International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(14s) 2026 pp. 922-930
https://doi.org/10.70917/ijcisim-2026-4298
Article

A Survey on Optimised Parallel Architectures for RSA Cryptographic Implementations on FPGA and ASIC

Vasudeva G.[1], Likhith Gowda H. R.[2], Bharathi Gururaj[3], Brunda T. S.[4], Dhruthi S.[5]

[1]Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
Email: devan.vasu921@gmail.com
[2] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
[3]Department of Electronics and Communication Engineering, KS Institute of Technology, Bengaluru, Karnataka, India.
Email: bharathigururaj@gmail.com
[4] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
[5] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.

Abstract: Although the RSA public-key cryptosystem is one of the most widely used asymmetric encryption algorithms for securing digital communications, the computational complexity associated with large-integer modular exponentiation makes it difficult to implement for real-time and resource-constrained applications. This paper provides a detailed overview of optimised architectures for implementing RSA cryptographic functions on FPGAs and ASICs. The three interconnected research streams, namely, hardware-accelerated implementations of RSA on FP-GAs and ASICs, algorithmic optimisations for accelerating modular exponentiation based on Chinese Remainder Theorem (CRT) and parallel software-level implementations of modular exponentiation in multi-core and GPU environments are systematically reviewed. The mathematical underpinnings of these optimisations are also discussed, such as Multidimensional CRT, Gaussian inte-ger based modulo samplers and robust reconstruction algorithms. Twenty primary research works from 2000 to 2026 are analyzed and key performance trends, architectural trade-offs, and open research challenges are identified. A comparison of reported implementations is given to inform future design choices in this area.
Keywords: RSA, FPGA, ASIC, modular exponentiation, Montgomery multiplication, Chinese remainder theorem, parallel architecture, hardware accelerator, cryptography

1. INTRODUCTION

Public-key cryptography is a critical component of the secu-rity of modern digital communication. Of the many algorithms in this family, the one introduced by Rivest, Shamir, and Adleman in 1977 (RSA) [1] has remained a standard algorithm for secure key exchange, digital signatures, and encrypted communication over the Internet. It is based on the difficulty of factoring large integers, usually 1024 bits or larger for practical deployments.

But, the security provided by RSA is very expensive in terms of computation. The basic operation of modular expo-nentiation, which is used to encrypt $C = Me \bmod n$ and

decrypt $M = Cd \bmod n$ requires hundreds to thousands of modular multiplications for operands larger than 1024 bits. This computation has non-trivial latency even on modern processors and thus is a bottleneck in high throughput systems like web servers, Internet of Things (IoT) networks, or edge computing nodes. This limitation is

going to be even more pronounced in the future as the key lengths increase to ensure security against increasingly powerful adversaries, such as the future threat of quantum computers using Shor's algorithm [2]. There are two main approaches that have been developed to overcome this challenge. The first is algorithmic: the Chinese Remainder Theorem (CRT) takes advantage of the factored form of the RSA modulus to split one big modular exponentiation into two smaller ones, which can be computed concurrently, and which give a theoretical speedup of roughly 4 ×. The second is architectural: dedicated hardware on FPGAs and ASICs takes advantage of parallelism, pipelining, and optimised arithmetic datapaths, such as Montgomery multi-plication, to gain orders of magnitude over software.

This survey unites these two strategies into one. What archi-tectural and algorithmic decisions determine the performance of FPGA implementations of RSA? (ii) What are the reasons why the performance of these implementations varies?Why is the performance of these implementations different? (iii) What are the implications of these results for the design of high-speed FPGA implementations of RSA? (ii) What are the measurable throughput gains of the integration of the CRT at the hardware level? (iii) What is the current state of parallel software-based RSA, and how does it compare to hardware approaches? (iv) What are the mathematical principles that are still underutilized in hardware? (v) Which are the key open research questions?

