International Journal of Psychosocial Rehabilitation, Vol. 23, Issue 05,
2019 ISSN: 1475-7192

MAC unit suitable for Reconfigurable Systems based on 6-Input LUT

[1]Satya Ranjan Das, [2]Priyabrata Pattanaik

Abstract--- *this work presents the designing of an alternative multiply-accumulate (MAC) unit fused by redundant arithmetic. A "double carry-save output encoding" is used in this design. It is a suitable structure for reconfigurable systems which is based on 6 input LUT. A high performance system could be obtained without any pipelining by the employment of (6,3) counters in the reduction of accumulation operations and partial products as a logic depth which is provided is least in amount. The carry propagation does not affects this proposed system as MAC structure implements a redundant arithmetic scheme. The MAC unit designed here comprise of an accumulate output of 40 bits and a multiplier of s16x16 bits. The synthesis is done on 'AlteraTM Stratix III FPGA board' and better performance in comparison with conventional pipelines is achieved.*

Index Terms--- *multiply accumulate unit, LUT, FPGA, XilinxTM, counters.*

I. INTRODUCTION

Mathematical operations namely, convolution, matrix multiplication, filtering operations etc use "Multiply-accumulate (MAC) units". Thus, "digital signal processors" and coprocessors have MAC units as their crucial elements [1][2]. The applications of signal processing require reconfigurable systems such as FPGAs. This work presents a MAC unit of low latency and high throughput for FPGAs based on "high performance 6-input look up table (LUT)". LUT elements of four inputs are needed for building conventional FPGAs i.e., mapping of synthesized logic circuit is done into memory blocks of four inputs in FPGA. LUT elements having six inputs was used for building "Stratix families of AlteraTM" and "Virtex families of XilinxTM" [3][4]. A faster efficiency of logic is achieved by LUT structures having higher inputs because mapping of complicated building blocks is done into depth of less logic. A delay for operations of addition is constant, which popularizes redundant number systems. A constant delay is included in operations of addition that are not dependent in size of digits of input operands in redundant number systems [5]. The carry free arithmetic is created by two ways. Namely, carry save arithmetic and signed digit arithmetic. Hardware implementations are plenty that employ carry free fast arithmetic for operations [6]–[8]. The performance of carry-save arithmetic is similar to signed digit arithmetic because redundant arithmetic operations form basis of both of them. Also elements of conventional adders can be used for building carry-save arithmetic. The partial products are efficiently reduced by using carry save adder trees [9][10]. This work generates a MAC unit based carry save addition for employment of double carry save encoding at output. The traditional arithmetic (carry-save) results a sumbit for every

Department of Engineering, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar,
satyadas@soa.ac.in, priyabratapattanaik@soa.ac.in

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

digit and a redundant output based on one carry, i.e. value set of (0,1,2) is given to each digit [11], [12]. This work presents an implementation by a value set (0,1,2,3) is given to each digit for representing each output digit. This representation is an advantage for FPGAs based on LUT of 6 inputs, because (6,3) counters are used for arithmetic operations. The partial product reduction recommends using (6,3) counters in LUT devices of 6 inputs because of a single atomic delay of (6,3) counters [13]. There are 6 inputs in (6,3) counter in every parallel input, which is capable of being synthesized in one LUT delay. Also (4,2) compressors are employed in a reduction technique of partial products, which is an efficient block of VLSI design. However, a single level cell cannot be used for synthesis of LUT designs of 6-inputs. A signed digit arithmetic is an alternative technique of partial product reduction [14], [15].

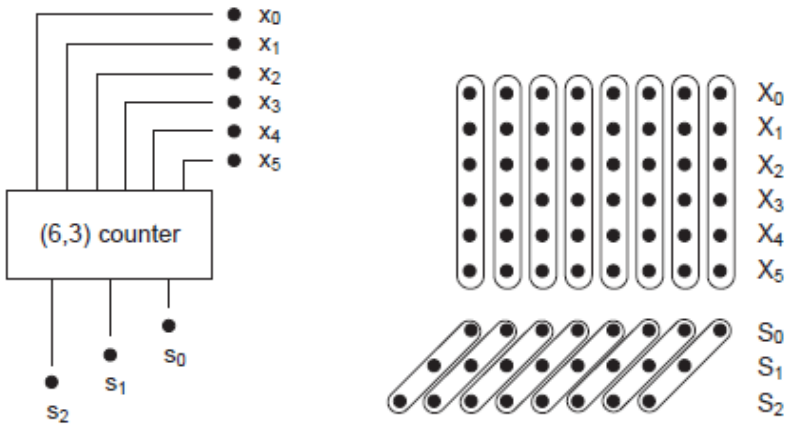

Fig. 1 Representation of (6,3) counter
(a) Single bit, and (b) Multi bit

The synthesis of MAC unit is done for "AlteraTM's Stratix-III FPGA" and comparison is done with conventional MAC units that are made by using several hardware, software and pipelined multipliers etc. The analysis of hardware cost, throughput and speed metrics is done.

