e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 11
September 2017

VLSI Design of a New High Throughput Finite Field Redundant Multiplier

A. Shravya,
PG Scholar,
Department of ECE,
Sreenidhi Institute of Science and Technology.

Mr. G. Prasad Acharya,
Associate Professor,
Department of ECE,
Sreenidhi Institute of Science and Technology.

Abstract: Redundant basis (RB) multipliers over Galois Field (GF (2^m)) have gained huge popularity in Elliptic Curve Cryptography (ECC) mainly because of their negligible hardware cost for squaring and modular reduction. Elliptic Curve Cryptography (ECC) is an approach to public key cryptography based on the algebraic structure of elliptic curves over finite fields. In this paper, we have proposed a novel recursive decomposition algorithm for RB multiplication to obtain high throughput digit serial implementation. Through efficient projection of Signal Flow Graph (SFG) of the proposed algorithm, a highly regular Processor Space Flow Graph (PSFG) is derived. In this project we are deriving three novel multipliers which not only involve significantly less time complexity than the existing ones but also require less area and less power consumption compared with the others. Both theoretical analysis and synthesis results confirm the efficiency of proposed multipliers over the existing ones. The synthesis results for Field Programmable Gate Array (FPGA) and Application Specific Integrated Circuit (ASIC) realization of the proposed designs and competing existing designs are compared. The extension for the project is Dadda multiplier. The Dadda multiplier is a hardware multiplier design similar to the Wallace multiplier, but it is slightly faster (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes). Simulation and synthesis results are obtained by using Xilinx ISE 13.2, which when compared with proposed and extension results in the reduction of area, increasing the speed.

Keywords: ASIC, FPGA, digit serial, finite field multiplication, PSFG, high throughput, redundant basis.

I. INTRODUCTION

Finite field multiplication over $GF(2^m)$ is a basic operation frequently encountered in modern cryptographic systems such as the ECC and error control coding. Moreover, multiplication over a finite field can be used further to perform other field operations, e.g., division, exponentiation and inversion. Multiplication over $GF(2^m)$ can be implemented on a general purpose machine, but it is expensive to use a general purpose machine to implement cryptographic systems in cost sensitive consumer products. Besides, a low end microprocessor cannot meet the real time requirement of different applications since word length of these processors is too small compared with the order of typical finite fields used in cryptographic systems. Most of the real time applications therefore, need hardware implementation of finite field arithmetic operations for the benefits like low cost and high throughput rate.

In this paper, we aim at presenting efficient digit level serial/parallel designs for high throughput finite field multiplication over based on RB. We have proposed an efficient recursive decomposition scheme for digit level RB multiplication and based on that we have derived parallel algorithms for high throughput digit serial multiplication. We have mapped the algorithm to three different high speed architectures by mapping the parallel algorithm to a regular 2 dimensional Signal Flow Graph (SFG) array, followed by suitable projection of SFG to 1 dimensional Processor Space Flow Graph (PSFG) and the choice of feed forward cut set to enhance the throughput rate. Our proposed digit serial multipliers involve significantly less area time power complexities than the corresponding existing designs. Field Programmable Gate Array (FPGA) has evolved as a mainstream dedicated computing platform.

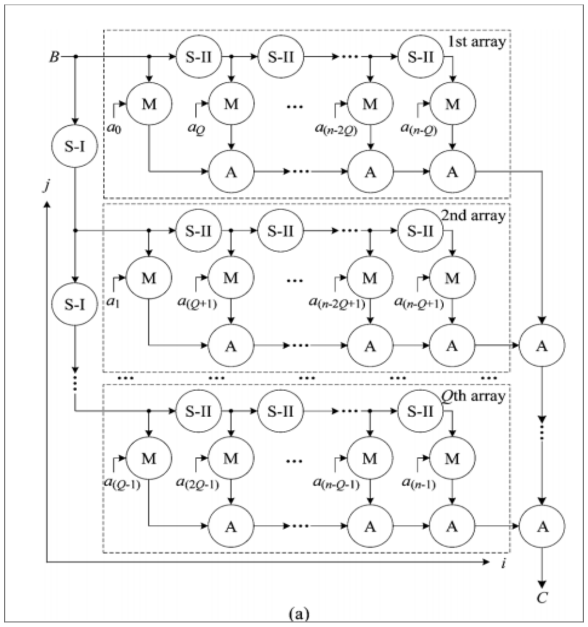
Dadda multipliers are a refinement of the parallel multipliers presented by Wallace multiplier. Dadda multiplier consists of three stages. The partial product matrix is formed in the first stage by AND gates. In the second stage, the partial product matrix is reduced to a height of two. Dadda replaced Wallace Pseudo adders with parallel (n, m) counters. A Parallel (n, m) counter is a circuit which has n inputs and produce m outputs which provide a binary count of the ONEs present at the inputs. A full adder is an implementation of a (3, 2) counter which takes 3 inputs and produces 2 outputs. Similarly a half adder is an implementation of a (2, 2) counter which takes 2 inputs and produces 2 outputs.

