ISSN: 2319-345X
IJMRBS
International Journal of
Management Research and
Business Strategy
www.ijmrbs.org
E-mail
editor@ijmrbs.org
editor.ijmrbs@gmail.com

ISSN 2319-345X www.ijmrbs.com

Vol. 9, Issuse. 4, Nov 2021

EFFICIENTVLSIIMPLEMENTATIONOFASEQUENTIAL FINITE FIELD MULTIPLIER USING REORDERED NORMAL BASIS IN DOMINO LOGIC

P.NAGASUDHAKAR1,S.NAZMA2

Abstract- An efficient VLSI implementation of a finite field multiplier in GF(2m) is described in this paper.. Serial-in, parallel-out architecture with a reordered normal basis for multiplication is offered. Main building block of multiplier is implemented in domino logic in order to shorten critical route time. Keeper control circuits are used to reduce dynamic power consumption by restricting keeper transistor contention currents during initial evaluations, resulting in lower dynamic power consumption. The 65-nm CMOS technology was used to implement the multiplier's semicustom layout. Simulations after the fact indicated that the multiplier can multiply accurately up to 3.85 GHz and consumes marginally less power than the static counterpart in CMOS. This is a significant improvement (also implemented with custom placement and route). Binary field multiplication in elliptic curve cryptography requires a multiplier of the current size recommended by the National Institute of Standards and Technology (NIST). Similar finite field multipliers with regular structures can also benefit from the proposed design process.

Index Terms— SIPO finite field multiplier. Domino logic.Elliptic curve cryptography. Finite field arithmetic reordered normal basis (RNB).

INTRODUCTION

Field operations, such as elliptic curve cryptography (ECC) and Elgamal cryptosystem, necessitate efficient computations of finite field arithmetic in these cryptographic applications. It is possible to carry out arithmetic operations on the field's elements without ever leaving the binary extension field GF(2m). m-bit sequences are all that is required to represent a finite field. As a vector space spanned by m linearly independent elements, called a basis, a field can be described as an object. It is critical to select a method of representation for field elements that is both accurate and efficient.implementationoffinitefieldoperations.Anumber of bases over finite fields have been proposed inthe literature, among which polynomial basis (PB) and normal basis (NB) are primarily used in practice. Although the use of PB is most commonin software implementations, NB offers a virtually cost-free squaring operation performed by a single Field element coordinates are shifted cyclically, making it the best choice for a hardware implementation. To execute sophisticated field operations (e.g., exponentiation and division), it is essential to have an efficient implementation of field multiplication, which can be used in conjunction with other operations (such as field

1AssistantProfessor,DeptofECE,CBIT,Proddutur,AP,India.
2AssistantProfessor,DeptofECE,CBIT,Proddutur,AP,India.

multiplication). Every field in GF has been confirmed to have an NB (2m). A matrix multiplication is required for each product coordinate in order to model the multiplication process as a matrix-vector multiplication in NB. The amount of nonzero entries in the multiplication matrix has a direct impact on the hardware complexity of the operation. The complexity of NB is represented by the numeric code CN. Type I and II ideal NBs are two subclasses of NB that exhibit the lowest possible CN for a given m. These two extreme CN values are denoted as 2m+1 and m2, respectively (ONBs). For the NB subclass including a type-II ONB, Gao and Vanstone were the first to propose a mathematical definition for reordered NB (RNB). By defining multiplication as a closed-form operation, RNB can reduce the complexity of the operation.matrix operation is preferred over the formulation formula. For applications where speed is a top priority, a fully parallel design would be the obvious choice. Additionally, high-order fields (m > 160) are suggested by cryptographic standards to assure excellent security. Even yet, given the fact that a parallel architecture has an area complexity of O(m2), it is not suitable for resource limited applications to have a large m. On the other hand, the area complexity of a fully serial (sequential) multiplier is O(m), which results in a much smaller structure. When compared to a completely parallel architecture, sequential multipliers use many more clock cycles to perform the same amount of multiplication in the same amount of time. As a result, it's in everyone's best interest to cut down on waste.A sequential multiplier's multiplication delay compensates for this flaw. SIPO RNB multiplier in GF is implemented using an efficient VLSI implementation in this study (2m). As Wu et al. have suggested, our approach is built on a sequential architecture. The structure of this design, which derives from an inherent characteristic of RNB, is remarkably regular. A high-speed custom-layout multiplier was previously built using the regularity of this architecture's core building component, which was implemented in domino logic. In terms of critical path, however, the improvement is negligible delay is obtained at the increase in the price of electricity. The domino logic circuit's main downside is this. With the custom-designed domino logic circuit in this work, we hope to further improve the multiplier's performance while simultaneously reducing the domino circuit's power consumption. Compared to its corresponding static CMOS implementation, the new solution greatly improves the maximum operating frequency and reduces power consumption to a comparable level.