The remainder of this paper is organised as follows. The background of RSA and the basics of arithmetic is given in

Section II. The hardware implementations based on FPGAs are surveyed in Section III. In Section IV, the algorithmic optimisations of CRTs are reviewed. The parallel software im-plementations are discussed in Section V. Section VI discusses mathematical foundations. A comparative analysis is given in Section VII. The research gaps are identified in Section VIII. Section IX concludes

2. BACKGROUND

A. RSA Algorithm

The following mathematical structure is the basis for the RSA cryptosystem. Two large prime numbers p and q are chosen, and their product $n = p \times q$ is made public.Two large prime numbers p and q are chosen, and $n = p \times q$ is made public.The public exponent e is chosen such that $\gcd(e, \varphi(n)) = 1$, where $\varphi(n) = (p - 1)(q - 1)$ is Euler's totient function. The private exponent d is the modular inverse of e with respect to $\varphi(n)$. The plaintext M is encrypted to produce the ciphertext $C = Me \bmod n$ and then it is decrypted to produce $M = Cd \bmod n$.

The computational difficulty is modular exponentiation which is usually performed by the square-and-multiply algo-rithm: about $1.5k$ modular multiplications with n-bit operands are needed for a k-bit exponent. With n = 1024, this is thou-sands of large-integer multiplications that require considerable hardware or software resources.

B. Montgomery Multiplication

The modular reduction, $a \bmod n$ is expensive in hardware because it involves trial division. Montgomery's algorithm [3] avoids this by working in a transformed domain. Given an odd modulus N and a constant $R = 2k$ chosen larger than N , the Montgomery form of x is $x\tilde{} = xR \bmod N$. Multiplication in this domain replaces division by N with division by R, which is trivially implemented as a right-shift. The transformation overhead is amortised across the many multiplications in modular exponentiation, making Montgomery multiplication the dominant algorithm for hardware RSA implementations.

C. Chinese Remainder Theorem in RSA

The Chinese Remainder Theorem (CRT) states that for pairwise coprime moduli $m_1, m_2, \ldots, m_k$, any integer x in $[0, m_1 m_2 \cdots m_k)$ can be uniquely reconstructed from its residues $x \bmod m_i$. Applied to RSA, since the private-key owner knows the prime factors p and q of n, decryption can be split as:

$$m_1 = C^{d \bmod (p-1)} \bmod p, \qquad (1)$$

$$m_2 = C^{d \bmod (q-1)} \bmod q. \qquad (2)$$

Garner's formula is used to recover the plaintext M from m_1 and m_2. Each sub-exponentiation is applied to operands of about half the size of n and the two computations are done in parallel, which gives a theoretical speedup of about 4 × over standard decryption.

D. FPGA vs. ASIC for Cryptographic Acceleration

FPGAs are reconfigurable, provide fast prototyping, and have moderate non-recurring engineering (NRE) costs, making them appealing for cryptographic research and flexible deploy-ment. ASICs provide high clock speed, low power and silicon area for a given design, but require significant NRE investment and lack flexibility. Both platforms share the same algorithmic optimisations, but have different implementation constraints.

3. FPGA-BASED RSA IMPLEMENTATIONS

A. Early FPGA Implementations

Early FPGA implementations of RSA were directed towards proving the feasibility of hardware acceleration of modular exponentiation. Gauthama Raman and Kaja Mohideen in [4] have realized RSA on FPGA by using a modified modular exponentiation method. The design, which was implemented in Verilog and was designed for multi-FPGA communication over an RS-232 link, eliminated the need to divide, and proved that FPGA-based RSA can be used in embedded multi-node secure communication.

Abdulrazzaq [5] performed a comparative study of 4 mod-ular multiplication algorithms implemented on FPGA: inter-leaved multiplication (142.42 μs), Montgomery multiplication (110.00 μs), faster Montgomery multiplication (85.51 μs), and modified interleaved multiplication (110.82 μs). The study also compared the performance of the FPGA with Java software: standard RSA in Java took 29.26 ms, while the decryption in Java using the CRT took 6.52 ms, and FPGA-based faster Montgomery took 85.51 μs. This showed that hardware ac-celeration gives about a 10× benefit over optimised software CRT, and more than 300× over naive software, thus giving a good empirical basis for comparing algorithmic and platform options.

B. High-Throughput FPGA Architectures

Xiao et al. [6] introduced a Variable Segmentation Mont-gomery Modular Multiplication (VSMMM) algorithm that adjusts the radix of operand segmentation to the datawidth of the DSP element on any particular FPGA.Their dual-path fully concurrent Montgomery Modular Multiplier (MMM) architec-ture achieved 256-bit MMM in 0.165 μs at 24,899.7 Mbps, and 1024-bit RSA decryption in 1.86 ms at 140.97 Mbps, sig-nificantly outperforming prior FPGA RSA implementations. The important point to note is that adaptive radix selection can maximize the utilization of the DSPs on any target device without the timing problems associated with fixed high-radix designs.