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 11
September 2017

The Dadda multiplier reduces the number of rows as much as possible on each layer at the same time Dadda multiplier do as few reductions as possible. Because of this, Dadda multipliers have less expensive reduction phase, but the numbers may be a few bits longer, thus requiring slightly bigger adders.

II. Derivation of Proposed High throughput Structures for RB Multipliers

A. *Proposed Structure I :*



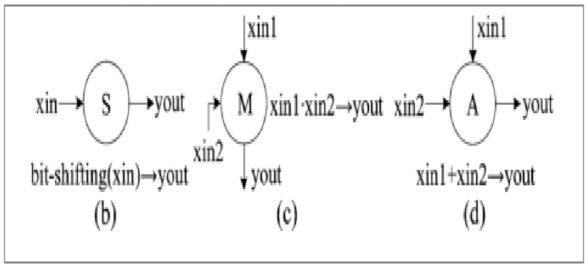
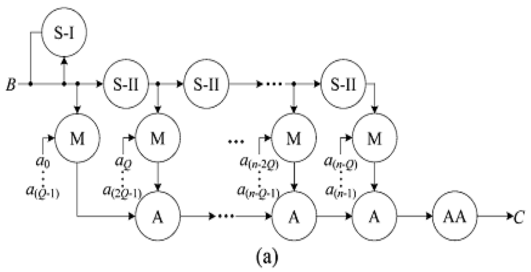
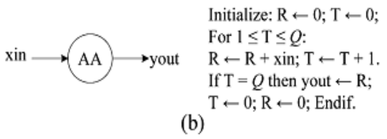


Fig. 1 Signal Flow Graph (SFG) for parallel realization of RB multiplication (a) The proposed SFG (b) Functional description of S node, where S I node performs circular bit shifting of one position and S II node performs circular bit shifting by Q positions (c) Functional description of M node (d) Functional description of A node

The RB multiplication can be represented by the 2 dimensional SFG (shown in Fig. 1) consisting of parallel arrays, where each array consists of bit shifting nodes (S node), multiplication nodes (M nodes) and addition nodes (A nodes). There are two types of S nodes (S I node and S II node). Function of S nodes is depicted in Fig. 1(b), where S I node performs circular bit shifting by one position and S II node performs circular bit shifting by Q positions for the reduction requirement. Functions of M nodes and A nodes are depicted in Fig. 1(c) and 1(d) respectively. Each of the M nodes perform an AND operation of a bit of serial input operand with bit shifted form of operand, while each of the A nodes performs an XOR operation. The final addition of the output of arrays of Fig. 1 can be performed by bit by bit XOR of the operands in (Q 1) number of A nodes as depicted in Fig. 1. The desired product word is obtained after the addition of parallel output of the arrays.

For digit serial realization of RB multiplier, the SFG of Fig. 1 can be projected along j direction to obtain a PSFG as shown in Fig. 2, where P input bits are loaded in parallel to multiplication nodes during each cycle period.

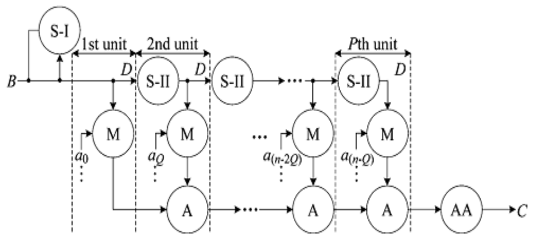




Initialize: R ← 0; T ← 0;
For 1 ≤ T ≤ Q:
R ← R + xin; T ← T + 1.
If T = Q then yout ← R;
T ← 0; R ← 0; Endif.

Fig. 2 Processor Space Flow Graph (PSFG) of digit serial realization of finite field RB multiplication (a) The proposed PSFG (b) Functional description of add accumulation (AA) node

The functions of nodes of PSFG are the same as those of corresponding nodes in the SFG of Fig. 1 except an extra add accumulation (AA) node. The function of the AA node is described in Fig. 2(b), to execute the accumulation operation for Q cycles to yield the desired result thereafter.



e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 11
September 2017

Fig. 3 Cut set retiming of PSFG of finite field RB multiplication over GF (2^m) Where 'D' denotes delay

For efficient realization of a digit serial RB multiplier, we can perform feed forward cut set retiming in a regular interval in the PSFG as shown in Fig. 3. As a result of cut set retiming of the Fig. 3, the minimum duration of each clock period is reduced to (T_A+T_B), where T_A and T_B denote the delay of an AND gate and an XOR gate respectively.

