JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AN IMPROVED GENERALIZED KARATSUBA MULTIPLIER IN GF(2^M) FIELD AND IT'S CONVOLUTION APPLICATION

RAZIA SULTANA
Assistant Professor
Haldia Institute of Technology
Haldia, India
razia04@gmail.com

Finite field or Galois field arithmetic is becoming increasingly a very important solution for calculations in many applications like wireless or mobile communication systems, satellite communication systems, storage systems, crypto systems, etc. This paper presents a modified multiplier based on Karatsuba multiplication algorithm. To optimize the Karatsuba multiplication algorithm, the product terms are splited into two alternative forms and all the terms are expressed in the repeated fashion. We generalize the modified algorithm for any input bit GF field application. We have compared our proposed multiplier with karatsuba multiplier with respect to complexity. We have also given the general equation for hardware requirement for any field. Less no. of additions is required by our proposed multiplier. The overall area of modified multiplier is improved by 53.75% (over without reduction) & 52.08% (over with reduction). Proposed multiplier is also faster than actual karatsuba multiplier by 3.63% (over without reduction) & 3.91% (over with reduction). The proposed architecture has been simulated and synthesized by Xilinx ISE 14.7 design suite for Vertex , Spartan and Artix 7 device family. The new architecture is simple & easy. Common Expression Elimination (CSE) is applied in the proposed Modified Karatsuba Multiplier (MKM).It is also faster than M-Term Karatsuba algorithm .and it is also applied to compute the circular convolution for DSP application. In Vertex5 FPGA device family, computation of 8-bit circular convolution using Modified Karatsuba Algorithm (MKA) is 23.69% faster than Karatsuba Algorithm (KA). It also consumes 68.69% less slices than existing KA based Convolution.

Keywords: Karatsuba Algorithm; Finite fields; FPGA; VLSI; polynomial multiplication; Circular Convolution.Common Expression Elimination (CSE).

1. INTRODUCTION:

Galois fields (GF) have gained wide spread applications in error correcting codes and cryptographic algorithms. Further applications may be found in signal processing and pseudo random number generation. Modern applications in many cases call for VLSI implementations of the arithmetic modules in order to satisfy the high speed requirements. VLSI allows the designers to allocate complex systems consisting of several thousand or even millions transistors on one or very few chips. VLSI modules having Galois field multiplier can be classified into three categories: bit- serial multipliers [6], bit- parallel multipliers, and hybrid multiplier. Bit parallel architectures tend to be faster and only use combinatorial logic [5]. On the other hand, bit serial architectures require less area and uses registers in addition to combinatorial logic, and the hybrid multipliers, which are partially bit-serial and partially bit-parallel. Hybrid multipliers are faster than bit-serial ones, while their area is smaller than that of bit- parallel. For efficient VLSI implementation suitable hardware architecture is needed. It is obtained by using addition, multiplication, field operations, suitably in the architecture. Addition can be implemented with a very low space complexity, multiplication is required to be fast but it is implemented with a higher complexity. Efficient architectures require low complexity and fast multipliers. Assuming a basis representation of the field elements addition is a relatively inexpensive operation, whereas the other field operation, is costly in terms of gate count and delay.

In the polynomial multiplication, Karatsuba algorithm is used to make multiplication efficient which means algorithm saves multiplication at the cost of extra addition. Because multiplication is more costly than addition. Addition of two m-bit numbers require *m* no. of XOR gates. Koc *et al.* [8] have proposed a recursive algorithm for fast multiplication of large integers having a precision of 2^k computer words, where k is an integer. Their algorithm has been derived from the Karatsuba-Ofman algorithm and has the same asymptotic complexity. They have claimed that the running time of their algorithm is a little better that makes one third as many recursive calls. Murat Cenk *et al.* [9] gave improved formulas to multiply polynomials of small degree over F_2 using Chinese Remainder Theorem (CRT) that improve multiplication complexity. Gang Zhou *et al.* have presented complexity analysis and efficient FPGA (Field Programmable Gate Array) implementations of bit parallel mixed Karatsuba–Ofman multipliers in [10].

In this paper, a modified multiplier based on Karatsuba multiplication algorithm is proposed. To optimize the Karatsuba multiplication algorithm, the product terms are splited into two alternative forms and computed all the terms in the repeated fashion. We generalize the modified algorithm for any input bit GF field application. We have compared our proposed multiplier with karatsuba multiplier with respect to complexity. We have also given the general equation for hardware requirement for any field. Our proposed multiplier is 30.36% (over without reduction) & 31.30% (over with reduction) less no of XOR gates than karatsuba multiplier. The overall area (in terms of slices) is also improved by 53.75% (over without reduction) & 52.08% (over with reduction). MKM is also faster than KM by 3.63% (over without reduction) & 3.91% (over with reduction). The proposed design has been simulated and synthesized using Xilinx FPGA based Spartan and Vertex device family. The new architecture is simple and easy. It is also applied to compute circular convolution. In Spartan3E FPGA device family, computation of 8-bit circular convolution using MKA is 23.69% faster than KA. It also consumes 68.69% less slices than existing KA based convolution.

