ENGINEERING IN ADVANCED RESEARCH SCIENCE AND TECHNOLOGY

UGC Care Group-I
Vol.03, Issue.01
November-2022
Pages: -1152-1164

DENSITY OPTIMISED RNS BASED LOW LATENCY AND HIGH SECURED ED448

[1]CHITIKINA VENKATA SATISH, [2]Dr.B.Raja Rao
[1]M. tech, Dept. of ECE, Eluru College of Engineering and Technology, ELURU, AP
[2]Professor & H.O.D, Dept. of ECE, Eluru College of Engineering and Technology, ELURU, AP

ABSTRACT: In particular, concept objective is to create a system-on-chip (SoC) crypto-accelerator with an MCU such that achieve high area-time efficiency, rather than creating a very low area or ultra-high performance implementations at the high cost of the other. This implementation can also be integrated as an off-chip solution; however, other criteria, such as performance, are often as important as or more important than efficiency in the external crypto-chip design. This method proposes the first XILINX-based EdDSA architecture over Ed448. Moreover, complete signature and verification implementations with certificate handling are scarce. Hence, a direct comparison of the area utilization and performance is difficult since the implementations target different schemes and security levels, and they use different platforms and technologies. Further this concept is enhanced by using RNS system. Some redundant number systems, such as the residue number system, have interesting and potentially useful characteristics in the arithmetic operations of multiplication, addition and subtraction. The proposed architecture is based on the CRT. The need for an intermediate binary stage is eliminated. Overall, the proposed architectures facilitate the implementation of RNS based processors by reducing the latency and complexity introduced by the binary stage. This makes it more possible and more practical to build effective RNS based processors.

Keywords: elliptic curve cryptography, Public key cryptography, Residue Number system, Discrete Logarithm Problem

INTRODUCTION: In recent times, confidential information transmission over the internet is increased and recommended for higher data security. Cryptography serves as a renowned method to provide sensitive data transmission with a high degree of confidentiality. There are two widely accepted PKC (Public key cryptography) algorithms for cryptographic applications are Rivest-Shamir-Adleman (RSA) and elliptic curve cryptography (ECC) [1]. RSA is based on integer factorization, whose encryption strength depends on the key sizes. ECC is relevant to both the discrete logarithm algorithm and integer factorization families which were first introduced by Koblitz [2] and Miller [3]. ECC has the main features of Discrete Logarithm Problem (DLP) over various points on the elliptic curve which provides complex security. ECC requires a shorter key length than RSA to provide the same level of security. This smaller key size feature makes ECC the best suited for resource-constrained IoT devices as well as high-speed cryptographic processors [4]. ECC offers strong security per bit and provides an efficient hardware implementation in terms of power consumption and speed than other PKC algorithms[5]. In order to implement

the ECC algorithm, there are three choices: software, ASIC and FPGA. FPGA is a perfect hardware implementation platform for a prototype design, considering cost, time consumption, and hardware development facility. Our literature review consists of four segments. First, we place the design options and discuss design flow and its impact on ECC implementation. Second, we compile different approaches and algorithms used in the literature for implementing scalar multiplication. Third, we review and analyze best practices in the literature to implement ECC architectures in the different reconfigurable platforms. Fourth, we summarize the performance enhancement parameters for ECC. Besides, this paper provides a comparison of the different design parameters and hardware platform implementations of ECC. Digital signatures are an indispensable component of modern security protocols like TLS [6], where they are used to authenticate the server and optionally the client too. More specifically, TLS can provide server authentication by means of a certificate that binds an identity (e.g. the server's domain name) to a public key. The certificate contains besides the ID and public key also a collection of attributes, all of which is signed by a trusted third party called Certification Authority (CA). In the initial (i.e. handshake) phase of the TLS protocol, the client normally requests the server's certificate, and if he manages to verify the signature of the CA successfully, he is assured that the public key contained in the certificate is authentic and indeed belongs to the server. The next step is then to establish a shared secret between client and server, which can be done through either RSA-based key transport or via Diffie-Hellman key exchange. In any case, without prior authentication, the key establishment process would be vulnerable to a classical Man-In-The-Middle (MITM) attack.

LITERATURE SURVEY: As one of the first FPGA-based works in ECC-based digital signature, Glas et al. [12] proposed architecture for 128-bit security to integrate into a vehicle-to-vehicle communication system. Furthermore, Panjwani [13] presented a scalable hardware implementation in prime fields over NIST recommended field sizes up to 521 bit, employing hardware– software codesign approach. The work of Vliegen et al. [14] introduced a compact core over the NIST P-256 curve resis tant against simple power analysis (SPA) attacks. Moreover, Zhang and Bai [15] proposed a core with a security level 128 bit over the SM2 curve. Recently, a number of hardware implementations have been introduced to implement an elliptic curve point multiplication (ECPM) core over Curve25519. Sasdrich and Güneysu [16] proposed the first Curve25519 implementation using a DSP-based single-core architecture. This work has been extended by adding side-channel countermeasures in [17] and [18] to provide an evaluation against common physical attacks. In [19], fast and compact implementations of ECPM were proposed. This architecture employs a semisystolic bit-serial multiplier and carry-compact addition to provide a high-performance architecture. The work of Koppermann et al. [20], [21] presented a high-speed prime field multiplier with a latency of 92 μs for a point multiplication. In addition, in [22], a low-latency ECPM was proposed employing a pipelined arithmetic architecture on FPGA and ASIC platforms.

ELLIPTIC CURVE CRYPTOGRAPHY: Elliptic curve cryptography is used to implement public key cryptography. It was discovered by Victor Miller of IBM and Neil Koblitz of the University of Washington in the year 1985. ECC popularly used an acronym for Elliptic Curve Cryptography. It is based on the latest mathematics and delivers a relatively more secure foundation than the first generation public key cryptography systems for example RSA.

