WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

An Efficient and Overlap Free Urdhva Tiryagbyam Multiplier Using HDL

S.Hari Krishnan[1],D.Ganesh[2],G.Naga Sai[3],M.MadanMohan Reddy[4],T.Manjunath Reddy[5]

[1] Vice Principal, Sanskrithi School of Engineering, Puttaparthi, Andhra Pradesh, India

[2] *Department of Electronics and Communication Engineering, Sanskrithi School of Engineering, Puttaparthi, Andhra Pradesh*

DOI: https://doi.org/10.55248/gengpi.4.423.38171

A B S T R A C T

Numerous applications, including error-correcting codes and cryptography, heavily rely on arithmetic in the finite/Galois field. The two fundamental operations in the finite field GF are addition and multiplication. The bit parallel Karatsuba Multiplier over GF (2m) complexity (space) study and effective FPGA implementation are provided. Due to its carry-free characteristic, this is extremely intriguing for high performance systems. By using a Karatsuba multiplier, we can increase the process' efficiency. For tiny digital systems, this research suggests an optimal polynomial multiplication method. This idea suggests brand-new optimisations for the polynomial multiplier, the component of lattice-based cryptography that requires the greatest computing, with a focus on high-speed hardware.Polynomial multiplication is regarded as the most time- and space-consuming operation in ECC systems. This article suggests a novel hardware design for effective Finitefield multiplier implementation for ECC using field-programmable gate arrays (FPGAs). The original Karatsuba has been speed-optimized in the OKA. This approach divides inputs into odd and even orders rather than high and low sections in order to reduce the longest path time. In the proposed design, a 2-bit and a 4-bit binary polynomial multiplier with six input LUTs is implemented as an example DFG for an FPGA.This work can be improved by comparing the performance of the Karatsuba algorithm to that of other multiplication-related algorithms that have been developed. Additionally, an algorithm for improving multiplication operation efficiency and decreasing process cost, area, and latency can be created.

Keywords: Xilinx Software, FPGA

1. Introduction

An UrdhvaTiryagbyam multiplier is a type of ancient Indian multiplication technique, which involves a series of vertical and horizontal operations to efficiently calculate products. An efficient and overlap-free UrdhvaTiryagbyam multiplier using HDL refers to a modern implementation of this technique in digital hardware using hardware description language (HDL). This approach involves designing a circuit that can perform UrdhvaTiryagbyam multiplication in a highly optimized and parallelized manner, without any overlapping of operations. Such a circuit can be used in various digital systems and applications, such as signal processing, image and video processing, and scientific computing, to perform high-speed and accurate multiplication operations.

2. Literature Review

- We did a thorough evaluation of the literature on in order to contextualize our work and identity research gaps in the fields of multiplying of higher order bits. Moslem Heidarpur and Mitra Mirhassani[1] described the paper as I An Efficient and High-Speed Overlap-Free Karatsuba-Based Finite-Field Multiplier for FGPA. Implementation results indicated that the proposed method is, on average, 30% faster than Karatsuba and 20% faster than the OKA.
- Sun mi park,Ku-young chang,Dowon Hong.,Changhoseo[2].These authors described the paper Combining algorithmic techniques with platform dependent strategies can be used to develop designs which are highly optimized for FPGAs, However, internal and exterior variations may make accuracy difficult for conventional approaches. The results of an enhanced convolutional neural network-based algorithm are encouraging.
- AjithaS.S,Retheesh.D[3] This research visvalizes the Efficient implementation of bit parallel finite field multipliers. This paper mainly two multipliers namely classical and Karatsuba multiplier is more efficient than the other multiplier .Karatsuba improve the performance of the process

3. Methodology

POLYNOMIAL MULTIPLICATION:

Polynomial multiplication is a mathematical operation used to multiply two polynomials together. A polynomial is a mathematical expression consisting of variables and coefficients, with the variables raised to non-negative integer exponents.

To multiply two polynomials, we use the distributive property of multiplication over addition. Specifically, we multiply each term of one polynomial by each term of the other polynomial and then combine like terms.

For example, let's say we want to multiply the polynomials (2x + 3) and (x - 1). We start by multiplying the first term of the first polynomial (2x) by each term of the second polynomial (x and -1), then we multiply the second term of the first polynomial (3) by each term of the second polynomial, like so:

(2x + 3) * (x - 1) = (2x * x) + (2x * -1) + (3 * x) + (3 * -1) = 2x^2 - 2x + 3x - 3 = 2x^2 + x - 3

Therefore, the product of (2x + 3) and (x - 1) is 2x^2 + x - 3.

Polynomial multiplication is a basic concept which is taught to the students since their school life. But, it is very important topic when it comes to computer science field, as in algorithms, is we are supposed to design an algorithm for this polynomial multiplication, then, definitely we all were a simple algorithm which will have a time complexity of $O(N^2)$.

- We have 2 polynomials of of degree (n-1) each.
- Take an array of size 2n-2.
- Run 2 for loops, each will run through 0 to n-1.
- We assign final product array like this, p[i + j] = A[i] * B[j].

improvement is that divide and conquer approach was used to solve the problem.

