A NEW ARCHITECTURE OF MODIFIED BOOTH RECORDER FOR ADD MULTIPLY OPERATOR USING CARRY SAVE ADDER

K ArunaManjusha[1], B. Naresh[2] and T. S. Arulanath[1]
[1]Department of Electronics and Communication Engineering, Marri Laxman Reddy Institute of Technology, Hyderabad, India
[2]Department of Electronics and Communication Engineering, Institute of Aeronautical Engineering, Hyderabad, India
E-Mail: manjusha.aruna@gmail.com

ABSTRACT

In most of the digital systems addition and multiplication are the crucial arithmetic functions. So generally this is heavily impact on overall performance of digital systems. In existing one adds and multiply operations are done separately. In this paper we are introducing a structured and efficient recoding technique and exploring three different schemes by incorporating them in Fused Add multiply designs. It represents an area efficient design, fast addition and multiplication using Radix based modified booth technique. This technique mainly used to reduce the partial products for the design of many parallel multipliers.

Keywords: arithmetic circuit, fused add-multiply operation, modified booth, save addition, carry look-ahead addition.

INTRODUCTION

In digital signal processing system one of the major important parts is speed performance multiplier. As the scale of integration keeps growing, more and more sophisticated signal processing systems are being implemented on a VLSI chip. These signal processing applications not only demand great computation capacity but also consume considerable amount of energy. While performance and Area remain to be the two major design tolls, power consumption has become a critical concern.

Today's VLSI system design the need for low-power VLSI system arises from two main forces. First, with the steady growth of operating frequency and processing capacity per chip, large currents have to be delivered and the heat due to large power consumption must be removed by proper cooling techniques. Second, battery life in portable electronic devices is limited. Low design directly leads to prolonged operation time in these portable devices. Multiplication is a fundamental operation in most signal processing algorithms. Multipliers have large area, long latency and consume considerable power. Therefore low-power multiplier design has been an important part in VLSI system design. There has been extensive work on low-power multipliers at technology, physical, circuit and logic levels. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas. As a result, a whole spectrum of multipliers with different area speed constraints has been designed with fully parallel.

EXISTING SYSTEMS

In the existing system, several low power approaches have proposed for scan based BIST. In the existing system Add and Multiply units are separately used to perform the operation Z=X. (A+B). The existing system is shown in below diagram. It takes the inputs A and B first then it forwarded into an adder which gives the results Y=A+B and another input X both are given to the multiplier in order to get the final output.

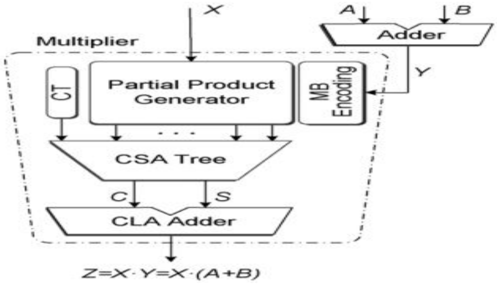


Figure-1. Structure for add and multiply operations.

Adder

We are give two inputs i.e., A and B for this block because of Addition purpose and generate the result as Y. Then these Results are further given to the input of multiplier block.

Modified booth encoding technique

Modified Booth encoding Signal is generated for reduce the number of bits in Multiplier. The inputs Encoder are Y2j+1, Y2j, Y2j-1 and the outputs are derived by and, x-or & not gate. The outputs are One$_j$, two$_j$, S$_j$. The truth table is shown in bellow.

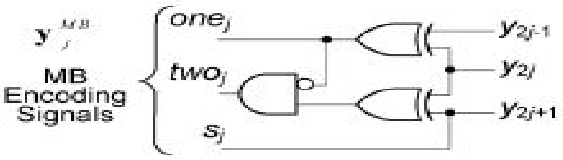


Figure-2. Modified encoding technique.

Table-1. Truth table for modified booth encoding algorithm.

Binary				MB encoding			Input carry
Y_{2j+1}	y_{2j}	Y_{2j-1}	$y^{Mb}{}_j$	Sign= s_j	x1=one $_j$	x2=two$_j$	$c_{in,j}$
0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	2	0	0	1	0
1	0	0	-2	1	0	1	1
1	0	1	-1	1	1	0	1
1	1	0	-1	1	1	0	1
1	1	1	0	1	0	0	0

CSA tree

A carry-save adder is a type of digital adder, used in computer micro architecture to compute the sum of three or more n-bit numbers in binary It differs from other digital adders in that it outputs two numbers of the same dimensions as the inputs, one which is a sequence of partial sum bits and another which is a sequence of Carry bits. The CSA is consists of n full adders.