I. EXISTINGSYSTEM

The wide range of applications for finite field computation in error control coding, coding theory, and, most importantly, cryptography, make it extremely valuable. As public-key cryptography's use grows in resource-constrained situations, the need for a power-efficient solution has grown as well. For elliptic curve cryptography (ECC) public key protocols, operations like scalar multiplication and the elliptic curve group operations point addition and point doubling are necessary. FFI operations such as addition and multiplication form the base of this hierarchy. Since more difficult operations like exponentiation and inversion can be performed with sequential usage of multiplication, finite field multiplication is critical in field computation. The choice of the representation of field elements has a direct impact on the efficiency of a multiplication operation. It has been proposed in literature and practice to utilize a variety of different bases over finite fields, including polynomial base, normal base, dual base and redundant base, all of which are already in use. Software developers frequently utilize polynomial basis since it takes less processing power than other methods. However, a simple circular shift over the coordinates performs a low-cost squaring operation on normal basis.of field elements, thus making it suitable for hardware implementation. This advantage has been widely exploited to accelerate the inversion operation by performing a series of field squaring and field multiplication operations based on Fermat’s Little Theorem (FLT). In normal basis, multiplication operation is generally modeled as a matrix-vector multiplication where a matrix multiplication is required to be carried out to generateeachelementoftheproductcoordinates.It is evident that the computational complexity of multiplicationoperationsdependsonthenumberof nonzero elements inside the multiplication matrix. This quantity is referred to as the complexity of normal basis and is denoted by CN. It has been shown that CN is a function of field size m and selected irreducible polynomial and can vary between a lower bound of 2m + 1 and a higher bound of m2. For two subclasses of normal basis known as type I and II Optimal Normal Basis (ONB)thecomplexityofnormalbasisisminimal, i.e. 2m+ 1. Reordered Normal Basis (RNB) is a permutation oftype-II ONB, first presented byGao et al.. This representation system can facilitate the hardware implementation of multiplication operation byexpressing it as aclosed form formula instead of a matrix operation.

A.Word-The RNB Multiplier's level architecture

An m-bit word-level multiplier architecture is shown in Fig. 1. The xax-cell architecture, depicted in the figure, is characterized by its regularity, consisting of parallel connections to a single repeating unit. Within a dashed box, we can see this circuit, which consists of two XOR gates and one AND gate.

