IMPLEMENTATION OF KARATSUBA ALGORITHM USING POLYNOMIAL MULTIPLICATION

SUDHANSHU MISHRA

Department of Electronics and Telecommunication,VeerSurendraSai University of Technology, Burla
Sambalpur-768018, Odisha, India
sudhanshu.mishra37@gmail.com

MANORANJAN PRADHAN

Department of Electronics and Telecommunication,VeerSurendraSai University of Technology, Burla
Sambalpur-768018, Odisha, India
manoranjan66@rediffmail.com

Abstract

Efficiency in multiplication is very important in applications like signal processing, cryptosystems and coding theory. This paper presents the design of a fast multiplier using the Karatsuba algorithm to multiply two numbers using the technique of polynomial multiplication. The Karatsuba algorithm saves coefficient multiplications at the cost of extra additions as compared to the ordinary multiplication method. The Karatsuba algorithm is more efficient for multiplication of large numbers.

Keywords**:** Karatsuba algorithm; FPGA; VLSI, polynomial multiplication.

1. Introduction

This paper presents the implementation of a fast multiplier using the Karatsuba algorithm to multiply two numbers using the technique of polynomial multiplication and comparison of combinational path delay and space requirements with that of a normal multiplier.

The authors Gang Zhou *et al.* have presented complexity analysis [both in application-specific integrated circuits (ASICs) and on field-programmable gate arrays (FPGAs)] and efficient FPGA implementations of bit parallel mixed Karatsuba–Ofman multipliers in [Zhou *et al.*, (2010)]. By introducing the common expression sharing and the complexity analysis on odd-term polynomials, they have achieved a lower gate bound than previous ASIC discussions. They have extended the analysis by using 4-input/6-input lookup tables (LUT) on FPGAs. They have evaluated the LUT complexity and area-time product tradeoffs on FPGAs with different computer-aided design (CAD) tools. They claim that their bit parallel multipliers consume the least resources among known FPGA implementations.

Karatsuba's multiplication algorithm uses three single digit multiplications to perform one two-digit multiplication. If Karatsuba's multiplier algorithm is applied recursively, it takes only 3^n single-digit multiplications to multiply a pair of 2^n-digit numbers. This is a significant improvement compared to 4^n single-digit multiplications using simple multiplication. In their paper [Liu *et al.*, (2003)], the authors have used tensor products to express the Karatsuba algorithm in both recursive and iterative form.

The authors [Koc, Erdem, (2003)] 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.

The Karatsuba multiplier can be used in avariety of applications like cryptographic techniques, digital signal processing and other computational areas involving multiplication.

2. Karatsuba algorithm

The basic step of Karatsuba algorithm can be used to compute the product of two large numbers *a* and *b*using three multiplications of smaller numbers, each with about half as many digits as a or b along with some additions and digit shifts.

Let a and b represent *n*-digit strings in some radix *R*. For any positive integer *m* less than *n*, the two numbers can be divided as follows:

$$a = a_i R^m + a_0. \quad (1)$$

$$b = b_i R^m + b_0. \quad (2)$$

where a_0 and b_0 are less than R^m. The product is then

$$ab = (a_1 R^m + a_0)(b_1 R^m + b_0). \quad (3)$$

or, $$ab = a_1 b_1 R^{2m} + a_1 b_0 + a_0 b_1 R^m + a_0 b_0. \quad (4)$$

or, $$ab = u_2 R^{2m} + u_1 R^m + u_0. \quad (5)$$

Where,

$$u_2 = a_1 b_1,$$

$$u_1 = a_1 b_0 + a_0 b_1,$$

and $$u_0 = a_0 b_0.$$

These formulae require four numbers of multiplications. But, it can be observed that the value of the product ab can be determined using only three numbers of multiplications, at the cost of a few more number of additions in the following manner:

After obtaining,

$u_2 = a_1 b_1$and $u_0 = a_0 b_0$,

the value of u_1 can be determined as:

$$u_1 = (a_1 + a_0)(b_1 + b_0) - u_2 - u_0. \quad (6)$$

since

$$u_1 = (a_1 b_0 + a_0 b_1) = (a_1 b_1 + a_1 b_0 + a_0 b_1 + a_0 b_0) - a_1 b_1 - a_0 b_0$$

or, $$u_1 = (a_1 + a_0)(b_1 + b_0) - a_1 b_1 - a_0 b_0. \quad (7)$$

2.1 *Example*

Let the product of numbers, 7654 and 6789, be determined using Karatsuba algorithm. For calculating the product of 7654 and 6789, the values of *R* and *m* can be chosen as 10 and 2 respectively.