II. ARCHITECTURE OF REDUNDANT MAC

The MAC unit proposed in this work uses a modified Booth encoding by using radix 4 for generation of partial product. A (6, 3) counter tree unifies operations of addition and multiplication. The addition operands are reduced by using counters. Figure 1(a) depicts addition of six input operands producing output values lying between 0 and 6. Figure 1(b) shows addition of operands of 6 of 8 bit binary inputs and are fed into an array (6,3) and there is a reduction from six to three in the addition of input operands.

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

Table 1 Metrics of Reduction Operator

	Binary signed-digit		(4,2) compressor		(6, 3) counter	
Bit-width	Delay (ns)	Area (LUT)	Delay (ns)	Area (LUT)	Delay (ns)	Area (LUT)
16-bit	2,77	80	1,84	48	1,15	48
24-bit	3,45	120	1,91	72	1,13	72
32-bit	3,27	160	1,95	96	1,15	96
64-bit	3,19	320	1,93	192	1,15	192

There are certain advantages of using (6,3) counters and techniques of implementation for FPGA based on 6 inputs LUT [16]. Generally, it is not preferred to use a "carry-save reduction tree" in FPGA system for partial product reduction because more area is required in carry propagation schemes [17][18]. However, there is a linear dependency of carry propagate schemes on width of bit and the efficiency of carry propagation decreases with the growth of bit-width. The area requirement is doubled by carry save scheme when comparison is done with schemes of carry-propagation [18]. The FPGA multipliers can use counter trees and binary signed digit adder trees if area is not the concern. The schemes of performance and area requirements for different reduction operators is shown in table 1. The best performance among different reduction operators for FPGA based on 6 inputs LUT is given in table 1. The same quantity of resources are required by (6,3) counters and (4,2) compressors where (6,3) counters are faster. More delay and area of an operator is required by digit scheme of binary scheme. All the redundant operators are compared; (4,2) reduction array, (6,3) counter array and digit operators of binary signed that have half input operands and similar throughput performance. A regular carry save scheme is proposed which is suited for operations of multiplication-addition that does not involve carry propagation [19]. There are two components of redundant number format whereas multiple add operations are done without carry propagation by a scheme of regular carry save. There are three components in a redundant number format but there are three components in double carry save scheme. A registered feedback path comprise of an accumulator output and a multiplier path in a MAC unit. The architecture proposed here includes a multiplication in which a standard Booth encoding scheme is used having sign extension up to outputs of 40 bits. The multiple operations of multiply-accumulate should not be overflown by output of accumulate. Fig. 2 depicts modified Booth encoding of radix-4 with sign extension. A generated partial product is represented by each line. The rules written in table II shows the generated partial products. The multiplier bits are used for generation of each partial product. The partial product has sign bit equal to ei in every line. When sign Y is inverted then si equals 1 in each line. Thus, the sign Y is positive if it is 0. The output of multiplier is extended from 32 bits to 40 bits by padding 1's to booth encoding stage's last partial product. This is done for preventing output from overflowing after operations of multiply-accumulate. The Booth encoding scheme has all the details explained [20]. The number of partial products are halved by Booth encoding. Thus, there are only eight partial products where last line results in s7 of another operand

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

that is added together. This results in output of Booth encoding into addition of 9 operands. O's are padded in empty slot of each addition operand rows for implementation of hardware easily.