The rest of the paper is organized as follows. Basics of Galois Field arithmetic is presented in section-II. A new method for implementations of Karatsuba multipliers has been proposed in Section-III. Results & discussion are provided in Section-IV. Section-V describes application of proposed algorithm to compute the circular convolution and finally the paper is concluded in Section-VI.

2. GALOIS FIELD ARITHMETIC:

Galois field defines as $GF(p^m)$ which is a field with p^m numbers of elements (p is a prime number) [7]. Furthermore, order of Galois field is the number of elements in the Galois field. Addition and multiplication are two basic operations mainly done in Galois field arithmetic. Addition and subtraction of elements of GF(2^m) are simple XOR operations of the two operands. Each of the elements in the GF is first represented as a corresponding polynomial. Multiplication operation over the Galois field is a more complex operation than the addition operation. For m=4, the product term is represented as follows:

$$A(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \quad (1)$$

$$B(x) = b_3x^3 + b_2x^2 + b_1x + b_0 \quad (2)$$

$$A(x)\times B(x) = (a_3x^3 + a_2x^2 + a_1x + a_0) \times (b_3x^3 + b_2x^2 + b_1x + b_0)$$

$$= (a_3b_3)x^6 + (a_3b_2 + a_2b_3)x^5 + (a_3b_1 + a_2b_2 + a_1b_3)x^4 + (a_3b_0 + a_0b_3 + a_2b_1 + a_1b_2)x^3$$

$$+ (a_2b_0 + a_0b_2 + a_1b_1)x^2 + (a_1b_0 + a_0b_1)x + (a_0b_0)$$

The result has seven coefficients which must be reverted back into a 4-tuple to achieve closure. This can be done by substituting the value of x^6, x^5 and x^4 with their polynomial representations and summing terms.

$$A(x) \times B(x) = (a_3b_3 + a_3b_0 + a_2b_1 + a_1b_2 + a_0b_3)\, x^3 + (a_3b_3 + a_3b_2 + a_2b_3 + a_2b_0 + a_1b_1 + a_0b_2)\, x^2 + (a_3b_2 + a_2b_3 + a_3b_1 + a_2b_2 + a_1b_3 + a_1b_0 + a_0b_1)\, x + (a_3b_1 + a_2b_2 + a_1b_3 + a_0b_0). \quad (3)$$

Eqn. (3) is often expressed in matrix form.

$$\begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_3 + a_0 & a_3 + a_2 & a_1 + a_2 \\ a_2 & a_1 & a_3 + a_0 & a_3 + a_2 \\ a_3 & a_2 & a_1 & a_3 + a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad (4)$$

The multiplication results in eqn.(3) can be implemented as logical ANDs and the additions as logical XORs. Thus, the expression requires only 16 AND and 15 XOR to implement.

GF multipliers are dependent on addition and multiplication. Addition is easy and it equates to a bit-wise XOR of the m-tuple and is realized by an array of mXOR gates. The GF multiplier much more complicated and is the key to developing efficient of GF field computational circuits. We have used the Verilog HDL language to code all the designs.

Karatsuba Multiplier (KM)

In this section, we introduce the fundamental Karatsuba algorithm which can successfully be applied to polynomial multiplication. The Karatsuba Algorithm was introduced by Karatsuba in 1962. The fundamental Karatsuba multiplication for polynomial in $GF(2^m)$ is a recursive divide-and-conquer technique. It is considered as one of the fastest way to multiply long numbers. For polynomial multiplication with original Karatsuba method both operands have to be divided into two equal parts. Then each sub operands is divided again into two parts. The process will continue until this become single. Then we get the followings by splitting the polynomials using KM:

If $A(x)$ and $B(x)$ are field polynomials with degrees 3 over a field GF (2^4).