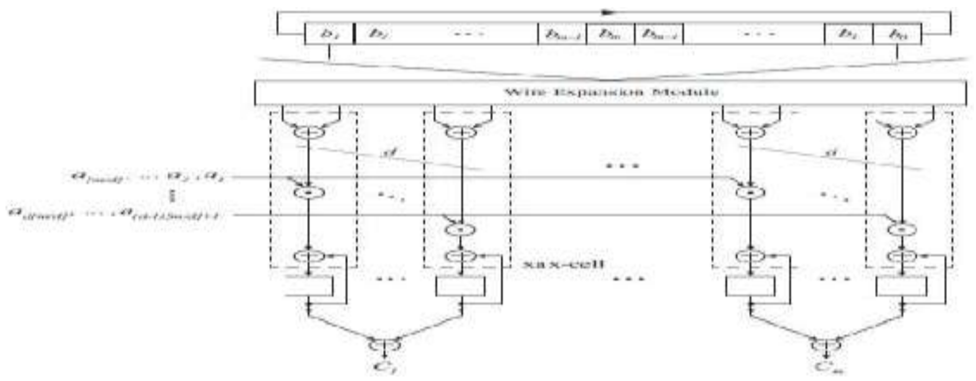

Figure 1: Word-level RNB multiplier composed of xax-cells

One of the input coefficients is used to set a circular shift register displayed at the top of the picture, while the other input is supplied into the multiplier in the form of a digit-serial sequence. Clocks go through w cycles.coordinateoftheproductCcanbeobtainedby summinguptheoutputsofdaccumulationunits. Fig. 2shows theproposeddesign foranxax-cell. ThestaticPUNconsistsofasinglepMOS transistor that charges the dynamic node Q during theprechargephase.TransistorsN4□N15forma PDNresponsibleforrealizingcombinationalfunction ((b1 _ b2) : a) and discharge the dynamic nodewhencertaincombinationsofinputvaluesare applied.Fourinvertergatesalsoexistinthemodule (notshowninthefigure)whichgeneratethe complementsofinputsignalsforthePDN.The PDNisconnectedtoafootertransistor,N16,which reducestheleakagecurrentduetothestacking effectandopensapathtothegroundduringthe evaluation phase. Transistors P2 and N2 generate a controlsignaltothenMOSkeeperdependingon thestatusofthedynamicnodeandclocksignal. TransistorsP1andN1formtheoutputinverting stage, providing the output current drawn from the module.Theproposeddominocircuitoperatesin two phases as follows:

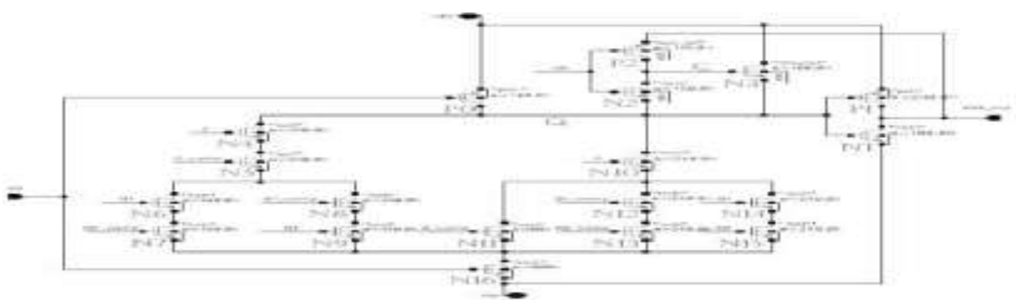
Figure 2: Existing design for XOR-AND-XOR cell in domino logic

II. **PROPOSEDSYSTEM**

A. Design of the Multiplier's Main Building Block In Domino Logic

Figure 3 shows that the xax-module contains the critical path of the SIPO design, which can be viewed. The xax-module contains two XOR gates, as well as an AND gate, for this path. Namin et al. have created and implemented the critical path of the multiplier in domino logic in an effort to reduce the multiplication delay. The maximum operating frequency can be increased by employing domino logic for the main building block of the multiplier, however the power dissipation is degraded as a result of this strategy. Domino logic circuits have a higher internal switching activity, which contributes to the increased power consumption. As a result, compared to a static CMOS counterpart, the proposed device would use significantly more dynamic power.High-speed domino, XOR-based domino, conditional-keeper domino, single-phase domino and current comparison domino have all been presented in the past few years to minimize the negative consequences caused by employing domino logic architectures over static CMOS. It is important to use design methodologies that take into account the transistor stacking effect, smaller noise margins, evaluation time, and leakage current in deep sub-micron technologies. These solutions, on the other hand, are better suited to high fan-in circuits, such as multiplexers, comparators, and more generic OR-like cells, where the Pull-Down Network (PDN) comprises a large number of parallel pathways to the ground. These strategies require a considerable number of transistors compared to the overall number of transistors in the design of the compact xax-module, resulting in significant power and area overheads. The multiplier in question is not a candidate for such procedures. An attempt is made to reduce the power dissipation problem here by lowering initial contention current demand.

