CLEI Electronic Journal, Volume 27, Issue 3, paper 3, August 2024

High-Performance Elliptic Curve Cryptography: A SIMD Approach to Modern Curves*

Armando Faz Hernandez
Institute of Computing, University of Campinas.
1251 Albert Einstein, Cidade Universitária. Campinas, SP. Brazil.
armfazh@ic.unicamp.br

and

Julio López
Institute of Computing, University of Campinas.
1251 Albert Einstein, Cidade Universitária. Campinas, SP. Brazil.
jlopez@ic.unicamp.br

Abstract

Cryptography based on elliptic curves is endowed with efficient methods for public-key cryptography. Recent research has shown the superiority of the Montgomery and Edwards curves over the Weierstrass curves as they require fewer arithmetic operations. Using these modern curves has, however, introduced several challenges to the cryptographic algorithm's design, opening up new opportunities for optimization.

Our main objective is to propose algorithmic optimizations and implementation techniques for cryptographic algorithms based on elliptic curves. In order to speed up the execution of these algorithms, our approach relies on the use of extensions to the instruction set architecture. In addition to those specific for cryptography, we use extensions that follow the Single Instruction, Multiple Data (SIMD) parallel computing paradigm. In this model, the processor executes the same operation over a set of data in parallel. We investigated how to apply SIMD to the implementation of elliptic curve algorithms. As part of our contributions, we design parallel algorithms for prime field and elliptic curve arithmetic. We also design a new three-point ladder algorithm for the scalar multiplication $P + kQ$, and a faster formula for calculating $3P$ on Montgomery curves. These algorithms have found applicability in isogeny-based cryptography. Using SIMD extensions such as SSE, AVX, and AVX2, we develop optimized implementations of the following cryptographic algorithms: X25519, X448, SIDH, ECDH, ECDSA, EdDSA, and qDSA. Performance benchmarks show that these implementations are faster than existing implementations in the state of the art.

Our study confirms that using extensions to the instruction set architecture is an effective tool for optimizing implementations of cryptographic algorithms based on elliptic curves. May this be an incentive not only for those seeking to speed up programs in general but also for computer manufacturers to include more advanced extensions that support the increasing demand for cryptography.

Keywords: cryptography, elliptic curves, SIMD, parallel computing, secure software.

1 Introduction

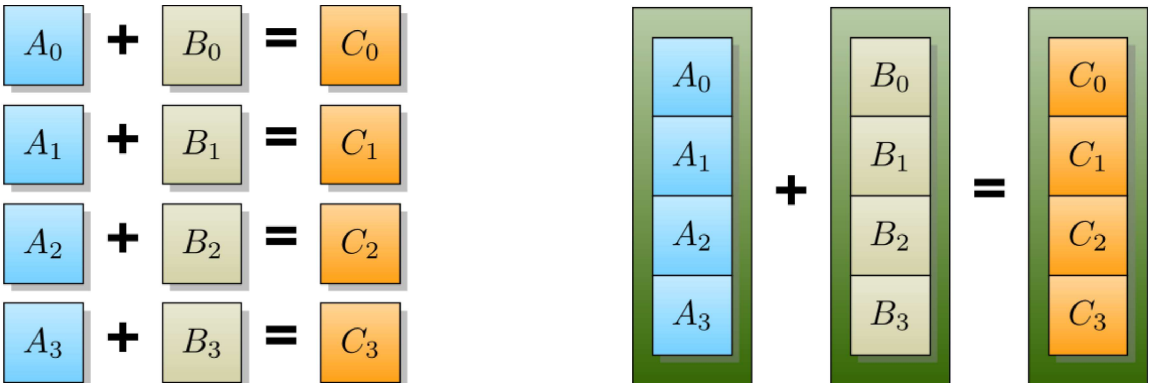
Extensive research efforts have focused on delivering public-key cryptography securely and efficiently. Cryptography based on elliptic curves provides efficient methods due to the use of keys shorter than the ones used in well-known cryptosystems, such as the Rivest-Shamir-Adleman (RSA) [3] and in those based in

* Summary of a doctoral thesis authored by Armando Faz Hernandez and supervised by Julio López. The full text [1] can be found at `https://hdl.handle.net/20.500.12733/6756`. This is the extended version of a previous summary [2].

the Discrete Logarithm Problem [4]. Elliptic curve cryptography has been endorsed by international standardization agencies [5, 6, 7]. Despite these standards have been in circulation for several years, a recent line of research proposes a shift to new elliptic curves with the aim of improving efficiency while preserving high-security guarantees.

With the avalanche of novel elliptic curve proposals, such as the ones highlighted by [8], new challenges have appeared. Former investigations focused on elliptic curves given in the Weierstrass model; however, there is still room for optimizing the operations of alternative elliptic curve models, such as the *Montgomery* curves [9] and the *Edwards* curves [10]. New algorithms for these curves must likely be adapted, or otherwise reformulated considering the upsides and downsides of each model. New improvements could arise by analyzing the algorithms from theoretical, computational, and practical standpoints. Therefore, the pathway for designing cryptographic algorithms, their implementation, and their put in practice is currently in progress.