Nishanthi et al. [7] proposed a 1024-bit RSA encryption core using Verilog HDL on Xilinx Artix-7 FPGA for IoT applications. The design was implemented using Montgomery multiplication and square-and-multiply algorithm, and the encryption latency was about 0.012 ms and the throughput was above 80 Mbps at about 172 mW power consumption. UART and Bluetooth interfaces were added to support real-time secure communication between the IoT nodes. Another weakness of software-only implementations is that they are

susceptible to timing-based side-channel attacks, which are inherently prevented by the hardware implementation.

Ghayoula et al. [8] worked on medical imaging applications using a VHDL implementation of the RSA-1024 on the Artix-7 FPGA, with a focus on flexibility in message block size, which can be small, such as for control packets, or large, such as for medical image files. The parameterised library components for Montgomery multiplication and modular ex-ponentiation were used to implement the algorithms, allowing easy adaptation to various security levels and highlighting the applicability of hardware accelerated RSA in domain-specific embedded systems.

C. Architectural Trends

There are some common patterns in the FPGA implemen-tations. The universal adoption of Montgomery multiplication is due to its division-free nature and its pipelining-friendly property. The two most popular implementation languages are Verilog and VHDL, and the most popular evaluation target is the mid-range Xilinx Artix-7 FPGA. The 1024-bit key size is most commonly attacked, although parameterised designs for 512- to 2048-bit operations are becoming more common. One of the main challenges in the works related to IoT is power consumption.

4. CRT-BASED ALGORITHMIC OPTIMISATIONS

A. CRT in Hardware: Foundational Work

The seminal paper introducing the use of CRT for high speed RSA hardware was by Großschadl [9]. The RSA crypto chip was a VLSI custom circuit that included a 1056 × 16-bit word-serial multiplier using Barrett's modular reduction with FastMM algorithm, which cost 1056 × 16 bits.The RSA crypto chip was a VLSI custom circuit with a 1056 × 16-bit word-serial multiplier using Barrett's modular reduction with FastMM algorithm, costing 1056 ×16 bits. The multiplier core worked at 200 MHz and took 227 clock cycles to multiply 1024 bits in a modular unit, resulting in a decryption rate of 560 kbit/s in normal mode. The reconfigurable multi-plier datapath was designed to perform either one 1024-bit exponentiation or two 512-bit exponentiations concurrently, providing the possibility of operating in the CRT mode. In the CRT mode, the two-half width exponentiations are executed simultaneously, which results in a decryption rate of 3.5 times or almost 2 Mbit/s. This work was the first to make the parallelism of CRTs a first-class architectural feature, and the 3.5× speedup, which was slightly lower than the 4× theoretical because of the overhead of Garner's formula, became a standard benchmark.

B. Software CRT Optimisations

Suryawanshi et al. [10] have done quantitative analysis of the performance of the RSA-CRT, and found that the decryption time for CRT is about 4 times that of the standard decryption for the same key size, due to the cubic dependence of the cost of multiplication on the length of operands. This is

to support the use of CRT as the default implementation for private-key operations.

Somsuk et al. [11] introduced an optimised strategy in selecting the CRT equation for RSA decryption. Instead of a fixed formulation of the CRT, their approach calculates the best possible pair of equations for each decryption operation, based on the number of modular multiplications, modular squares, and modular inverses required. The proposed ap-proach consistently outperforms standard CRT by 10–30%, demonstrating that the CRT speedup ceiling has not yet been fully exploited and that equation selection is a meaningful optimisation variable.

Pavani [12] explored three RSA variants—standard RSA, RSA-CRT, and N-Prime RSA with multiple keys—from a unified perspective. N-Prime RSA generalises the two-prime CRT approach by using more than two prime factors, further reducing the cost of each sub-exponentiation and offering speedups beyond the 4× achievable with two primes, at the cost of more complex key generation.