With the auxiliary variables

$$D_0 = a_0b_0 \;\; , D_1 = a_1b_1$$

$$D_2 = a_2b_2 \;\; , D_3 = a_3b_3$$

$$D_{0,1} = (a_0 + a_1)(b_0 + b_1)$$

$$D_{0,2} = (a_0 + a_2)(b_0 + b_2)$$

$$D_{1,3} = (a_1 + a_3)(b_1 + b_3)$$

$$D_{3,2} = (a_3 + a_2)(b_3 + b_2)$$

$$D_{0,1,2,3} = (a_0 + a_1 + a_2 + a_3)(b_0 + b_1 + b_2 + b_3)$$

Field multiplication can be performed into two steps. Firstly, we perform an ordinary polynomial multiplication of two field elements. Secondly, a reduction operation with an irreducible polynomial is need to be performed in order to obtain the (m - 1) degree polynomial. It is noticed that once the irreducible polynomial $p(x) = x^4 + x + 1$ has been selected, the reduction step can be accomplished by using XOR gates only [9]. From the irreducible polynomial $p(x)$ we can replace $x^4 = x+1$, $x^5 = x^2 + x$ and $x^6 = x^3 + x^2$ to obtain $C'(x)$ as follows:

$$C'(x) = A(x)\,B(x) \bmod p(x)$$

$$C'(x)=(D_{0,1,2,3}-D_{1,3}-D_{2,0}-D_{3,2}-D_{0,1}+D_0+D_1+D_2)x^3+(D_{0,2}+D_{3,2}+D_1-D_0)x^2+(D_{0,1}+D_{1,3}+D_{3,2}-D_0)x+(D_{1,3}-D_1-D_3+D_2+D_0) \quad (5)$$

3. MODIFIED KARATSUBA MULTIPLIER (MKM):

In this section our Modified Karatsuba Algorithm (MKA) has been discussed. In MKA all techniques are same as fundamental basic Karatsuba multiplier except the splitting techniques. To optimize the Karatsuba Multiplication Algorithm, the product terms are splited into two alternative forms. This reduction technique requires small area and less delay than others existing multiplication algorithms. The results are compared by using Xilinx based synthesis tools on different FPGA device family like Spartan & Vertex. Our synthesis results are better than existing basic Karatsuba algorithm which is shown in the following section. Assume $A(x)$ and $B(x)$ are two field polynomials with degree 7 in $GF(2^8)$.

$$A(x) = a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$$

$$B(x) = b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0$$

When we compute $C(x) = A(x)\cdot B(x)$ as the following:

$$\begin{aligned}C'(x) &= (a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0)\cdot(b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3\\ &+b_3x^3+b_2x^2+b_1x+b_0)\\ &=(a_7b_7)x^{14}+(a_7b_6+a_6b_7)x^{13}+(a_7b_5+a_5b_7+a_6b_6)x^{12}+(a_7b_4+a_6b_5+a_5b_6+a_4b_7)x^{11}+\\ &+(a_7b_3+a_6b_4+a_5b_5+a_4b_6+a_3b_7)x^{10}+(a_2b_7+a_3b_6+a_4b_5+a_5b_4+a_6b_3+a_7b_2)x^9+(a_1b_7\\ &+a_2b_6+a_3b_5+a_4b_4+a_5b_3+a_6b_2+a_7b_1)x^8+(a_0b_7+a_1b_6+a_2b_5+a_3b_4+a_4b_3+a_5b_2+a_6b_1\\ &+a_7b_0)x^7+(a_0b_6+a_1b_5+a_2b_4+a_3b_3+a_4b_2+a_5b_1+a_6b_0)x^6+(a_0b_5+a_1b_4+a_2b_3+a_3b_2+\\ &a_4b_1+a_5b_0)x^5+(a_0b_4+a_1b_3+a_2b_2+a_3b_1+a_4b_0)x^4+(a_0b_3+a_1b_2+a_2b_1+a_3b_0)x^3+(a_0b_2+\\ &a_2b_0+a_1b_1)x^2+(a_0b_1+a_1b_0)x+a_0b_0\end{aligned}$$

Then we get the following expression by splitting the coefficients of $C(x)= A(x)B(x)$ polynomial using MKA.

Then $C'(x)$ is computed by using the relationship $C'(x)=C(x) \bmod p(x)$. Using the irreducible primitive polynomial $p(x) = x^8 + x^4 + x^3 + x^2 + 1$, terms x^{14}, x^{13}, x^{12}, x^{11}, x^{10}, x^9 and x^8 are replaced in $C(x)$. The simplified expression of $C'(x)$ is as follows:

$$C(x) = D_7x^{14}+(D_{7,6}-D_7-D_6)x^{13}+(D_{7,5}-D_7-D_5+D_6)x^{12}+[(D_{7,4}-D_4-D_7)+(D_{6,5}-D_6-D_5)]x^{11}+[(D_{7,3}-D_7-D_3)+(D_{6,4}-D_6-D_4)+D_5]x^{10}+[(D_{2,7}-D_2-D_7)+(D_{3,6}-D_3-D_6)+(D_{4,5}-D_4-D_5)]x^9+[(D_{1,7}-D_1-D_7)+(D_{2,6}-D_2-D_6)+(D_{3,5}-D_3-D_5)+D_4]x^8+[(D_{0,7}-D_0-D_7)+(D_{1,6}-D_1-D_6)+(D_{2,5}-D_2-D_5)+(D_{3,4}-D_3-D_4)]x^7+[(D_{0,6}-D_0-D_6)+(D_{1,5}-D_1-D_5)+(D_{2,4}-D_2-D_4)+D_3]x^6+[(D_{0,5}-D_0-D_5)+(D_{1,4}-D_1-D_4)+(D_{2,3}-D_2-D_3)]x^5+[(D_{0,4}-D_0-D_4)+(D_{1,3}-D_1-D_3)+D_2]x^4+[(D_{0,3}-D_0-D_3)+(D_{1,2}-D_1-D_2)]x^3+(D_{0,2}-D_0-D_2+D_1)x^2+(D_{0,1}-D_0-D_1)x+D_0 \quad (6)$$

Here operands are splited into two alternative terms. Employing auxiliary variables, we can obtain the following expression.

$$D_0= a_0b_0,\ D_{0,1}= (a_0+a_1)\cdot(b_0+b_1),\ D_1= a_1b_1,\ D_{2,0}= (a_2+a_0)\cdot(b_2+b_0)$$

$$D_2 = a_2b_2,\ D_{0,3} = (a_0+a_3)\cdot(b_0+b_3),\ D_3 = a_3b_3,\ D_{1,2}= (a_1+a_2)\cdot(b_1+b_2)$$

$$D_4= a_4b_4,\ D_{0,4} = (a_0+a_4)\cdot(b_0+b_4),\ D_{1,3}= (a_1+a_3)\cdot(b_1+b_3),\ D_{0,5}= (a_0+a_5)\cdot(b_0+b_5)$$

$$D_5= a_5b_5,\ D_{0,6}= (a_0+a_6)\cdot(b_0+b_6),\ D_6= a_6b_6,\ D_{1,5}= (a_1+a_5)\cdot(b_1+b_5)$$

$$D_{2,4}= (a_2+a_4)\cdot(b_2+b_4),\ D_{2,5}= (a_2+a_5)\cdot(b_2+b_5),$$

$$D_{4,3} = (a_4+a_3)\cdot(b_4+b_3),\ D_{1,7}= (a_1+a_7)\cdot(b_1+b_7)$$

$$D_{2,6}= (a_2+a_6)\cdot(b_2+b_6),\ D_{5,3}= (a_5+a_3)\cdot(b_5+b_3)$$

$$D_{2,7}= (a_2+a_7)\cdot(b_2+b_7),\ D_{6,3}= (a_6+a_3)\cdot(b_6+b_3) \quad (7)$$

$$D_{4,5}= (a_4+a_5)\cdot(b_4+b_5),\ D_{7,3}= (a_7+a_3)\cdot(b_7+b_3)$$

$$D_{6,4}= (a_6+a_4)\cdot(b_6+b_4),\ D_{7,4}= (a_7+a_4)\cdot(b_7+b_4)$$

$$D_{6,5}= (a_6+a_5)\cdot(b_6+b_5),\ D_{7,5}= (a_7+a_5)\cdot(b_7+b_5)$$

$D_{7,6} = (a_7+a_6)\cdot(b_7+b_6)$, $D_7 = a_7b_7$

Figure1 shows the block diagram of Modified Karatsuba multiplier for degree-3 polynomials.

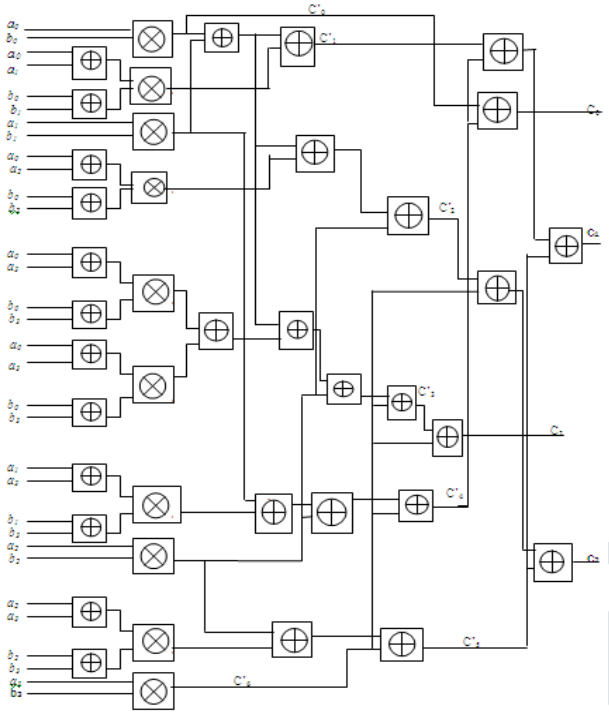


Fig.1: Block diagram of Modified Karatsuba multiplier for degree-3 polynomial

The general form of the coefficients of partial product terms of our proposed multiplier is given below.