From the computational perspective, a compelling approach for improving performance is using extensions to the instruction set architecture. There exist extensions that support the *Single Instruction, Multiple Data* (SIMD) paradigm characterized in Flynn's taxonomy [11] of parallel computing. In this model, a *vector instruction* encodes an operation that is executed over several data units simultaneously. Figure 1 contrasts the number of instructions used in scalar processing versus vector processing.



(a) Scalar Processing: Four scalar instructions perform the workload

(b) Vector Processing: One vector instruction performs the same workload

Figure 1: SIMD vector instructions

Historically, SIMD processing has been shown effective in the high-performance computing area applied to graphics processing, scientific computing, mathematical simulation, among other domains. In the early days, SIMD execution units were exclusive of large workstations and supercomputers; but nowadays, SIMD vector units [12, 13, 14] can be found at commodity computers and portable devices. Figure 2 shows the increasing addition of extensions to the instruction set architecture and their applicability to different domains. Hundreds of instructions have been added in order to support SIMD processing for integer and floating-point arithmetic. As can be seen, more recently instructions target more specific domains, for example, the inclusion of extensions tailored to accelerate cryptographic algorithms [15].

1.1 Problem Statement

The widespread availability of SIMD execution units in commodity computers, Internet servers, and mobile devices motivates their application to the implementation of cryptographic algorithms. Nonetheless, a few resources explain how to use SIMD units efficiently, and even fewer are dedicated to the case of elliptic curve cryptography, and the secure software development required in this domain. Moreover, it is unclear to what extent these computational resources can help to improve efficiency.

It is interesting to know how to effectively apply SIMD processing to the implementation of elliptic curve cryptography. Especially to the algorithms derived from the recent proposals of elliptic curves and making use of the most advanced vector instructions [14] and other extensions found in contemporary computer architectures. For this reason, it is imperative to investigate how to design new algorithms and data structures (or adapt the existing ones) so that implementations take full advantage of SIMD processing.

Thesis Statement We claim that the execution of algorithms for elliptic curve cryptography can be accelerated through a combination of algorithmic optimizations, implementation techniques, and the use of SIMD processing and other hardware extensions.

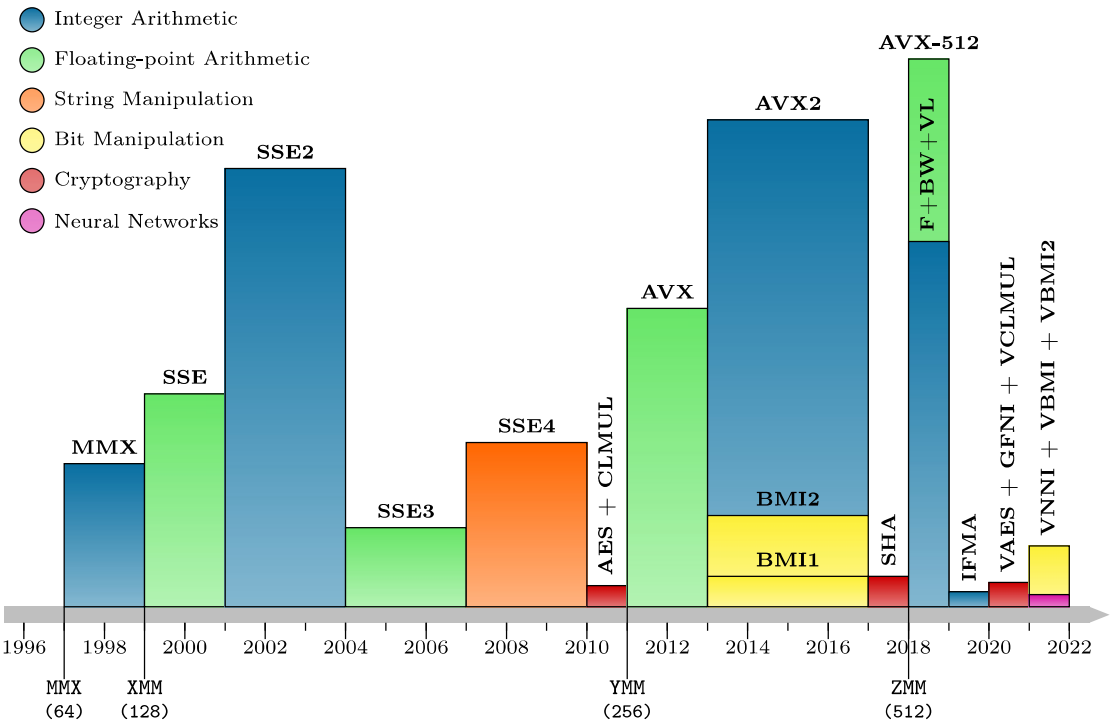


Figure 2: Evolution of SIMD Instructions. Each bar represents an instruction set showing its release date (in the horizontal dimension), the number of new instructions (in the vertical dimension), and the main application (in the chromatic dimension). The milestones show the release date of large vector registers.

1.2 Aims

To support this assumption, we investigate algorithmic optimizations and look for implementation techniques for elliptic curve algorithms emphasizing the application of SIMD parallel processing.