C. CRT as a Unified Framework for Montgomery-Type Algo-rithms

Xu et al. [13] demonstrated that Montgomery-type modular reduction algorithms can be derived from and analysed within the CRT formalism. Starting from Qin's Identity—a number-theoretic result from a 13th-century Chinese mathematical treatise—they showed that the Montgomery REDC algorithm emerges directly from a two-modulus CRT construction. This unified CRT framework enables correctness validation and error detection in recently proposed Montgomery variants, and may simplify the co-design of CRT decomposition and multiplication datapaths in future RSA accelerators.

5. PARALLEL SOFTWARE IMPLEMENTATIONS

A. OpenMP-Based Parallelisation on Multi-Core CPUs

Ayub et al. [14] presented a comprehensive survey of RSA parallelisation strategies from 1978 to 2019, followed by their own OpenMP-based implementation focusing on parallelising modular exponentiation in a High Performance Computing (HPC) environment. The parallel implementation reduced the execution times significantly as compared to the serial RSA. The work pointed out that the square-and-multiply algorithm is inherently sequential, but can be parallelized at a coarser level, such as by dividing the sub-exponentiations of RSA-CRT into independent threads. An open-source implementation was published to enable further research [14].

Al-Ardhi and Abdullah [15] studied the impact of thread count on the parallelisation of RSA using OpenMP more systematically. They implemented the partitioning of the ex-ponentiation operations over the available cores, and obtained significant speedup compared to serial RSA. The study showed that after a certain point, the increase in thread count does not provide a significant benefit because of the overhead of synchronisation, and that the modular reduction step is the main limiting factor for further parallelisation. The use of algorithmic CRT decomposition in conjunction with OpenMP thread parallelism was suggested for the best possible perfor-mance.

B. GPU-Based and Heterogeneous Parallelisation

Liu et al. [16] proposed a variant of the RSA scheme (PMKRSA) which divides plaintext into multiple chunks, each of which is encrypted using a different key pair. This chunking method is naturally suited to be parallelised on a GPU, where each chunk can be handled by a different CUDA core.Implemented on CPU-only and CPU+GPU (CUDA) plat-forms, results for 100 MB files showed encryption time drop from 0.2476 s on CPU to 0.0009 s on GPU, and decryption from 9.4476 s to 0.0618 s, speedups of approximately 275× and 153× respectively. Beyond performance, the multi-key structure provides a security advantage as compromising a single key pair reveals only one message chunk.

C. FPGA-Accelerated Homomorphic Encryption

Agrawal et al. [17] presented a hardware arithmetic li-brary for Ring Learning with Errors (RLWE)-based somewhat homomorphic encryption (SHE) on FPGA. While not RSA-specific, this work implements the Residue Number System (RNS), Chinese Remainder Theorem, and NTT-based poly-nomial multiplication—arithmetic primitives shared between RSA and post-quantum cryptographic schemes. The FPGA prototype delivered a 4200× and 2950× speedup over software for homomorphic multiplication and homomorphic addition, respectively, demonstrating the potential of hardware CRT and NTT acceleration for cryptographic workloads in general.

6. MATHEMATICAL FOUNDATIONS

A. Multidimensional CRT for Integer Vectors

In addition to the scalar CRT, there is an area of work that has emerged, the Multidimensional Chinese Remainder Theo-rem (MD-CRT), which is the reconstruction of integer vectors from their remainders, modulo integer matrices. Xiao, Xia, and Wang [18] introduced the MD-CRT and its powerful variant for integer vectors, which have exact reconstruction formulas and have conditions on the error bounds of remainders for robust reconstruction when the remainders are corrupted, a property essential for fault-tolerant cryptographic hardware. Xiao, Huo, and Xia [19] extended this to a general set of moduli without commutativity restrictions, with a closed-form reconstruction algorithm and application to multidimensional frequency estimation.

Guo and Xia [20] further generalised the MD-CRT to reconstruct multiple integer vectors simultaneously from mul-tiple residue sets, resolving fundamental questions about the uniquely determinable range and establishing conditions for maximal dynamic range. A companion paper [21] constructed new families of pairwise coprime integer matrices of arbitrary dimension with closed-form least common right multiples—a combinatorial prerequisite for practical MD-CRT-based sys-tems.

B. Gaussian Integers and Complex-Valued CRT

Gong, Gan, and Liu [22] proposed multi-channel modulo samplers using complex-valued Gaussian integer moduli for sampling band-limited complex signals, exploiting a complex-valued analogue of CRT to enable high dynamic range signal recovery at low sampling rates. Li et al. [23] extended this to a maximum likelihood estimation framework for complex-valued robust CRT with Gaussian integer moduli, applicable to multichannel self-reset ADC systems.

