VLSI Implementation of an efficient Fused Add-Multiply Unit using Constant-time addition

S. Durgadevi
Dept. of Electronics and Communication Engineering
College of Engineering Guindy, Anna University
Chennai, Tamilnadu, India

Dr. R. Seshasayanan, Associate Professor,
Dept. of Electronics and Communication Engineering
College of Engineering Guindy, Anna University
Chennai, Tamilnadu, India

***Abstract*— Digital Signal Processing (DSP) applications widely use complex arithmetic operations. This paper introduces an implementation of one of the complex arithmetic operations, the Fused Add-Multiply (FAM) operation. In general, a separate adder and a multiplier are used to perform Add-Multiply operation. The use of separate adder and multiplier adds significant area and delay. So, an efficient Fused Add-Multiply Unit is implemented using the Radix-8 Modified Booth Recoder. The Modified Booth Recoder design is based on the redundant logic and Constant-time addition. Instead of deriving the adder output as in conventional methods, the recoder recodes the adder input directly to MB form thus decreasing delay. The multiplier unit uses the Radix-8 Modified Booth algorithm. The delay and the area of the redundant adder/recoder have been reduced by 2% and 57% respectively. In comparison with the conventional design, the overall delay of the FAM has been reduced by 15%. On analyzing the Radix-8 FAM unit, it has been observed that the newly modified design yields better performance in terms of area and delay.**

***Key words*—Modified Booth Recoder, Constant-time Addition, Redundant Number system, Fused Add-Multiply Unit.**

I. INTRODUCTION

Digital Signal Processing is extensively used in the domains of multimedia, signal processing, etc. Since most of the DSP applications are based on intensive kernels, a large number of arithmetic operations are carried out. Depending on the allocation and architecture of these arithmetic units, the performance of the DSP systems varies. By sharing the common data among different arithmetic operations, significant performance improvement can be achieved. This improvement can be observed from the implementation of Divide-Add Fused [1] operation and fused floating point operations [2].

In most of the DSP applications, an addition operation is often successive to multiplication operation. To perform multiply-add or add-multiply or multiply-add-multiply operations, instead of using separate blocks, if a single dedicated unit is used, better performance can be achieved. This can be observed from the Multiply-Accumulator (MAC) and Multiply-Add (MAD) unit designs in [3] [4]. Other than the MAC/MAD operations, many DSP applications depend on Add-Multiply (AM) operations. The direct design of the AM unit requires that the output of the adder to be first driven to the input of the multiplier. The use of separate adder and multiplier increases both the critical path delay and area. To minimize the carry propagation delay, Carry Look Ahead (CLA) adder or some other efficient adder can be used. But this increases the area. So to improve the performance of the AM unit, designs that share data were implemented based on the fusion technique and Carry free addition [5]. The fusion technique is employed based on the direct recoding of the addition of two numbers to its Modified Booth (MB) form (equivalent to Carry Save form [6]). The use of constant-time addition ensures that the execution delay is independent of the input bit widths. In [7], the authors Lyu and Matula introduced a novel signed-bit recoder that transforms the redundant numbers to MB recoding form. In [8], a two-stage recoder has been proposed that converts a carry save form number to its MB form. In this, the first stage converts an input number in the carry save form to signed digit form, and then in the second stage, recoding is done to match with the form that MB digits request. In [9], Zimmerman and Tran presented an improved design of [7], which yields optimized design in terms of both critical path and area. The authors Daumas and Matula in [10], had proposed a recoder that transforms input in carry save form to the respective borrow-save form with the critical path fixed.

The existing recoding schemes may provide efficient implementation, but the disadvantage is that they use complex manipulations at bit level with the circuits implemented in gate level. In [11], the authors proposed an efficient Sum to Modified Booth (S-MB) recoder for implementing AM unit using a Radix-4 algorithm. The S-MB recoder in [11] is efficient and structured. With the increase in the radix number, the number of partial products gets reduced and hence the hardware and delay. So the main focus of this work is the design and implementation of Radix-8 Modified Booth Recoder that yield better performance when implemented with Add-Multiply Unit (AM). Compared to the Radix-4 design [11], the modified Radix-8 MB Recoder design is simple, structured, better in performance and can be easily modified for any higher radix. This proposed FAM unit can be used in Signal Processing applications such as Fast Fourier Transform (FFT). Figure 1 shows the conventional and modified design of AM Unit. The Fused Add-Multiply design is implemented using structural Verilog HDL and synthesis is done using Xilinx 9.1 tool.