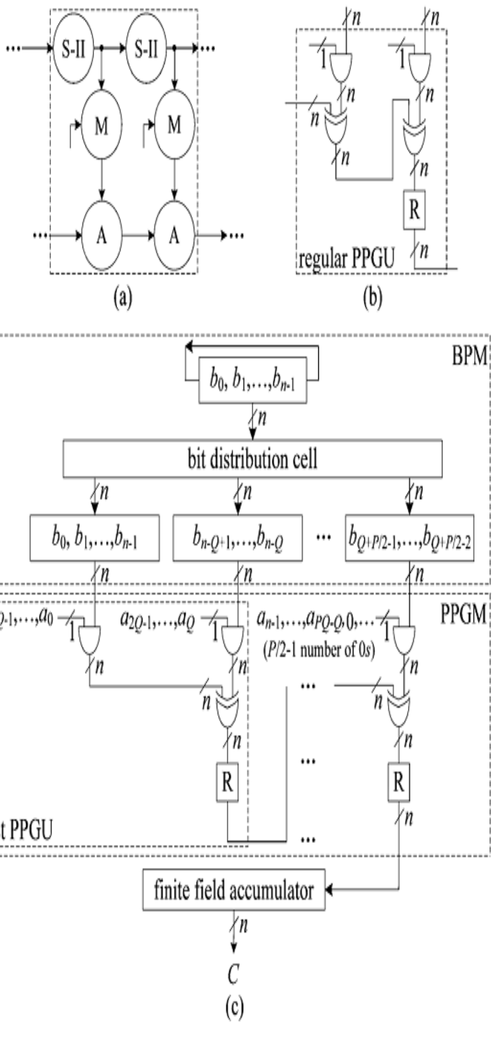


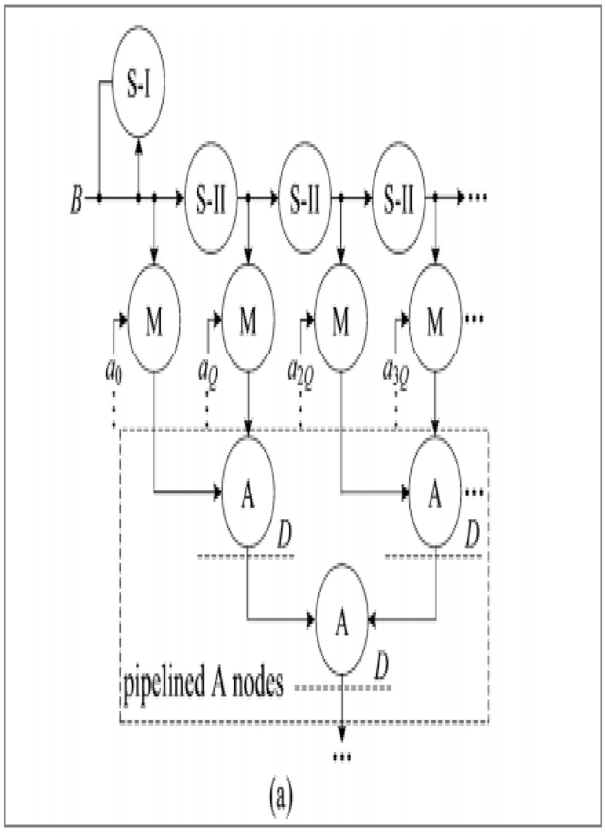
Fig. 4 Proposed Structure I for RB multiplier (a) Proposed cut set retiming of PSFG when (b) Detailed internal structure of merged regular PPGU (c) Corresponding PS I for the case d=2

For example, to obtain the proposed structure for d=2 , a pair of S nodes, a pair of M nodes and a pair of A nodes of the PSFG of Fig. 3 can be merged to form a macro node as shown within the dashed lines in Fig. 4. Each of these macro nodes can be implemented by a new PPGU to obtain a PPGM of p/2 PPGUs as shown in Fig. 4(b), which consists of two AND cells and two XOR cells (the first PPGU requires only one XOR cell).

The critical path of the structure of Fig. 4(c) amounts to (T_A+2T_X). The first output of desired product is available from this structure after a latency of (P/2+Q) cycles, while the successive outputs are available thereafter in each Q cycles of duration (T_A+2T_X).

