in the implementation of carry generation block with that of CLA. There are many prefix adders. In this paper we use Brent-Kung adder, the structure is shown in figure 4. It is done in three steps: The fundamental step is to pre-calculate the propagation and generation signals.

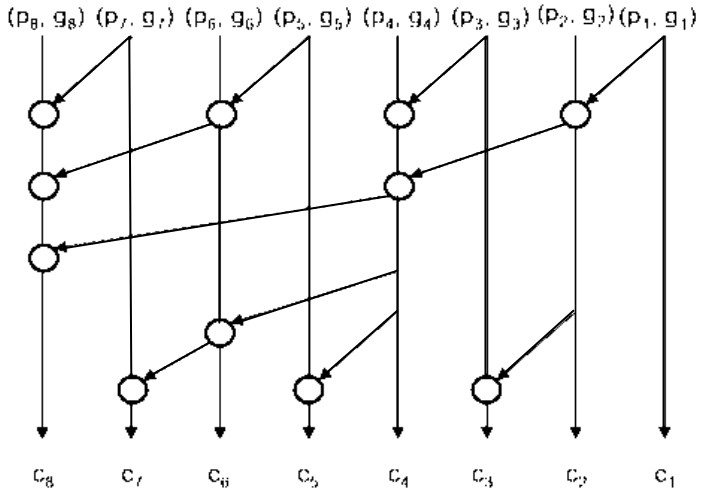

Figure 4: Brent-Kung adder

In the first stage propagation and generation signals are calculated by using the following equations

$$p_{0,\,i} = a_i \oplus b_i$$

$$g_{0,\,i} = a_i \wedge b_i$$

In the later stages, these propagation and generation signals are calculated by taking the previous stage signals as inputs. The equations to generate these signals:

$$p_{j,\,i} = p_{j-1,\,i} \wedge p_{j-1,\,i-1}$$

$$g_{j,\,i} = g_{j-1,\,i} \vee (p_{j-1,\,i} \wedge g_{j-1,\,i-1})$$

In the second step prefix graphs are used to calculate the carries, and it describes the structure that performs this part. The third step is to generate the sum. The equations to generate the sum and carries are as below:

$$c_i = g_{j,\,i}$$

$$s_i = p_{0,\,i} \oplus c_{i-1}$$

By using prefix adder at the last stage to the carry-save output to form the final result Z = X.Y=X. (A+B) as shown in figure 2.

6. Results and Discussion

6.1 Simulation Environment

The three recoding schemes are implemented in verilog HDL for both cases of even and odd bit-width of the input numbers. Xilinx ISE 10.1i is used for simulation and Quartus II 9.1 is used to evaluate the delay and power consumption.