Elliptic Curves

In 1985, cryptographic algorithms were proposed based on elliptic curves. An elliptic curve is the set of points that satisfy a specific mathematical equation. They are symmetrical.

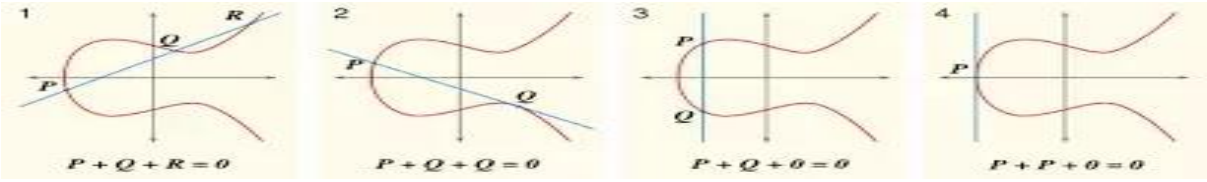

Fig: **Elliptic Curves**

Cryptography supports a number of security aims to provide the privacy of information, non-alteration of information and so on. Because of the high security benefit of cryptography it is broadly used today.

EDWARDS ELLIPTIC CURVE CRYPTOGRAPHY:

Edwards curves were first proposed in 2007 9. Compared to elliptic curves in normal form, Edwards curves can be easier to implement securely because they support complete addition formulas without exceptional cases (division by zero). That is, Edwards curves are faster and simpler to handle than NIST curves 4. Most existing implementations of (twisted) Edwards curves support at most 128-bit security 10, 11, and relatively few have high-security levels. Ed448-Goldilocks is an example of an elliptic curve with high-security level (approximately 224 bits), as suggested by Hamburg 4. The arithmetic is expressed as

$$E : y^2 + x^2 = 1 + dx^2y^2$$

defined over the field $\mathbb{F}_{2^{448}-2^{224}-1}$ with curve parameter $d = -39\,081$. The curve satisfies the SafeCurves policies and criteria, which mitigate security vulnerabilities and flaws in existing NIST curves 12.

EXISTING METHOD:

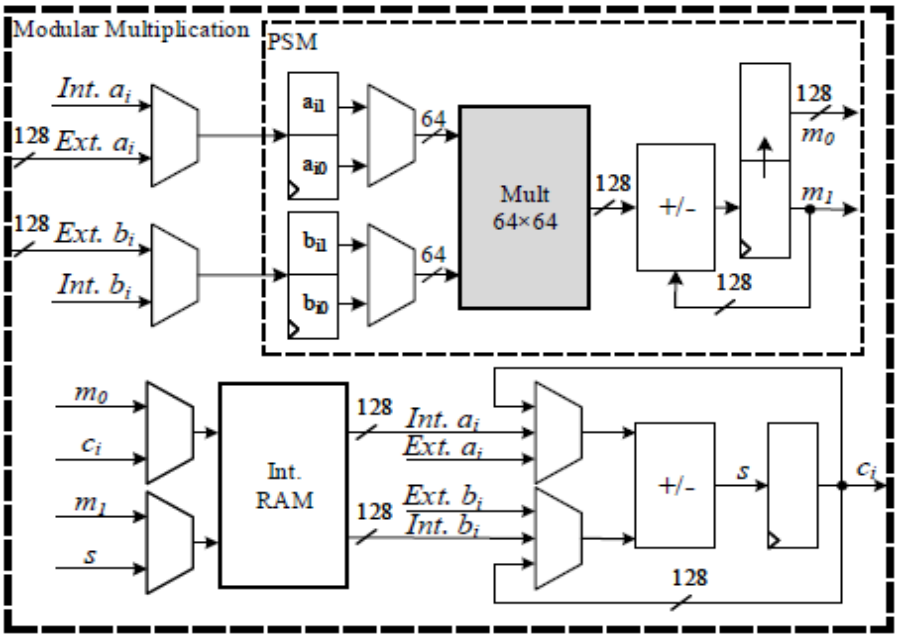

Fig.1 Modular multiplier in the proposed efficient Ed448 scheme

To reduce the computation complexity, we propose performing ECPM over Montgomery curve instead of the Edwards curve. Algorithm 1 describes our proposed Ed448 point multiplication, including four major steps:
Step 1: The base point should be mapped from Edwards domain to Montgomery domain such that:

$$x_{Mont} = y_{Ed}^2 / x_{Ed}^2$$
$$y_{Mont} = y_{Ed} \cdot (2 - x_{Ed}^2 - y_{Ed}^2) / x_{Ed}^3$$

However, as the base point is constant, we assume the Montgomery base point is available without any cost.

_ Step 2: The efficient Montgomery ladder in projective coordinates should be performed to achieve the Montgomery domain result.
_ Step 3: Since computation is implemented using restricted-X coordinate on the Montgomery curve, we have to recover the Y -coordinate result proposed
Step 4: A map from the Montgomery domain is implemented to achieve a result in Edwards domain such that:

$$x_{Ed} = \frac{4 \cdot (x_{Mont}^2 - 1) \cdot y_{Mont}}{(x_{Mont}^2 - 1)^2 + 4 \cdot y_{Mont}^2}$$
$$y_{Ed} = \frac{x_{Mont} \cdot ((x_{Mont}^2 - 1)^2 - 4 \cdot y_{Mont}^2)}{2 \cdot (x_{Mont}^2 + 1) \cdot y_{Mont}^2 - x_{Mont} \cdot (x_{Mont}^2 - 1)^2}$$

Since the dual isogenous map has a degree of 4, the Montgomery ladder in step 2 should be performed for two fewer iterations

HARDWARE ARCHITECTURE: Although Ed448 is implemented over an extended field size of 448-bit long, it provides impressive flexibility to design an efficient architecture for different platforms. Moreover, its special prime with golden ratio 2224 makes it suitable in many security applications employing fast Karatsuba multiplication.