978-1-4799-6818-3/15/$31.00 © 2015 IEEE

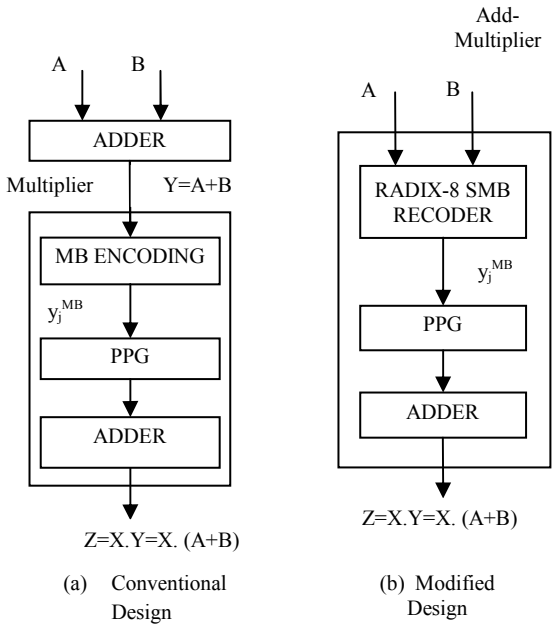

Fig. 1 Add Multiply Unit (a) Conventional Design (b) Modified Design

The rest of the paper is organized as follows: In Section II, the recoding logic (a) redundant binary number system and b) radix-8 modified booth multiplier is discussed. In Section III, design of Radix-8 Modified Booth recoder and its implementation is discussed. In Section IV, Experimental analysis is given. In Section V, summary is presented.

II. RECODING LOGIC

A. Redundant Binary Number system

A radix-r redundant signed-digit number system has digits from digit set S = {-β, - (β-1),… -1, 0, 1,…, α}, where, $1 \leq \beta$, $\alpha \leq r - 1$. The digit set S containing more than r values gives multiple representations for any number in signed digit format. Hence, it is named Redundant Number System (RNS). An attractive property of redundant signed-digit numbers is carry-free addition used in [7] and [10]. Constant time addition [14] works well only if the output is in redundant representation. The way in which the redundancy is introduced can greatly affect the performance of digit set conversion and constant-time addition. Beginning at the LSB of the given number, each digit can be rewritten as an interim sum in the new digit set and a transfer into the next higher digit position. Adding the interim sum and the incoming transfer digit yields a new digit and creates no new transfer. Figure 2 shows an addition example, using the redundant binary number system. To add two n bit binary numbers in [11], two bits are grouped but here three bits are grouped in radix-8 design so as to limit the carry propagation. More generally the group size can be greater than 3. A larger group size reduces the hardware complexity, but adds propagation delay as the group size increases.

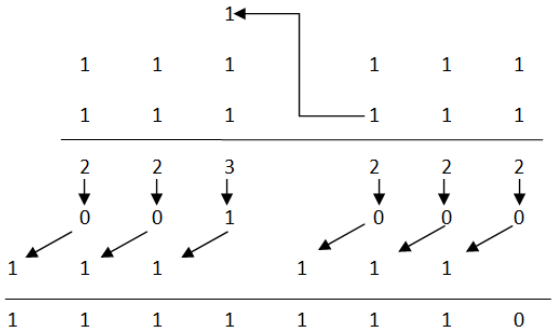

Fig. 2 Addition implemented considering redundancy and look-ahead scheme

A single stage propagation of transfers can be eliminated by a simple look-ahead scheme; that is, instead of first computing the carry bit based on the digits x_{i-1}, y_{i-1} and then combining it with the interim sum, we can determine s_i directly from x_i, y_i, x_{i-1} and y_{i-1}. This may make the adder logic somewhat complex, but in general results in higher speed.