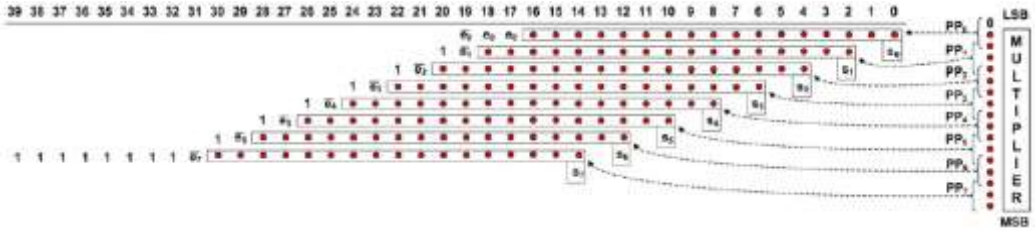

Fig. 2 Modified scheme of Booth encoding

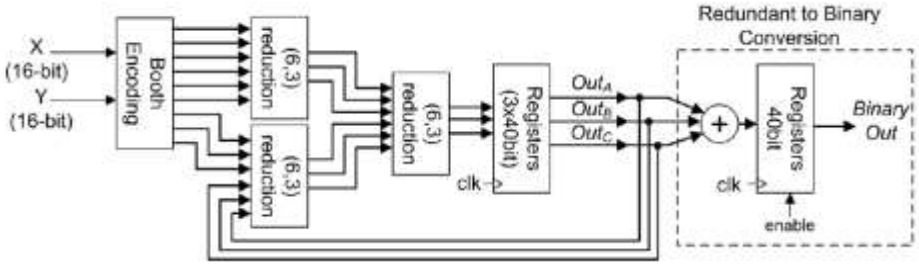

Fig. 3 Proposed architecture of multiply accumulate

Table II Modified Booth encoding of radix 4

x_{2i+1}	x_{2i}	x_{2i-1}	Partial Product	Booth Selector Output
0	0	0	0	0
0	0	1	Y	y_j
0	1	0	Y	y_j
0	1	1	$2Y$	y_{j-1}
1	0	0	$-2Y$	$\overline{y_{j-1}}$
1	0	1	$-Y$	$\overline{y_j}$
1	1	0	$-Y$	$\overline{y_j}$
1	1	1	-0	0

The output of a MAC comprise of three outputs, i.e. having a value set (0,1,2,3). For the operation of accumulation, (6,3) counter tree is fed by output of three bits. The proposed architecture is used for multiply-add operation which has one stage of booth encoding and two stages of (6,3) counter. A single LUT logic

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

depth can be used for implementing booth encoding and counter stages and there are 3 LUT delays in critical path of total system. It is possible to achieve minimum register delay by a high throughput. The figure 3 shows that explicit pipeline stage is not present in operation of redundant multiply accumulate. High speed operation of structure is required without using a pipeline in operation as opposite to conventional structures of multiply-accumulate. A three operand is there for redundant to a normal stage of binary conversion by a clock delay. Most of the DSP applications do not require normal binary conversion. After 100 operations of multiply-add, there is a need of redundant to binary conversion for an FIR of 100 taps.

III. RESULTS AND DISCUSSION

The comparison between resource requirement and performance is done by constructing conventional MAC units by using: "MAC units with hardware and software multipliers, units with pipeline stages etc". A generic pipelined MAC unit is shown in figure 4 having various stages of pipeline in which comparison with proposed structure is done. Stage A is pipeline stage's first stage, having a multiplier. The stage of pipeline between adder and multiplier operation is the second stage of pipeline. A pipeline stage at bit level does not exists for add operator,

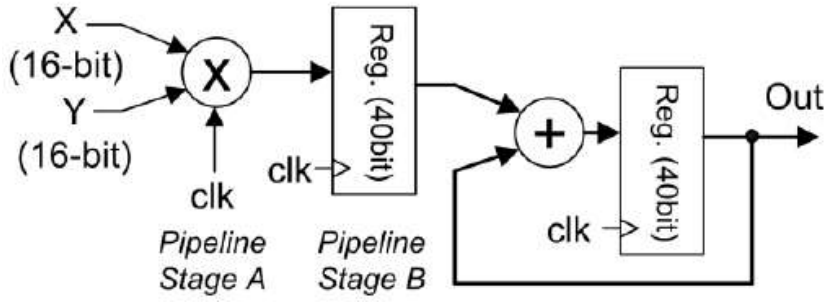

Fig. 4 Various combinations of pipeline in a regular MAC unit

Because sufficiently fast operation is provided by fast carry chains for addition of 40 bits. The given pipeline stages shows the measurement of performance metrics and table III tabulates all this. The register count and "Adaptive LUT units (ALUT)" determines resource usage. Logic elements are used for synthesizing multiplier in a MAC unit and soft multiplier is the name given to structure and hardwire multiplication units are used for synthesizing multiplier and structure is called hard multiplier. DSP sub-blocks are composed in Altera Stratix III and also full hardwired multiply blocks [4], which is known as "full hardware multiply add unit".

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

Table III Comparison in MAC units

Structure	*Resource Usage*	Performance	Clock Delay
Soft Multiplier (no pipeline)	258 ALUT + 72 Reg.	124 MHz	1
Soft Multiplier 1-level pipeline: (B =1)	259 ALUT + 104 Reg.	160 MHz	2
Soft Multiplier 2-level pipeline: (A =1; B =1)	263 ALUT + 190 Reg.	210 MHz	3
Hardware Multiplier (no pipeline)	2 DSP Block 40 ALUT + 40 Reg.	141 MHz	1
Hardware Multiplier 1-level pipeline: (B =1)	2 DSP Block 40 ALUT + 80 Reg.	261 MHz	2
Full Hardware Multiply-Add Unit [18]	4 DSP Block	393 MHz	2
Proposed (Fig. 3)	418 ALUT + 178 Reg.	286 MHz	1+1