The proposed architecture consists of three stages:

(i) the top stage includes FSM, controller, and ROM,

(ii) the lower stage

includes the field arithmetic logic unit, and

(iii) the middle

stage includes hash function, reduction handlers, memory unit, and secret key buffer. Redundant representation is employed in the proposed Ed448 architecture. Hence, we decompose an integer into four chunks in radix 2448=4 = 2112. Therefore, the data path is considered 128-bit to allow several operations before causing an overflow.
1) Modular Multiplication: Suppose A (and B) is decomposed such that A = A1_2224+A0, Ai = a2i+1_2112+a2i, and ai = ai1_264+ai0. Employing Karatsuba multiplication for the top level results in: C = A_B = (A10 _B10−A0B0)2224+(A1B1+A0B0) (8) where A10 = (A1 + A0) and B10 = (B1 + B0). The authors in [17] introduced the refined Karatsuba identity formula to decrease the number of required addition. Applying refined Karatsuba

identity in middle-level can decompose A0B0, A1B1, and A10B10 to reduce a 225_225-bit multiplication to three 114_114-bit multiplications. For example, A0B0 is decomposed such that:

$$A_0B_0 = (1 - 2^{112}) \cdot (a_0b_0 - 2^{112}a_1b_1) + 2^{112}(a_{10}b_{10})$$

We implement a pipelined schoolbook multiplier (PSM) illustrated in Fig. 1. To control the execution sequence efficiently, we design a specific controller and an internal memory for multiplier using a dedicated ROM and RAM. In our proposed architecture, the partial multiplications are compute with PSM by selecting the operands from the input registers with two multiplexers. Results of the partial multiplications are accumulated into a 256-bit register. Triggering the enable command starts the multiplier to read from the external memory and write the intermediate data to its internal memory. Eventually, applying the last reduction stage prepares the product to store in the external memory. We also design a precise scheduling to increase efficiency presented in Fig. 2. Since PSM requires 4 cycles to read the input registers, the next decomposed part in middle-level is started each 4-cycle. The ith middle-level recombination can be performed after i_3_4 + 5 clock cycles.

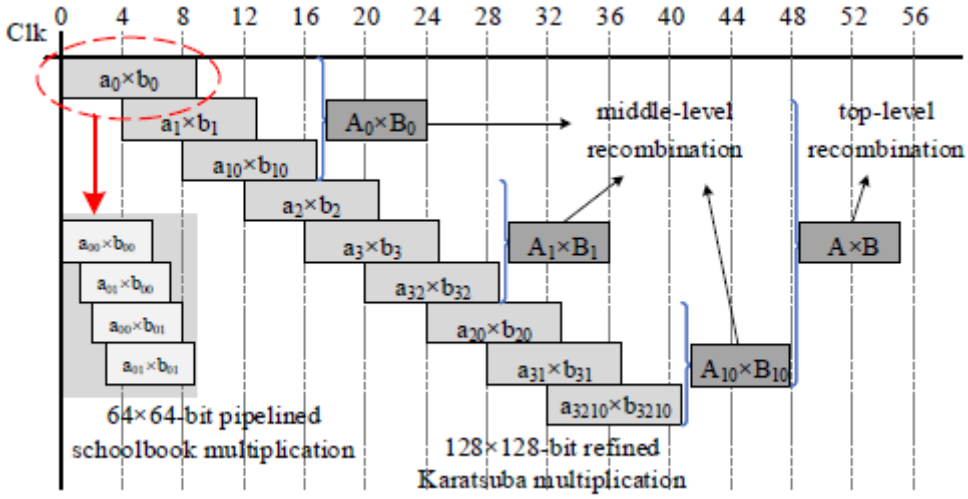

Fig. 2. Timing diagram of the scheduling scheme for A *B

PROPOSED ARCHITECTURE:

RESIDUE NUMBER SYSTEMS: Residue number systems are based on the congruence relation, which is defined as follows. Two integers a and b are said to be congruent modulo m if m divides exactly the difference of a and b; it is common, especially in mathematics tests, to write a ≡ b (mod m) to denote this. Thus, for example, 10 ≡ 7 (mod 3), 10 ≡ 4 (mod 3), 10 ≡ 1 (mod 3), and 10 ≡−2 (mod 3). The number m is a modulus or base , and assume that its values exclude unity, which produces only trivial congruence's. If q and r are the quotient and remainder, respectively, of the integer division of a by m—that is, a = q.m + r—then, by definition, have a ≡ r (mod m). The number r is said to be the residue of a with respect to m, and usually denote this by r = |a|m . The set of m smallest values, {0, 1, 2, . . . , m − 1}, that the residue may assume is called the set of least positive residues modulo m . Unless otherwise specified, assume that these are the only residues in use. Suppose have a set, {m1 , m2 , . . . , mN }, of N positive and pair wise relatively prime moduli5 . Let M be the product of the moduli. Then every number X < M