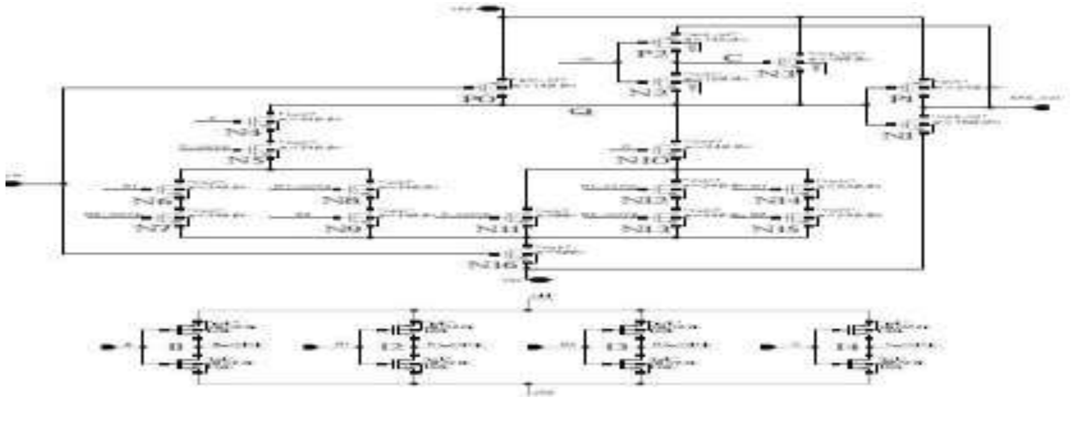
Figure 3: Proposed design for XOR-AND-XOR function in domino logic

Depending on the value of the input signals, contention may occur between the Pull-UpNetwork (PUN) and the Pull- Down Network (PDN) of a domino circuit during the evaluation phase. This contention, though short in time, forms a conducting path from VDD, across PUN and PDN, to ground causing high amplitude current spikes. The basic idea is to limit the contention current by utilizing a new conditional keeper to compensate for the power overhead caused by the higher switching activity of the circuit. Fig. 3shows a schematic of the circuit designed to implement XOR-AND-XOR function in domino logic.ThiscircuitisresponsibletorealizeanXOR operation between two separate B coordinates, followed by an AND applied to the result and one of the B coordinates. To round things off, we'll add an accumulation unit by pairing an XOR with another flip-flop. Using the variables in this circuit, this circuit performs the logic function ((b1 b2): a). Only a single pMOS transistor is used to charge the dynamic node Q to VDD during the pre-charging phase. On the other hand, the pull-down network comprises of 12 transistors (N4-N15) that discharge the dynamic node when the input values are appropriate. An N16 footer transistor connected to the PDN lowers leakage current owing to the stacking effect and provides a path to ground during the testing process. Based on the voltage of the dynamic node (P2) and the logic state of the clock signal (N2), a control signal is generated by transistors P2 and N2. To power the output flip-flop, we use transistors P1 and N1 as the output inverters. The input signals are shown in the schematic diagram. referred to as B1, B2, and A and C. the module's input signals are generated by four inverter gates (I1-I4) illustrated below. A "comp" at the end of a signal's name indicates that it is its complement. The dynamic circuit is divided into two parts: Pull-up transistor P0 steadily charges the dynamic node during the precharging period. Node C is swiftly charged to VDD if the dynamic node is originally in a low state by P2, which activates the keeper transistor to speed up the precharging process. It is at this point that P2 flips to a low state, which discharges node C and shuts down both the keeper transistor and the dynamic node, as the voltage on the dynamic node climbs to this point. Because of this, the dynamic node is completely charged at the end of the precharge phase, and the keeper is held off until the beginning of the next phase in order to minimize delays and power usage. The clock signal is raised to a high level at the start of the evaluation phase, keeping the pull-up transistor off. The logic values of the input signals could lead to two alternative outcomes at this point. In the first scenario, the dynamic node is discharged through the PDN network by forming a conducting channel to the ground. Voltage drops below VDD in this situation.□ Vth;N2, the source and drain junctions of transistor N3 are reversed and the accumulated charge on node C is fully discharged through N3. This prevents the keeper transistor from being turned on. In the second scenario, the dynamicnode is evaluated to a high state. N2 is turned onin the case that the leakage current reduces thevoltageofthedynamicnode.Thebehavioroft he circuitshownin Fig.6.2is explainedin moredetail in our recent work.