$$D_i = \sum_{j=0,2n} a_j b_j \qquad for \;\; i = 0 \; and \; 2n$$

$$D_i = \sum_{j=0}^{\frac{i-1}{2}} (a_j + a_{i-j})(b_j + b_{i-j}) + \sum_{j=0}^{i} a_j b_j \quad for \; i = 1 \; to \; n \qquad (8)$$

$$D_i = \sum_{j=i-n}^{\frac{i-1}{2}} (a_j + a_{i-j})(b_j + b_{i-j}) + \sum_{j=i-n}^{n} a_j b_j \quad for \; i = n+1 \; to \; 2n-1$$

TABLE 1: Complexity comparison between KM and MKM for different GF field

Degree n	bit m	Complexity of Karatsuba Multiplier (KM)		Complexity of Modified Karatsuba Multiplier (MKM)		Improvement (%)	
		# Mul.	# Addition	# Mul.	# Addition	Mul.	Addition
1	2	3	4	3	4	0.00	0.00
2	3	6	13	6	12	0.00	7.69
3	4	9	24	10	23	-11.11	4.17
4	5	15	46	15	37	0.00	19.57
5	6	21	59	21	54	0.00	8.47
6	7	28	99	28	74	0.00	25.25
7	8	36	100	36	97	0.00	3.00
8	9	45	139	45	123	0.00	11.51
9	10	55	174	55	152	0.00	12.64
10	11	66	265	66	184	0.00	30.57
11	12	78	221	78	219	0.00	0.90

Table1 shows the comparison of no. of multiplications and additions required by the two different design techniques. It is shown that all the cases our proposed algorithm is required lesser no. of additions. We have also given formula to calculate the no. of addition and multiplication for different bit lengths (m).

i. No. of multipliers $= \dfrac{m(m+1)}{2}$

ii. No. of additions $= N_{m-1} + (3m-1)$

Where N_{m-1} is the no. of additions in (m-1) steps

4. RESULTS & DISCUSSION:

We have studied the performance of each multiplier over $GF(2^8)$ employing the Xilinx ISE simulation tool. Multipliers are implemented on Vertex5 5vsx50tff1136-1 device. These multipliers are compared based on number of slices, number of 4-input LUTs, bonded I/O blocks and delay.

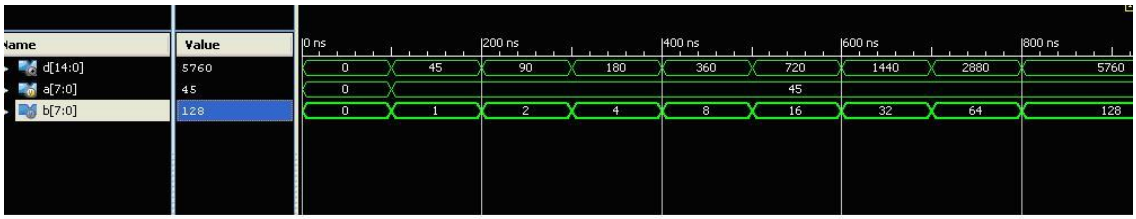


Fig. 2: Simulation results of Modified Karatsuba Multiplier without reduction

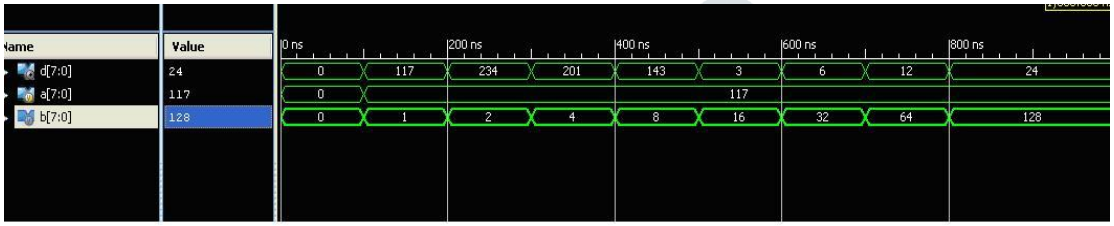


Fig. 3: Simulation results of Modified Karatsuba Multiplier after reduction

Figure 2 & 3 shows the input-output waveform of proposed multiplier for both without reduction and with reduction in $GF(2^8)$.