An objective of our study is to close the gap between theory and practice. For instance, in addition to proposing parallel algorithms, we also cover their implementation in software. We highlight some issues arisen during development and propose some solutions for them. The design of our proposed algorithms considers the capabilities and limitations of the computer architecture studied.

Our research aims to enlighten a pathway for applying SIMD efficiently. Current computer architectures support hundreds of SIMD instructions including SSE, AVX, AVX2, AVX512, and others. Moreover, the number of instructions is gradually increasing in the upcoming computer architectures (cf. Figure 2). Part of this research is to give guidance on the use of SIMD instructions, to identify some of their limitations, and to show how to apply them to elliptic curve cryptography.

2 Contributions

Our contributions are the union of several layers of improvements comprising algorithmic optimizations for elliptic curve cryptography, practical implementation techniques for SIMD vector processing of mathematical field operations, and the immediate applicability of our findings into current information security standards. Now, we briefly describe these contributions. More details can be found in the full text available at [1].

2.1 Algorithmic Optimizations

For Montgomery curves, we introduced a new *Three-point Ladder Algorithm* that calculates the x-coordinate of $P + kQ$, where P, Q are points on an elliptic curve and k is an integer. Our algorithm improves in three aspects. First, it saves one third of arithmetic operations than the previously-known three-point ladder algorithm [16, 17]. Second, when P and Q are known in advance, the algorithm allows faster execution by employing precomputation. Third, when precomputation is used, fetching precomputed values from memory requires non-secret indexes, which naturally prevents against side-channel attacks. Figure 3 exemplifies the operation of the multiplication algorithm. We showed immediate application of this algorithmic optimization in several cryptographic schemes such as in Isogeny-based Cryptography, the Diffie-Hellman (DH) protocol [18], and the qDSA digital signature scheme.

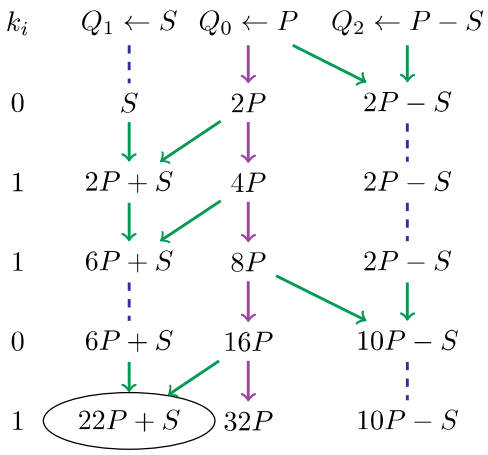


Figure 3: New Three-point Ladder Algorithm. Example of the execution flow for calculating $22P + S$ given P, S, $P - S$ and $k = 22$.

Application to Isogeny-based Cryptography Isogeny-based Cryptography is a branch of cryptography looking for secure algorithms resistant against adversaries with quantum computing power. In this scenario, elliptic curves can be seen as the vertex set of a regular graph, and the edges are *isogenies*, algebraic maps that link two elliptic curves. Given a starting elliptic curve, it is easy to reach to other curve by computing an isogeny; however, finding the path of isogenies that connects two arbitrary elliptic curves in the graph is known to be hard using either a classical or a quantum computer [19].

Formulas to compute isogenies are known [20] and rely on the knowledge of the kernel of an isogeny. Such a kernel is often calculated as $P + kQ$, where P and Q are points on an elliptic curve, and k is a secret integer chosen uniformly at random. An algorithm that computes this operation using only the x-coordinate of the points also requires knowledge of a third point $P - Q$, hence the name of three-point ladder.

We found direct application of our algorithm for computing the kernel of isogenies. Previous algorithms requires two differential additions plus on differential doubling operations. In contrast, ours only requires one differential addition and one doubling per bit of the integer scalar. Theoretically, our algorithm saves around a third of the arithmetic operations to compute the same result. In practice, we verify these improvements in the Supersingular Isogeny Diffie-Hellman (SIDH) protocol [19]. The algorithm and its implementation were published in the IEEE Transactions of Computing journal [21], and Table 1 is reproduced from that article showing the timings measured in Haswell and Skylake micro-architectures.

Table 1: Timings of SIDH-751

Operation		Haswell			Skylake		
		CLN [16]	FLOR [21]	Speedup	CLN [16]	FLOR [21]	Speedup
Key Generation	Alice	48.3	38.0	1.27×	35.7	26.9	1.33×
	Bob	54.5	42.8	1.27×	39.9	30.5	1.31×
Shared Secret	Alice	45.7	34.3	1.33×	33.6	24.9	1.35×
	Bob	52.8	39.6	1.33×	38.4	28.6	1.34×

[1] Timings of the SIDH v2 implementation are 10^6 clock cycles.

SQISign is another quantum-resistant algorithm that requires the calculation of isogenies. SQISign is a digital signature scheme [22], recently proposed in the NIST Post-Quantum Competition [23], has one of the shortest key and signature sizes in comparison to other contenders. However, the execution of this scheme is slow and needs more optimizations to be used in practical scenarios such as in the TLS protocol.