CLA adder

A carry look-ahead adder (CLA) is a type of adder used in digital logic. A carry look-ahead adder improves speed by reducing the amount of time required to determine carry bits. It can be contrasted with the simpler, but usually slower, ripple carry adder for which the carry bit is calculated alongside the sum bit, and each bit must wait until the previous carry has been calculated to begin calculating its own result and carry bits The carry-look ahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits. The Kogge-Stone adder and Brent-Kung adder are examples of this type of adder.

By using this adder next further stage has no need to wait for the previous stages output carry. The carry look-ahead logic block calculates the input carries for the blocks which are present in the next stage. This concept reduces the waiting time for the input carries. The propagation delay of the CSA adder is less compared with the ripple carry adder.

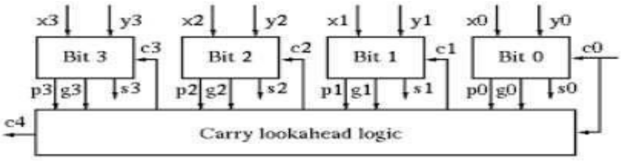


Figure-3. CLA adder.

PROPOSED SYSTEM

In this system, we focus on AM units which implement the operation Z=X. (A+B). The conventional design of the AM operator requires that its inputs and are first driven to an adder and then the input X and the sum Y=A+B are driven to a multiplier in order to get. The drawback of using an adder is that it inserts a significant delay in the critical path of the AM. As there are carry signals to be propagated inside the adder, the critical path depends on the bit-width of the inputs.

In order to decrease this delay, a Carry-Look-Ahead (CLA) adder can be used which, however, increases the area occupation and power dissipation. An optimized design of the AM operator is based on the fusion of the adder and the MB encoding unit into a single data path block by direct recoding of the sum Y=A+B to its MB representation. The fused Add-Multiply (FAM) component contains only one adder at the end (final adder of the parallel multiplier). As a result, significant area savings are observed and the critical path delay of the recoding process is reduced and decoupled from the bit-width of its inputs. In this work, we present a new technique for direct recoding of two numbers in the MB representation of their sum.

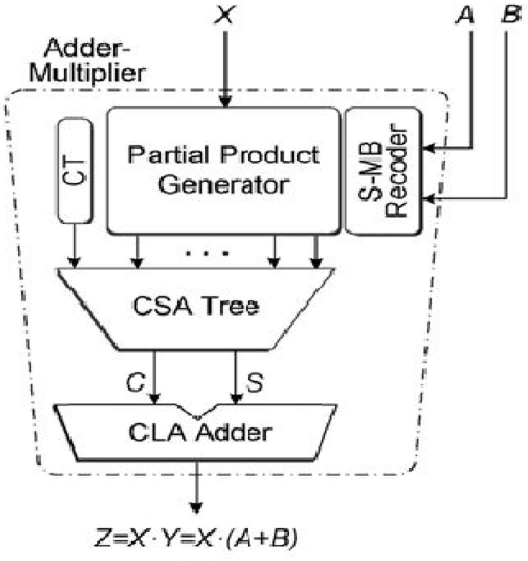


Figure-4. Structure for modified booth adder and multiplier.

S-MB3 recoding scheme even and odd number of bits

The third scheme implementing the proposed recoding technique is S-MB3. It is illustrated in detail for even and odd bit-width of input numbers