B. Modified Booth Form

Of all the multipliers, the Modified Booth multiplier is the efficient one due to reduced number of partial products arrays. It is a redundant signed digit radix-4 encoding technique. Its main advantage is that it reduces the number of partial products to half in multiplication. Also radix-8 modified booth multiplier is a redundant signed digit radix-8 encoding technique. It depends on encoding the two's complement number (multiplier) to reduce the number of partial products to n/3.This is a major advantage compared to radix-4 encoding scheme.

Consider the multiplication of two numbers X and Y, *n* bits each. The multiplier Y can be decomposed in MB form as:

$$Y = < y_{n-1}, y_{n-2}, y_{n-3}, \dots, y_1, y_0 >$$

$$= \sum 2^{3i} y_i^{MB} \quad (1)$$

where y_i^{MB} is an encoding of i^{th} group, with the encoding in the range $y_i^{MB} \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$. Each digit correspond to the four consecutive bits $y_{3i+2}, y_{3i+1}, y_{3i}, y_{3i-1}$ with one bit overlapped and considering that $y_{3i} = 0$. A radix-4 multiplier generates n/2 partial products while radix-8 generates n/3 partial products. So, a radix-8 based multiplier generates fewer partial products when compared to a radix-4 based multiplier. This reduces the adder in the accumulation of the partial products. But in radix-8 there lies complexity in the computation of each partial product. In particular, generation of partial products for $y_i^{MB} = \pm 3$ requires the computation of the hard multiple 3X and -3X. This cannot be obtained by simply shifting or complementing the multiplicand. This will add extra delay to the design. So, based on the timing and application of the design, the choice of whether radix-4 or radix-8 is made.

Assume that X and Y are bit vectors of widths n-2 and 3m-1 respectively. The multiplier Y is now partitioned into m 3-bit slices, $y[3i+2:3i],\ i = 0,1,..,m-1$, and encoded as

$$y_i^{MB} = y[3i-1] + y[3i] + 2y[3i+1] - 4y[3i+2] \quad (2)$$

In grouping, the least significant block uses only three bits of the multiplier, and the fourth bit is assumed to be zero. The overlap is necessary to know the operation occurred in the previous block, as the MSB of the previous block acts as a sign bit. Using the Equation 2, MB digits are formed. Then, referring to the encoding table given in Table II, respective operations are done to compute the partial products.

III. MODIFIED BOOTH RECODER

A. Design of signed bit full adders

In the design of radix-4 modified booth recoder [11], signed HAs implementing the relation 2c-s=p+q, 2c-s=-p+q and FAs $2c_0$-s=p-q+c_i , -$2c_0$ + s=-p-q+c_i have been used. Here to design Radix-8 Recoder, only signed bit FAs is used. More specifically, this Radix-8 Recoder is designed using a Full Adder (FA) and a signed bit Full Adder (FA*). FA implements the relation $2c_0$ +s=p+q+c_i while FA* implements the relation $2c_0$-s=p-q+c_i. Table I shows the truth table of FA*, which uses p, q and c_i as the binary inputs and c_o, s as the output carry and sum with the relation 2.c_o-s=p-q+c_i producing the output values {-1, 0, +1, +2}.

TABLE I
FA* OPERATION

Inputs			Output Value	Outputs	
p(+)	*q(-)*	*c_i(+)*		*c_o(+)*	*s(-)*
0	0	0	0	0	0
0	0	1	+1	1	1
0	1	0	-1	0	1
0	1	1	0	0	0
1	0	0	+1	1	1
1	0	1	+2	1	0
1	1	0	0	0	0
1	1	1	+1	1	1

B. Modified Recoding Technique

Starting from the LSB, three consecutive bits of the input A (a_{2j-1}, a_{2j}, a_{2j+1}) and B (b_{2j-1}, b_{2j}, b_{2j+1}) are recoded into a single Modified Booth (MB) digit using the designed recoder as in the Figure 3. In radix-4, three bits are grouped but here in radix-8, four bits are grouped to form a MB digit. The most significant of them is negatively weighted while the three least significant of them is positively weighted as in Equation 1. So using signed bit arithmetic, the bits are converted to MB form. Assuming both A and B to be in 2's complement form, the MB digit is formed using the Equation 2. Based on the equation $b_{3j+1} = 2b_{3j+1} - b_{3j+1}$, b_{3j+1} is driven to the FA* as negatively signed, assuming $b_{-1} = 0$ and $c_0 = 0$.