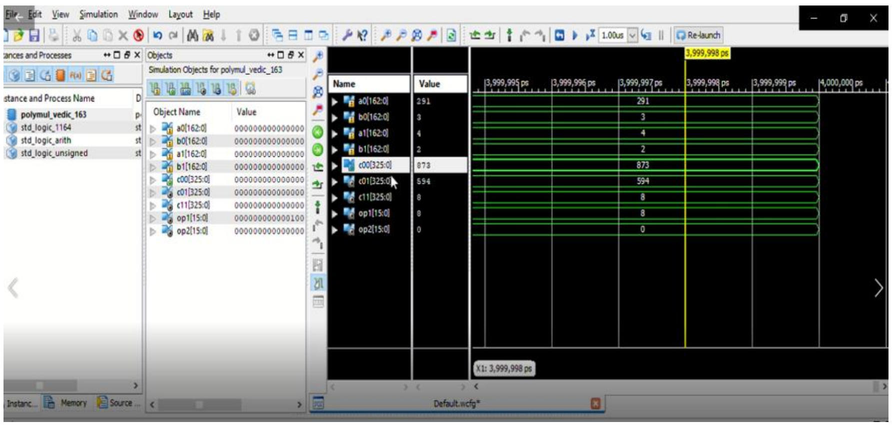
Steps are as follows:

- Divide each polynomial in 2 halves: upper and lower.
- For first polynomial we denote upper and lower halves with D1 and D0 respectively. For second polynomial we denote the same by E1 and E0.
- Upper half of each polynomial is in the form: a1* x ^ (n/2 -1) + a2 * x ^ (n/2-2) + ……., where a1, a2…. are coefficients of each term, and are actually like a ((n-1) in subscript), a((n-2) in subscript) and goes till a(n/2 in subscript).
- Lower half of each polynomial is in the form: a1* x ^ (n/2 -1) + a2 * x ^ (n/2–2) + ……., where a1, a2…. are coefficients of each term, and are actually like a ((n/2 -1) in subscript), a((n/2-2) in subscript) and goes till a(0 in subscript).
- Now, to obtain the final multiplied polynomial, we have to do, **((E1*D1) * (x^n) + (E1*D0 + E0*D1) * (x ^ n/2) + E0D0)** < = *it represents the final multiplied polynomial.*
- Here, we divided the polynomial in the 4 sub problems. So, its recurrence relation will be **T(n) = 4*T(n/2) + cn,** where c is constant.
- After solving the above relation we get Time complexity as **$O(N^2)$,** at each level the work is **4^i *c * T(n/(2 ^ i)) => 2 ^i * c * T(n).** The algorithm goes till log2(n) levels. Therefore work is **c * 4 ^ log2(n) = c * n ^log2(4) = $O(N^2)$** [by log property].

4.Outputs and Results:

The input data is given in the form of bits in Xilinx tool software.By giving various higher order data bits we can do multiplicaion for the data.

The below figure shows the output for the given input data.



Object Name	Value
a0[162:0]	000000000000000
b0[162:0]	000000000000000
a1[162:0]	000000000000000
b1[162:0]	000000000000000
c00[325:0]	000000000000000
c01[325:0]	000000000000000
c11[325:0]	000000000000000
op1[15:0]	000000000000100
op2[15:0]	000000000000000

Name	Value
a0[162:0]	291
b0[162:0]	3
a1[162:0]	4
b1[162:0]	2
c00[325:0]	873
c01[325:0]	594
c11[325:0]	8
op1[15:0]	8
op2[15:0]	0

5.Conclusion

In conclusion, the design and implementation of an efficient and overlap-free UrdhvaTiryagbyam multiplier using HDL can provide a significant advantage in terms of speed, accuracy, and area utilization over conventional multiplication techniques. By leveraging the principles of UrdhvaTiryagbyam multiplication, the circuit can perform multiplication operations in a highly optimized and parallelized manner, resulting in faster and more efficient computation. Moreover, the use of HDL allows for easy customization and integration of the multiplier into larger digital systems. Therefore, the development of an efficient and overlap-free UrdhvaTiryagbyam multiplier using HDL holds great promise for various digital applications that require high-speed and accurate multiplication operations.

Acknowledgements

Mr.S.Hari Krishnan, guide of the project has guided us throughout the project in each and every situation.

The team G.Naga Sai,M.Madan Mohan Reddy,T.Manjunath Reddy, D.Ganesh(myself) has worked in development of code and various modules of the project.

Reference:

[1] C. P. Rentería-Mejía, A. López-Parrado, J. VelascoMedina, “Hardware Design of FFT Polynomial Multipliers”, 978-1-4799-2507-0/14/$31.00 ©2014 IEEE.

[2] Lo Sing Cheng, Ali Miri, Tet HinYeap, “EFFICIENT FPGA IMPLEMENTATION OF FFT BASED MULTIPLIERS”, 0-7803-8886-0/05/$20.00 ©2005 IEEE.

[3] E. Theochari, K. Tatas, D. J. Soudris, K. Masselos, K. Potamianos, “A REUSABLE IP FFT CORE FOR DSP APPLICATIONS”, 0-7803-8251-X/04/$17.00 © 2004 IEEE.

[4] A.Ronisha Prakash, S.Kirubaveni, “Performance Evaluation of FFT Processor Using Conventional and Vedic Algorithm”, 2013 IEEE International Conference on Emerging Trends in Computing, Communication and Nanotechnology (ICECCN 2013), 978-1-4673-5036- 5/13/$31.00 © 2013 IEEE 89.

[5] Ali Chamas Al Ghouwayel, Amin Haj-Ali and Zouhair El-Bazzal, “Towards a Triple Mode Common Operator FFT for SoftWare Radio Systems”, 19th International Conference on Telecommunications (ICT 2012), 978-1- 4673-0747-5/12/$31.00 ©2012 IEEE.