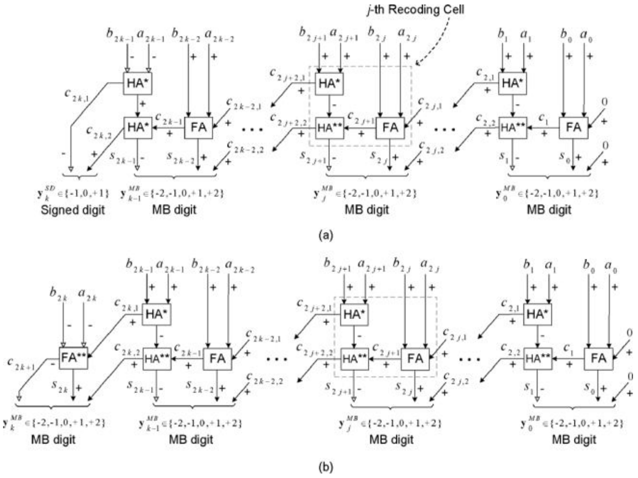
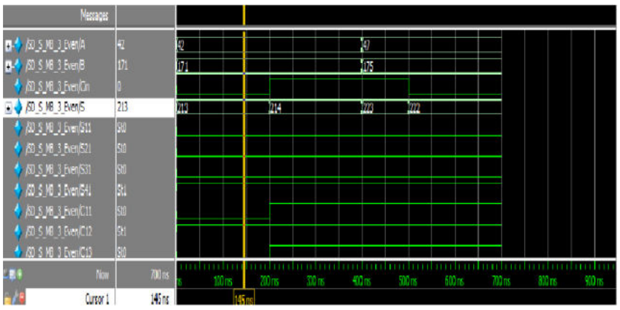

j-th Recoding Cell
b_{2k-1} a_{2k-1} b_{2k-2} a_{2k-2}
b_{2j+1} a_{2j+1} b_{2j} a_{2j}
b_1 a_1 b_0 a_0
b_{2k} a_{2k}
HA*
HA**
FA
FA**
$c_{2k,1}$ $c_{2k,2}$ c_{2k-1} $c_{2k-2,1}$ $c_{2k-2,2}$ c_{2k+1}
$c_{2j+2,1}$ $c_{2j+2,2}$ c_{2j+1} $c_{2j,1}$ $c_{2j,2}$
$c_{2,1}$ $c_{2,2}$ c_1
s_{2k} s_{2k-1} s_{2k-2} s_{2j+1} s_{2j} s_1 s_0
$\mathbf{y}_k^{SD} \in \{-1,0,+1\}$
Signed digit
$\mathbf{y}_{k-1}^{MB} \in \{-2,-1,0,+1,+2\}$
$\mathbf{y}_j^{MB} \in \{-2,-1,0,+1,+2\}$
$\mathbf{y}_0^{MB} \in \{-2,-1,0,+1,+2\}$
$\mathbf{y}_k^{MB} \in \{-2,-1,0,+1,+2\}$
MB digit
(a)
(b)

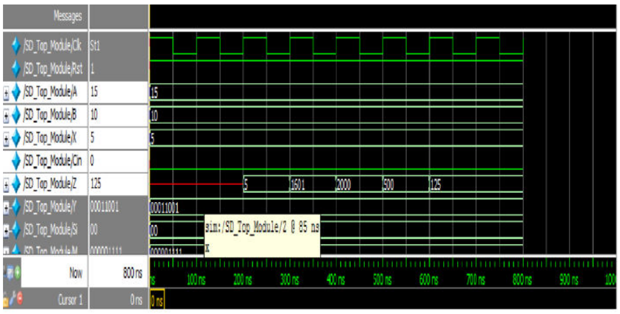
Figure-5. S-MB3 recording for even and odd no. of bits.

4. RESULTS AND DISCUSSIONS



```
Messages
/SD_S_MB_3_Even/A     42    42  47
/SD_S_MB_3_Even/B     171   171  175
/SD_S_MB_3_Even/Cin   0
/SD_S_MB_3_Even/S     213   213  214  223  222
/SD_S_MB_3_Even/S11   St0
/SD_S_MB_3_Even/S21   St0
/SD_S_MB_3_Even/S31   St0
/SD_S_MB_3_Even/S41   St1
/SD_S_MB_3_Even/C11   St0
/SD_S_MB_3_Even/C12   St1
/SD_S_MB_3_Even/C13   St0
Now        700 ns
Cursor 1   145 ns
```

Figure-6a. SMB3 even.



```
Messages
/SD_Top_Module/Clk   St1
/SD_Top_Module/Rst   1
/SD_Top_Module/A     15         15
/SD_Top_Module/B     10         10
/SD_Top_Module/X     5          5
/SD_Top_Module/Cin   0
/SD_Top_Module/Z     125        5  1601  2000  500  125
/SD_Top_Module/Y     00011001   00011001
/SD_Top_Module/Si    00         00
/SD_Top_Module/M     000001111  000001111
sim:/SD_Top_Module/Z @ 85 ns
X
Now        800 ns
Cursor 1   0 ns
```

Figure-6.b. SMB3 odd.