has a unique representation in the residue number system, which is the set of residues {|X |mi : 1 ≤ i ≤ N }. The number M is called the dynamic range of the RNS, because the number of numbers that can be represented is M . For unsigned numbers, that range is [0, M − 1]. Residue number systems are based on the *congruence* relation, which is defined as follows. Two integers *a* and *b* are said to be *congruent modulo m* if *m* divides exactly the difference of *a* and *b*; it is common, especially in mathematics tests, to write *a* ´ *b* (mod *m*) to denote this. Thus, for example, 10 ´7 (mod 3)*;* 10 ´4 (mod 3)*;* 10 ´1 (mod 3), and 10 ¡2 (mod 3). The number *m* is a *modulus* or *base*, and we shall assume that its values exclude unity, which produces only trivial congruences If *q* and *r* are the quotient and remainder, respectively, of the integer division of *a* by *m*|that is, *a* = *q*:*m* + *r*|then, by definition, we have *a* ´ *r* (mod *m*). The number *r* is said to be the *residue* of *a* with respect to *m*, and we shall usually denote this by *r* = *jajm*. The set of *m* smallest values, *f*0*;* 1*;* 2*; : : : ;m* ¡ 1*g*, that the residue may assume is called the set of *least positive residues modulo m*. Unless otherwise specified, we shall assume that these are the only residues in use. Suppose we have a set, *fm*1*;m*2*; : : : ;mNg*, of *N* positive and pairwise relatively prime moduli5. Let *M* be the product of the moduli. Then every number *X* ≺ *M* has a unique representation in the residue number system, which is the set of residues *fjXjmi* : 1 · *i* · *Ng*. A partial proof of this is as follows. Suppose *X*1 and *X*2 are two different numbers with the same *residue-set*. Then *jX*1*jmi* = *jX*2*jmi* , and so *jX*1 ¡ *X*2*jmi* = 0. Therefore *X*1 ¡ *X*2 is the least common multiple (lcm) of *mi*. But if the *mi* are relatively prime, then their lcm is *M*, and it must be that *X*1 ¡ *X*2 is a multiple of *M*. So it cannot be that *X*1 ≺ *M* and *X*2 ≺ *M*. Therefore, the set *fjXjmi* : 1 · *i* · *Ng* is unique and may be taken as the representationof *X*. We shall write such a representation in the form *hx*1*; x*2*; : : : ; xNi*, where *xi* = *jXjmi* , and we shall indicate the relationship between *X* and its residues by writing *X* »= *hx*1*; x*2*; : : : ; xNi*. The number *M* is called the *dynamic range* of the RNS, because the number of numbers that can be represented is *M*. For unsigned numbers, that range is [0*;M* ¡ 1]. Representations in a system in which the moduli are not pairwise rela- tively prime will be not be unique: two or more numbers will have the samerepresentation. As an example, the residues of the integers zero through fifteen relative to the moduli two, three, and five (which are pairwise rel- atively prime) are given in the left half of Table 1.1. And the residues of the same numbers relative to the moduli two, four, and six (which are not pairwise relatively prime) are given in the right half of the same table. Observe that no sequence of residues is repeated in the first half, whereas there are repetitions in the second. The preceding discussions (and the example in the left-half of Table 1.1) define what may be considered *standard residue number systems*, and it is with these that we shall primarily be concerned. Nevertheless, there are useful examples of \non-standard" RNS, the most common of which ar the *redundant residue number systems*. Such a system is obtained by, essentially, adding extra (redundant) moduli to a standard system. The dynamic range then consists of a \legitimate" range, defined by the non- redundant moduli and an \illegitimate" range; for arithmetic operations, initial operands and results should be within legitimate range. Redundant number systems of this type are especially useful in fault-tolerant computing. The redundant moduli mean that digit-positions with errors may be excluded from computations while still retaining a sufficient part of the dynamic range. Furthermore, both the detection and correction of errors are possible: with *k* redundant moduli, it is possible to detect up to *k* error and to correct up to *bk*=2*c* errors. A different form of redundancy can be introduced by extending the size of the digit-set corresponding to a mod lus, in a manner similar

to RSDs. For a modulus m, the normal digit se is $f0; 1; 2; : : : ;m ¡ 1g$; but if instead the digit-set used is $f0; 1; 2; : : : ; emg$, where $em \ , \ m$, then some residues will have redundant representations.

	Relatively prime moduli			Relatively non-prime moduli		
N	$m_1 = 2$	$m_2 = 3$	$m_3 = 5$	$m_1 = 2$	$m_2 = 4$	$m_3 = 6$
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	0	2	2	0	2	2
3	1	0	3	1	3	3
4	0	1	4	0	0	4
5	1	2	0	1	1	5
6	0	0	1	0	2	0
7	1	1	2	1	3	1
8	0	2	3	0	0	2
9	1	0	4	1	1	3
10	0	1	0	0	2	4
11	1	2	1	1	3	5
12	0	0	2	0	0	0
13	1	1	3	1	1	1
14	0	2	4	0	2	2
15	1	0	0	1	3	3

Table 1: Residues for various moduli

In the RNS based on the set of pairwise relatively prime moduli m1, m2,...,mn (M = Qn i=1 mi), an integer X ∈ [0,M − 1] is represented by an n-tuple of residue digits X → (RNS)(x1, x2,...,xn), where xi = |X|mi is the residue of X modulo mi, i = 1, 2, ...,n . The CRT provides the traditional formula for residue-to-binary conversion

$$\forall X \in [0, M-1] \quad X \rightarrow (\mathrm{RNS})(x_1, x_2, \ldots, x_n):$$
$$X = \left| \sum_{i=1}^{n} N_i \cdot x_i \right|_M,$$
where
$$N_i = M_i \cdot \left| \frac{1}{M_i} \right|_{m_i}, \qquad M_i = \frac{M}{m_i}, \qquad i = 1, 2, \ldots, n.$$

Eq. (1) is the main tool to compute non-modular operations in RNS which require the conversion of the operands. For instance, magnitude comparison between X → (RNS)(x1, x2,...,xn) and Y → (RNS) (y1, y2,...,yn) can be performed by CRT as follows:

Case (a): Magnitude comparison

Step 1. Residue-to-binary conversion of X and Y (Eq. (1)).