We investigate the impact of our algorithm in SQISign. The verification of a signature validates that the challenge isogenies are computed honestly, to do so a number of isogenies are computed by first computing its kernel. Like for SIDH, our three-point ladder algorithm performs this task. So, we took the reference implementation of SQISign [22] and measure the time taken to verify a signature. The measurements are shown in Table 2 for the security parameters of SQISign Level 1. As can be seen, verification is 9 % faster due to the use of our three-point ladder algorithm. Other operations such as key generation and signing still require more investigation to improve their execution time.

Table 2: Timings of SQISign Level 1 with Three-point Ladder

Three-point Ladder Algorithm	Jao-De Feo [19]	Ours [21]
Signature Verification	20×10^6 cycles	18×10^6 cycles

For Montgomery curves, we showed an *optimized formula for tripling points*, that is, given a point P, to calculate $3P$. This operation is relevant for multi-base scalar multiplication methods as well as in the SIDH protocol. SIDH requires to evaluate $3^n P$ for an integer $n > 0$. By applying our formula, we reduce the total number of field operations by an observable margin. We acknowledge some trade-offs against formulas independently proposed in [24].

Application to Diffie-Hellman Protocol The Diffie-Hellman protocol can be efficiently instantiated with Montgomery curves. Montgomery curves allow a faster computation because operations use only the x-coordinates of points, so operations are correct up to the sign. The central operation in elliptic curve cryptography is called scalar multiplication. So, given an integer k and a point $P = (x_P, y_P)$ on the curve, a scalar multiplication algorithm calculates $Q = kP = (x_Q, y_Q)$. Montgomery devised a faster algorithm for computing x_Q given only k and x_P. Fortunately, the x-coordinate is more than enough to accomplish the Diffie-Hellman protocol.

We noted that the three-point ladder can be used to compute kP. Our approach relies on an auxiliary point S, so we first compute $Q' = S + kP$ using the three-point ladder. For security, it is required to remove any low order points, so a multiplication by h, the curve's order cofactor, one can get rid of the point S. We apply this to the X25519 and X448 Diffie-Hellman protocols and we also specialize the algorithm to the case when the input point is known in advance. Details about these contributions were published in the SAC 2017 paper [25].

Application to qDSA The qDSA signature scheme also uses the fast arithmetic of Montgomery curves. In this case, signing requires to perform a scalar multiplication kP using only the x-coordinates of the points. We applied a similar approach as described for Diffie-Hellman resulting on 35 % faster signatures by employing the three-point ladder algorithm. Moreover, we proposed an alternative signature verification procedure that checks for a stronger signature verification. More details can be found in the SPACE 2017 paper [26].

2.2 Implementation Techniques

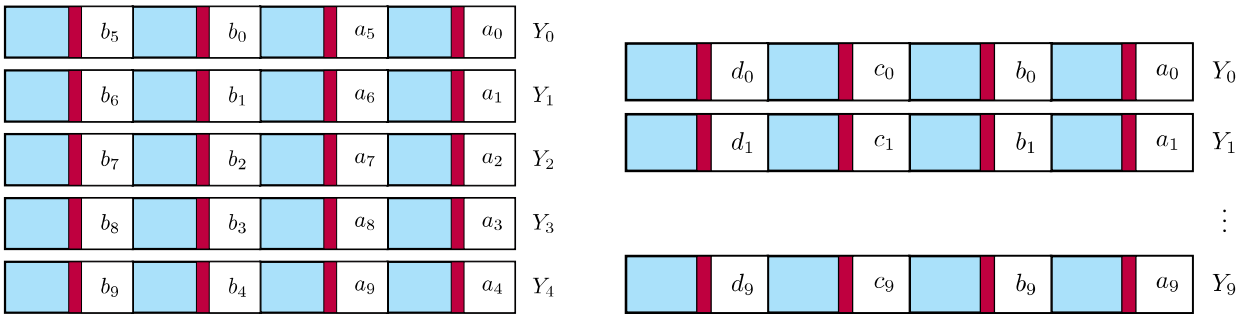
On the availability of SIMD and other extensions to the instruction set architecture, we speed up implementations of arithmetic operations over prime fields and elliptic curves.

SIMD Implementation of Prime Field Arithmetic We initially focus on the SIMD parallel processing of prime field arithmetic. To do so, we showed *data structures* and *representation of numbers* suitable for parallel processing. Our study covered four families of prime moduli corresponding to the ones used in recent proposals of new elliptic curves. For each family, we showed how to perform field operations using scalar and vector instructions. Our benchmark analysis showed that improvements in performance are more significant when operating over larger numbers.

For smaller prime fields, manipulating data inside vector registers using permutation instructions has a negative effect on performance. This is explained because in the targeted computer architecture, the AVX2 permutation instructions have a higher latency than other vector instructions. Thus, the vectorized implementation of smaller prime fields suffers of a notorious performance overhead limiting the amount of improvement.

A better approach that employs vector units more efficiently is taking the SIMD's essence to higher abstraction levels. We follow the notion of *n-way operations* using the n words of a vector register for calculating *n field operations* in parallel. This approach was motivated due to the overheads of using SIMD instructions to perform single field operations. Figure 4 shows how to distribute the individual words of a prime field element into a vector register. For example, note that one can store data units in such a way to perform either two-way or four-way operations. Using the AVX-512 instruction set, one can extend this idea further to prepare data for performing eight-way operations.