C. Relevance to Cryptographic Hardware

The advances in MD-CRT and Gaussian integer CRT are relevant to cryptographic hardware in two principal ways. First, Residue Number System arithmetic—a natural hard-ware realisation of the scalar CRT principle already exploited in FPGA homomorphic encryption accelerators [17]—may benefit from MD-CRT extensions toward more flexible RNS bases and error-resilient implementations. Second, multi-prime RSA [12] can benefit from MD-CRT reconstruction algo-rithms for more efficient Garner-formula computation across many primes. These connections remain largely unexploited in current hardware implementations and represent a promising future research direction.

7. COMPARATIVE ANALYSIS

Table I summarises the twenty surveyed works across key dimensions: platform, key size, technique, and principal re-sults.

Several patterns emerge from Table I. FPGA throughput has progressed from sub-1 Mbps in early ASIC designs [9] to over 140 Mbps in modern FPGA designs [6], driven by architectural advances in multiplier design. CRT consistently delivers 3.5–4× speedup across software and hardware contexts, with further improvement possible through optimised equation se-lection [11]. FPGA implementations outperform software-only approaches by 1–3

orders of magnitude, while GPU-based implementations [16] approach hardware-level performance for bulk operations but require a modified cryptographic structure. The most advanced mathematical work on MD-CRT and Gaussian integer CRT [18], [20], [23] has not yet been translated into RSA hardware, representing a significant research opportunity.

8. RESEARCH GAPS AND FUTURE DIRECTIONS

Based on the survey, the following research gaps are iden-tified.

Limited modern ASIC-specific designs. The seminal ASIC work by Großschadl [9] remains the primary reference for custom silicon RSA in this survey. Modern deep-submicron processes and new algorithmic insights (VSMMM, MD-CRT) offer significant untapped potential for high-performance ASIC RSA cores.

Incomplete hardware–CRT integration. Most FPGA im-plementations use standard modular exponentiation without on-chip CRT decomposition. Designs that deeply integrate CRT decomposition with the Montgomery multiplication

TABLE I

COMPARATIVE SUMMARY OF SURVEYED RSA AND CRT IMPLEMENTATIONS

Ref	Year	Authors	Platform	Key Size	Technique	Key Results
[9]	2000	Großschadl	ASIC (VLSI)	1024-bit	Barrett/FastMM + CRT	560 kbit/s (std); ≈2 Mbit/s (CRT); 3.5×
[4]	2014	Gauthama Raman et al.	FPGA	N/S	Modified mod. exp.	Multi-FPGA secure comm.; area-efficient
[5]	2018	Abdulrazzaq	FPGA	32–1024	Montgomery, Interleaved, CRT	Faster Montgomery: 85.5 μs; FPGA ≈ 300× over software
[14]	2019	Ayub et al.	HPC Multi-core	1024-bit	OpenMP exponentiation	Significant speedup vs serial; scales with cores
[17]	2020	Agrawal et al.	FPGA (RLWE)	N/A (HE)	RNS, CRT, NTT	4200× (HE mult.); 2950× (HE add.)
[18]	2020	Xiao, Xia, Wang	Mathematical	N/A	Exact & Robust MD-CRT	Exact and robust reconstruction with error bounds
[12]	2021	Pavani	Software	Variable	RSA, RSA-CRT, N-Prime	≈ 4× (CRT); further gain with N-prime
[16]	2021	Liu et al.	CPU + GPU(CUDA)	Multi-key	PMKRSA, chunked RSA	Enc: 0.0009 s (GPU) vs 0.2476 s (CPU); ≈ 275×
[22]	2021	Gong, Gan, Liu	Signal Proc.	N/A	Gaussian integer CRT	Complex CRT-based high-DR signal recovery
[19]	2021	Xiao, Huo, Xia	Mathematical	N/A	Robust MD-CRT	Closed-form reconstruction; real-vector generalisation
[6]	2022	Xiao et al.	FPGA	1024-bit	VSMMM, dual-path MMM	1.86 ms, 140.97 Mbps; MMM: 24,899.7 Mbps