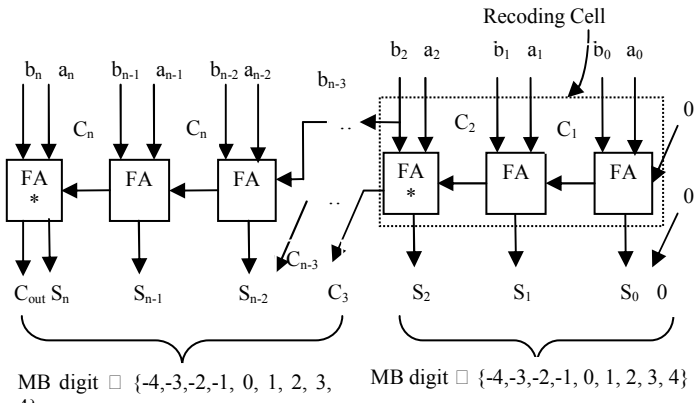

Fig. 3 S-MB Recoding Scheme

C. Partial Product Generation

Partial products are generated based on the radix-8 encoding scheme [13] shown in the Table II. 2X is obtained by shifting X to the left by one position. 4X is obtained by shifting X to the left by two positions. –X is obtained by taking 2's complement of X. –2X is obtained by shifting –X to the left by one position. –4X is obtained by shifting –X to the left by two positions. 3X is obtained by adding the pre-computed value 2X and X. –3X is obtained by adding –2X and –X. This is implemented using a 16:1 Multiplexer, with each input of the multiplexer carrying an operation of the multiplicand multiples.

TABLE II
RADIX-8 MODIFIED BOOTH ENCODING TABLE

y_{3i+2}	y_{3i+1}	y_{3i}	y_{3i-1}	y_i^{MB}
0	0	0	0	0
0	0	0	1	+1X
0	0	1	0	+1X
0	0	1	1	+2X
0	1	0	0	+2X
0	1	0	1	+3X
0	1	1	0	+3X
0	1	1	1	+4X
1	0	0	0	-4X
1	0	0	1	-3X
1	0	1	0	-3X
1	0	1	1	-2X
1	1	0	0	-2X
1	1	0	1	-1X
1	1	1	0	-1X
1	1	1	1	0

D. Accumulation of Partial Products

A carry-save adder is a type of digital adder, used to compute the sum of three or more than three bit numbers in binary. Carry save adder tree [12] is used to accumulate the partial products. The CSA is one of the most efficiently used techniques to speed up digital designs that deal with multiple operands for addition and multiplication. A CSA reduces three operands into two operands without any carry propagation rather saving the carry in the next significant bit position. The partial products are added using this carry save adder by properly shifting to the left. However, there will be some carry propagation at the last stage. At that stage fast carry look-ahead adder is used to reduce the delay. Figure 4 shows a basic carry save adder scheme.