Armed with n-way field operations, we turned our attention to investigate parallel algorithms for elliptic curve arithmetic that benefit from them.



(a) $\langle A, B\rangle$: Data storage for a two-way operation

(b) $\langle A, B, C, D\rangle$: Data storage for a four-way operation

Figure 4: Data structures for SIMD operations

SIMD Implementation of Elliptic Curve Arithmetic We target the formulation of parallel algorithms for elliptic curves in three different models: Weierstrass, Montgomery, and Edwards curves.

For Weierstrass curves, we adapt the $\mathbb{F}_q$-complete formulas for point addition, in such a way that point addition is performed by two parallel units, enabling the application of two-way field operations. The explicit algorithms are described in the SBSeg 2016 paper [27] together with timings for the P-384 curve.

For Montgomery curves, we showed how to calculate the Montgomery ladder step using two parallel units. The common implementation strategy for these two curve models consists on using the 256-bit AVX2 vector unit for simulating two 128-bit parallel units. Thus, each 128-bit unit can also be seen as two 64-bit parallel units that are dedicated to field arithmetic. We follow this approach because it reduces the use of expensive permutation instructions; thus, minimizing the overheads observed in the vectorization of smaller prime fields. The findings for Montgomery curves were published at Latincrypt 2017 paper [28].

For Edwards curves, we focused on parallel algorithms for point addition, point doubling, and scalar multiplication. Our implementation strategy is to perform four-way operations using the 256-bit vector unit. Then, we developed parallel algorithms for point addition (and doubling) using four-way field operations. Additionally, we constructed a four-way point addition that allowed us to perform parts of the scalar multiplication in parallel. The design of all parallel algorithms has the purpose to minimize the use of costly permutation instructions. The combined application of these strategies results on the acceleration of scalar multiplications that ultimately inject an speed up to the higher level cryptographic algorithms.

Optimized Implementation of Cryptographic Algorithms Building on top of the improvements on the prime field arithmetic and the elliptic curve arithmetic, we found their direct applicability for speeding up some cryptographic algorithms. We developed vectorized implementations of the ECDH and ECDSA with the P-384 curve; the X25519, X448, and Supersingular Isogeny Diffie-Hellman protocols; and the EdDSA and qDSA digital signature schemes. In all cases, we observed improvements on performance by using vector instructions.

We show timings of our implementations of the Diffie-Hellman protocol and digital signatures. Part of these results were published in the ACM Transactions on Mathematical Software journal [29]. For the Diffie-Hellman protocol, Table 3 shows timings of the X25519 and X448 protocols.

Table 3: Timings of the X25519 and X448 protocols

Instance	Operation	Haswell	Skylake	Tiger Lake
X25519	Key Generation	43.7	34.5	18.2
	Shared Secret	121.0	99.4	50.7
X448	Key Generation	129.0	107.7	53.7
	Shared Secret	428.1	364.2	168.1

[1] Entries are 10^3 clock cycles.

For digital signatures, the dominant operation of EdDSA is elliptic curve scalar multiplication. Figure 5 shows a breakdown of the internal operations of the EdDSA signature scheme. We contrast the timings measured by our vectorized implementation. Ed25519's latency gets reduced by 19 % for signing and 24 % for signature verification. Table 4 shows timings for the operations of the Ed25519 and Ed448 schemes.

In February 2023, the National Institute of Standards and Technology [30] has approved the standardization of EdDSA. In practical terms, it means that EdDSA is endorsed to be used on Internet communications

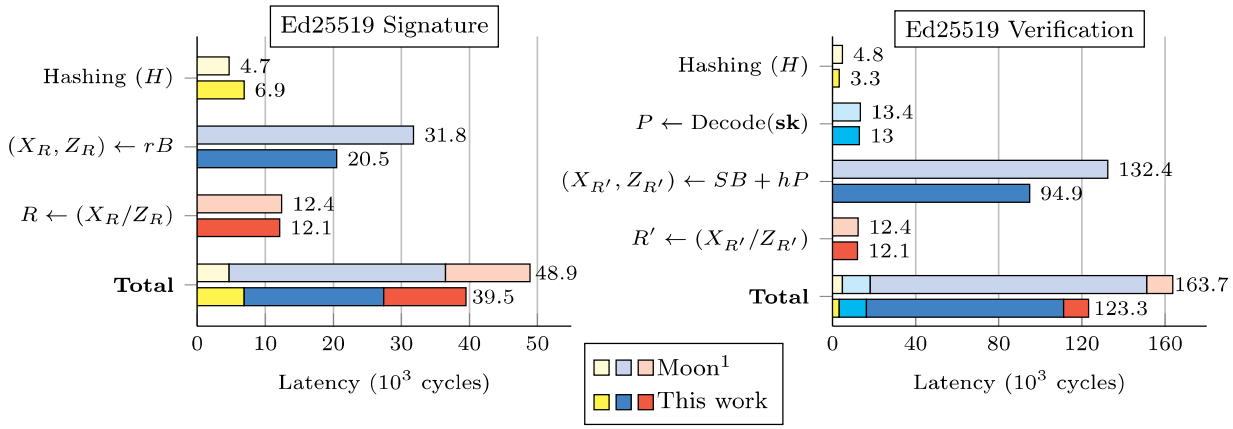


Figure 5: Performance comparison of Ed25519

massively. This is relevant to secure communication protocols such as SSL/TLS, SSH, VPN, and others. Our contributions on accelerating the performance of this algorithm are pertinent.