$R = 10$and $m = 2$

$2178 = 21 \times 10^2 + 78$

$5423 = 54 \times 10^2 + 23$

$u_2 = 21 \times 54 = 1134$

$u_0 = 78 \times 23 = 1794$

$u_1 = (21 + 78)(54 + 23) - u_2 - u_0$

or, $u_1 = (99 \times 77) - 1134 - 1794$

or, $u_1 = 7623 - 1134 - 1794 = 4695$

Therefore, the product of 2178and 5423 can be calculated as:

$$2178 \times 5423 = (1134 \times 10000) + (4695 \times 100) + 1794$$

or, $$2178 \times 5423 = 11340000 + 469500 + 1794 = 11811294$$

3. General method of polynomial multiplication

Usually multiplication of polynomials is done in the following manner:

Let there be two degree-d polynomials with n = d+ 1 coefficients:

$$A(x) = \sum_{i=0}^{d} a_i \, x^i \qquad (8)$$

and

$$B(x) = \sum_{i=0}^{d} b_i \, x^i \qquad (9)$$

Then the product of *A(x)* and *B(x)* can be written as

$$C(x) = A(x)B(x) = \sum_{i=0}^{d} \sum_{j=0}^{d} a_i \; b_j \qquad (10)$$

The polynomial *C(x)* can be obtained with n^2 multiplications and $(n - 1)^2$ additions.

4. Karatsuba algorithm for degree-1 polynomials

This section describes the mathematical procedures of Karatsuba algorithm for degree-1 polynomials using simple algebraic manipulations.

Let there be two degree-1 polynomials, *A(x)* and *B(x)* given by:

$$A(x) = a_1x + a_0 \text{ and } B(x) = b_1x + b_0$$

Then the product*C(x) = A(x) B(x)* can be determined in the following manner:

$$C(x) = (a_1b_1)\, x^2 + (a_0b_1 + a_1b_0)\, x + a_0b_0\,. \qquad (11)$$

The coefficient of *x* in the above polynomial can be written as:

$$(a_0b_1 + a_1b_0) = ((a_0 + a_1)\,(b_0 + b_1) - a_0b_0 - a_1b_1). \qquad (12)$$

Let there be three auxiliary variables D_0, $D_1$and $D_{0,\,1}$ given by:

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

Then the polynomial *C(x)* can be written as:

$$C(x) = D_1x^2 + (D_{0,1} – D_0 – D_1)\, x + D_0. \qquad (13)$$

5. Karatsuba algorithm for degree-2 polynomials:

The technique mentioned above can be extended and applied for any polynomial having 2^i number of coefficients, where i > 0. But for polynomials having 2^j n number of coefficients, where j ≥ 0, n > 1 and n belongs to the set of odd integers, the above method cannot be applied directly.
This section describes the polynomial multiplication of two degree-2 multiplication, that is, when j = 0 and n =3, using Karatsuba decomposition technique.
\
Let there be two degree-2 polynomials, *A(x)* and *B(x)* given by:

$$A(x) = a_2x^2 + a_1x + a_0 \text{ and } B(x) = b_2x^2 + b_1x + b_0$$

Then the product *C(x) = A(x) B(x)* can be determined in the following manner:

$$C(x) = (a_2b_2)\, x^4 + (a_1b_2 + a_2b_1)\, x^3 + \\ (a_0b_2 + a_2b_0 + a_1b_1)\, x^2 + (a_0b_1 + a_1b_0)\, x + a_0b_0\,. \qquad (14)$$

The coefficients of *x*, (part of) $x^2$and x^3 in the above polynomial can be written as:

$$(a_0b_1 + a_1b_0) = ((a_0 + a_1)\,(b_0 + b_1) – a_0b_0 – a_1b_1),$$

$$(a_0b_2 + a_2b_0) = ((a_0 + a_2)\,(b_0 + b_2) - a_0b_0 – a_2b_2)$$

and

$$(a_1b_2 + a_2b_1) = ((a_1 + a_2)\,(b_1 + b_2) – a_1b_1 – a_2b_2)$$

The auxiliary variables D_0, D_1, D_2, $D_{0,1}$, $D_{0,2}$, $D_{1,2}$are given by:

$D_0 = a_0b_0$, $D_1 = a_1b_1$, $D_2 = a_2b_2$, $D_{0,1} = (a_0 + a_1)\,(b_0 + b_1)$ $D_{0,2} = (a_0 + a_2)\,(b_0 + b_2)$, *and* $D_{1,2} = (a_1 + a_2)\,(b_1 + b_2)$
The polynomial *C(x)* is given by:

$$C(x) = D_2x^4 + (D_{1,2} - D_1 - D_2)x^3 + \\ (D_{0,2} – D_2 - D_0 + D_1)x^2 + (D_{0,1} - D_1 - D_0)x + D_0. \qquad (15)$$

6. Karatsuba algorithm for polynomials of arbitrary degree

This section provides a generalization of the techniques presented above so as to multiply two polynomials of any arbitrary degree with n number of coefficients using the Karatsuba algorithm:
Let there be two degree-d polynomials with n number of coefficients such that n = d + 1 given by:

$$A(x) = \sum_{i=0}^{d} a_i\, x^i$$

and

$$B(x) = \sum_{i=0}^{d} b_i\, x^i$$

Then the following auxiliary variables can be computed:

$$D_i = a_i b_i \; [\ \forall i = 0, 1,2,\ldots, n\text{-}1]$$

$$D_{p,q} = (a_p + a_q)\,(b_p + b_q)\; [\ \forall i = 1,2,\ldots, 2n\text{-}3, \quad \text{and } \forall s, t \text{ such that } s + t = i \text{ and } t > s \geq 0\ ]$$

Then the product $C(x) = A(x)B(x) = \sum_{i=0}^{2n-2} c_i\ x^i$ can be determined by using the following values of c_i :

$$c_0 = D_0$$
$$c_{2n-2} = D_{n\text{-}1}$$

$$c_i = \begin{cases} \sum\limits_{p+q=i, q>p\geq 0} D_{p,q} - \sum\limits_{p+q=i, q>p\geq 0} (D_p + D_q)\,, & for\ odd\ values\ of\ i, 0 < i < 2n-2 \\ \sum\limits_{p+q=i, q>p\geq 0} D_{p,q} - \sum\limits_{p+q=i, q>p\geq 0} (D_p + D_q) + D_{i/2}, & for\ even\ values\ of\ i, O < i < 2n-2 \end{cases} \quad (16)$$

7. Synthesis results

The proposed fast multiplier using Karatsuba algorithm is coded in VHDL. It is synthesised and simulated using Xilinx ISE 10.1 software tool It has been implemented on Spartan 2s200pq208 FPGA device with speed grade of -6.

Table 1. Comparison of device utilization and combinational path delay of 8×8 Karatsuba Multiplier and normal Multiplier.

device (Spartan 2 xc2s200pq208)	number of slices	number of 4 input LUTs	number of bonded IOBs	maximum combinational path delay
8×8 (Karatsuba Multiplier)	26 out of 2352 (1%)	45 out of 4704 (0%)	31 out of 140 (22%)	12.338ns
8×8 (Normal Multiplier)	38 out of 2352 (1%)	73 out of 4704 (1%)	32 out of 140 (22%)	15.656ns

Table 1. and figure 1 show the comparison of device utilization and combinational path delay of 8×8 Karatsuba Multiplier and normal Multiplier. The number of slices and combinational path delay for 8×8 Karatsuba multiplier are 26 out of 2352 (1%) and 12.338ns respectively. Whereas, the number of slices and combinational path delay for 8×8 normal multiplier are 38 out of 2352 (1%) and 15.656ns respectively.

The above observations justify that the proposed Karatsuba multiplier using polynomial multiplication uses less number of slices and at the same time the maximum combinational path delay is also less. Hence, the proposed 8×8 multiplier has speed improvement using lesser space than the corresponding normal 8×8 multiplier.