TABLE 2: Comparison of resource utilization between KM and MKM in $GF(2^8)$.

Name of Parameters	Karatsuba Multiplier		Modified karatsuba Multiplier		Improvement over withou reduction (%)	Improvement over with reduction (%)
	without reduction	with reduction	without reduction	with reduction		
No. of XOR gates	112	115	78	79	30.36	31.30
No. of Slice LUTs (out of 32640)	80	96	37	46	53.75	52.08
No. of LUT Flip Flop pairs used (out of 37)	80	96	37	46	53.75	52.08
No. of bonded IOBs (out of 480)	31	24	31	24	0.00	0.00
Max. combinationa path delay (ns)	7.82	9.611	7.536	9.235	3.63	3.91

Table2 shows the simulation results of proposed multiplier and original karatsuba multiplier both for without reduction and with modulo reduction in GF(2^8) field. Our proposed multiplier is 30.36% (over without reduction) & 31.30% (over with reduction) less no of XOR gates than Karatsuba multiplier. The overall area (in terms of slices) is also improved by 53.75% (over without reduction) & 52.08% (over with reduction). MKM is also faster than KM by 3.63% (over without reduction) & 3.91% (over with reduction).

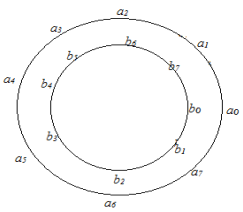
TABLE 3: Comparison of resource utilization between KM and MKM (applied CSE method)

PARAMETER	BIT	CSE METHOD APPLIED IN PROOSED KARATSUBA MULTIPLIER	M-TERM KARATSUBA LIKE BINARY POLYNOMIAL MULTIPLIER
NO. OF LUT(OUT OF 63400)	4	12	29
NO. OF IOB(OUT OF 210)		19	19
PROPAGATION PATH DELAY(nS)		1.870 nS	2.767nS
NO. OF LUT(OUT OF 63400)	7	32	57
NO. OF IOB(OUT OF 210)		27	27
PROPAGATION PATH DELAY(nS)		2.37nS	2.80nS

Table3 shows the simulation results of CSE method applied in proposed multiplier and M-term Karatsuba like binary polynomial Multiplier for without reduction in GF(2^4) & GF(2^7) field. Our proposed multiplier is faster than M-term Karatsuba like Binary polynomial Multiplier[13].

5. APPLICATION

In this Section, computation of circular convolution by employing proposed Modified Karatsuba Algorithm is presented. Assume A and B are the two sequences, where $A=\{a_0,a_1,a_2,a_3,a_4,a_5,a_6,a_7\}$ and $B=\{b_0,b_1,b_2,b_3,b_4,b_5,b_6,b_7\}$. All the points of A are placed on the outer circle in the counter clockwise direction. Starting at the same point as A, all points of B are placed on the inner circle in clockwise direction.



Expression of d_0 is obtained by multiplying the corresponding samples points and then adding the product terms.

$$d_0=a_0b_0+a_7b_1+a_6b_2+a_5b_3+a_4b_4+a_3b_5+a_2b_6+a_1b_7 \quad (8)$$

Applying Modified Karatsuba Algorithm (MKA) in equation (8) we can obtain,

$$d_0=a_0b_0+(a_7+a_1)(b_7+b_1)+a_7b_7+a_1b_1+(a_5+a_3)(b_5+b_3)+ a_5b_5+a_3b_3+(a_2+a_6)(b_2+b_6)+a_2b_2+a_6b_6+a_4b_4 \quad (9)$$

Similarly the expressions of $d_1,d_2,d_3,d_4\ d_5,d_6$ and d_7 are obtained and they are as follows:

$$d_1=a_0b_1+a_1b_0+a_2b_7+a_3b_6+a_4b_5+a_5b_4+a_6b_3+a_7b_2$$
$$=(a_0+a_1)(b_0+b_1)+a_0b_0+a_1b_1+(a_2+a_7)(b_2+b_7)+a_2b_2+a_7b_7+$$
$$(a_3+a_6)(b_3+b_6)+a_3b_3+a_6b_6+(a_5+a_4)(b_5+b_4)+a_5b_5+a_4b_4 \quad (10)$$