Table 4: Timings of Ed25519 and Ed448 schemes

Instance	Operation	Haswell	Skylake	Tiger Lake
Ed25519	Key Generation	42.8	34.8	18.4
	Signing	48.6	39.5	20.1
	Verification	156.0	123.3	77.1
Ed448	Key Generation	126.7	104.9	54.8
	Signing	132.7	110.1	57.4
	Verification	465.8	409.5	193.6

[1] Entries are 10^3 clock cycles.

Implementations using other Extensions In addition to the SIMD extensions, we studied the efficient application of other hardware extensions such as BMI2, ADX, and SHA-NI instructions.

We developed efficient implementations of field arithmetic using the MULX instruction and the ADCX/ADOX instructions, which are part of, respectively, the BMI and ADX instruction sets. Using these instructions, our implementations render better performance than using basic instructions. Nonetheless, our vectorized implementations offer superior improvements to prime fields of larger size.

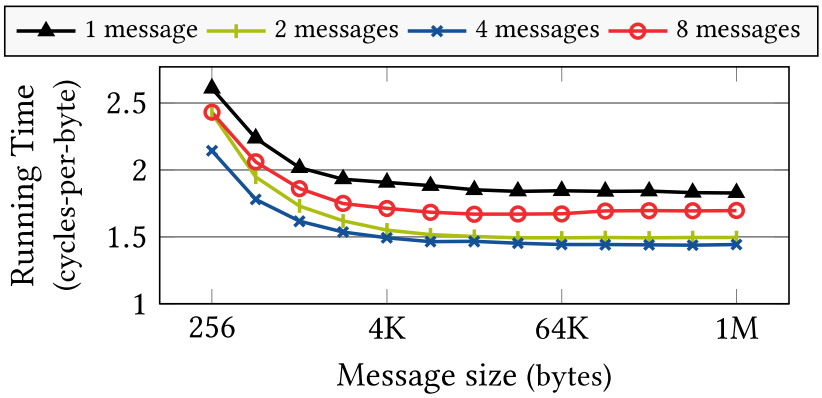
The availability of SHA-NI allowed us to evaluate the performance of the SHA-256 cryptographic hash function. We developed a pipelined implementation that performs a four-way version of the SHA-256 function. We applied this function to the XMSS and XMSS$^{\mathrm{MT}}$ hash-based signatures, which are in the portfolio of quantum-resistant algorithms. Using SHA-NI, we observed that signature operations run up to four times faster than using a non-hardware aided implementation. Moreover, the performance is slightly better than implementing a four-way version with SIMD vector instructions. More details about these results are published in the ACM AsiaPKC paper [31].

2.3 Software Libraries

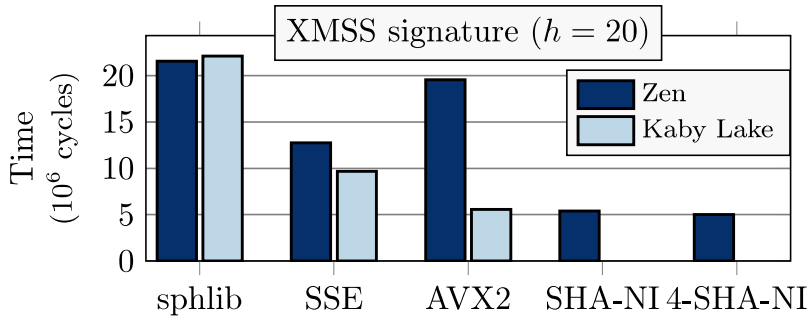
We developed a set of software libraries that exhibit implementation techniques and optimizations of the cryptographic algorithms mentioned above. Source code was optimized for SIMD processing, and performance benchmarks provide evidence of their superiority. To enable reproducibility, our libraries are available at public repositories and released under permissive software licenses. The code is also available at an institutional repository:

`https://gitlab.ic.unicamp.br/ra142685/phd_libs/`

Third-party Usage Due to its improved efficiency, our x64 implementation of X25519 was included in the implementation of the Wireguard protocol [32]. Wireguard offers a VPN-like secure communication tunel to connect between remote machines. Wireguard was recently included in the kernel of Linux.



(a) Performance comparison SHA-256 hash (multiple buffer)



(b) Performance comparison of XMSS

Figure 6: Performance benchmark on Zen and Kaby Lake micro-architectures

As part of the EverCrypt project, Protzenko et al. [33] formally verified a x64 implementation of X25519, which is based on our implementation published at SAC 2017 [25] conference. Formal methods allow proving the correctness of the code and brings higher assurances to the development of cryptographic software.

Our three-point ladder algorithm and other implementation techniques were adopted by SIKE [34] and SQISign [22], contenders of the NIST's Post-Quantum Cryptography Standardization project [23].