Step 2. Compare the binary representations of X and Y . Recently, the 'diagonal function' has been proposed as a new tool to perform non-modular operations in the RNS [6]. In the RNS of moduli m1, m2,...,mn, the 'diagonal function' is defined as [7]: ∀X ∈ [0,M − 1] X → (RNS)(x1, x2,...,xn):

$$\forall X \in [0, M-1] \quad X \rightarrow (\mathrm{RNS})(x_1, x_2, \ldots, x_n):$$
$$D(X) = \left| \sum_{i=1}^{n} k_i \cdot x_i \right|_{SQ},$$

2

where: • SQ = Pn i=1 Mi, is the 'diagonal modulus' of the RNS (Mi = M/mi, i = 1, 2,...,n); • ki = |− 1/mi|SQ, i = 1, 2,...,n (|1/mi|SQ is the multiplicative inverse of mi modulo SQ). From a geometric point of view the 'diagonal function' exploits the geometrical disposal in multidimensional discrete space of the integers in residue representation [6]. If the RNS of n-moduli m1, m2, ...,mn is associated to the n-dimensional discrete space in which

each dimension corresponds to one modulus, each integer X → (RNS)(x1, x2,...,xn) corresponds to the point identified by the n-tuple (x1, x2,...,xn). In this way, integers will be arranged in the n-dimensional discrete space on diagonals parallel to the main diagonal of the space. Moreover, if the diagonals are labeled with increasing integer values as they are traced when the integers of the RNS are counted from 0 to M − 1, the 'diagonal function' provides the label of the diagonal to which an integer belongs [5].

RESULTS:

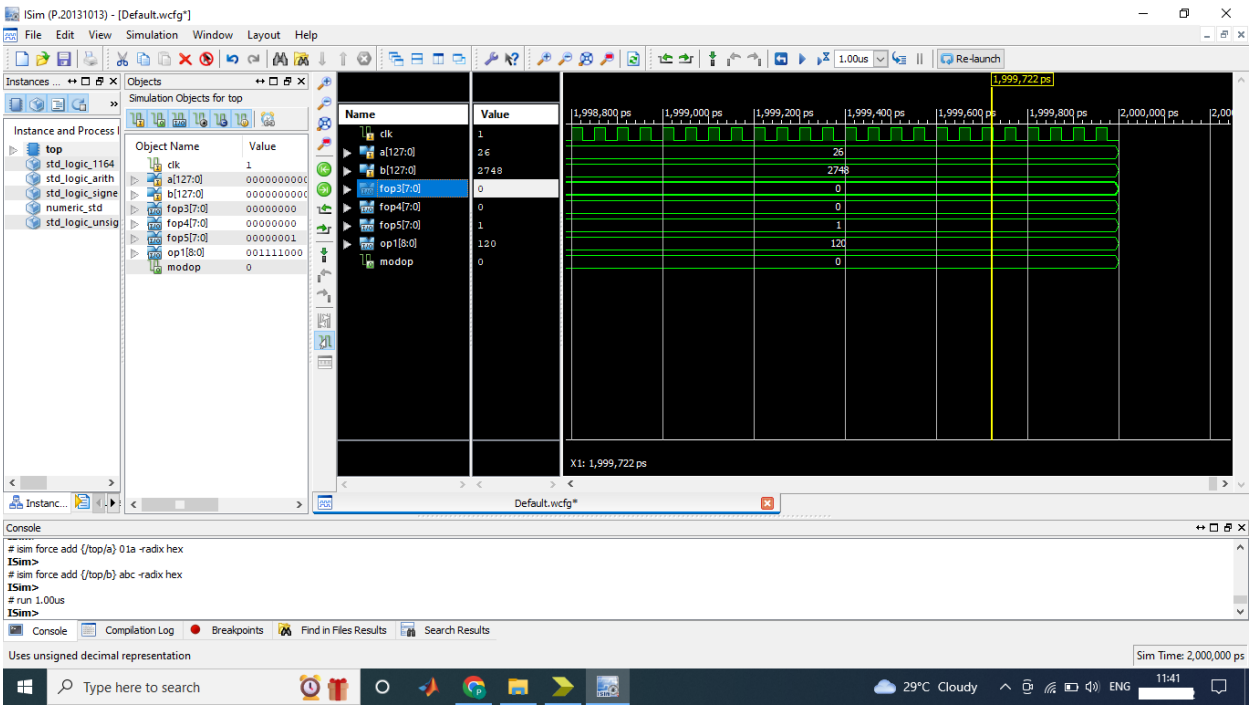

Fig: Proposed simulation RESULT

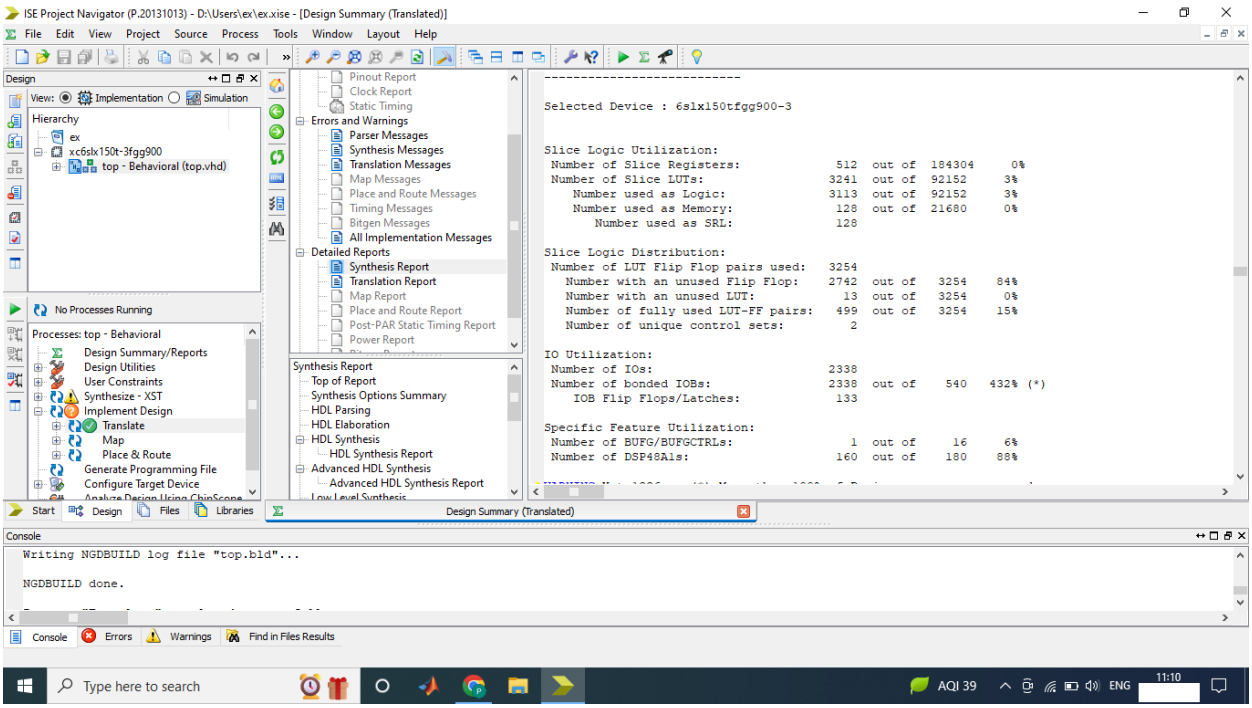

Fig: Total Gate density for existing method

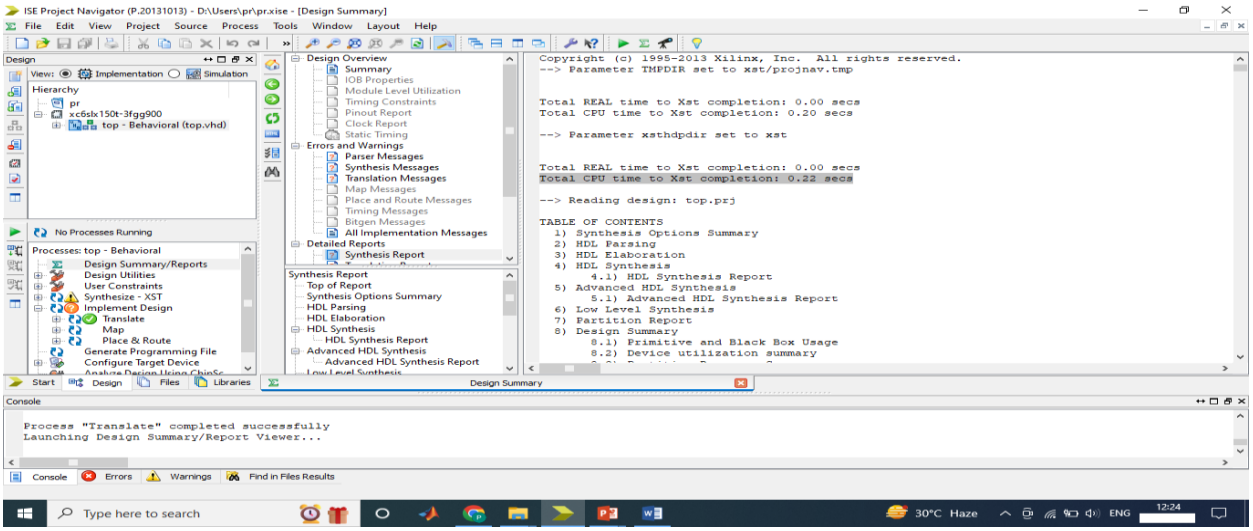

Fig: Total execution time for existing method

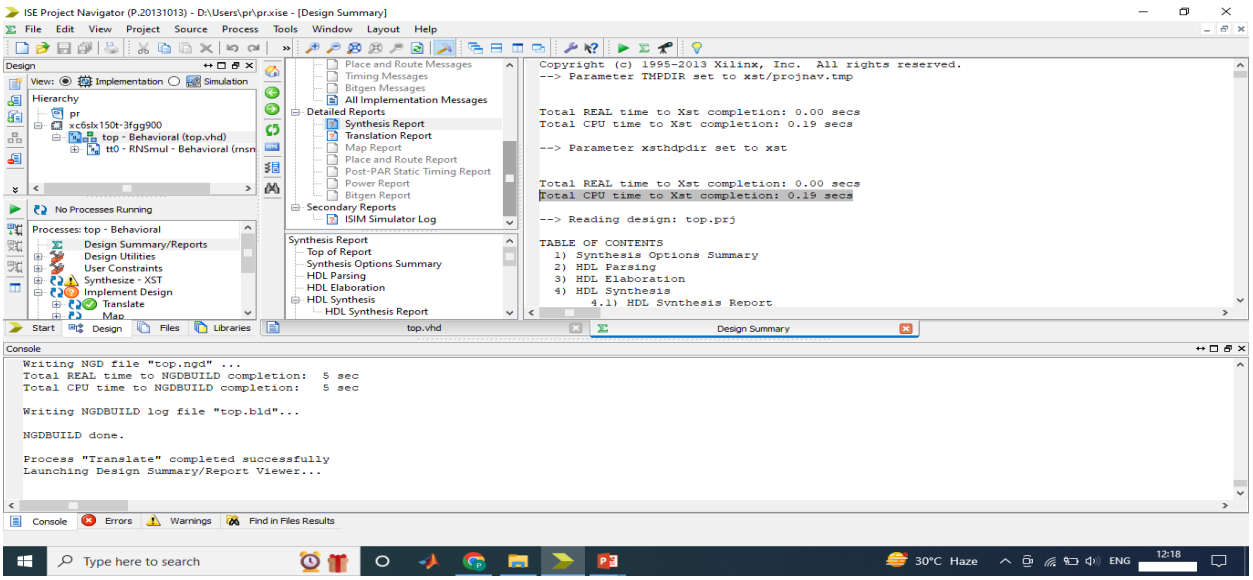

Fig: Total execution time for proposed method

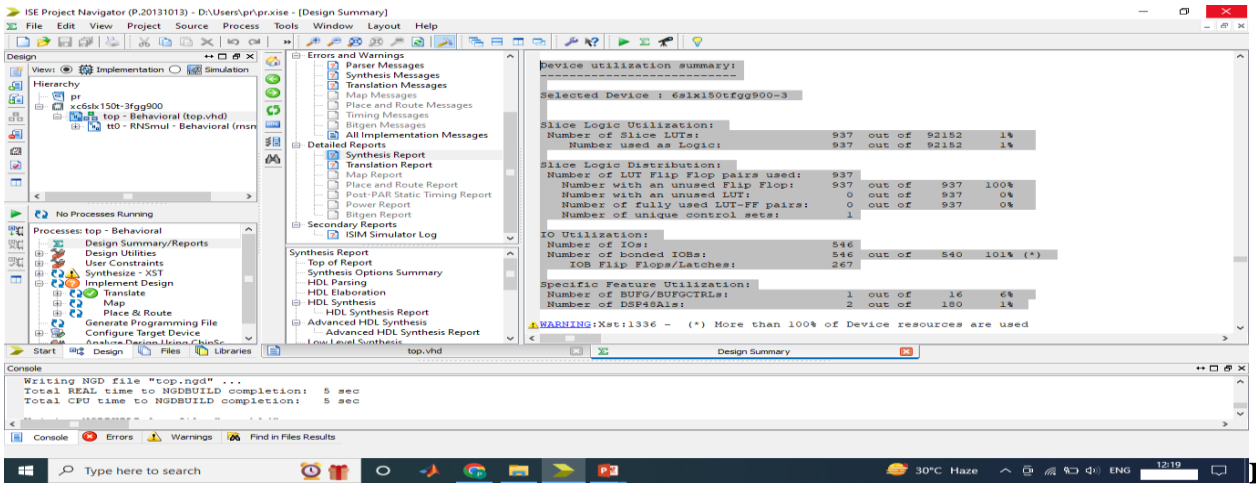

Fig: Total gate density for proposed method

CONCLUSION:

A new class of RNS architectures for the implementation of the 'Modular multiplicaiton' is presented in this paper. The architectures exploit properties of the geometrical disposal in multi-dimensional discrete spaces of integers in residue representation. Compared to the traditional architectures for magnitude based on the 'diagonal function' and on the Chinese Remainder Theorem, the superiority of the new architectures has been shown in terms of waste of hardware and time delay. The proposed method allows more efficient and problematic operations in RNS, such as multiplication, number comparison, and modular values, to be performed. In addition, according to the simulation results, the proposed moduli sets reduce circuit delay in comparison with balanced moduli sets, although, in several cases, require less hardware resources than balanced moduli sets.

FUTURE SCOPE:

This low power, fast and area efficient multiplier can be used for cryptography design, MAC design as an extension to this paper. The hardware implementations of the approximate multiplier including one for the unsigned and two for the signed operations can be done. It can be downloaded into FPGA for further improvements and observations