B. *Proposed Structure II :*

We can further transform the PSFG of Fig. 3 to reduce the latency and hardware complexity of PS I. To obtain the proposed structure, (P 1) serially connected A nodes of the PSFG of Fig. 3 are merged into a pipeline form of (P 2) A nodes as shown within the dashed box in Fig. 5(a). These pipelined A nodes can be implemented by a pipelined XOR tree, as shown in Fig. 5(b). Since all the AND cells can be processed in parallel, there is no need of using extra "0"s on the input path to meet the timing requirement in systolic pipeline. The critical path and throughput of PS II are the same as those of PS I. Similarly, PS II can be easily extended to larger values of 'd' to have low register complexity structures.



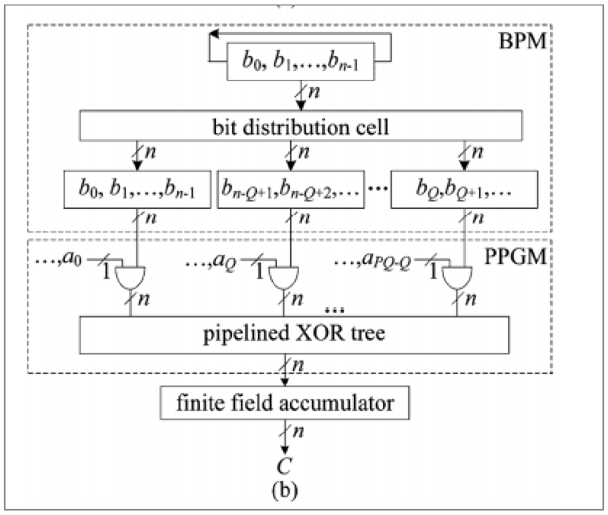
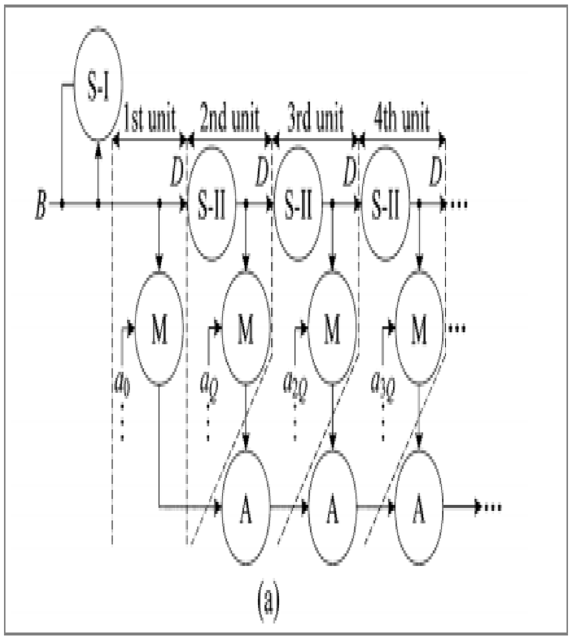


Fig. 5 Proposed Structure II for RB multiplier, where 'R' denotes a register cell (a) Modified PSFG (b) Structure of RB multiplier

C. *Proposed Structure III :*

Since the S nodes of Fig. 3 perform only the bit shifting operations they do not involve any time consumption. Therefore, we can introduce a novel cut set retiming to reduce the critical path further, as shown in Fig. 6(a). It can be observed that the cut set retiming allows to perform the bit addition and bit multiplication concurrently, so that the critical path is reduced to max $\{T_A, TX\} = T_X$, i.e., the throughput of the design is increased.