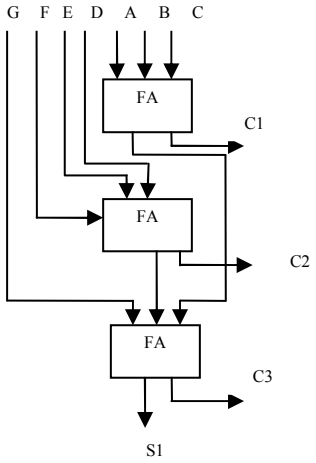

Fig. 4 General CSA Tree adding 6 digits

IV. PERFORMANCE EVALUATION

In order to validate the performance of the proposed system, the architecture of both existing and proposed systems is described in Verilog HDL and targeted to Xilinx Virtex 5 XC5VLX30-3FF324 device. The existing and proposed architecture is compared in terms of hardware complexity and timing delay. Since only the conventional and signed full adders are used, complex manipulations at the bit level are highly reduced. Comparison of delay and area for the design in [9], radix-4 and radix-8 recoder is given in the Table III. The delay comparison of the FAM Unit is given in Table IV. From the Figure 5 and 7, reduction in the delay can be observed for the radix-8 design. Figure 6 shows the reduction in the number of slice LUTs. Figure 8 shows the Modified Booth Recoder output. Figure 9 shows the output of the entire AM unit implemented using Radix-8.To get the accurate timing result, Post Place and Route (PAR) is done. In the proposed circuit, since the same recoding cell is used for the remaining bits with executions in parallel, the critical path delay is the execution time of one single recoding cell. This is due to property of constant time addition. Delay has been reduced by 15%. However there will be some slight increase in the area when we go for higher radix implementation. This will be due to the hard multiple calculations for higher radices.

TABLE III
PERFORMANCE COMPARISON OF MB RECODER

Design	No of slice LUTs	Delay (ns)	Area delay product
Work in [9]	26	4.454	115.804
Radix-4 MB Recoder Unit	11	3.607	39.677
Radix-8 MB Recoder Unit	11	4.358	47.938
Percentage reduction (%)	57	2	59

TABLE IV
DELAY COMPARISON OF VARIOUS BLOCKS IN AM UNIT

Design	Partial Product Generation	Carry Save Adder	S-MB Design
Radix-8 AM Unit	7.144	6.415	10.511
Radix-4 AM Unit	5.077	9.137	12.340
Percentage reduction (%)	-	30	15