[11]	2023	Somsuk et al.	Software	Variable	Optimal CRT equations	10–30% faster than standard CRT
[10]	2024	Suryawanshi et al.	Software	Variable	RSA-CRT analysis	≈ 4× speedup confirmed
[15]	2024	Al-Ardhi, Abdullah	Multi-core CPU	1024-bit	OpenMP parallel RSA	Speedup with threads; diminishing returns at high count
[8]	2025	Ghayoula et al.	FPGA (Artix-7)	1024-bit	VHDL, Montgomery	Medical imaging; flexible block size
[13]	2025	Xu et al.	Mathematical	N/A	CRT-Montgomery framework	Unified framework; error detection in prior designs
[20]	2025	Guo, Xia	Mathematical	N/A	Generalised MD-CRT	Max dynamic range; multi-vector reconstruction
[21]	2025	Guo, Xia	Mathematical	N/A	Coprime integer matrices	Closed-form LCRM; new matrix families
[7]	2026	Nishanthi et al.	FPGA (Artix-7)	1024-bit	Verilog, Montgomery	0.012 ms latency; >80 Mbps; 172 mW; IoT
[23]	2026	Li et al.	Signal Proc.	N/A	MLE Complex CRT	Robust estimation; SR-ADC application

datapath—as done for ASIC in [9]—are needed for modern targets.

MD-CRT in cryptographic hardware. The rich MD-CRT theory [18]–[20] has not been applied to RSA hardware. RNS-based RSA implementations exploiting MD-CRT reconstruc-tion could offer new throughput-area trade-offs.

Post-quantum readiness. Shor's algorithm [2], if realised on a sufficiently powerful quantum computer, would break RSA. Hybrid architectures accelerating both classical RSA and lattice-based post-quantum algorithms—sharing NTT, CRT, and modular arithmetic primitives—are an important emerging direction.

IoT and edge constraints. Lightweight FPGA implemen-tations trading throughput for power efficiency, suitable for battery-powered edge nodes, remain underexplored beyond the work of Nishanthi et al. [7].

Side-channel resistance. None of the surveyed software-parallel implementations explicitly address side-channel vul-nerabilities such as timing attacks. Hardware implementations offering both high throughput and formal side-channel resis-tance are needed.

Benchmarking standardisation. The surveyed works use inconsistent benchmarking conditions—different key sizes, clock frequencies, and target devices—making direct compar-

ison difficult. A standardised benchmark suite would signifi-cantly benefit the community.

9. CONCLUSION

This paper has surveyed optimised parallel architectures for RSA cryptographic implementations on FPGA and ASIC, covering twenty primary research works spanning 2000 to 2026. Three research streams were identified and reviewed: hardware FPGA and ASIC implementations, CRT-based algo-rithmic optimisations, and parallel software implementations. Mathematical foundations in MD-CRT and Gaussian integer arithmetic that underpin future advances were also examined.

The key findings are as follows. Montgomery multiplication is the universal hardware primitive for modular arithmetic in RSA. CRT integration at the hardware level delivers 3.5–4× decryption speedup with room for further improvement through optimal equation selection. Modern FPGA implemen-tations have achieved throughputs

exceeding 140 Mbps at 1024 bits. GPU parallelisation offers near-hardware perfor-mance for bulk operations. Substantial mathematical advances in MD-CRT remain to be translated into RSA hardware implementations.

The field is at an inflection point: traditional RSA must evolve toward hybrid post-quantum designs, domain-specific

constraints from IoT and healthcare are reshaping performance targets, and the mathematical tools for deeper CRT integra-tion are available but not yet applied. Future work bridging the mathematical–hardware gap and developing standardised, domain-aware, and side-channel-resistant RSA accelerators will be of significant practical value.

ACKNOWLEDGMENT

The author thanks the Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, for supporting this research.