Table-2. SMB 3 based ADD multiply operator (even).

Device utilization summary			
Logic utilization	**Used**	**Available**	**utilization**
Number of 4 input LUTs	14	7.168	1%
Logic distribution			
No.of occupied slices	9	3.584	1%
No.of slices containing only related logic	9	9	100%
No.of slices containing unrelated logic	0	9	0%
Total Number of 4 input LUTs	14	7.168	1%
No.of bonded LUTs	26	141	18%
Total equivalent gate count for design	877		
Additional JTAG count for IOBs	1.248		

Table-3. SMB 3 based ADD multiply operator (odd).

Device utilization summary			
Logic utilization	**Used**	**Available**	**utilization**
Number of 4 input LUTs	14	7.168	1%
Logic distribution			
No.of occupied slices	9	3.584	1%
No.of slices containing only related logic	9	9	100%
No.of slices containing unrelated logic	0	9	0%
Total Number of 4 input LUTs	14	7.168	1%
No.of bonded LUTs	29	141	20%

Total equiivakent gate count for design	87		
Additional JTAG count for IOBs	1.248		

It considers that c0, 1 = 0 and c0, 2. It builds the digits, 0 ≤ j ≤ k -1, based on s2j+1, s2j and c2j, 2. The negatively signed bit s2j+1 is produced by a HA** in which drive c2j+1 and the output sum (negatively signed) of the HA* of the recoding cell with the bits a2j+1, b2j+1 as inputs. The carry and sum outputs of the HA** are calculated

CONCLUSION AND FUTURE SCOPE

This paper has proposed a multiplier, fused Add-multiply unit which can sacrifice the accuracy of addition and multiplication operations for saving the power consumption. While performing addition and multiplication operation in the single-precision mode it was also capable of reducing the area, critical path and hardware complexity. In future it can be implemented in FPGA using Verilog code. The proposed architectures show the best performance compared with the previous method of the FAM unit.

The power saving may be increased if the following conditions are considered in the future low power VLSI design. The bit size may be increased i.e., number of bits considered may be increased in the encoding scheme using Modified Booth Technique. The power consumption can be reduced by improving the partial product compression ratio. This concept of Fusing Technique can also be implemented in Radix-8 for area efficiency and low delay.

ACKNOWLEDGEMENT

The authors would like to thank the management of MLR Institute of Technology for proving this best opportunity.

REFERENCES

[1] C. S. Wallace. 1964. A suggestion for a fast multiplier. IEEE Trans. Electron. Comput. EC-13(1): 14-17.

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

[3] Z. Huang and M. D. Ercegovac. 2005. High-performance low-power left-to right array multiplier design. IEEE Trans. Comput. 54(3): 272-283.

[4] A. Amaricai, M. Vladutiu and O. Boncalo. 2010. Design issues and implementations for floating-point divide-add fused. IEEE Trans. Circuits Syst. II-Exp. Briefs. 57(4): 295-299.

[5] E. E. Swartzlander and H. H. M. Saleh. 2012. FFT implementation with fused floating-point operations. IEEE Trans. Comput. 61(2): 284-288.

[6] Hsin-Lei Lin, Robert C. Chang, Ming-Tsai Chan. 2004. Design of a Novel Radix-4 Booth Multiplier, IEEE Asia-Pacific Conference on Circuits and Systems. 2: 837-840.

[7] M. Sheplie. 2004. High performance array multiplier. IEEE transactions on very large scale integration systems. 12(3): 320-325.

[8] Robert C. Ghest, Saratoga; Hua-ThyeChua, Cupertino; John M. Birkner, Santa Clara. 1979. High speed combinatorial digital multiplier, United States Patent, US 4153938A.

[9] Vincent P. Heuring, Harry F. Jordon. 2003. Computer Systems Design and Architecture, Pearson Education, Singapore.

[10] Y.N. Ching. 2005. Low-power high-speed multipliers. IEEE Transactions on Computers. 54(3): 355-361.