III.**SIMULATION RESULTS AND PERFORMANCE COMPARISON BETWEEN DIFFERENTVLSIIMPLEMENTATIONS**

This section compares the proposed VLSI implementation's properties to those of other implementations in the literature. Calibre PEX was used to extract the complete multiplier's parasitic information in order to do accurate simulations. Components such as parasitic capacitances and resistors were removed from the physical layout at this point. The circuit's power consumption and maximum operational frequency were then determined by simulations in Cadences Analog Environment with Spectre simulator. An first pre-simulation of the circuit was necessary to create the test data set and assure proper operation. MATLAB was used to model the multiplier's operational behavior. The golden product coordinates were generated by feeding a huge array of random 233-bit paired vectors into the MATLAB function. Two files were used to store the input and output data. Verilog-A was used to read the input files and feed an input pair into the multiplier for each multiplication operation during the

analog simulations.

Figure 4: The proposed layout for a 233-bit sequential RNB multiplier designed in domino logic.

Before loading new data into the multiplier, the output coordinates were sampled and saved in an output file. The MATLAB code was used to create a golden set of outputs that were then compared to these outputs. At a clock rate of 3.84 GHz, the simulation demonstrated that the circuit worked appropriately. It was found that the multiplier's power consumption averaged out to 13:01 mW/GHz across 100 successive multiplications. The primary goal of this study, as stated earlier in Section 6:1, is to compare Demonstrate that the new domino logic circuit may significantly minimize the multiplication delay of the multiplier while keeping the total power consumption. TSMC's standard cell libraries were utilized to implement the configuration of the static CMOS multiplier in order to ensure a fair and accurate comparison

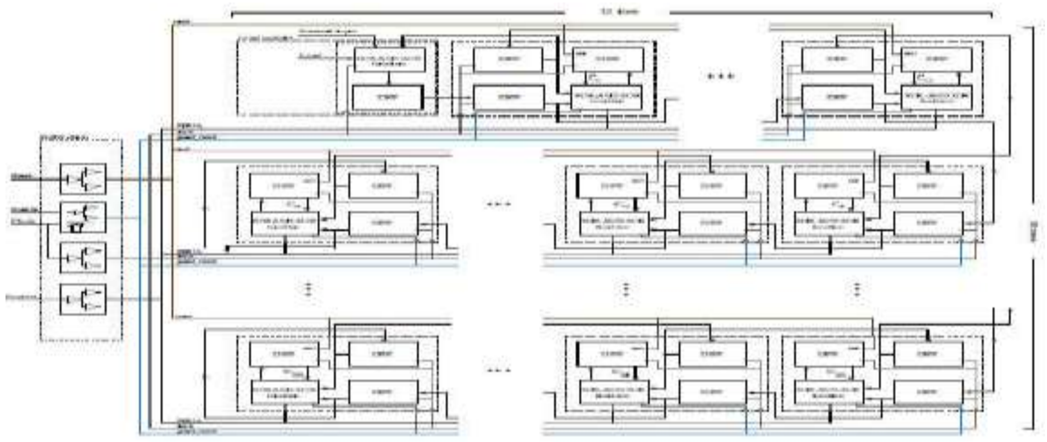

Figure 5: Block diagram of a full 233-bit RNB multiplier

The layout of the static design was constructed based on the same structure shown in Fig. 5.