3 Conclusions

Based on the experimentation performed in our investigation, we conclude that the application of SIMD vector instructions does reduce the execution time of both prime field operations and elliptic curve arithmetic resulting in observable improvements in high-level cryptographic algorithms. However, we remark that in order to get better performance several changes in the algorithms are needed. Some of them are naturally motivated by the SIMD parallel computing paradigm, but others arose from the instruction set used.

Our investigation provided explicit optimizations and implementation techniques that resulted in a faster execution of cryptographic algorithms than those existent in the state of the art. Our software implementations render better performance when using AVX2 vector instructions for the X25519 and X448 Diffie-Hellman protocols, and the Ed25519 and Ed448 digital signature schemes. We also identified trade-offs and limitations of these developments, which can provide insights for future improvements. We hope our work and the ideas presented motivate to students and researchers on future projects.

References

[1] A. Faz-Hernandez, "High-Performance Elliptic Curve Cryptography: A SIMD Approach to Modern Curves," Ph.D. dissertation, University of Campinas, Campinas, Brazil, sep 2022, https://hdl.handle.net/20.500.12733/6756.

[2] A. Faz-Hernandez and J. López, "High-Performance Elliptic Curve Cryptography: A SIMD Approach to Modern Curves (Thesis Summary)," in *Anais do XXXVI Concurso de Teses e Dissertações*. Porto Alegre, RS, Brasil: SBC, 2023, pp. 20–29. [Online]. Available: https://doi.org/10.5753/ctd.2023.230156

[3] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: https://doi.org/10.1145/359340.359342

[4] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," pp. 10–18, 1985. [Online]. Available: https://doi.org/10.1007/3-540-39568-7_2

[5] NIST, "Digital Signature Standard (DSS)," National Institute of Standards and Technology, Tech. Rep., Jan. 2000, http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf.

[6] American National Standards Institute, "Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)," Tech. Rep. ANSI X9.62-1998, 1998. [Online]. Available: https://webstore.ansi.org/Standards/ASCX9/ANSIX9621998

[7] "IEEE Standard Specifications for Public-Key Cryptography," pp. 1–228, Aug. 2000. [Online]. Available: https://doi.org/10.1109/IEEESTD.2000.92292

[8] D. J. Bernstein and T. Lange, "SafeCurves: choosing safe curves for elliptic-curve cryptography," http://safecurves.cr.yp.to accessed 20 March 2015, 2015.

[9] P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization," *Mathematics of Computation*, vol. 48, no. 177, pp. 243–264, 1987. [Online]. Available: https://doi.org/10.1090/S0025-5718-1987-0866113-7

[10] D. J. Bernstein, P. Birkner, M. Joye, T. Lange, and C. Peters, "Twisted Edwards Curves," in *Progress in Cryptology – AFRICACRYPT 2008*, ser. Lecture Notes in Computer Science, S. Vaudenay, Ed., vol. 5023. Springer Berlin Heidelberg, 2008, pp. 389–405. [Online]. Available: https://doi.org/10.1007/978-3-540-68164-9_26

[11] M. Flynn, "Very high-speed computing systems," *Proceedings of the IEEE*, vol. 54, no. 12, pp. 1901–1909, Dec 1966. [Online]. Available: https://doi.org/10.1109/PROC.1966.5273

[12] S. Thakkar and T. Huff, "Internet Streaming SIMD Extensions," *Computer*, vol. 32, no. 12, pp. 26–34, 1999, http://doi.org/10.1109/2.809248. [Online]. Available: https://doi.org/10.1109/2.809248

[13] ARM, "ARM NEON Intrinsics," https://developer.arm.com/architectures/instruction-sets/intrinsics.

[14] Intel Corporation, "Intel® Advanced Vector Extensions Programming Reference," https://software.intel.com/sites/default/files/m/f/7/c/36945, Jun. 2011.

[15] S. Gueron, "Intel's New AES Instructions for Enhanced Performance and Security," in *Fast Software Encryption: 16th International Workshop, FSE 2009 Leuven, Belgium, February 22-25, 2009 Revised Selected Papers*, O. Dunkelman, Ed. Berlin, Heidelberg: Springer, 2009, pp. 51–66. [Online]. Available: https://doi.org/10.1007/978-3-642-03317-9_4

[16] C. Costello, P. Longa, and M. Naehrig, "Efficient Algorithms for Supersingular Isogeny Diffie-Hellman," in *Advances in Cryptology – CRYPTO 2016*, M. Robshaw and J. Katz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 572–601. [Online]. Available: https://doi.org/10.1007/978-3-662-53018-4_21

[17] D. Jao, L. De Feo, and J. Plût, "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies," *Journal of Mathematical Cryptology*, vol. 8, no. 3, pp. 209–247, Sep. 2014. [Online]. Available: https://doi.org/10.1007/978-3-642-25405-5_2

[18] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976. [Online]. Available: https://doi.org/10.1109/TIT.1976.1055638

[19] D. Jao and L. De Feo, "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies," in *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 – December 2, 2011. Proceedings*, B.-Y. Yang, Ed. Berlin, Heidelberg: Springer, 2011, pp. 19–34. [Online]. Available: https://doi.org/10.1007/978-3-642-25405-5_2

[20] J. Velú, "Isogénies entre courbes elliptiques," *C. R. Acad. Sci. Paris Sér*, vol. 273, no. A-B, pp. A238–A241, 1971.