The comparison is done with non-pipelined MAC units except “full hardware multiply-add unit” and the structure proposed is much faster as illustrated in table III. If adder unit which is based on software is utilized with hardware multiplier, then performance is good. However, there is no pipeline in proposed structure and an extra delay of clock is required for converting redundant output to binary and thus, (1+1) is clock delay. Shallow stages of pipeline are employed for building benchmark circuits and it is faster to employ proposed scheme rather than multiply-add pipelined units. Altera’s Megafunction library is used for synthesizing soft multipliers that are developed for systems of comparison. The proposed scheme has a performance close to MAC unit based on hardware multiplier. However, the scheme proposed is about 10 percent faster. When compared to the scheme proposed, only “full hardware multiply-add unit” [4]. However, 4 DSP sub-blocks are required having a high requirement of resources in the scarcity of DSP blocks. About 18x18 bit operations are done by full multiply-add block where it is scalable to use proposed scheme to higher bit widths. This results in good performance of metrics among various carry propagate multiplier by “double carry save redundant representation”. The LUT elements in proposed structure are highest among all those having requirement of moderate register when compared with pipelined MAC units. It is advantageous to use proposed MAC unit when least delay of clock is desired having requirement of higher LUT.

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019

IV. CONCLUSION

This work proposes a MAC unit based on alternative redundant number which has low clock delay and high value of throughput. This structure is free of carry propagation and the logic depth is least here. The structure has a "modified Booth encoding stage" which is followed by "symmetrical (6,3) counter tree. The logic depth provided by this proposed structure is lowest, which also provides fast operation of multiply add without using pipeline in a single stage. This system is useful for FPGA structures based on 6 input LUT.

REFERENCES

[1] W.-K. Chen and R. Priemer, "Fundamentals of Digital Signal Processing," in Analog and VLSI Circuits, 2019.
[2] W. Kamp and A. Bainbridge-Smith, "Multiply accumulate unit optimised for fast dot-product evaluation," in ICFPT 2007 - International Conference on Field Programmable Technology, 2007.
[3] Xilinx, "Virtex-5 FPGA Configuration User Guide," Xilinx Inc., 2011.
[4] Altera, "Logic Array Blocks and Adaptive Logic Modules in Stratix III Devices," in Stratix III Device Handbook, 2009.
[5] S. Gorgin and G. Jaberipur, "Fully redundant decimal arithmetic," in Proceedings - Symposium on Computer Arithmetic, 2009.
[6] S. Gorgin and G. Jaberipur, "A fully redundant decimal adder and its application in parallel decimal multipliers," Microelectronics J., vol. 40, no. 10, pp. 1471–1481, 2009.
[7] R. Hutty and R. Hutty, "Decimal Arithmetic," in Programming in Z80 Assembly Language, 2015, pp. 83–89.
[8] A. Kaivani and G. Jaberipur, "Fully redundant decimal addition and subtraction using stored-unibit encoding," Integr. VLSI J., vol. 43, no. 1, pp. 34–41, 2010.
[9] D. Kythe and P. Kythe, "Digital Arithmetic," in Algebraic and Stochastic Coding Theory, 2012.
[10] B. Mitchell, My Digital Arithmetic. 2016.
[11] R. A. Javali, R. J. Nayak, A. M. Mhetar, and M. C. Lakkannavar, "Design of high speed carry save adder using carry lookahead adder," in Proceedings of International Conference on Circuits, Communication, Control and Computing, I4C 2014, 2014, pp. 33–36.
[12] Design and Architectures for Digital Signal Processing. 2013.
[13] J. Ramsden, Nanotechnology Cookbook. 2012.
[14] A. A. Bawaskar, V. Alagdeve, and R. Keote, "High performance redundant binary multiplier," in International Conference on Communication and Signal Processing, ICCSP 2016, 2016, pp. 1277–1281.
[15] S. Dutt, A. Chauhan, R. Bhadoriya, S. Nandi, and G. Trivedi, "A High-Performance Energy-Efficient Hybrid Redundant MAC for Error-Resilient Applications," in Proceedings of the IEEE International Conference on VLSI Design, 2015, vol. 2015-February, no. February, pp. 351–356.
[16] Don Clark, "Intel Completes Acquisition of Altera - WSJ," The Wall Street Journal, 2015. .
[17] K. Mohammad and S. Agaian, "Efficient FPGA implementation of convolution," in Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, 2009.
[18] C. D. Moreno et al., "Efficient mapping on FPGA of convolution computation based on combined CSA-CPA accumulator," in 2009 16th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2009, 2009.
[19] J. Sitch, "High-speed digital signal processing for optical communications," in European Conference on Optical Communication, ECOC, 2008.
[20] A. . Fallis, Principles of CMOS VLSI Design, a Systems Perspective, vol. 53, no. 9. 2013.

Received: 10 Nov 2018 | Revised: 20 Dec 2018 | Accepted: 02 Jan 2019