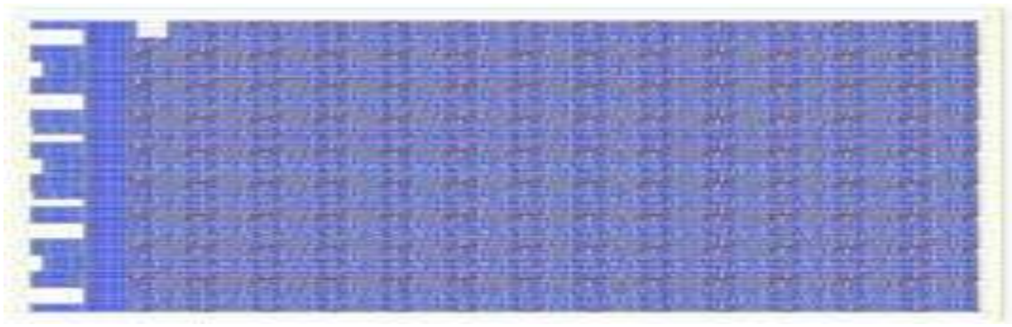

Figure 6.8: The static CMOS layout for a 233-bit sequential RNB multiplier

The final layout of the multiplier is presented in Fig. 6. Note that the Load module was implemented in static CMOS and then was incorporated in the layout to provide the same functionality as its counterpart in domino logic. To ensure consistency in all of the measurements, the same set of random inputs was applied to the static multiplier during the simulations conducted. This realization has a maximum operating frequency of 2:94 GHz and requires 158:44ns to finish a single multiplicationoperation.Includingthepowerrings, the size of the layout is 153m, 71_m, equal to an area of 10; 863_m2. The required area is reduced to9;574m2whennotconsideringtheouter rings.

IV. **CONCLUSION**

A 233-bit SIPO finite field multiplier VLSI implementation was provided. The National Institute of Standards and Technology (NIST) now recommends a field size of 233 for embedded ECC applications. By cascading a certain number of blocks, the proposed architecture can be easily extended to any arbitrary size multiplier while still maintaining a very regular pattern of a single building block implemented in domino logic. As a means of reducing the excessive power dissipation caused by the domino circuit due to increased internal switching activity, modifications were made to the original design of this building block. Place and route simulations confirmed that the design worked correctly up to a clock range of 3.85 GHz, allowing it to operate at significantly greater speeds and consume slightly less power than the static CMOS version. There are additional regular architectures that can benefit from the same design process without sacrificing performance or power consumption.

REFERENCES

[1] Public-Key Cryptography: IEEE Standard Specifications, IEEE Standard 1363-2000, Aug. 2000, pp. 1–228. [1]

[2] Introduction to Finite Fields and Their Applications: 2nd ed.

[3] The optimal normal bases in GF(pn) were first published in Discrete Appl. Mathematics, Volume 22, Number 2, Feb. 1989, Pages 149–161.

[4] "On the optimal normal basis generators' order," Math. Comput., vol. 64, no. 211, pages 1227–1233, 1995, is the work of S. Gao and S. A. Vanstone."

[5] "Computational method and apparatus for finite field arithmetic," J.K. Omura& J.L. Massey,

[6] U.S. Patent No. 4,597,627, issued on May 6, 1986;

[7] Improved VLSI designs for multiplication and inversion of GF(2m) over normal bases are shown in the Proceedings of the 13th Annual IEEE Int. ASIC/SOC Conference, which was held in September 2000.G. B. Agnew, R. C. Mullin, I. M. Onyszchuk, and S. A. Vanstone, “An implementation for a fast public-keycryptosystem,” J. Cryptol., vol. 3, no. 2, pp. 63–79, Jan. 1991.

[8] G.-L. Feng, “A VLSI architecture for fast inversion in GF(2m),” IEEE Trans. Comput., vol. 38, no. 10, pp. 1383–1386, Oct. 1989.

[9] A. Reyhani-Masoleh and M. A. Hasan, “Low complexity word-level sequential normal basis multipliers,” IEEE Trans. Comput., vol. 54, no. 2, pp. 98–110, Feb. 2005.