REFERENCES:

[1] Kerry C.F. and Gallagher P.D., "Digital signature standard (DSS)," FIPS PUB, pp. 186-4, 2013. https://doi.org/10.6028/nist.fips.186-4

[2] Edwards H.M., "A normal form for elliptic curves," Bulletin of the American Mathematical Society, vol. 44, no. 03, pp. 393- 423, 2007. https://doi.org/10.1090/s0273-0979-07-01153-6

[3] Laska M., "An algorithm for finding a minimal Weierstrass equation for an elliptic curve," Mathematics of Computation, vol. 38, no. 157, pp. 257-257, 1982. https://doi.org/10.1090/s0025-5718-1982-0637305-2

[4] DuPont B., Franck C., and Großschädl J., "Fast and Flexible Elliptic Curve Cryptography for Dining Cryptographers Networks," Mobile, Secure, and Programmable Networking, pp. 89-109, 2021. https://doi.org/10.1007/978-3-030-67550-9_7

[5] Kirlar B.B., "Efficient message transmission via twisted Edwards curves," Mathematica Slovaca, vol. 70, no. 6, pp. 1511- 1520, 2020. https://doi.org/10.1515/ms-2017-0444

[6] Islam M.M., Hossain M.S., Hasan M.K., Shahjalal M., and Jang Y.M., "Design and Implementation of High-Performance ECC Processor with Unified Point Addition on Twisted Edwards Curve," Sensors, vol. 20, no. 18, 2020. https://doi.org/10.3390/s20185148

[7] Semmouni M.C., Nitaj A., and Belkasmi M., "Bitcoin security with a twisted Edwards curve," Journal of Discrete Mathematical Sciences and Cryptography, pp. 1-19, 2020. https://doi.org/10.1080/09720529.2019. 1681673