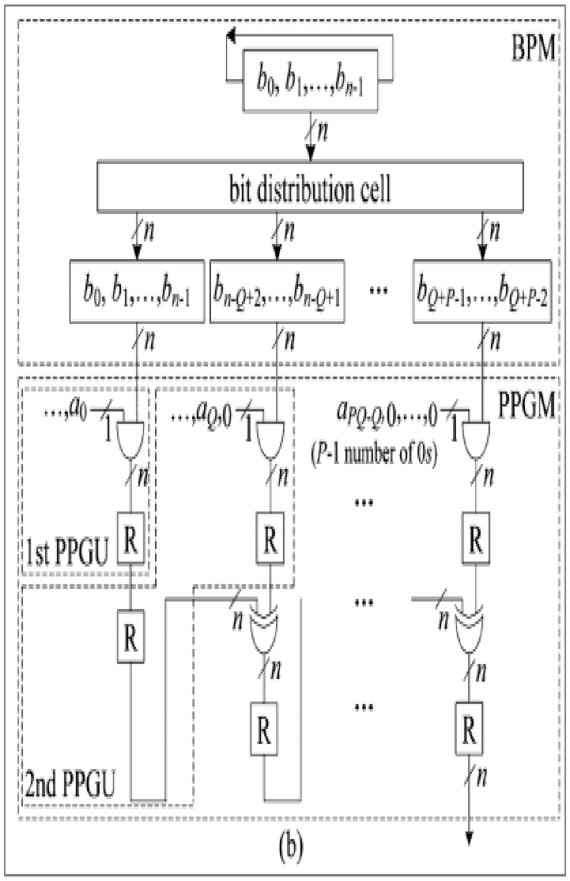


Fig. 6 Novel cut set retiming of PSFG and its corresponding structure Proposed structure III (a) Cut set retiming (b) BPM and PPGM of Proposed Structure III

III. Extension Work

The extension can be done using Dadda multiplier, to reduce the delay. The process of Dadda multiplication is as follows:

The entire 16 × 16 multiplication requires six stages. Always the first stage is partial products stage, which is obtained by simple multiplication of multiplicand with multiplier. The number of rows (height) present at this stage is 16. Now reduce the number of rows further in such a way that final stage contains only two rows. For this, Dadda introduces a sequence of intermediate matrix heights that provides the minimum number of reduction stages for a given size multiplier. This sequence determined by working back from the final two row matrix, limit the height of each intermediate matrix to the largest integer that is no more than 1.5 times the height of its successor.

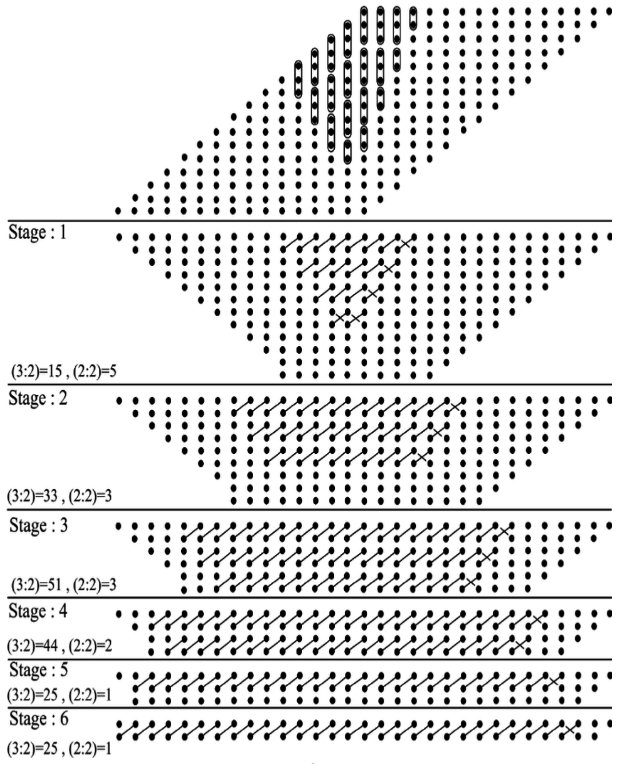


Fig. 7 Schematic of Dadda Multiplier

Dadda proposed a method of reduction which achieves the reduced two rowed partial products in a minimum number of reduction stages. Dadda succeeded this, by placing the (3,2) and (2,2) counters in maximum Critical path in optimal manner. For an N bit multiplier and multiplicand, there results N by N partial products. These partial products are arranged in the form a Matrix. Dadda reduced these Matrix height to a two rowed matrix, through a sequence a reduction stages.

Algorithm:

1. Multiply (that is AND) each bit of one of the arguments, by each bit of the other, yielding N2 results.
2. Reduce the number of partial products to two layers of full and half adders. For this Dadda reduction scheme uses the following algorithm:
(a) Let d1= 2 and dj+1 = [3.dj / 2], where dj is the matrix height for the j th stage from the end. Find the largest j such that at least one column of the matrix has more than dj bits.
(b) Employ (3, 2) and (2, 2) counters to obtain a reduced matrix with no more than dj elements in any column.
c) Until a matrix with only two rows is generated. Let j=j 1 and repeat step (b)
3. Group the wires in two numbers, and add them with a conventional adder.