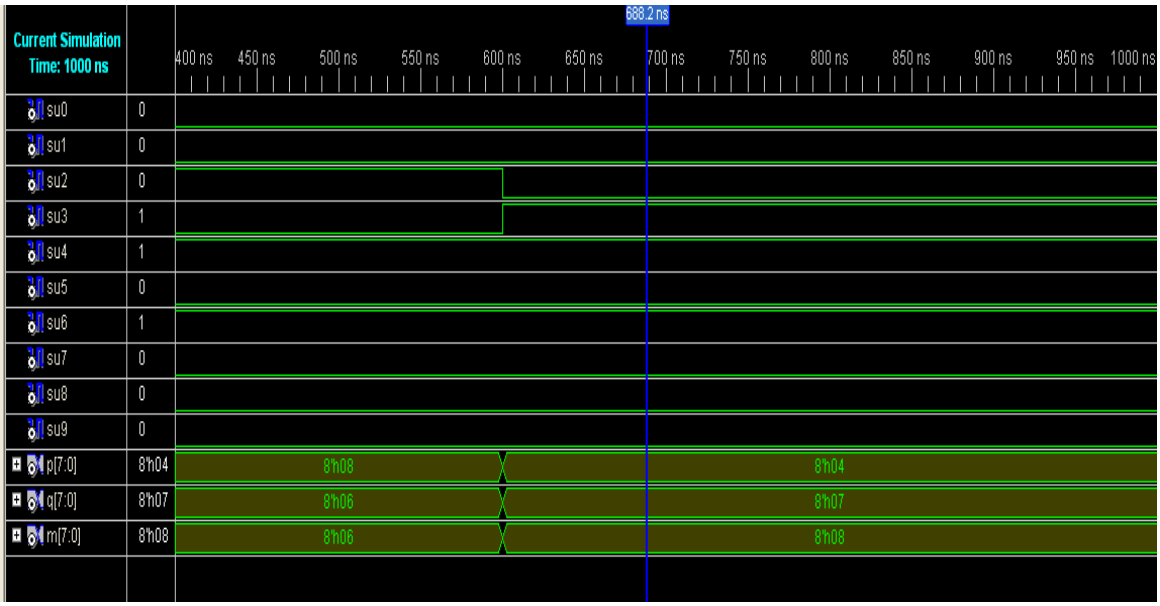

Figure 5: Simulation result of S-MB1 for even bit-width

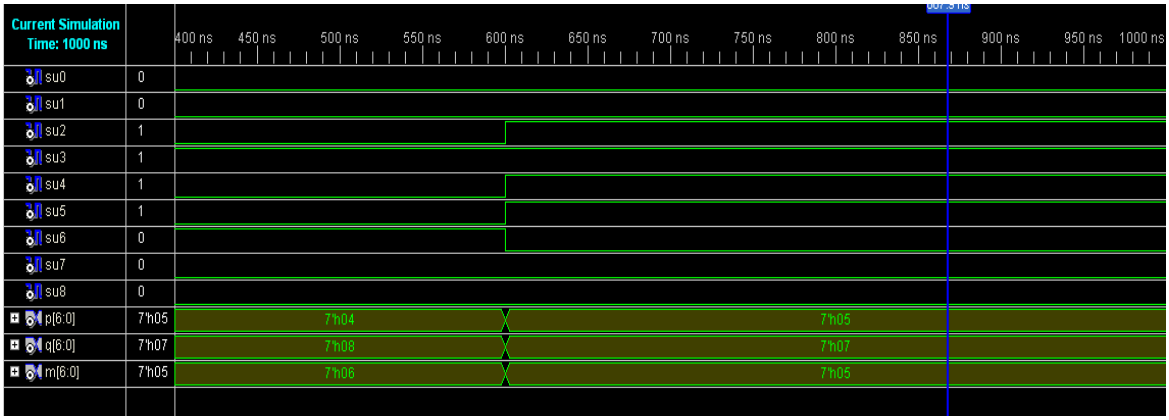

Figure 6: Simulation result of S-MB1 for odd bit-width

6.2 Performance Evaluation

The three different schemes are compared in terms of delay and power consumption is given below.

Design	SMB1		SMB2		SMB3	
	even	Odd	even	odd	Even	Odd
With CLA	20.65	19.42	20.71	19.55	20.79	19.88
With prefix adders	17.95	17.15	19.02	17.3	19.15	17.6

Table 2: Delay comparison of the proposed recoding scheme with existing scheme

Design	SMB1		SMB2		SMB3	
	Even	Odd	even	odd	Even	odd
With CLA	33.17	32.59	33.11	32.57	33.11	32.57
With prefix adders	33.09	32.34	33.02	32.41	33.02	32.41

Table 3: Power consumption comparison of the proposed recoding scheme with existing scheme

7. Conclusion

The design of Add-Multiply operator is used to implement the direct recoding of the sum of two numbers to its MB form. When compared to the existing schemes, the proposed recoding schemes deliver considerable improvements in both delay and power consumption. Regarding the overall performance S-MB2, S-MB3 based schemes will suit for low power applications where the power consumption is less compared to S-MB1.

REFERENCES

[1] Kostas Tsoumanis, Kiamal Pekmestzi, "*An Optimized Modified Booth Recoder for Efficient Design of the Add-multiply Operator*," IEEE Trans. Circuits and Systems-I, regular papers, vol.61, no.4, April 2014.

[2] Chandrasekhar T. Kukade, Raghavendra B. Deshmukh, R. M. Patrikar,"*A Novel parallel multiplier for 2's compliment numbers using Booth's Recoding Algorithm,*"2014 International Conference on Electronic Systems, Signal Processing and Computer Technologies.

[3] A. Amaricai, M. Vladutiu, and O. Boncalo,"*Design issues and implementations for floating-point divide-add fused*," IEEE Trans. Circuits syst. II-Exp. Briefs, vol. 57, no. 4, pp. 295-299, Apr.2010.

[4] M. Daumas and D. W. Matula,"*A Booth multiplier accepting both a redundant or a non redundant input with no additional delay*," in Proc. IEEE Int. Conf. on Application-Specific Syst., Architectures, and Processors, 2000, pp. 205-214.

[5] C. S. Wallace," *A Suggestion for a fast multiplier*," IEEE Trans. Electron. Comput, vol. EC-13, no. 1, pp. 14-17, 1964.

G. Sushma: She is currently pursuing M.Tech in the stream of VLSI, Sree Vidyanikethan Engineering College, Tirupati. Her areas of interest are VLSI Design, digital systems.

M. Naresh Babu: He is currently working as an Assistant Professor in ECE Department, Sree Vidyanikethan Engineering College, Tirupati. He completed M.E in Applied Electronics at Satyabhama University, Chennai, TamilNadu. His areas of interest are VLSI Design, Embedded Systems.