[8] Skuratovskii R. and Osadchyy V., "The Order of Edwards and Montgomery Curves," WSEAS TRANSACTIONS ON MATHEMATICS, vol. 19, pp. 253-264, 2020. https://doi.org/10.37394/23206.2020.19.25

[9] Hisil H. and Renes J., "On Kummer Lines with Full Rational 2-torsion and Their Usage in Cryptography," ACM Transactions on Mathematical Software, vol. 45, no. 4, pp. 1-17, 2019. https://doi.org/10.1145/3361680

[10] Mehrabi M.A. and Doche C., "Low-Cost, LowPower FPGA Implementation of ED25519 and CURVE25519 Point Multiplication," 2019. https://doi.org/10.3390/info10090285

[11] Faz-Hernández A., López J., and Dahab R., "Highperformance Implementation of Elliptic Curve Cryptography Using Vector Instructions," ACM Transactions on Mathematical Software, vol. 45, no. 3, pp. 1-35, 2019. https://doi.org/10.1145/3309759 [12] Hu Z., Gnatyuk S., Kovtun M., and Seilova N., "Method of Searching Birationally Equivalent Edwards Curves Over Binary Fields," Advances in Intelligent Systems and Computing, pp. 309-319, 2018. https://doi.org/10.1007/978-3-319-91008-6_31 [13] Islam M.M., Hossain M.S., Hasan M.K., Shahjalal M., and Jang Y.M., "FPGA Implementation of High-Speed Area- Efficient Processor for Elliptic Curve Point Multiplication Over Prime Field," IEEE Access, vol. 7, pp. 178811-178826, 2019. https://doi.org/10.1109/access.2019.2958491 [14] Seo H. and Kim H., "MoTE-ECC based encryption on MSP430," Journal of Information and Communication Convergence Engineering, vol. 15, no. 10, pp. 160-164, 2017. https://doi.org/10.6109/jicce.2017.15.3.160 [15] Franck C. and Großschädl J., "Efficient Implementation of Pedersen Commitments Using Twisted Edwards Curves," Mobile, Secure, and Programmable Networking, pp. 1-17, 2017. https://doi.org/10.1007/978-3-319-67807-8_1 [16] Liu Z., Großschädl J., Hu Z., Jarvinen K., Wang H., and Verbauwhede I., "Elliptic Curve Cryptography with Efficiently Computable Endomorphisms and Its Hardware Implementations for the Internet of Things," IEEE Transactions on Computers, vol. 66, no. 5, pp. 773-785, 2017. https://doi.org/10.1109/tc.2016.2623609 [17] Karati S. and Das A., "Batch Verification of EdDSA Signatures," Security, Privacy, and Applied Cryptography Engineering, pp. 256-271, 2014. https://doi.org/10.1007/978-3-319-12060-7_17

[18] Liu Z., Weng J., Hu Z., and Seo H., "Efficient Elliptic Curve Cryptography for Embedded Devices," ACM Transactions on Embedded Computing Systems, vol. 16, no. 2, pp. 1-18, 2017. https://doi.org/10.1145/2967103 [19] Naresh V.S., Reddi S., and Allavarpu V.D., "Blockchain-based patient centric health care communication system," International Journal of Communication Systems, vol. 34, no. 7, pp. 34-34, 2021. https://doi.org/10.1002/dac.4749 [20] Saini A., Zhu Q., Singh N., Xiang Y., Gao L., and Zhang Y., "A Smart-Contract-Based Access Control Framework for Cloud Smart Healthcare System," IEEE Internet of Things Journal, vol. 8, no. 7, pp. 5914-5925, 2021. https://doi.org/10.1109/jiot.2020.3032997

[21] Jasem F.M., Sagheer A.M., and Awad A.M., "Enhancement of digital signature algorithm in bitcoin wallet," Bulletin of Electrical Engineering and Informatics, vol. 10, no. 1, pp. 449-457, 2021. https://doi.org/10.11591/eei.v10i1.2339

[22] Sadiq A., Javed M.U., Khalid R., Almogren A., Shafiq M., and Javaid N., "Blockchain Based Data and Energy Trading in Internet of Electric Vehicles," IEEE Access, vol. 9, pp. 7000-7020, 2021. https://doi.org/10.1109/access.2020.3048169

[23] Arulprakash M. and Jebakumar R., "Peoplecentric collective intelligence: decentralised and enhanced privacy mobile crowd sensing based on blockchain," The Journal of Supercomputing, 2021. https://doi.org/10.1007/s11227-021-03756-x

[24] Kavin B.P., Ganapathy S., Kanimozhi U., and Kannan A., "An Enhanced Security Framework for Secured Data Storage and Communications in Cloud Using ECC, Access Control and LDSA," 2020. https://doi.org/10.1007/s11277-020-07613-7

[25] Benil T. and Jasper J., "Cloud based security on outsourcing using blockchain in E-health systems," Computer Networks, vol. 178, pp. 107344-107344, 2020. https://doi.org/10.1016/j.comnet.2020.107344

[26] Wang H., He D., and Ji Y., "Designated-verifier proof of assets for bitcoin exchange using elliptic curve cryptography, "Future Generation Computer Systems, vol. 107, pp. 854-862, 2020. https://doi.org/10.1016/j.future.2017.06.028

[27] Kumar M., Chand S., and Katti C.P., "A Secure End-to-End Verifiable Internet-Voting System Using Identity-Based Blind Signature," IEEE Systems Journal, vol. 14, no. 2, pp. 2032-2041, 2020. https://doi.org/10.1109/jsyst.2019.2940474