Flow Chart:

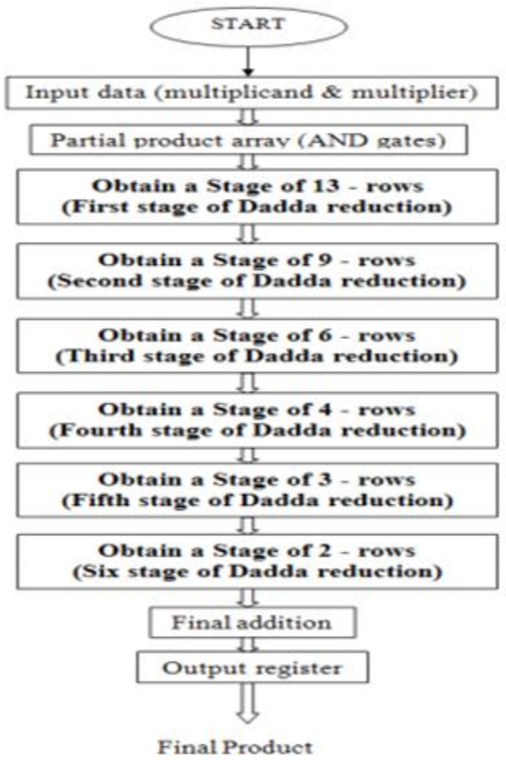

START
Input data (multiplicand & multiplier)
Partial product array (AND gates)
Obtain a Stage of 13 - rows
(First stage of Dadda reduction)
Obtain a Stage of 9 - rows
(Second stage of Dadda reduction)
Obtain a Stage of 6 - rows
(Third stage of Dadda reduction)
Obtain a Stage of 4 - rows
(Fourth stage of Dadda reduction)
Obtain a Stage of 3 - rows
(Fifth stage of Dadda reduction)
Obtain a Stage of 2 - rows
(Six stage of Dadda reduction)
Final addition
Output register
Final Product

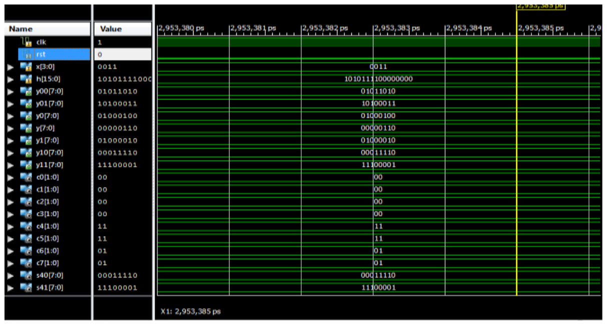
Fig. 8 Flow Chart for Dadda Multiplier

IV. RESULTS

The written Verilog HDL Modules have successfully simulated and synthesized using Xilinx ISE 13.2.

Simulation Results:

For Simulation results shown in Fig. 9, we have taken clk=1, rst=0, a, b, c as inputs, c, y has been obtained as outputs.



Name	Value
clk	1
rst	0
x[3:0]	0011
h[15:0]	1010111100000000
y00[7:0]	01011010
y01[7:0]	10100011
y0[7:0]	01000100
y[7:0]	00000110
y1[7:0]	01000010
y10[7:0]	00011110
y11[7:0]	11100001
c0[1:0]	00
c1[1:0]	00
c2[1:0]	00
c3[1:0]	00
c4[1:0]	11
c5[1:0]	11
c6[1:0]	01
c7[1:0]	01
s40[7:0]	00011110
s41[7:0]	11100001

Fig. 9 Simulation result for Proposed Structure I

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 11
September 2017

For Simulation results shown in Fig. 10, we have taken clk=1, rst=0, a, b, as inputs, y has been obtained as outputs.

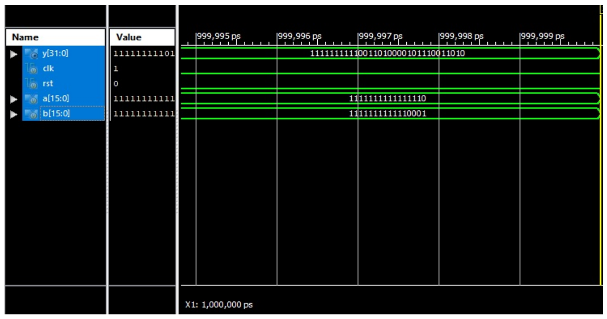


Fig. 10 Simulation result for Proposed Structure II

For Simulation results shown in Fig. 11, we have taken clk=0, rst=0, a, b, as inputs, y has been obtained as outputs.

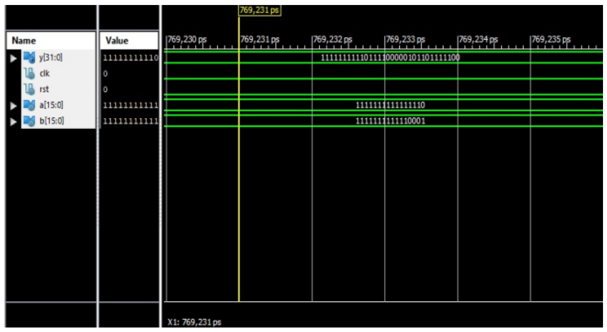


Fig. 11 Simulation result for Proposed Structure III

For Simulation results shown in Fig. 12, we have taken clk=1, rst=0, a, b, as inputs, c has been obtained as outputs.