[21] A. Faz-Hernández, J. López, E. Ochoa-Jiménez, and F. Rodríguez-Henríquez, "A Faster Software Implementation of the Supersingular Isogeny Diffie-Hellman Key Exchange Protocol," *IEEE Transactions on Computers*, vol. 67, no. 11, pp. 1622–1636, Nov 2018. [Online]. Available: https://doi.org/10.1109/TC.2017.2771535

[22] L. De Feo, D. Kohel, A. Leroux, C. Petit, and B. Wesolowski, "SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies," in *Advances in Cryptology – ASIACRYPT 2020*, S. Moriai and H. Wang, Eds. Cham: Springer International Publishing, 2020, pp. 64–93. [Online]. Available: https://doi.org/10.1007/978-3-030-64837-4_3

[23] NIST, "Post-Quantum Cryptography Standardization," National Institute of Standards and Technology, Gaithersburg, MD, USA, Dec. 2016, https://www.nist.gov/pqcrypto.

[24] C. Costello and H. Hisil, "A Simple and Compact Algorithm for SIDH with Arbitrary Degree Isogenies," in *Advances in Cryptology – ASIACRYPT 2017*, T. Takagi and T. Peyrin, Eds. Cham: Springer International Publishing, 2017, pp. 303–329. [Online]. Available: https://doi.org/10.1007/978-3-319-70697-9_11

[25] T. Oliveira, J. López, H. Hışıl, A. Faz-Hernández, and F. Rodríguez-Henríquez, "How to (Pre-)Compute a Ladder," in *Selected Areas in Cryptography – SAC 2017*, C. Adams and J. Camenisch, Eds. Cham: Springer International Publishing, 2018, pp. 172–191. [Online]. Available: https://doi.org/10.1007/978-3-319-72565-9_9

[26] A. Faz-Hernández, H. Fujii, D. F. Aranha, and J. López, "A Secure and Efficient Implementation of the Quotient Digital Signature Algorithm (qDSA)," in *Security, Privacy, and Applied Cryptography Engineering*, S. S. Ali, J.-L. Danger, and T. Eisenbarth, Eds. Cham: Springer International Publishing, 2017, pp. 170–189. [Online]. Available: https://doi.org/10.1007/978-3-319-71501-8_10

[27] A. Faz-Hernández and J. López, "Speeding up Elliptic Curve Cryptography on the P-384 Curve," in *XVI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, vol. 16. Sociedade Brasileira de Computação – SBC, Nov 2016, pp. 170–183. [Online]. Available: https://doi.org/10.5753/sbseg.2016.19306

[28] ——, "Fast Implementation of Curve25519 Using AVX2," in *Progress in Cryptology – LATINCRYPT 2015*, ser. Lecture Notes in Computer Science, K. Lauter and F. Rodríguez-Henríquez, Eds. Springer International Publishing, Aug. 2015, vol. 9230, pp. 329–345. [Online]. Available: https://doi.org/10.1007/978-3-319-22174-8_18

[29] A. Faz-Hernández, J. López, and R. Dahab, "High-performance Implementation of Elliptic Curve Cryptography Using Vector Instructions," *ACM Transactions on Mathematical Software (TOMS)*, vol. 45, no. 3, pp. 1–35, July 2019. [Online]. Available: https://doi.org/10.1145/3309759

[30] NIST, "Digital Signature Standard (DSS)," National Institute of Standards and Technology, Tech. Rep. FIPS PUB 186-5, Feb. 2023, https://doi.org/10.6028/NIST.FIPS.186-5.

[31] A. Faz-Hernandez, J. López, and A. K. D. S. de Oliveira, "SoK: A Performance Evaluation of Cryptographic Instruction Sets on Modern Architectures," in *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop*, ser. APKC '18. New York, NY, USA: ACM, Jun. 2018, pp. 9–18. [Online]. Available: https://doi.org/10.1145/3197507.3197511

[32] J. A. Donenfeld, "Wireguard Linux Compat," Feb. 2018, patch to Wireguard: https://git.zx2c4.com/wireguard-linux-compat/commit/src/crypto/curve25519-x86_64.h?id=186be2742c948351c27bc068102252e10a28959b.

[33] J. Protzenko, B. Parno, A. Fromherz, C. Hawblitzel, M. Polubelova, K. Bhargavan, B. Beurdouche, J. Choi, A. Delignat-Lavaud, C. Fournet, N. Kulatova, T. Ramananandro, A. Rastogi, N. Swamy, C. M. Wintersteiger, and S. Zanella-Beguelin, "EverCrypt: A Fast, Verified, Cross-Platform Cryptographic Provider," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 983–1002. [Online]. Available: https://doi.org/10.1109/SP40000.2020.00114

[34] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. D. Feo, B. Hess, A. Jalali, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, J. Renes, V. Soukharev, D. Urbanik, and G. Pereira, "Supersingular Isogeny Key Encapsulation," Nov. 2017. [Online]. Available: https://sike.org