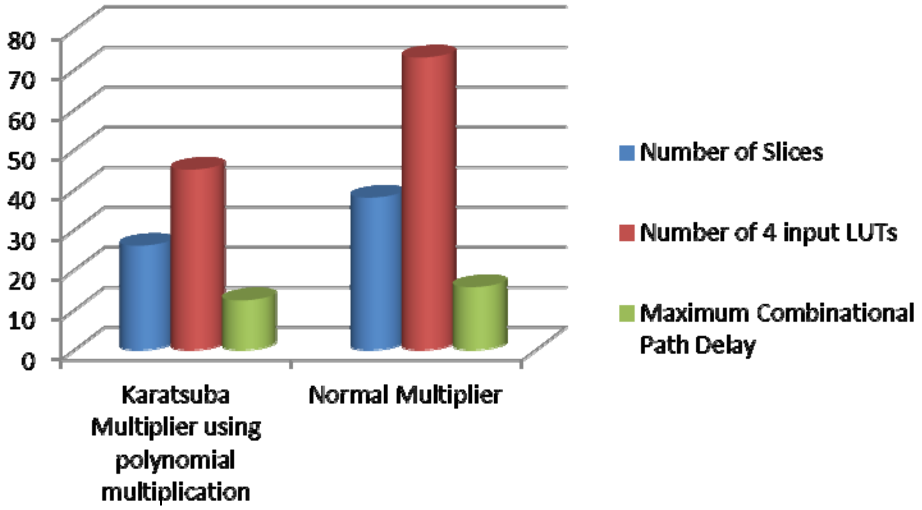

Fig.1. Comparison of device utilization and combinational path delay of 8×8 Karatsuba Multiplier and normal Multiplier

The simulation results of proposed 8×8 Karatsuba algorithm using polynomial multiplication have been shown in figure 2. The figure shows the decimal equivalent of multiplication of two 8-bit numbers to give the product. Ports ‘a’ and ’ b’ are the input ports that accept the numbers to be multiplied while the port ‘d’ is the output port where the product of the two aforesaid numbers are obtained. For example, the product of 8 and 10 (decimal equivalents), specified at the ports ‘a’ and ‘b’ (input ports) respectively, is obtained at port ‘d’ (output port). Similarly, products of all other specified numbers are obtained.

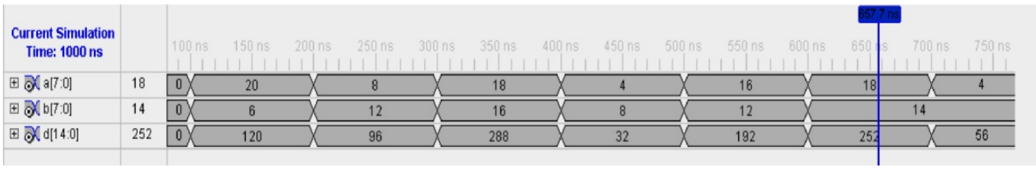

Fig. 2. Simulation results of Karatsuba algorithm using polynomial multiplication

8. Conclusion

The device utilization and combinational path delay of proposed 8×8 Karatsuba algorithm using polynomial multiplication has been compared with normal 8×8 multiplier. It has been observed that the proposed multiplier has better time performance over normal multiplier. This may be useful for digital signal processing techniques and cryptosystem applications.

9. References

[1] J. von zur Gathen and J. Shokrollahi, (2005): Efficient FPGA-based Karatsuba multipliers for polynomials over F_2. Selected Areas in Cryptography, Lecture Notes in Computer Science, 3897, pp. 359–369.

[2] Koc, Cetin K.; Erdem, Serdar S., (2003): 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, 1063-6889.

[3] Lima, Juliano B., *et al.*, (2010): A Karatsuba-Based Algorithm for Polynomial Multiplication in Chebyshev Form. IEEE Transactions on Computers, 59(6), pp. 835-841.

[4] Liu, Chin-Bou; Huang, Chua-Huang; Lei, Chin-Laung. (2003): Design and Implementation of long-digit Karatsuba’s multiplication algorithm using tensor product formulation. Ninth workshop on compiler techniques for high performance computing.

[5] Montgomery, P.L., (2005): Five, six, and seven-term Karatsuba-like formulae. IEEE Transactions on Computers, 54 (3), pp. 362–369.

[6] Rebeiro, C.; Mukhopadhyay,D, (2008): Power attack resistant efficient FPGA architecture for Karatsuba multiplier. 21st International Conference on VLSI Design, pp. 706–711.

[7] Weimerskirch, A.; Paar, C., (2006): Generalizations of the Karatsuba algorithm for efficient implementations. [Online].Available: http://eprint.iacr.org/2006/224.pdf

[8] Zhou, Gang; Michalik,Harald; Hinsenkamp, László (2010): 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 (VLSI) Systems, 18 (7), pp.1057-1066.