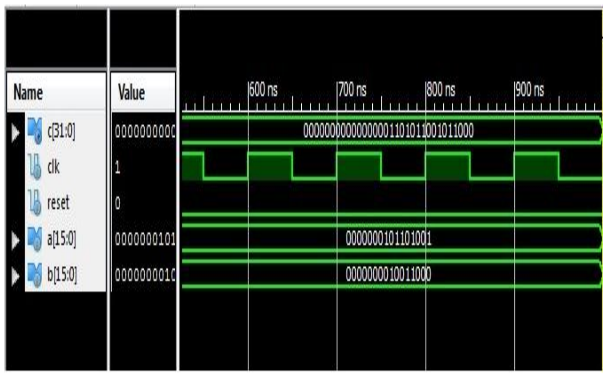


Fig. 12 Simulation result for Dadda Multiplier

RTL Schematic:

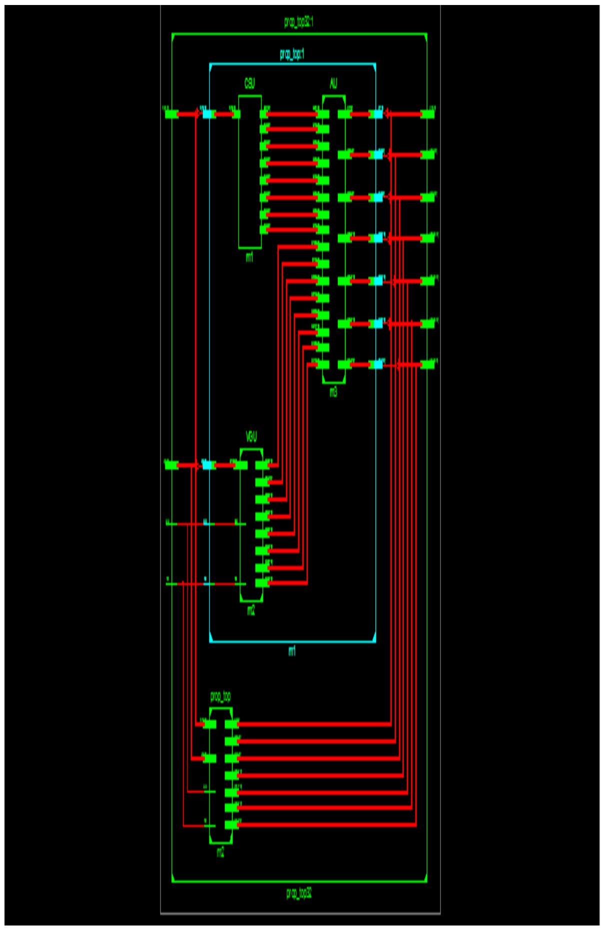


Fig. 13 RTL Schematic for Proposed Structures

The RTL Schematic of Proposed Structure is shown in Fig.13 which consists of a combinational multipliers and an accumulator. Combinational multiplier is a multiplier which is used to compute the final result by computing partial products in parallel. An accumulator is a storage device from where the final output is obtained.

Device Utilization Summary:

Table 1: Device Utilization Summary for Proposed Structures

Device Utilization Summary(estimated value)			
Logic Utilization	Used	Available	Utilization
Number of Slices	95	4656	2%
Number of Slice Flip flops	32	9312	0%
Number of 4 input LUTs	170	9312	1%
Number of	50	232	21%

bonded IOBs			
Number of GCLKs	1	24	4%

Table 2: Device Utilization Summary for Dadda Multiplier

Device Utilization Summary(estimated value)			
Logic Utilization	Used	Available	Utilization
Number of Slices	68	4656	1%
Number of 4 input LUTs	120	9312	1%
Number of bonded IOBs	34	232	14%
Number of GCLKs	1	24	4%

Table 3: Comparison of the Proposed and Extension results

	AREA			DELAY (ns)
	No. of Slices	No. of LUTs	No. of FFs	
PROPOSED	95	170	50	4.063ns
EXTENSION	68	120	34	4.04ns

Comparison of the Proposed and Extension results vary in area and delay which shows that there is a decrease in them and the performance is more better. The delay for proposed structure is 4.063ns whereas, for dadda mutiplier is 4.04ns.