$$d_2=a_0b_2+a_1b_1+a_2b_0+a_3b_7+a_4b_6+a_5b_5+a_6b_4+a_7b_3$$
$$=a_1b_1+(a_0+a_2)(b_0+b_2)+a_0b_0+a_2b_2+(a_7+a_3)(b_7+b_3)+a_7b_7+a_3b_3$$
$$+(a_4+a_6)(b_4+b_6)+a_4b_4+a_6b_6+a_5b_5 \quad (11)$$

$$d_3=a_0b_3+a_1b_2+a_2b_1+a_3b_0+a_4b_7+a_5b_6+a_6b_5+a_7b_4$$
$$= (a_0+a_3)(b_0+b_3)+a_0b_0+a_3b_3+(a_1+a_2)(b_1+b_2)+a_1b_1+a_2b_2$$
$$+(a_4+a_7)(b_4+b_7)+a_4b_4+a_7b_7+(a_6+a_5)(b_6+b_5)+a_5b_5+a_6b_6 \quad (12)$$

$$d_4= a_0b_4 +a_1b_3+a_2b_2+a_3b_1+a_4b_0+a_5b_7+a_6 b_6+a_7b_5$$
$$=(a_0+a_4)(b_0+b_4)+a_0b_0+a_4b_4+(a_1+a_3)(b_1+b_3)+a_1b_1+a_3b_3+$$
$$(a_5+a_7)(b_5+b_7)+a_5b_5+a_7b_7+ a_2b_2+a_6b_6 \quad (13)$$

$$d_5=a_0b_5+a_1b_4+a_2b_3+a_3b_2+a_4b_1+a_5b_0+a_6b_7+a_7b_6$$
$$=(a_0+a_5)(b_0+b_5)+a_0b_0+a_5b_5+(a_1+a_4)(b_1+b_4)+a_1b_1+a_4b_4$$
$$+(a_6+a_7)(b_6+b_7)+a_6b_6+a_7b_7+(a_2+a_3)(b_2+b_3)+ a_2b_2+a_3b_3 \quad (14)$$

$$d_6= a_0b_6 +a_1b_5+a_2 b_4+a_3b_3+a_4b_2+a_5b_1+a_6 b_0+a_7b_7$$
$$=(a_0+a_6)(b_0+b_6)+a_0b_0+a_6b_6+(a_1+a_5)(b_1+b_5)+a_1b_1+a_5b_5+$$
$$(a_2+a_4)(b_2+b_4)+a_2b_2+a_4b_4+a_7b_7+a_3b_3 \quad (15)$$

$$d_7=a_0b_7+a_1b_6+a_2b_5+a_3b_4+a_4b_3+a_5b_2+a_6b_1+a_7b_0$$
$$=(a_0+a_7)(b_0+b_7)+a_0b_0+a_7b_7+(a_1+a_6)(b_1+b_6)+a_1b_1+a_6b_6$$
$$+(a_2+a_5)(b_2+b_5)+a_2b_2+a_5b_5+(a_3+a_4)(b_3+b_4)+a_4b_4+a_3b_3 \quad (16)$$

TABLE 5: Comparison of device utilization and combinational path delay to compute circular convolution using KA and MKA for 8-bit input

Name of Parameters	Circular Convolution using KA	Circular Convolution using MKA	Improvement (%)
No. of XOR gates	122	65	46.72
No. of Slice LUTs (out of 32640)	99	31	68.69
No. of bonded IOBs (out of 480)	24	24	0.00
Max. combinational path delay (ns)	9.614	7.336	23.69