References

1. R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.
2. P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in Proc. 35th Annual Symposium on Foundations of Computer Science, IEEE, 1994, pp. 124–134.
3. P. L. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, vol. 44, no. 170, pp. 519–521, 1985.
4. M. R. Gauthama Raman and S. Kaja Mohideen, "Modified modular exponentiation for a faster implementation of RSA algorithm on FPGA," in Int. Journal of Engineering Research & Technology (IJERT), NCI-CCT'14 Conference Proceedings, 2014.
5. Z. A. Abdulrazzaq, "FPGA based RSA algorithm implementation," M.Sc. Dissertation, Dept. of Computer and Information Engineering, University of Mosul, 2018.
6. H. Xiao, S. Yu, B. Cheng, and G. Liu, "FPGA-based high-throughput Montgomery modular multipliers for RSA cryptosystems," IEICE Elec-tronics Express, vol. 19, no. 9, pp. 1–6, 2022.
7. K. S. Nishanthi, M. Venkateswar Reddy, G. Bhargav Ram, and
8. R. Venu Gopal, "RSA encryption core implementation on FPGA for secure data transmission in IoT networks," Int. Journal of Emerging Research in Science, Engineering, and Management, vol. 2, no. 3,
9. pp. 90–102, Mar. 2026.
10. R. Ghayoula, S. Bahroun, W. Amara, J. Fattahi, A. Smida, I. El Gmati, and W. Mohamed, "Optimized RSA-1024 FPGA implementation on Artix-7," in Advances in Cryptography – Foundations, Techniques and Applications, IntechOpen, 2025.
11. J. Großschadl, "The Chinese remainder theorem and its application in a high-speed RSA crypto chip," Graz University of Technology, 2000.
12. Y. Suryawanshi, S. Khade, D. B. Bhoyar, and A. R. Bhagat Patil, "Chinese remainder theorem based performance analysis of RSA cryp-tosystem," Int. Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 8, 2024.
13. K. Somsuk, S. Atsawaraungsuk, C. Suwannapong, S. Khummanee, and C. Sanemueang, "The optimal equations with Chinese remainder theorem for RSA's decryption process," 2023.
14. K. Pavani, "Enhancing public key cryptography using RSA, RSA-CRT and N-prime RSA with multiple keys," in Proc. Third Int. Conf. on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV 2021), IEEE, 2021.
15. G. Xu, Y. Jia, and Y. Yang, "Chinese remainder theorem approach to Montgomery-type algorithms," arXiv:2402.00675v3 [cs.CR], Feb. 2025.
16. Md. A. Ayub, Z. A. Onik, and S. Smith, "Parallelized RSA algorithm: An analysis with performance evaluation using OpenMP library in high performance computing environment," in Proc. 22nd Int. Conf. on Computer and Information Technology (ICCIT), Dec. 2019.
17. R. A. Al-Ardhi and M. F. Abdullah, "Enhancing parallel implementation of RSA algorithm using OpenMP," vol. 29, no. 2, 2024.
18. J.-J. Liu, K.-T. Tsang, and Y.-H. Deng, "A variant RSA acceleration with parallelisation," arXiv:2111.11924v1 [cs.DC], Nov. 2021.
19. R. Agrawal, L. Bu, A. Ehret, and M. A. Kinsy, "Fast arithmetic hardware library for RLWE-based homomorphic encryption," arXiv:2007.01648v1 [cs.CR], Jul. 2020.
20. L. Xiao, X.-G. Xia, and Y.-P. Wang, "Exact and robust reconstructions of integer vectors based on multidimensional Chinese remainder the-orem (MD-CRT)," IEEE Transactions on Signal Processing, vol. 68,
21. pp. 5349–5364, 2020.
22. L. Xiao, H. Huo, and X.-G. Xia, "Robust multidimensional Chinese remainder theorem for integer vector reconstruction," IEEE Transactions on Signal Processing, pp. 2364–, 2021.

23. G. Guo and X.-G. Xia, “A generalized multidimensional Chinese remain-der theorem (MD-CRT) for multiple integer vectors,” IEEE Transactions on Signal Processing, vol. 73, pp. 5136–, 2025.
24. G. Guo and X.-G. Xia, “A construction of pairwise co-prime integer matrices of any dimension and their least common right multiple,” IEEE Transactions on Signal Processing, vol. 73, pp. 2187–, 2025.
25. Y. Gong, L. Gan, and H. Liu, “Multi-channel modulo samplers con-structed from Gaussian integers,” IEEE Signal Processing Letters, vol. 28, pp. 1828–1832, 2021.
26. X. Li, S. Sun, Q. Liao, and X.-G. Xia, “Maximum likelihood estimation-based complex-valued robust Chinese remainder theorem and its fast algorithm,” IEEE Journal on Miniaturization for Air and Space Systems, vol. 7, no. 1, Mar. 2026.