V. CONCLUSION

We have proposed a novel recursive decomposition algorithm for RB multiplication to derive high throughput digit serial multipliers. By suitable projection of SFG of proposed algorithm and identifying suitable cut sets for feed forward cut set retiming, three novel high throughput digit serial RB multipliers are derived to achieve significantly less area time complexities than the existing ones. The extension for the project is Dadda multiplier. The Dadda multiplier is a hardware multiplier design similar to the Wallace multiplier, but it is slightly faster (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes).Simulation and synthesis results are obtained by using Xilinx ISE 13.2, which when compared with proposed and extension results in the reduction of area, increasing the speed.

REFERENCES

[1] High Throughput Finite Field Multipliers Using Redundant Basis for FPGA and ASIC Implementations Jiafeng Xie, Pramod Kumar Meher, *Senior Member, IEEE*, and Zhi Hong Mao, *Senior Member, IEEE*
[2] I. Blake, G. Seroussi, and N. P. Smart, Elliptic Curves in Cryptography, ser. London Mathematical Society Lecture Note Series.. Cambridge, U.K.: Cambridge Univ. Press, 1999.
[3] N. R. Murthy and M. N. S. Swamy, "Cryptographic applications of brahmaqupta bha skara equation," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 53, no. 7, pp. 1565–1571, 2006.
[4] L. Song and K. K. Parhi, "Low energy digit serial/parallel finite field multipliers," J. VLSI Digit. Process, vol. 19, pp. 149–C166, 1998.
[5] P. K. Meher, "On efficient implementation of accumulation in finite field over and its applications," IEEE Trans. Very Large Scale Integer. (VLSI) Syst., vol. 1 7, no. 4, pp. 541–550, 2009.
[6] L. Song, K. K. Parhi, I. Kuroda, and T. Nishitani, "Hardware/software co design of finite field data path for low energy Reed Solomn codecs," IEEE Trans. Very Large Scale Integer. (VLSI) Syst., vol. 8, no. 2, pp. 160–172, Apr. 2000.
[7] G. Drolet, "A new representation of elements of finite fields yielding small complexity arithmetic circuits," *IEEE Trans. Comput.*, vol. 47, no. 9, pp. 938–946, 1998.
[8] C. Y. Lee, J. S. Horng, I. C. Jou and E. H. Lu, "Low complexity bit parallel systolic Montgomery multipliers for special classes of" *IEEE Trans. Comput.*, vol. 54, no. 9, pp. 1061–1070, Sep. 2005.
[9] P. K. Meher, "Systolic and super systolic multipliers for finite field based on irreducible trinomials," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 4, pp. 1031–1040, May 2008.
[10] J. Xie, J. He, and P. K. Meher, "Low latency systolic Montgomery multiplier for finite field based on pentanomials," *IEEE Trans. Very Large Scale Integer. (VLSI) Syst.*, vol. 21, no. 2, pp. 385–389, Feb, 2013.
[11] A. H. Namin, H. Wu, and M. Ahmadi, "An efficient finite field multiplier using redundant representation," *ACMTrans. Embedded Comput. Sys*, vol. 11, no. 2, Jul. 2012, Art. 31.
[12] North Carolina State University, *45 nm FreePDK wiki* [Online]. Available: http://www.eda.ncsu.edu/wiki/FreePDK45:Manual
[13] P. K. Meher, "Systolic and non systolic scalable modular designs of finite field multipliers for Reed Solomon Codec," *IEEE Trans. Very Large Scale Integer. (VLSI) Syst.*, vol. 17, no. 6, pp. 747–C757, Jun. 2009.

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 11
September 2017

[14] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*.
New York: Wiley, 1999.
[15] J. L. Massey and J. K. Omura, "Computational method and apparatus for finite field arithmetic," U.S. patent application, 1984.
[16] S. Gao and S. Vanstone, "On orders of optimal normal basis generators," *Math. Comput*, vol. 64, no. 2, pp. 1227–1233, 1995.
[17] A. Reyhani Masoleh and M. A. Hasan, "Low complexity word level sequential normal basis multipliers," *IEEE Trans. Comput.*, vol. 54, no. 2, pp. 98–C110, Feb. 2005.
[18] A. H. Namin, H. Wu, and M. Ahmadi, "A word level finite field multiplier using normal basis," *IEEE Trans. Comput.*, vol. 60, no. 6, pp. 890–895, Jun. 2011.
[19] H. Wu, M. A. Hasan, I. F. Blake, and S. Gao, "Finite field multiplier using redundant representation," *IEEE Trans. Comput.*, vol. 51, no. 11, pp. 1306–1316, Nov. 2002.