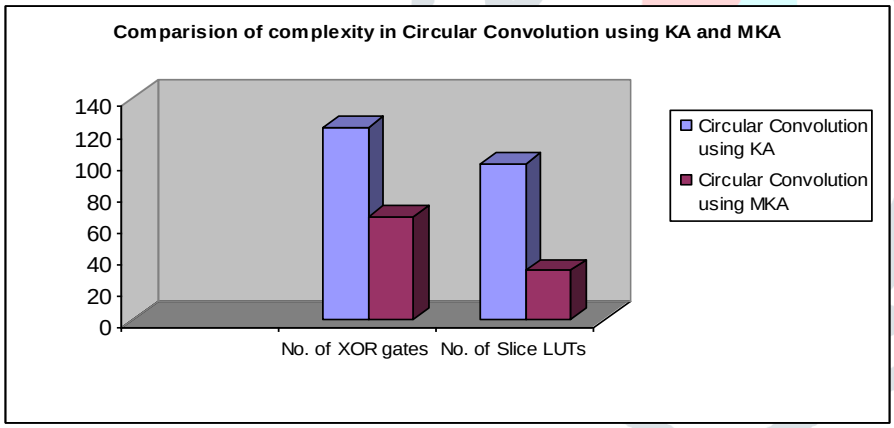


Fig. 9: Area occupied (% slices) between circular Convolution using KA and MKA

The circular convolution algorithm is coded using Verilog HDL language. It is simulated and synthesized using Xilinx ISE 14.3i software tool. Table 5 shows the comparison of device utilization and combinational path delay to compute circular convolution using KA and MKA. It is observed that circular convolution based on MKA requires least amount of area and path delay. Figure 9 shows the resource utilization in terms of % of XOR gates and slices, necessary for the implementation. In Vertex5 FPGA device family, computation of 8-bit circular convolution based on MKA is 23.69% faster than KA. It also consumes 68.69% less slices than existing KA based convolution.

6. CONCLUSION

In this paper, modified Karatsuba multipliers for degree 7 polynomials have been implemented on FPGA platform. The device utilization and combinational path delay of MKM have been compared with standard 8×8 KM. It has been observed that the proposed multiplier has better timing performance than standard KM. The new architecture is very simple and easy. This feature is advantageous to have a suitable trade-offs between area and speed for implementing circular convolution algorithm in VLSI. In FPGA device family, computation of 8-bit circular convolution using MKA is 23.69% faster than KA. It also consumes 68.69% less slices than existing KA based convolution. MKM may also be used to design cryptosystems. Proposed multiplier is faster and hardware efficient compared to existing Karatsuba multiplier.

References

1. Z. J. Shi and H. Yun, "Software implementations of elliptic curve cryptography," *International Journal of Network Security*, vol. 7, no. 1, (2008), pp. 141-150.
2. T. Zhang and K.K. Parhi, "Systematic Design of Original and Modified Mastrovito Multipliers for General Irreducible Polynomials," *IEEE Trans. Computers*, vol. 50, no. 7, (2001), pp. 734-749.
3. C. Paar, P. Fleischmann, and P. Roeise, "Efficient Multiplier Architectures for Galois Fields GF(2^{4n})" , *IEEE Trans. Computers*, vol. 47, no. 2, (1998), pp. 162-170.
4. C. A. Wang, T. K. Truong, H. M. Shao, L. J. Deutsch, J. K. Omura, and I. S. Reed, "VLSI architectures for computing multiplications and inverses in GF(2^m)", *IEEE Transactions on Computers*, Vol. 34, no. 8, (1985), pp.709- 717.
5. R. Masoleh and M.A. Hasan, "A New Construction of Massey- Omura Parallel Multiplier over GF(2^m)", *IEEE Trans. Computers*, vol. 51, no. 5, (2002), pp. 511-520.
6. E. R. Berlekamp, "Bit-Serial Reed-Solomon Encoder", *IEEE Trans. Inform. Theory*, vol. IT-28, (1982), pp. 869-874.
7. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers", in Doklady Akad. Nauk SSSR, vol. 145, (1962), pp.293-294.
8. C. K. Koc, and S. S. Erdem, "A Less Recursive Variant of Karatsuba-Ofman Algorithm for Multiplying Operands of Size a Power of Two", *Proceedings of the 16th IEEE Symposium on Computer Arithmetic*, (2003), pp.1063-1069.
9. M. Cenk and F. O. Zbudak, "Improved Polynomial Multiplication Formulas over F_2 Using Chinese Remainder Theorem", *IEEE Transactions on Computers*, vol. 58, no. 4, (2009), pp. 572- 576.
10. G. Zhou, H. Michalik and L. Hinsenkamp, "Complexity analysis and Efficient Implementations of Bit Parallel Finite Field Multipliers Based on Karatsuba-Ofman Algorithm on FPGAs", *IEEE Transactions on Very Large Scale Integration Systems*, vol. 18, no. 7, (2010), pp.1057-1066.
11. X.N. Xie, G.L. Chen and Y. Li, **"**Novel bit-parallel multiplier for GF(2m) defined by all-one polynomial using generalized Karatsuba algorithm", *Journal Information Processing Letters* , vol.114 no. 3, (2014), pp.140-146.
12. J. Samanta, R. Sultana and J. Bhaumik, "FPGA Based Modified Karatsuba Multiplier" *Int. Conf. on VLSI and Signal Processing (ICVSP14-IEEE Sponsored)* at IIT KGP, (2014), pp:1-6.
13. M.Thirumoorthi, M. Heidarpur,M. Mirhassani," An optimized M-term Karatsuba Like Binary Polynomial Multiplier for Finite Field Arithmetic ",IEEE transactions on very large scale integration(VLSI systems).