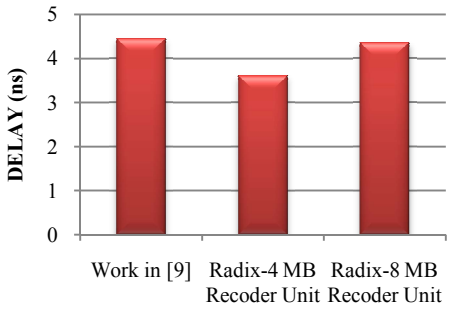

Fig. 5 Delay Comparison of MB Recoder unit

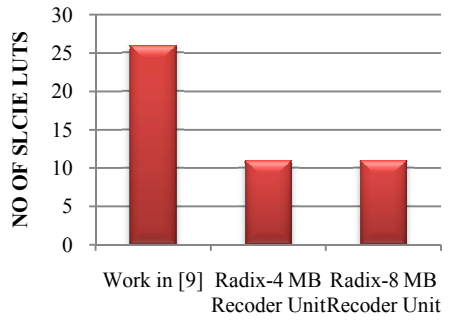

Fig. 6 Comparison of number of slices occupied by MB Recoder

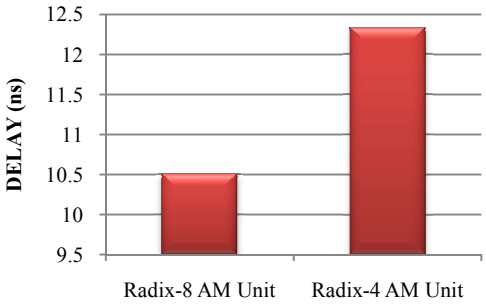

Fig. 7 Delay Comparison of AM Unit

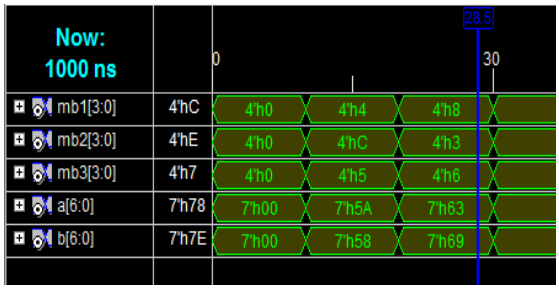

Fig. 8 MB Recoder Output

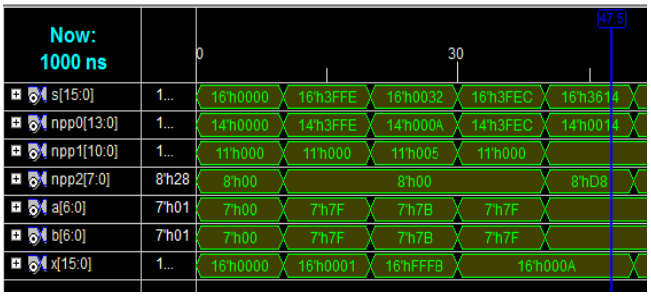

Fig. 9 Fused Add-Multiply Output

V. Conclusion

In this paper, a modified approach has been presented to design the radix-8 modified booth recoder for directly recoding the sum of two numbers. The proposed S-MB algorithm is simple to be modified for the signed and unsigned numbers. Simulation and synthesis for FPGAs are accomplished on XILINX ISE Software (Xilinx ISE 9.1i version) for Virtex 5 series FPGA, target device (XC5VLX30). It can be seen from the Table III that the delay and the area of the modified booth recoder has been reduced by 2% and 57% respectively. The performance of a multiplier unit is mostly affected by partial product accumulation unit. Since in radix-8 design, the partial products are reduced by n/3, performance improvement can be seen in the designed Add-Multiply unit when compared to the radix-4 Add-Multiply unit. This can be observed from the Table IV. The performance of the proposed S-MB technique is evaluated by comparing with the existing technique. It can be observed from the implementation that the delay gets reduced by 15% using this constant-time addition based FAM unit. The proposed recoding schemes, when implemented in FPGA, yield considerable performance improvements in comparison with the most efficient recoding schemes found in literature. In future, this concept can be extended for higher radix based on the application.

References

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

[2] E. Swartzlander and H. H. M. Saleh, "FFT implementation with fused floating-point operations," *IEEE Trans.Comput., v*ol. 61, no. 2, pp. 284–288, Feb. 2012.

[3] O. Kwon, K. Nowka, and E. E. Swartzlander, "A 16-bit by 16-bit MAC design using fast 5: 3 compressor cells," *J. VLSI Signal Process. Syst.,* vol. 31, no. 2, pp. 77–89, Jun. 2002.

[4] L.-H. Chen, O. T.-C. Chen, T.-Y.Wang, and Y.-C. Ma, "A multiplication-accumulation computation unit with optimized ompressors andminimized switching activities," in *Proc. IEEE Int, Symp. Circuits and Syst.*, Kobe, Japan, 2005, vol. 6, pp. 6118–6121.

[5] Klaus Schneider and Adrian Willenbucher "A New Algorithm for Carry-Free Addition of Binary Signed-Digit Numbers," *IEEE 22nd International Symp. Field-Programmable Custom Computing Machines, 2014*

[6] Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*. Oxford: Oxford Univ. Press, 2000.

[7] C.N. Lyu and D.W.Matula, "Redundant binary Booth recoding," in *Proc. 12th Symp. Comput. Arithmetic*, 1995, pp. 50–57

[8] W.C.Yeh, C.N, Jen, "A High performance Carry-Save to Signed-Digit Recoder for Fused Addition and Multiplication," *IEEE Trans.Comput.,2000.*

[9] R. Zimmermann and D. Q. Tran, "Optimized synthesis of sum-of-products," in *Proc. Asilomar Conf. Signals, Syst. Comput., Pacific Grove*, Washington, DC, 2003, pp. 867–872.

[10] 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.

[11] Kostas Tsoumanis, Sotiris Xydis, Constantinos Efstathiou, Kiamal Pekmestzi, and Nikos Moschopoulos,"An Optimized Modified Booth Recoder for efficient designof the Add-Multiply operator", *IEEE Trans. Circuits Syst. I- Regular papers*, vol. 61, no. 4, Apr 2014.

[12] Shoab Ahmed Khan, *Digital Design of Signal Processing Systems, A Practical Approach,* John Wiley & Sons, ltd, 2011.

[13] Paladugu Srinivas Teja, "Design of radix-8 booth multiplier using koggestone adder for high speed Arithmetic applications," *EEIEJ*, Vol. 1, No. 1, February 2014.s

[14] D S. Phatak, Tom Goff, Israel Koren, "Constant-Time Addition and Simultaneous Format Conversion Based on Redundant Binary Representations", *ASILOMAR, 33rd Conference on Signals, Systems and Computers*, 1999.